



SLVS-EC Receiver IP

User Guide

FPGA-IPUG-02125-2.0

August 2024

Disclaimers

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

Inclusive Language

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

Contents

Contents.....	3
Abbreviations in This Document.....	6
1. Introduction.....	7
1.1. Overview of the IP.....	7
1.2. Quick Facts.....	7
1.3. Features.....	7
1.4. Licensing and Ordering Information.....	7
1.4.1. Ordering Part Number.....	7
1.5. IP Validation Summary.....	8
1.6. Minimum Device Requirements.....	8
1.7. Naming Conventions.....	8
1.7.1. Nomenclature.....	8
1.7.2. Signal Names.....	8
2. Functional Description.....	9
2.1. IP Architecture Overview.....	9
2.2. Clocking.....	10
2.2.1. Clocking Overview.....	10
2.2.2. Adjusting Reference Clock for Different Baud Rate.....	10
2.3. Reset.....	11
2.4. User Interfaces.....	12
2.5. MPCS Soft IP Wrapper.....	12
2.6. SLVS-EC Rx Core.....	13
2.6.1. Rx Protocol Management FSM.....	13
2.6.2. Packet Parser.....	14
2.6.3. Byte-to-Pixel Converter.....	14
2.7. Configuration, Control and Status Register (CSR).....	17
2.8. Timing Descriptions.....	17
2.8.1. SLVS-EC Rx with Native Interface.....	17
2.8.2. SLVS-EC Rx with AXI4-Stream Transmitter Interface.....	18
3. IP Parameter Description.....	20
3.1. General.....	20
4. Signal Description.....	22
4.1. Clock and Reset.....	22
4.2. Primary I/O.....	22
4.3. Native Interface.....	22
4.4. Packet Header Information.....	22
4.5. Control and Status.....	23
4.6. Miscellaneous.....	23
4.7. AXI4-Stream Transmitter Interface.....	23
4.8. APB Interface.....	24
5. Register Description.....	25
6. Designing with the IP.....	27
6.1. Generating and Instantiating the IP.....	27
6.1.1. Generated Files and File Structure.....	29
6.2. Design Implementation.....	30
6.3. Timing and Physical Design Constraints.....	30
6.4. Specifying the Strategy.....	30
6.5. Running Functional Simulation.....	30
7. Debugging.....	34
7.1. Debug Tools.....	34
7.1.1. Reveal Analyzer.....	34
7.1.2. QuestaSim.....	34

Appendix A. Resource Utilization.....	35
Appendix B. Limitation.....	36
References	37
Technical Support Assistance	38
Revision History	39

Figures

Figure 2.1. Lattice SLVS-EC Receiver Block Diagram	9
Figure 2.2. Clock Domain Crossing Diagram	10
Figure 2.3. Control Sequence for Initialization	12
Figure 2.4. Training Sequence Details.....	12
Figure 2.5. Rx Protocol Management FSM	13
Figure 2.6. Input/Output Behavior of Rx Protocol Management FSM Module	13
Figure 2.7. Byte-to-Pixel Converter Implementation.....	14
Figure 2.8. FIFO Implementation Diagram.....	15
Figure 2.9. Configuration, Control, and Status Register Implementation	17
Figure 2.10. SLVS-EC Rx Native Interface Output Timing Diagram	18
Figure 2.11. SLVS-EC Rx AXI4-Stream Interface Timing Diagram	19
Figure 6.1. Module/IP Block Wizard	27
Figure 6.2. IP Configuration	28
Figure 6.3. Check Generated Result.....	29
Figure 6.4. Simulation Wizard.....	31
Figure 6.5. Add and Reorder Source	31
Figure 6.6. Parse HDL files for simulation	32
Figure 6.7. Summary	32
Figure 6.8. Simulation Waveform	33
Figure 6.9. Simulation Log on a Complete Run	33

Tables

Table 1.1. Summary of the SLVS-EC Receiver IP	7
Table 1.2. Ordering Part Number	7
Table 1.3. IP Validation Level	8
Table 2.1. Pre-Configured MPCS PLL Setting for SLVS-EC Rx	10
Table 2.2. User Interfaces and Supported Protocols	12
Table 2.3. Byte-to-Pixel Converter Data Groups.....	14
Table 3.1. General Attributes.....	20
Table 4.1. Clock and Reset Ports.....	22
Table 4.2. Primary I/O Ports.....	22
Table 4.3. Native Interface Ports ¹	22
Table 4.4. Packet Header Information Ports ¹	22
Table 4.5. Control and Status Ports	23
Table 4.6. Miscellaneous Ports ¹	23
Table 4.7. AXI4-Stream Transmitter Interface Ports ¹	23
Table 4.8. APB Interface Ports	24
Table 5.1. Access Type Definition	25
Table 5.2. Summary of SLVS-EC Receiver Registers	25
Table 6.1. Generated File List	29
Table A.1. Resource Utilization	35

Abbreviations in This Document

A list of abbreviations used in this document.

Abbreviation	Definition
AMBA	Advanced Microcontroller Bus Architecture
APB	Advanced Peripheral Bus
AXI	Advanced eXtensible Interface
CIS	CMOS Image Sensor
CMOS	Complementary Metal-Oxide Semiconductor
CRC	Cyclic Redundancy Check
DSP	Digital Signal Processor
DUT	Device Under Test
ECC	Error Code Correction
IP	Intellectual Property
LMMI	Lattice Memory Mapped Interface
LSB	Least Significant Bit
MPCS	Multi-Protocol PCS
PCS	Physical Coding Sublayer
PMA	Physical Media Attachment
Rx	Receiver
SERDES	Serializer Deserializer
Tx	Transmitter
SLVS-EC	Scalable Low Voltage Signaling Embedded Clock

1. Introduction

1.1. Overview of the IP

This user guide describes the Lattice Scalable Low Voltage Signaling-Embedded Clock Receiver (SLVS-EC Rx) IP. The SLVS-EC Receiver IP provides the FPGA an interface to receive serial data from CMOS Image Sensors and offers a solution to convert the incoming serial data to a parallel pixel data format. This can receive up to eight lanes of differential serial data running at a maximum of 5 Gbps per lane. The SLVS-EC functions defined by the PHY Layer are supported by the Multi-Protocol PCS (MPCS) soft IP which instantiates the PMA/PCS IPs, and the Link Layer functions defined by the PHY Layer are supported.

1.2. Quick Facts

Table 1.1. Summary of the SLVS-EC Receiver IP

IP Requirements	Supported FPGA Family	CertusPro™-NX
Resource Utilization	Targeted Devices	LFPCNX-100
	Resources	See Appendix A. Resource Utilization .
Design Tool Support	Lattice Implementation	IP v1.2.3 – Lattice Radiant™ software 3.0 or later
	Synthesis	Lattice Synthesis Engine
		Synopsys Synplify Pro® for Lattice
Simulation	For a list of supported simulators, see the Lattice Radiant Software user guide.	

1.3. Features

Key features of the SLVS-EC Receiver IP include:

- Compliant with SLVS-EC Protocol specification v2.0
- Back compatibility with SLVS-EC Protocol specification v1.2
- Supports PHY layer functions using the embedded PMA/PCS in CertusPro-NX FPGA device
- Supports up to eight lanes each running at a maximum of 5 Gbps baud rate
- Supports AXI4-Stream Protocol as a selectable option for data interface
- Supports AMBA 3 APB Protocol v1.0 for register access to control the IP

1.4. Licensing and Ordering Information

An IP specific license string is required to enable full use of the SLVS-EC Receiver IP in a complete, top-level design.

The IP can be fully evaluated through functional simulation and implementation (synthesis, map, place and route) without an IP license string. This IP supports Lattice’s IP hardware evaluation capabilities. You can create versions of the IP to operate in hardware for a limited time (approximately four hours) without requiring an IP license string. A license string is required to enable timing simulation and to generate a bitstream file that does not include the hardware evaluation timeout limitation.

For more information about pricing and availability of the SLVS-EC Receiver IP, contact your [local Lattice Sales Office](#).

1.4.1. Ordering Part Number

Table 1.2. Ordering Part Number

Device Family	Part Number	
	Single Machine Annual	Multi-Site Perpetual
CertusPro-NX	SLVS-EC-RX-CPNX-US	SLVS-EC-RX-CPNX-UT

1.5. IP Validation Summary

[Table 1.3](#) shows the validation status for the SLVS-EC Receiver IP core. The ✓ mark indicates whether the IP has been validated for Simulation, Timing, or with Hardware.

Table 1.3. IP Validation Level

Device Family	IP Version	Validation Level		
		Simulation	Timing	Hardware
CertusPro-NX	1.2.3	✓	✓	-

1.6. Minimum Device Requirements

Refer to [Appendix A. Resource Utilization](#) for the minimum required resource to instantiate this IP.

1.7. Naming Conventions

1.7.1. Nomenclature

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

1.7.2. Signal Names

- `_n` are active low (asserted when value is logic 0)
- `_i` are input signals
- `_o` are output signals
- `_io` are bidirectional signals

2. Functional Description

2.1. IP Architecture Overview

SLVS-EC is a high-speed serial interface technology developed by Sony for the next-generation high-resolution CMOS Image Sensors. The interface provides a unidirectional wideband pixel data transfer from a CMOS Image Sensor to a Digital Signal Processor.

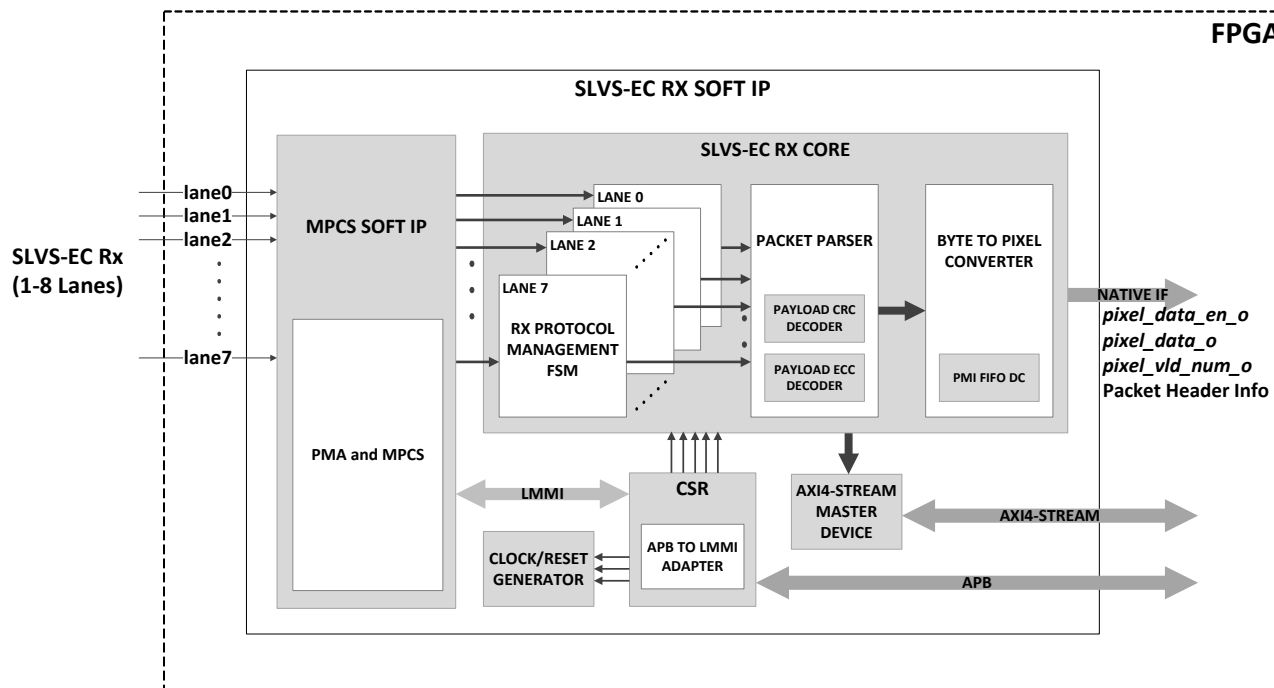


Figure 2.1. Lattice SLVS-EC Receiver Block Diagram

The SLVS-EC Receiver IP includes the following submodules:

- MPCS Soft IP Wrapper
- SLVS-EC Rx Core
- Transaction Layer

2.2. Clocking

2.2.1. Clocking Overview

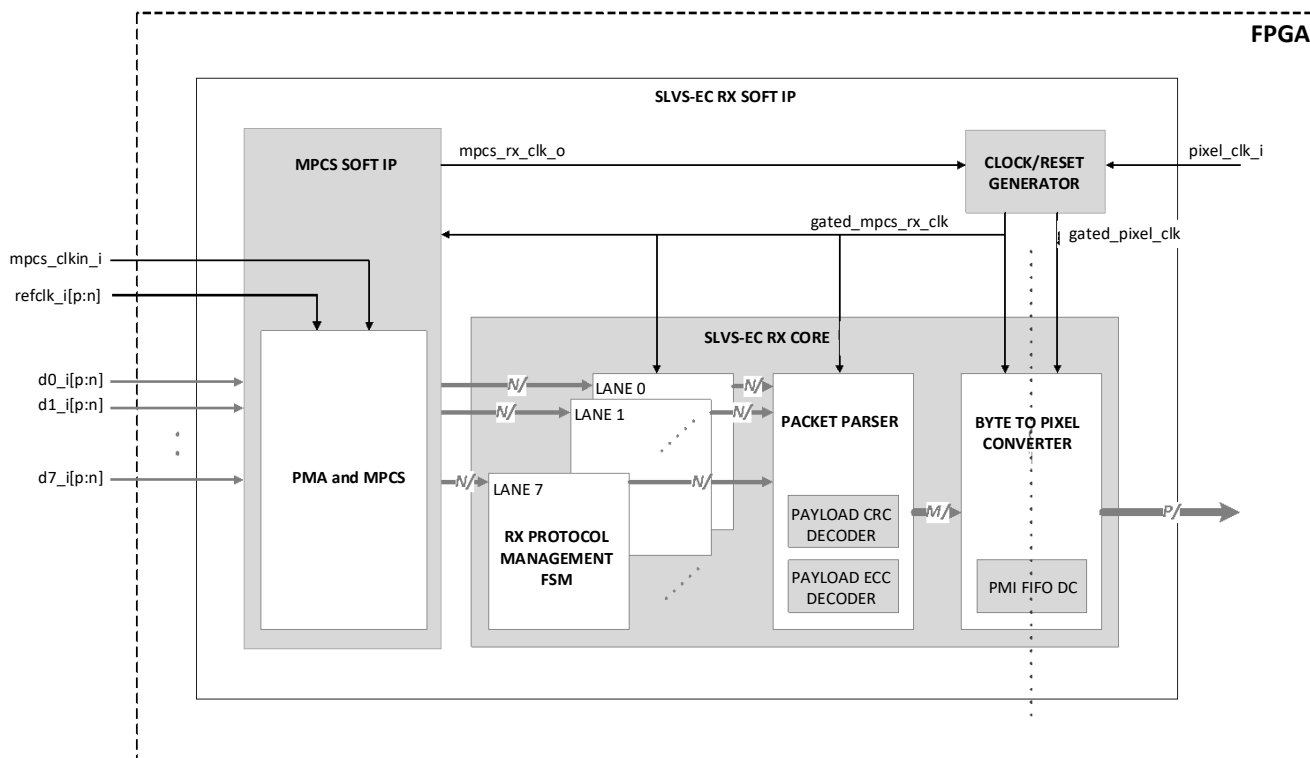


Figure 2.2. Clock Domain Crossing Diagram

Figure 2.2 shows the clocking scheme for the SLVS-EC Receiver soft IP. The output clock (mpcs_rx_clk_o) generated from MPCS soft IP is used to clock the Packet Parser module and the byte domain side of Byte-to-Pixel Converter. There is a FIFO inside Byte-to-Pixel Converter to handle the clock domain crossing between mpcs_rx_clk_o and pixel_clk_i.

The MPCS Rx clock frequency depends on the reference clock used as shown in Table 2.1. The Pixel clock frequency varies depending on the configuration (that is the number of Output Pixels, Data Type, and FIFO setting).

2.2.2. Adjusting Reference Clock for Different Baud Rate

The MPCS PLL settings for each baud grade are pre-configured for SLVS-EC as shown in Table 2.1.

Table 2.1. Pre-Configured MPCS PLL Setting for SLVS-EC Rx

Baud Grade	Bit Rate (Mbps)	Reference Clock (MHz) ¹	M	F	N	F _{VCO} (MHz)	F _{bit} (MHz)	F _{PMA} (MHz)	MPCS Rx Clock (MHz)
3	5000	125	2	4	10	10000	5000	500	125
2	2500	125	4	2	10	10000	2500	250	125
1	1200	125	8	1	10	10000	1250	125	125

Note:

- Frequency of Reference Clock can be varied depending on the target Bit Rate for each Baud Grade.

Simplified equations based on PLL settings are shown below:

$$\begin{aligned}
 \text{Internal Data Width Factor (bits)} &= 10 \text{ (Baud Grade 1); } 20 \text{ (Baud Grade 2); } 40 \text{ (Baud Grade 3)} \\
 \text{Reference Clock (MHz)} &= \frac{\text{Target Baud Rate}}{\text{Internal Data Width Factor}} \\
 \text{MPCS Rx Clock (MHz)} &= \text{Reference Clock}
 \end{aligned}$$

Example:

Case 1: Baud Grade 1, Target baud rate of 1250 Mbps

$$\begin{aligned} \text{Internal Data Width Factor (bits)} &= 10 \\ \text{Reference Clock (MHz)} &= \frac{1250 \text{ Mbps}}{10 \text{ bits}} = 125 \text{ MHz} \\ \text{MPCS Rx Clock (MHz)} &= 125 \text{ MHz} \end{aligned}$$

In the scenario when different baud rate is desired, the frequency of reference clock (refclk_n_i/refclk_p_i) can be adjusted. In order to do so, the target Baud Grade must be selected first in the IP user interface. To compute for the frequency of reference clock to be used, follow the equations above. Note however that there are only range of reference clock frequencies that are supported by MPCS/SERDES. For details, refer to [CertusPro-NX SERDES/PCS Usage Guide \(FPGA-TN-02245\)](#).

Example:

Case 2: Baud Grade 2, Target baud rate of 2304 Mbps

$$\begin{aligned} \text{Internal Data Width Factor (bits)} &= 20 \\ \text{Reference Clock (MHz)} &= \frac{2304 \text{ Mbps}}{20 \text{ bits}} = 115.2 \text{ MHz} \\ \text{MPCS Rx Clock (MHz)} &= 115.2 \text{ MHz} \end{aligned}$$

Case 3: Baud Grade 3, Target baud rate of 4752 Mbps

$$\begin{aligned} \text{Internal Data Width Factor (bits)} &= 40 \\ \text{Reference Clock (MHz)} &= \frac{4752 \text{ Mbps}}{40 \text{ bits}} = 118.8 \text{ MHz} \\ \text{MPCS Rx Clock (MHz)} &= 118.8 \text{ MHz} \end{aligned}$$

2.3. Reset

Listed below are the reset and initialization flow for the SLVS-EC Receiver IP:

- **Reset**
An asynchronous reset pin (active low) as system reset is used for resetting the SLVS-EC Receiver Soft IP. Internal reset logic is implemented to guarantee synchronous de-assertion all through-out the different clock domain among the MPCS soft IP and other soft logic blocks.
- **Initialization**
[Figure 2.3](#) shows the control sequence needed for the PHY initialization of SLVS-EC Receiver. Toggle mpcs_clkln_i, serial reference clock and wait for mpcs_ready_o to assert, to start the training sequence. A setup signal is sent from the Client-side to the SLVS-EC Rx for it to commence start-up and to be ready to receive the initialization sequence. CIS drives the link low before sending the training sequence. This training sequence is a series of repeated Sync codes and intermittent transmission of Deskew codes, to establish bit synchronization and adjust the skew between the lanes respectively. See [Figure 2.4](#) for a detailed definition of the training sequence. The PHY control codes, number of transfer times, and other training sequence items are to be set beforehand through APB register configuration (see [Table 5.2](#) for the list of attribute registers). After the training sequence has been received and processed successfully by the SLVS-EC Receiver, it sends a ready signal to the client-side to confirm that it is already in the state of receiving valid packets of data.

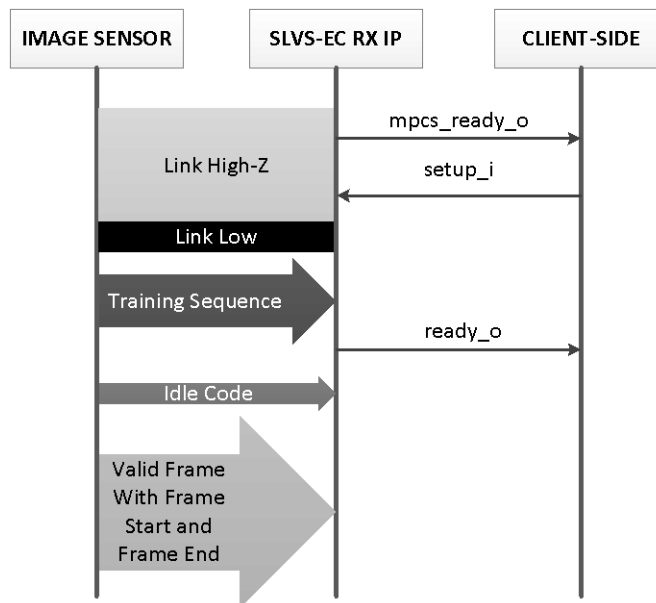


Figure 2.3. Control Sequence for Initialization

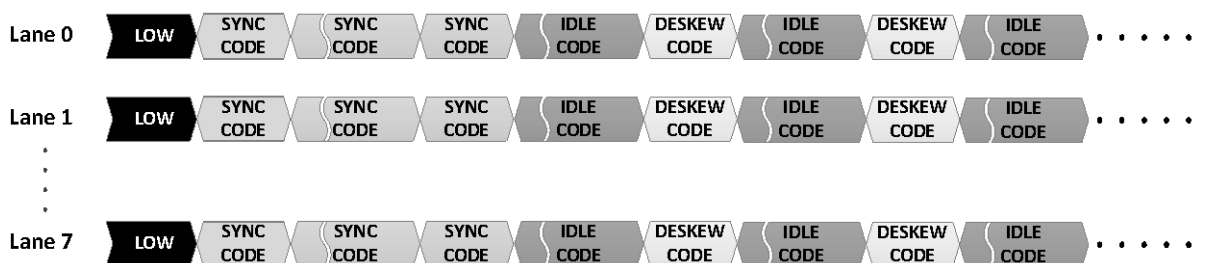


Figure 2.4. Training Sequence Details

2.4. User Interfaces

Table 2.2. User Interfaces and Supported Protocols

User Interface	Supported Protocols	Description
Device Transmitter	AXI4-Stream	The AXI4-Stream interface carries a stream of pixel data and packet header. The packet header information is considered as user sideband signals (m_tuser_o) if header analysis is enabled.
Control	APB	Use to configure IP CSRs, including MPCS LMMI registers. Refer to the Register Description section for more information.

2.5. MPCS Soft IP Wrapper

MPCS Soft IP supports a wide range of serial I/O protocols including the SLVS-EC. The PMA and PCS settings in this IP are pre-configured for SLVS-EC. This IP receives the serial data and performs mainly the word alignment, 8b/10b decoding, and lane-to-lane Rx deskew of the incoming data. Refer to the [MPCS Module User Guide \(FPGA-IPUG-02118\)](#) for further details.

2.6. SLVS-EC Rx Core

The SLVS-EC Rx Core, consisting of Rx Protocol Management FSM, Packet Parser, and Byte-to-Pixel Converter modules, receives the decoded byte data from the MPCS soft IP and handles the IP's Link Layer functions such as packet header and packet footer processing, link protocol management and data formatting.

2.6.1. Rx Protocol Management FSM

This module receives the parallel data from the MPCS soft IP wrapper and strips off the PHY control codes. [Figure 2.5](#) is the finite state machine diagram of the Rx Protocol Management FSM module. The FSM on the left side of the diagram is for the training sequence and on the right side of the diagram is the main FSM to determine the PHY control codes. The output of this module contains the packet header, valid payload, and payload/lane stuffing bytes, if there are any, as shown in [Figure 2.6](#).

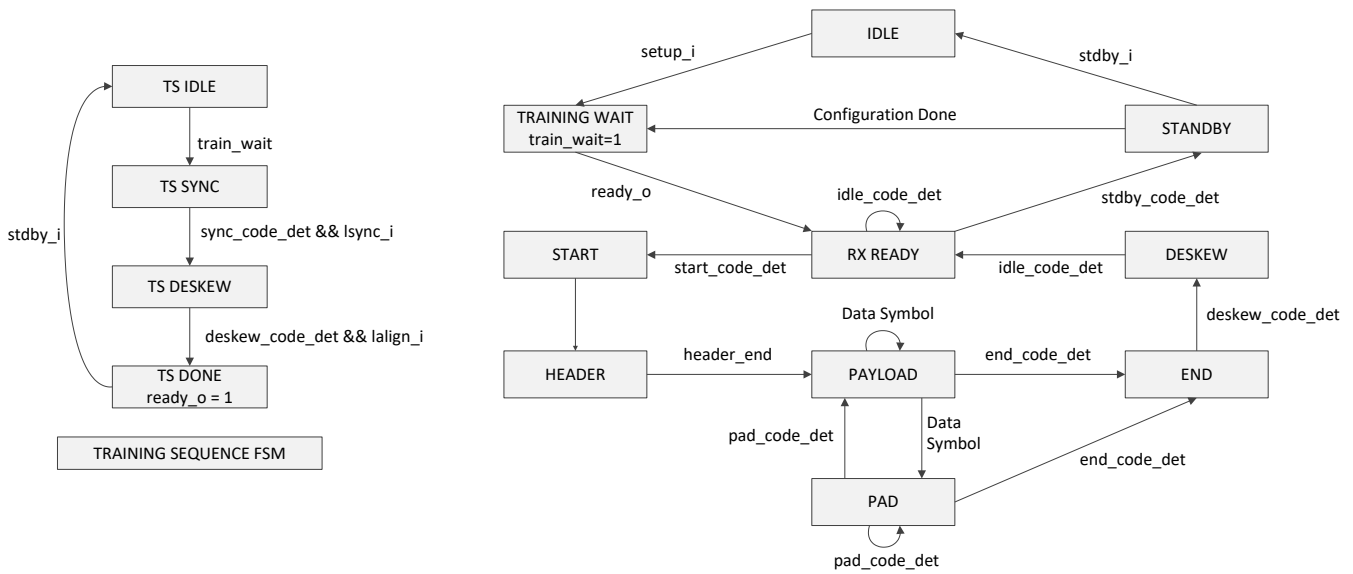


Figure 2.5. Rx Protocol Management FSM

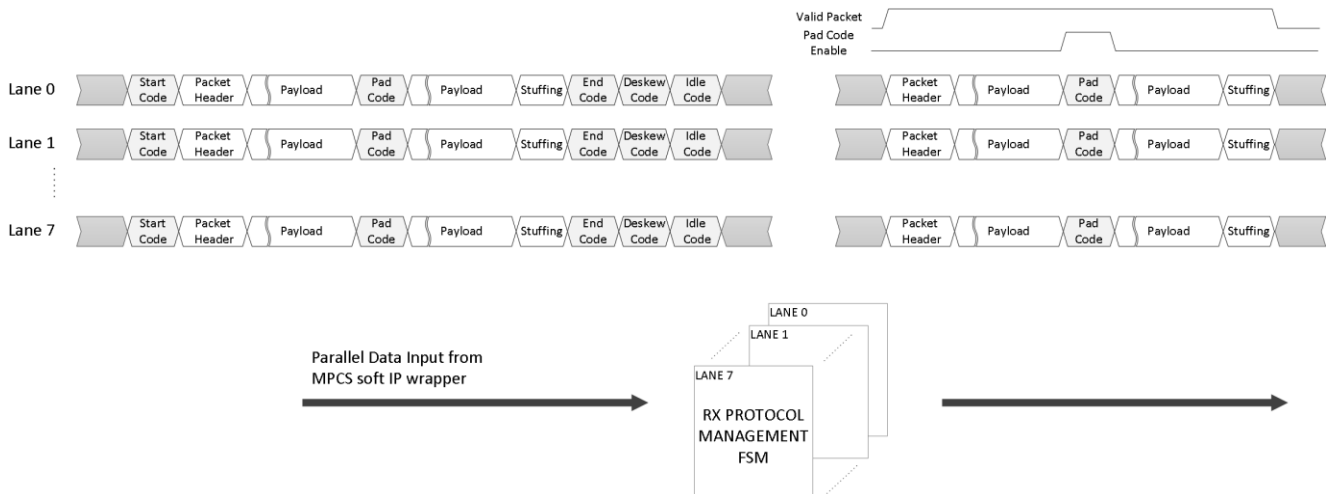


Figure 2.6. Input/Output Behavior of Rx Protocol Management FSM Module

Current IP only supports lane deskew during initialization/training phase. In cases when lane skew occurs during packet data transfer, you can perform the following procedure:

1. In the SLVS-EC Rx soft IP, add a custom deskew logic after the MPCS IP instance that will search for the START code from packet data of each lane.
2. Add single clock FIFO that will store these packets after START is detected.
3. Read out from FIFO after START is detected in all lanes.

2.6.2. Packet Parser

This module extracts the parallel payload data and parses the packet header and packet footer information of the received packet from the Rx Protocol Management FSM module. The Payload CRC decoder and Payload ECC decoder are also instantiated in this module for any payload error detection.

2.6.3. Byte-to-Pixel Converter

This module converts the received valid line data bytes from the Packet Parser module, into pixel data format. The PMI FIFO DC is used to transfer the valid line data from all lanes from the MPCS Rx clock domain (mpcs_rx_clk_o) to the pixel clock domain (pixel_clk_i). Figure 2.7 shows the diagram of the Byte-to-Pixel Converter Implementation. The input data is pipelined and grouped depending on the Data Type configuration to easily extract pixel data in the pixel domain (see Table 2.3).

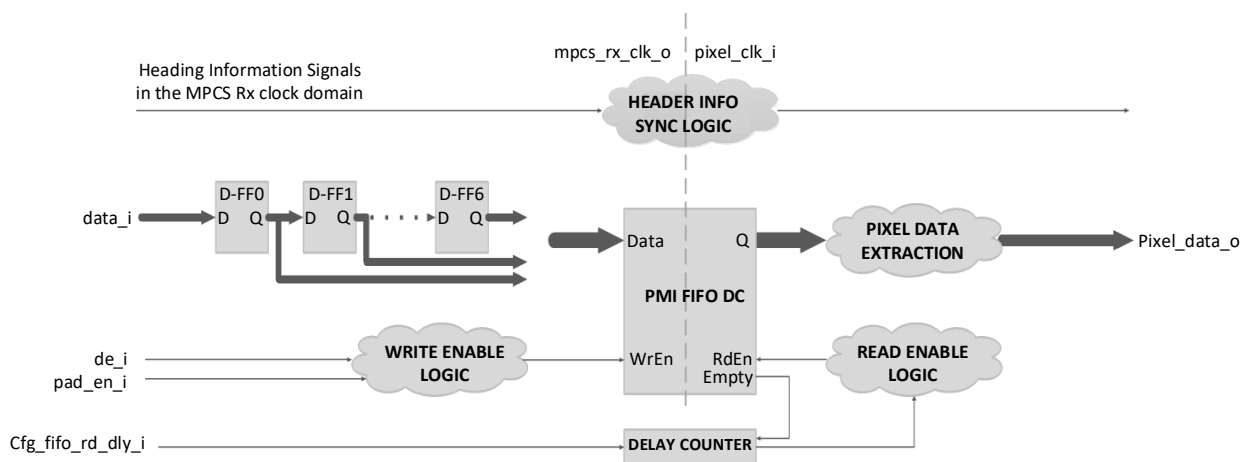


Figure 2.7. Byte-to-Pixel Converter Implementation

Table 2.3. Byte-to-Pixel Converter Data Groups

Data Type	Data Groups
RAW 8	1
RAW 10	5
RAW 12	3
RAW 14	7
RAW 16	2

The FIFO depth is defined based on the design configuration, as shown in the Attributes List in Table 3.1. In order to avoid the FIFO full or empty states, the Data Safe Zone is defined by setting data FIFO Delay attribute which handles the Overflow/Underflow Threshold, as shown in Figure 2.8. The FIFO delay value is used to trigger the start of FIFO read-out from FIFO not-empty state.

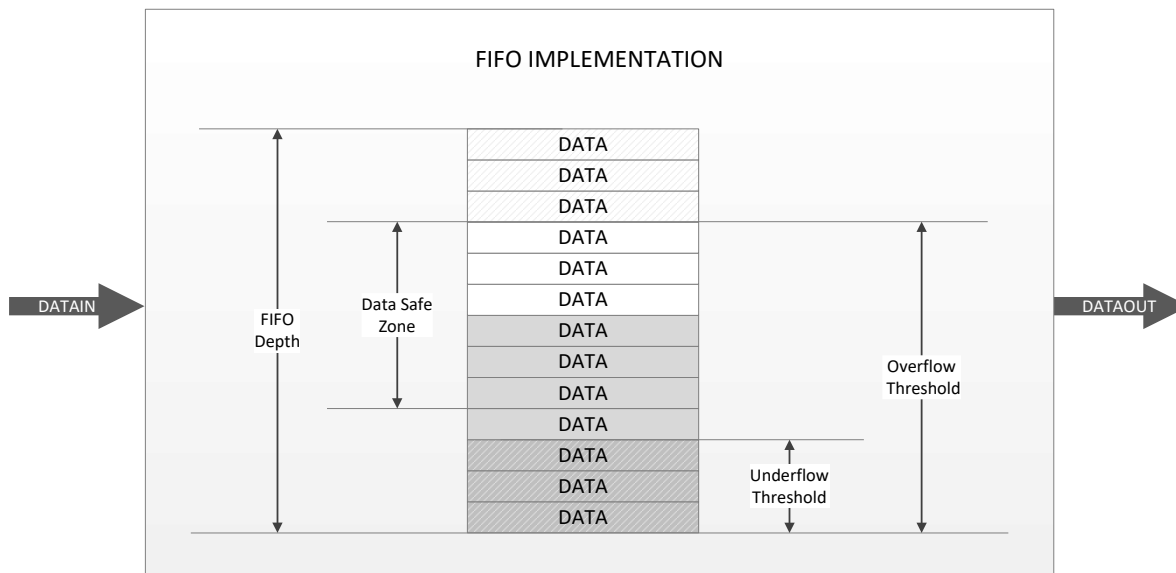


Figure 2.8. FIFO Implementation Diagram

The FIFO has the following three attributes: FIFO Width, FIFO Depth, and FIFO Delay. The FIFO Delay is used to synchronize data flows between byte and pixel sides. Data transfer on pixel side is started after the FIFO Delay is reached. FIFO Width, FIFO Depth, and FIFO Delay depend on several attributes listed below:

- Number of Rx Lanes (see [Table 3.1](#))
- MPCS Output Data Width – (based on Baud Grade setting)
 - Baud Grade 1: 8 Bits
 - Baud Grade 2: 16 Bits
 - Baud Grade 3: 32 Bits
- MPCS Rx Clock Frequency (mpcs_rx_clk_o, frequency is same as the frequency of the reference clock ref_clk_p/n_i used)
- Pixel Clock Frequency (pixel_clk_i, available on the user interface)
- Byte Count (converted from Pixel Count – see [Table 3.1](#))
- Number of Bits Per Pixel (Data Type – see [Table 3.1](#))
- Number of Output Pixels per Clock (see [Table 3.1](#))

FIFO Width is calculated as,

$$FIFO\ Width = 7 \times Number\ of\ Rx\ Lanes \times MPCS\ Output\ Data\ Width$$

The input data must be grouped based on the pixel to byte conversion multiplier in order to properly extract the pixel data in the pixel clock domain.

Denote the Data Flow Speed on byte side as MPCS Eff Rate and the Data Flow Speed on pixel side as Pixel Eff Rate. They are calculated as,

$$MPCS\ Eff\ Rate = MPCS\ Rx\ Clock\ Frequency \times Number\ of\ Rx\ Lanes \times MPCS\ Output\ Data\ Width$$

$$Pixel\ Eff\ Rate = Pixel\ Clock\ Frequency \times Number\ of\ Bits\ Per\ Pixel \times Number\ of\ Output\ Pixels\ per\ Clock$$

The FIFO Depth (minimum value) and FIFO Delay depend on the ratio (N_{ratio}) of the Data Flow Speed on the Byte side and Data flow speed on the Pixel side. Pixel Eff Rate should be greater than or equal to MPCS Eff Rate.

$$N_{ratio} = Pixel\ Eff\ Rate / MPCS\ Eff\ Rate$$

They can be calculated by the following formulas:
IF (MPCS Eff Rate == Pixel Eff Rate)

$$FIFO\ Depth = 16; FIFO\ Delay = 4$$

ELSE

$$FIFO\ Delay\ Temp = Ceiling\left[\frac{(N_{ratio} - 1)(Byte\ Count \times 8)}{No.\ of\ Lanes \times MPCS\ Output\ Data\ Width} \times \frac{1}{Data\ Group}\right]$$

$$FIFO\ Delay = (FIFO\ Delay\ Temp + 1) \times Data\ Group$$

$$FIFO\ Depth = 2^{log_2(Fifo\ Delay\ Temp + 4)}$$

If there are pad codes, add them to the Byte Count and recompute for new FIFO DELAY/DEPTH (for example, byte count is 12288 and additional 1200 bytes of pad codes per line, new byte count = 12288 + 1200 = 13488).

Sample Computations:

Case 1: Baud Grade = 2, Number of Rx Lanes = 8, Data Type = RAW10, MPCS Rx Clock = 115.2 MHz, Pixel Clock = 125 MHz, Pixel Count = 8432

MPCS Settings:

$$\begin{aligned} MPCS\ Output\ Data\ Width &= 16\ bits\ (Baud\ Grade\ 2) \\ MPCS\ Output\ Data\ (bytes) &= 2 \end{aligned}$$

Pixel Settings:

$$\begin{aligned} Output\ Pixels\ per\ clock &= 16\ (Table\ 3.1) \\ Pixel\ count\ per\ line\ (bytes) &= 10540\ (Byte\ Count) \\ Data\ Group &= 5\ (Table\ 2.3) \end{aligned}$$

Computations:

$$\begin{aligned} FIFO\ Width &= 7 \times 8\ lanes \times 16\ bits = 896\ bits \\ MPCS\ Eff\ Rate &= 115.2\ MHz \times 8\ lanes \times 16\ bits = 14,745.6\ Mbps \\ Pixel\ Eff\ Rate &= 125\ MHz \times 10\ bits \times 16\ output\ pixels\ per\ clock = 20,000\ Mbps \\ N_{ratio} &= \frac{20,000\ Mbps}{14,745.6\ Mbps} = 1.356336806 \end{aligned}$$

Since MPCS Eff Rate is not equal to Pixel Eff Rate:

$$\begin{aligned} FIFO\ Delay\ Temp &= Ceiling\left[\frac{(1.356336806 - 1) \times (10540 \times 8)}{8 \times 16} \times \frac{1}{5}\right] = 47 \\ FIFO\ Delay &= (47 + 1) \times 5 = 240 \\ FIFO\ Depth &= 2^{log_2(47+4)} = 64 \end{aligned}$$

Case2: Baud Grade = 3, Number of Rx Lanes = 4, Data Type = RAW8, MPCS Rx Clock = 118.8 MHz, Pixel Clock = 125 MHz, Pixel Count = 1920

MPCS Settings:

$$\begin{aligned} MPCS\ Output\ Data\ Width &= 32\ bits\ (Baud\ Grade\ 3) \\ MPCS\ Output\ Data\ (bytes) &= 4 \end{aligned}$$

Pixel Settings:

$$\begin{aligned} Output\ Pixels\ per\ clock &= 16\ (Table\ 3.1) \\ Pixel\ count\ per\ line\ (bytes) &= 1920\ (Byte\ Count) \\ Data\ Group &= 1\ (Table\ 2.3) \end{aligned}$$

Computations:

$$\begin{aligned} FIFO\ Width &= 7 \times 4\ lanes \times 32\ bits = 896\ bits \\ MPCS\ Eff\ Rate &= 118.8\ MHz \times 4\ lanes \times 32\ bits = 15,206.4\ Mbps \\ Pixel\ Eff\ Rate &= 125\ MHz \times 8\ bits \times 16\ output\ pixels\ per\ clock = 16,000\ Mbps \end{aligned}$$

$$N_{ratio} = \frac{16,000 \text{ Mbps}}{15,206.4 \text{ Mbps}} = 1.052188552$$

Since MPCS Eff Rate is not equal to Pixel Eff Rate:

$$FIFO \text{ Delay Temp} = \text{Ceiling} \left[\frac{(1.052188552 - 1) \times (1920 \times 8)}{4 \times 32} \times \frac{1}{1} \right] = 7$$

$$FIFO \text{ Delay} = (7 + 1) \times 1 = 8$$

$$FIFO \text{ Depth} = 2^{\text{ceil}(\log_2(7+4))} = 16$$

2.7. Configuration, Control and Status Register (CSR)

Figure 2.9 is the CSR Implementation that can be accessed through APB. MPCS registers can only be accessed through LMMI, thus an APB-to-LMMI converter is needed and the SLVS-EC Rx IP registers are then mapped to the MPCS LMMI registers. It is required to start APB register access after at least 22 ns. The MPCS register initialization is done after this duration.

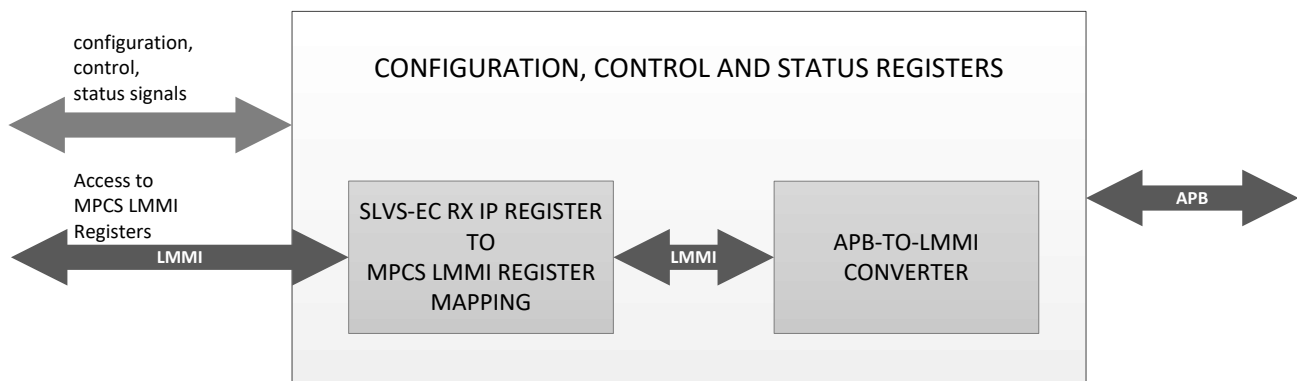


Figure 2.9. Configuration, Control, and Status Register Implementation

2.8. Timing Descriptions

2.8.1. SLVS-EC Rx with Native Interface

Figure 2.10 is the output timing diagram of SLVS-EC Rx with Native Interface. The output signals carrying the packet header information are available at the output if the Header Analysis attribute is enabled. The packet footer information is available if the Miscellaneous Signals attribute is enabled.

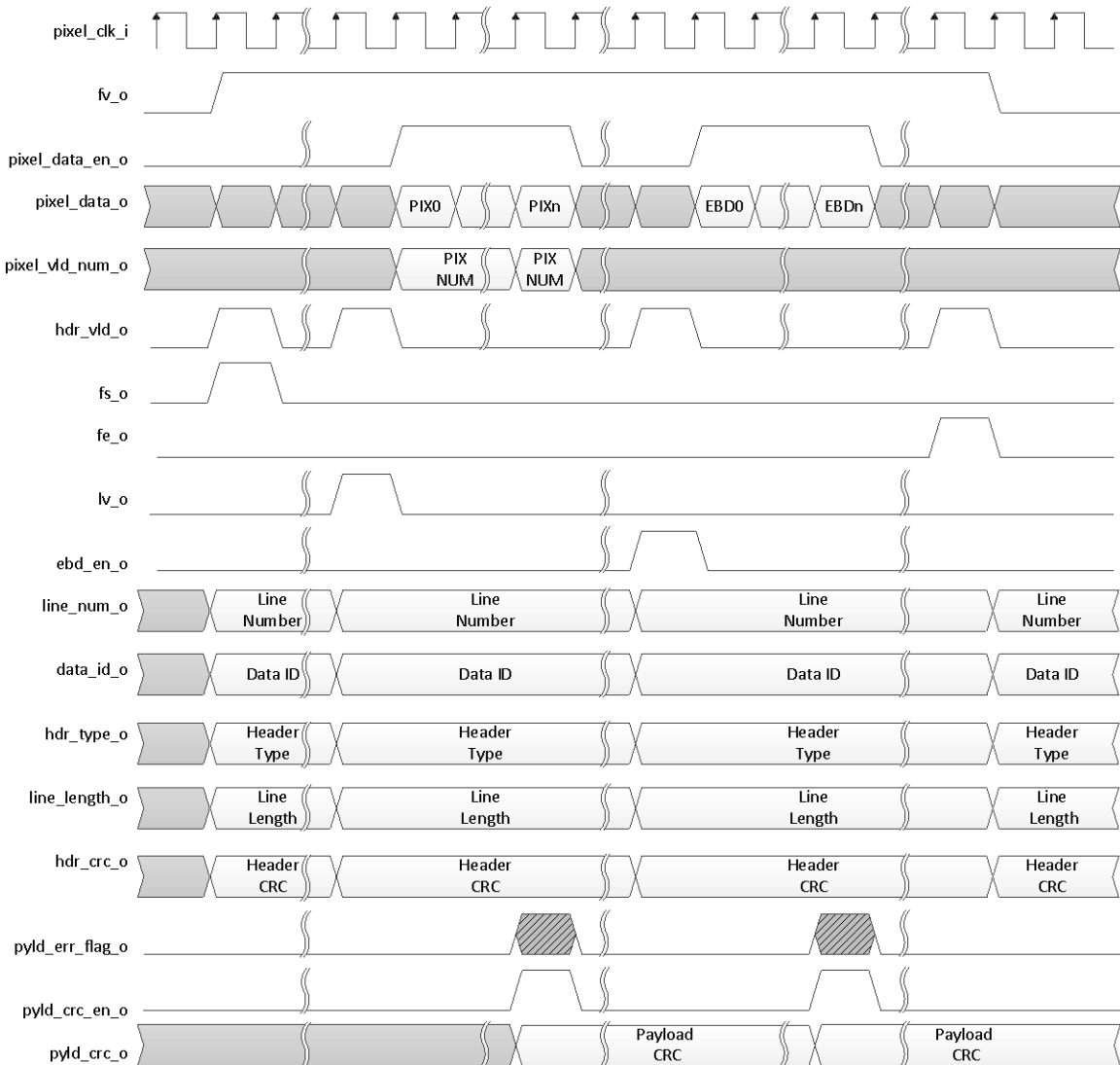


Figure 2.10. SLVS-EC Rx Native Interface Output Timing Diagram

2.8.2. SLVS-EC Rx with AXI4-Stream Transmitter Interface

Figure 2.11 shows the timing diagram of SLVS-EC Rx with AXI4-Stream interface. The AXI4-Stream Transmitter interface does not accept back-pressure. It assumes that `m_tready_i` is always HIGH. Transfer takes place once `m_tvalid_o` has been asserted.

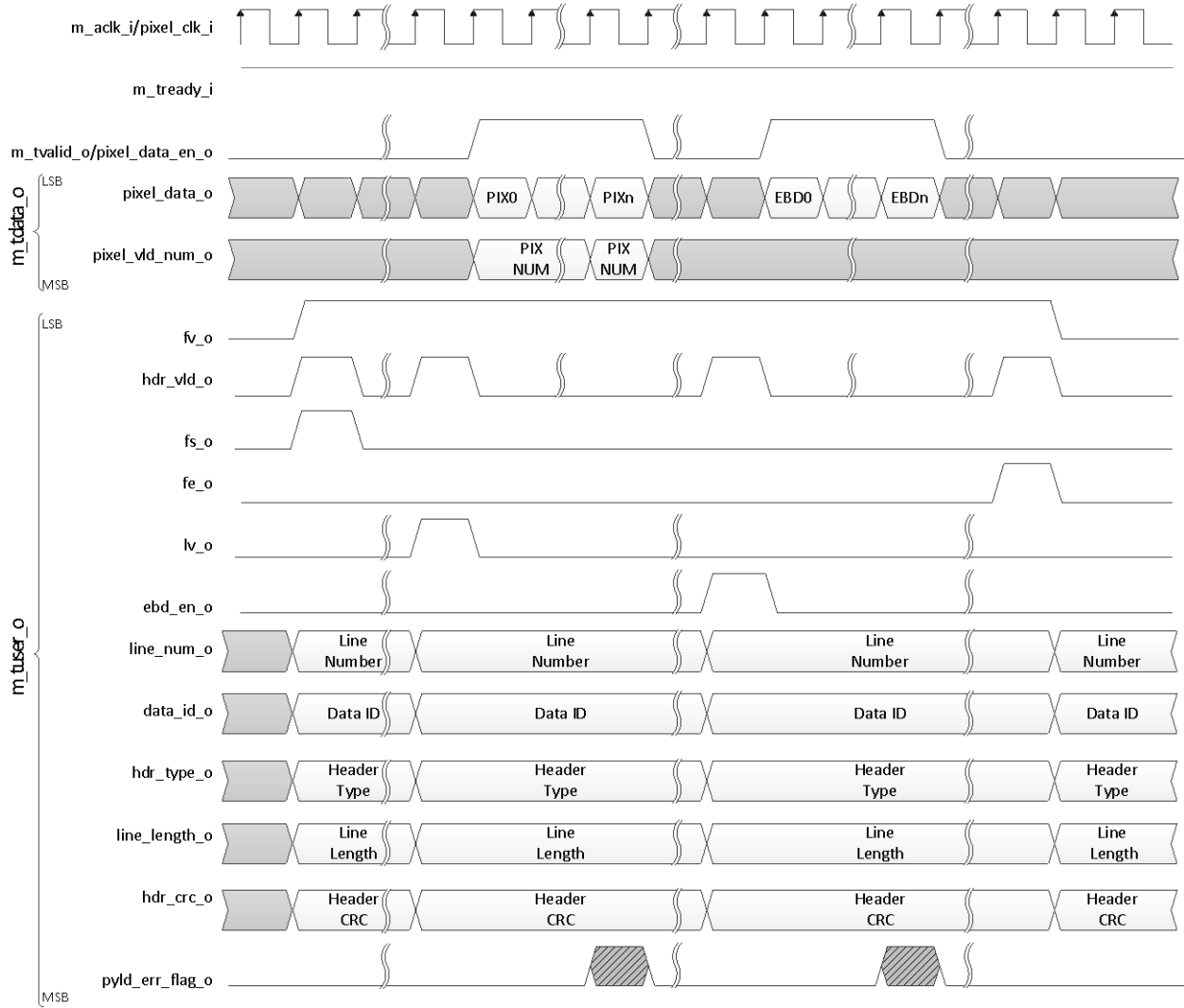


Figure 2.11. SLVS-EC Rx AXI4-Stream Interface Timing Diagram

3. IP Parameter Description

The configurable attributes of the SLVS-EC Receiver IP are shown in the following table. You can configure the IP by setting the attributes accordingly in the IP Catalog's Module/IP wizard of the Lattice Radiant software.

Wherever applicable, default values are in **bold**.

3.1. General

Table 3.1. General Attributes

Attribute	Selectable Values	Description
General		
Number of RX Lanes	1, 2, 4, 6, 8	Number of SLVS-EC Rx lanes.
PCS Lane ID	AUTO , 0, 1, 2, 3, 4, 5, 6, 7	Specifies the location of the first lane of the PCS instance. Refer to the MPCS Module User Guide (FPGA-IPUG-02118) for the rules in selecting the Lane ID.
Data Type (RAW Data Format)	8, 10, 12, 14, 16	Number of Bits per Pixel in RAW Data Format.
Data Interface Type		
Enable Native Interface	Checked , Unchecked	Select to use Native interface as outputs.
Enable AXI4-Stream Transmitter Interface	Checked, Unchecked	Select to use AXI4-Stream Transmitter interface as outputs.
Data Rate		
Baud Grade	1, 2, 3	Baud grade setting: <ul style="list-style-type: none"> 3: Baud Rate = up to 5000 Mbps 2: Baud Rate = up to 2500 Mbps 1: Baud Rate = up to 1250 Mbps
Reference Clock Frequency (MHz)	74.25 – 125	Reference clock frequency of the MPCS soft IP (refclk_n_i/refclk_p_i). The MPCS PLL settings for each baud grade is pre-configured. There is only a range of reference clock frequencies that are supported by MPCS/SERDES. For details, refer to the CertusPro-NX SERDES/PCS Usage Guide (FPGA-TN-02245) .
MPCS Bit Rate (Mbps)	Calculated	MPCS data rate per lane. Calculated values depend on the <i>Reference Clock</i> and <i>Baud Grade</i> .
MPCS Effective Data Rate (Mbps)	Calculated	Total effective MPCS output data rate. Calculated values depend on the <i>Reference Clock</i> , <i>Baud Grade</i> , and <i>Number of RX Lanes</i> .
Pixel Interface		
Number of Output Pixels per Pixel Clock	Calculated	Fixed value of output pixels per pixel clock depending on the <i>Data Type</i> , <i>MPCS Data Bytes</i> , and <i>Number of RX Lanes</i> . MPCS Data Bytes varies depending on the <i>Baud Grade</i> setting: <ul style="list-style-type: none"> 1 byte if Baud Grade = 1 2 bytes if Baud Grade = 2 4 bytes if Baud Grade = 3 <p>For RAW16 Data Type: If Number of RX Lanes = 1 and Baud Grade = 1, then the Number of Output Pixels per Pixel Clock = 1. Otherwise, the Number of Output Pixels per Pixel Clock = (MPCS data bytes × Number of RX Lanes) / 2.</p> <p>For other Data Types: The Number of Output Pixels per Pixel Clock = MPCS data bytes × Number of RX Lanes.</p>

Attribute	Selectable Values	Description
Pixel Clock Frequency (MHz)	10–250, 125	Input clock frequency for the pixel clock domain.
Effective Output Data Rate (Mbps)	Calculated	Effective output data rate in the pixel clock domain.
FIFO		
Manual Adjust	Checked, Unchecked	Enable to manually adjust <i>FIFO Depth</i> and <i>FIFO Read Delay</i> settings. Auto-computations of GUI <i>FIFO Depth</i> and <i>FIFO Read Delay</i> attributes do not consider the pad codes inserted in the packet data. Adjust manually to avoid FIFO Empty and Full states during packet transfer. Refer to the Byte-to-Pixel Converter section to get the calculation for <i>FIFO Depth</i> and <i>FIFO Read Delay</i> .
Pixel Count	1–16384, 1920	Number of pixels per line.
FIFO Depth	2–65536, 16	Set Byte-to-Pixel FIFO depth.
FIFO Read Delay	1–65536, 1	Set Byte-to-Pixel delay cycles for the FIFO read threshold.
Header Analysis		
Enable Header Analysis	Checked , Unchecked	Packet header information is parsed if enabled and is available in the output.
Miscellaneous		
Enable Miscellaneous Signals	Checked, Unchecked	Miscellaneous signals are available in the output if enabled.

4. Signal Description

This section describes the SLVS-EC Receiver IP ports.

4.1. Clock and Reset

Table 4.1. Clock and Reset Ports

Port	Type	Description
reset_n_i	Input	Active-low asynchronous reset.
refclk_n_i	Input	Differential Reference Clock (CLK-).
refclk_p_i	Input	Differential Reference Clock (CLK+).
mpcs_clkln_i	Input	Clock source for PMA inside the MPCS soft IP. Drives all calibration logic inside the PMA Controller and can be driven using any frequency as long as it is compliant to the CertusPro-NX SERDES/PCS Usage Guide (FPGA-TN-02245) .
mpcs_pwrndn_i[1:0]	Input	PHY power down: <ul style="list-style-type: none"> 2'b00: Operational 2'b10: Low-power state. Rx CDRPLL is powered down. The PHY starts in this low-power mode to perform calibration. 2'b11: Deep low-power state. Tx driver is in electrical Idle II (static low). All PLLs are powered down and clock shutdown.

4.2. Primary I/O

Table 4.2. Primary I/O Ports

Port	Type	Description
d_p_i[N-1:0]	Input	Differential Receive Serial Interface, Rx-, N: number of lanes.
d_n_i[N-1:0]	Input	Differential Receive Serial Interface, Rx+, N: number of lanes.

4.3. Native Interface

Table 4.3. Native Interface Ports¹

Port	Type	Description
pixel_clk_i	Input	Pixel clock input.
fv_o	Output	Frame valid output for active lines.
pixel_data_en_o	Output	Pixel data enable.
pixel_data_o[PIX_WIDTH-1:0]	Output	Pixel data output in LSB first data order. PIX_WIDTH = Bit Per Pixel (data type) × Number of Output Pixels per Pixel Clock
pixel_vld_num_o[7:0]	Output	Indicates valid number of output pixels per cycle.

Note:

1. Available if *Enable Native Interface* is checked.

4.4. Packet Header Information

Table 4.4. Packet Header Information Ports¹

Port	Type	Description
lv_o	Output	Indicates whether the line is valid or invalid.
fs_o	Output	Indicates the beginning of a frame.
fe_o	Output	Indicates the end of a frame.

Port	Type	Description
line_num_o[12:0]	Output	Indicates the line number.
data_id_o[3:0]	Output	Data ID for multiple stream transfer.
hdr_type_o[2:0]	Output	Type of header information.
line_length_o[15:0]	Output	Payload word count per line (hdr_type_o = 3'b001).
hdr_crc_o[15:0]	Output	Header CRC.
ebd_en_o	Output	Indicates the line with the embedded data.
hdr_vld_o	Output	Indicates the valid header information.

Note:

1. Available if *Enable Header Analysis* is checked.

4.5. Control and Status

Table 4.5. Control and Status Ports

Port	Type	Description
mpcs_ready_o	Output	Active high ready signal indicating that PHY has completed the calibration sequence for each specific lane. MPCS is ready to receive training sequence.
setup_i	Input	Active high setup signal from the user interface to commence start-up.
ready_o	Output	Active high ready signal indicating that Rx is ready for packet reception. Training sequence done.
pyld_err_flag_o	Output	Payload error detected.

4.6. Miscellaneous

Table 4.6. Miscellaneous Ports¹

Port	Type	Description
mpcs_rx_clk_o	Output	Output clock from the MPCS soft IP (MPCS Rx clock).
pyld_crc_o	Output	Packet footer information. Payload CRC.
pyld_crc_en_o	Output	Valid payload CRC.

Note:

1. Available if *Enable Miscellaneous Signals* is checked.

4.7. AXI4-Stream Transmitter Interface

Table 4.7. AXI4-Stream Transmitter Interface Ports¹

Port	Type	Description
m_ack_i	Input	Clock for the pixel domain logic.
m_tvalid_o	Output	TVALID indicates that the transmitter is driving a valid transfer. A transfer takes place when both TVALID and TREADY are asserted.
m_tready_i	Input	AXI4-Stream Transmitter Interface does not accept back-pressure and assumes an always high m_tready_i.
m_tdata_o[PIX_WIDTH+8-1:0]	Output	Pixel Output Data. PIX_WIDTH = Bit Per Pixel (data type) × Number of Output Pixels per Pixel Clock
m_tuser_o	Output	Sideband information transmitted alongside the data stream.

Note:

1. Available if *Enable AXI-4 Stream Interface* is checked.

4.8. APB Interface

Table 4.8. APB Interface Ports

Port	Type	Description
apb_pclk_i	Input	APB Clock which uses 125 MHz.
apb_preset_n_i	Input	Reset (active low).
apb_psel_i	Input	Select signal. Indicates that the completer device is selected and a data transfer is required.
apb_paddr_i	Input	Address signal.
apb_pwdata_i	Input	Write data signal.
apb_pwrite_i	Input	Direction signal. Write = 1, Read = 0.
apb_penable_i	Input	Enable signal. Indicates the second and subsequent cycles of an APB transfer.
apb_pready_o	Output	Ready signal. Indicates transfer completion. Completer device uses this signal to extend an APB transfer.
apb_prdata_o	Output	Read data signal.

5. Register Description

The behavior of registers to write and read access is defined by its access type, which is defined in [Table 5.1](#).

Table 5.1. Access Type Definition

Access Type	Behavior on Read Access	Behavior on Write Access
RO	Returns register value	Ignores write access
WO	Returns 0	Updates register value
RW	Returns register value	Updates register value
RW1C	Returns register value	Writing 1'b1 on register bit clears the bit to 1'b0. Writing 1'b0 on register bit is ignored.
RSVD	Returns 0	Ignores write access

Table 5.2. Summary of SLVS-EC Receiver Registers

Offset	Register Name	Access	Default Value	Register Description		
				Bit	Field	Field Description
Attribute Registers						
0x00	SYNC_SYMBOL	RW	D.10.5	[7:0]	sync_symbol	Set 8-bit symbol following the comma symbol (K28.5) within the Sync Code.
				[31:8]	RSVD	Reserved bits.
0x04	DESKEW_SYMBOL	RW	D.00.3	[7:0]	deskew_symbol	Set 8-bit symbol following the comma symbol (K28.5) within the Deskew Code.
				[31:8]	RSVD	Reserved bits.
0x0C	STDBY_SYMBOL	RW	D.03.0	[7:0]	stdby_symbol	Set 8-bit symbol following the comma symbol (K28.5) within the Standby code.
				[31:8]	RSVD	Reserved bits.
0x10	SYNC_LENGTH ¹	RW	24'd255	[23:0]	sync_length	Set the sync code transfer times.
				[31:24]	RSVD	Reserved bits.
0x14	DESKEW_LENGTH ¹	RW	8'd16	[7:0]	deskew_length	Set Deskew Code transfer times.
				[31:8]	RSVD	Reserved bits.
0x18	DESKEW_INTERVAL	RW	8'd16	[7:0]	deskew_interval	Set the Deskew Code transmission interval (number of times the Idle code is sent between Deskew codes).
				[31:8]	RSVD	Reserved bits.
0x1C	STDBY_LENGTH	RW	8'd16	[7:0]	stdby_length	Set the standby code transfer times.
				[31:8]	RSVD	Reserved bits.
0x20	INITIAL_LENGTH	RW	16'd2047	[7:0]	initial_length	Set the differential low output period during mode change and initialization.
				[31:8]	RSVD	Reserved bits.
0x24	IDLE_CODE	RW	D.00.0	[7:0]	idle_code	Set the symbol for the Idle Code.
				[31:8]	RSVD	Reserved bits.
Configuration Registers²						
0x28	LANE_WIDTH	RW	{depends on the number of lanes	[3:0]	lane_width	Set the number of SLVS-EC Rx lanes.

© 2024 Lattice Semiconductor Corp. All Lattice trademarks, registered trademarks, patents, and disclaimers are as listed at www.latticesemi.com/legal. All other brand or product names are trademarks or registered trademarks of their respective holders. The specifications and information herein are subject to change without notice.

Offset	Register Name	Access	Default Value	Register Description		
				Bit	Field	Field Description
			attribute}			1: 1 lane 2: 2 lanes 4: 4 lanes 6: 6 lanes 8: 8 lanes Others: Setting prohibited
				[31:4]	RSVD	Reserved bits.
0x2C	LINE_LENGTH	RW	{depends on the number of lanes attribute}	[15:0]	line_length	Set the number of pixels per line.
				[31:16]	RSVD	Reserved bits.
0x30	PIXEL_BIT	RW	{depends on the bits per pixel attribute}	[4:0]	pixel_bit	Set bits per pixel based on the data type. 8: RAW8 10: RAW10 12: RAW12 14: RAW14 16: RAW16 Others: Setting prohibited
				[31:5]	RSVD	Reserved bits
0x34	BAUD_GRADE	RW	{depends on the baud grade attribute}	[1:0]	baud_grade	Set grade of PHY baud rate. 1: Set Baud Grade 1 2: Set Baud Grade 2 3: Set Baud Grade 3 Others: Setting prohibited
				[31:2]	RSVD	Reserved bits
0x38	PACKET_FORMAT ³	RW	{29'd0,2'd0,1'd1}	[0]	crc_en	Set Payload CRC or packet footer. 0: Disables CRC 1: Enables CRC
				[2:1]	ecc_opt	Set Payload ECC Option. 0: Disables ECC 1: Enables ECC with 2 parity bytes, 18~226 block length in bytes 2: Enables ECC with 4 parity bytes, 20~228 block length in bytes Others: Setting prohibited
				[31:3]	RSVD	Reserved bits.
0x3C	FIFO_DELAY	RW	{depends on the fifo delay attribute}	[15:0]	fifo_delay	Set delay cycles for reading the FIFO.
				[31:16]	RSVD	Reserved bits.

Notes:

1. This register does not imply how many SYNC/DESKEW packets are needed to achieve alignment; rather, just states the number of packet transfer, and does not necessarily mean that the IP can assure alignment using the programmed value. Transmitter needs to send enough CODES, within the SLVS-EC Receiver requirement, to achieve alignment.
2. Reconfiguration of IP through registers is only supported if new target configuration is smaller/slower than initially set (i.e. BAUD_GRADE==3 to BAUD_GRADE==2; LANE_WIDTH==8 to LANE_WIDTH==6/4/2/1; and so on).
3. Packet footer (CRC) option and ECC for payload data cannot be enabled simultaneously. ECC is currently not supported, RO access to [2:1].

6. Designing with the IP

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

6.1. Generating and Instantiating the IP

You can use the Lattice Radiant software to generate IP modules and integrate them into the device's architecture. The steps below describe how to generate the SLVS-EC Receiver IP in the Lattice Radiant software.

To generate the SLVS-EC Receiver IP:

1. Create a new Lattice Radiant software project or open an existing project.
2. In the **IP Catalog** tab, double-click **SLVS-EC Receiver** under **IP, Audio_Video_And_Image_Processing** category. The **Module/IP Block Wizard** opens as shown in [Figure 6.1](#). Enter values in the **Component name** and the **Create in** fields and click **Next**.

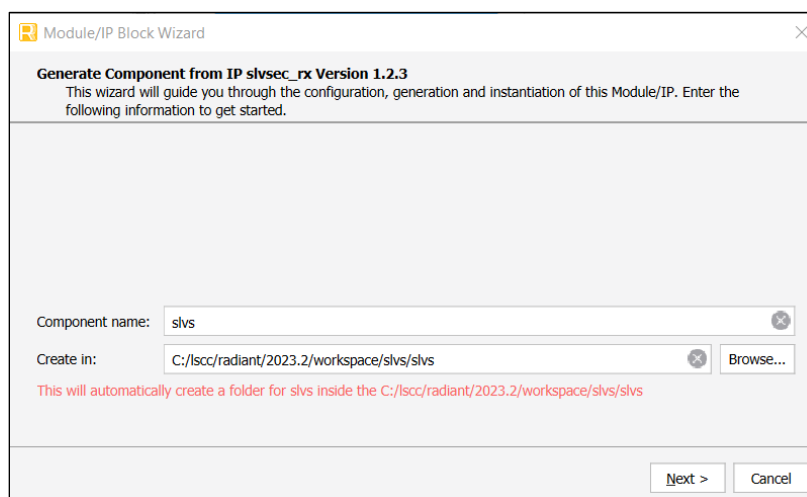


Figure 6.1. Module/IP Block Wizard

3. In the next **Module/IP Block Wizard** window, customize the selected SLVS-EC Receiver IP using drop-down lists and check boxes. [Figure 6.2](#) shows an example configuration of the SLVS-EC Receiver IP. For details on the configuration options, refer to the [IP Parameter Description](#) section.

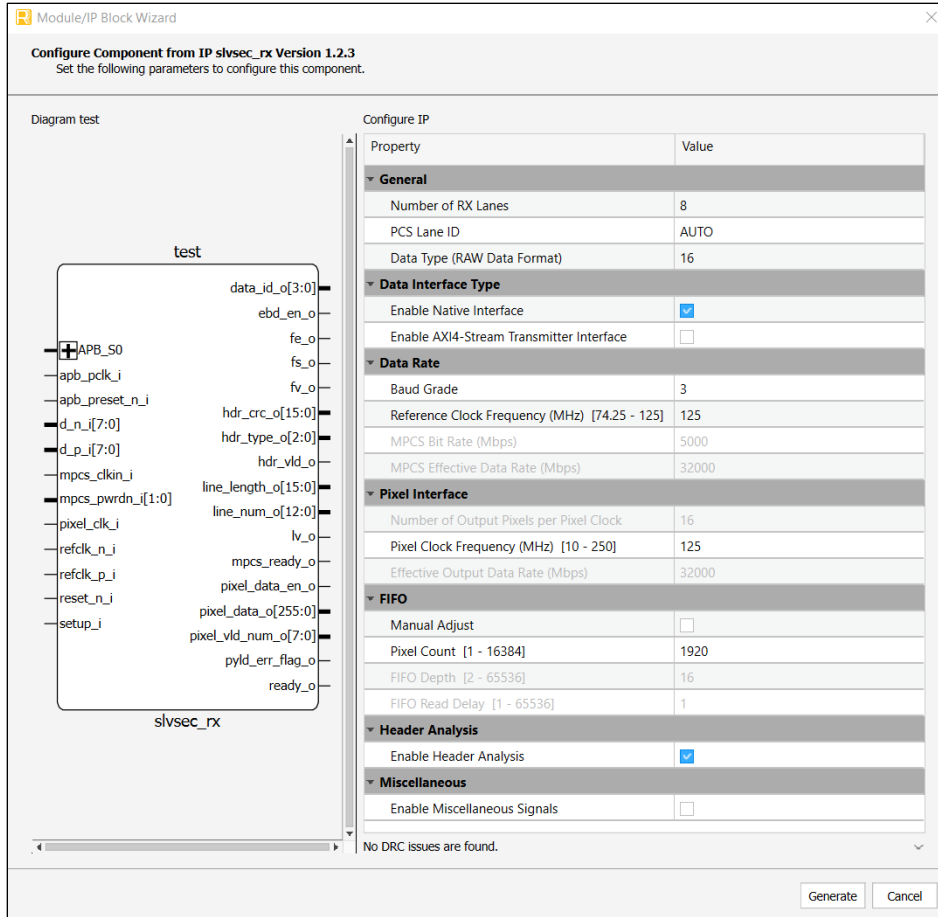


Figure 6.2. IP Configuration

4. Click **Generate**. The **Check Generating Result** dialog box opens, showing design block messages and results as shown in Figure 6.3.

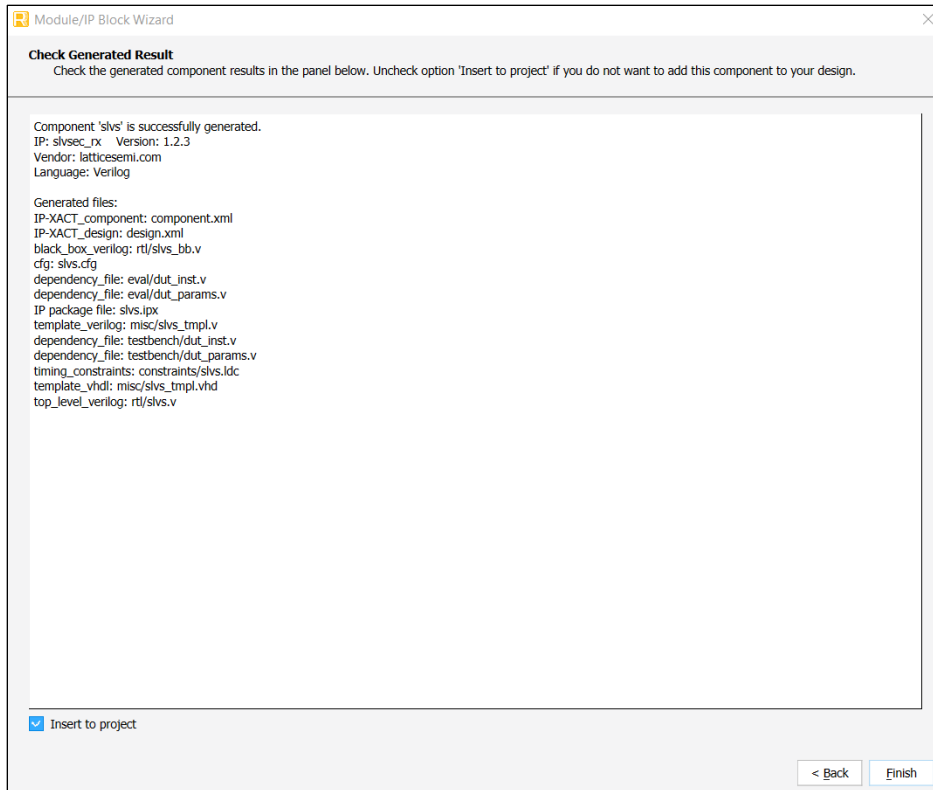


Figure 6.3. Check Generated Result

- Click **Finish**. All the generated files are placed under the directory paths in the **Create in** and the **Component name** fields shown in [Figure 6.1](#).

6.1.1. Generated Files and File Structure

The generated SLVS-EC Receiver module package includes the closed-box (<Component name>_bb.v) and instance templates (<Component name>_tmpl.v/vhd) that can be used to instantiate the core in a top-level design. An example RTL top-level reference source file (<Component name>.v) that can be used as an instantiation template for the module is also provided. You may also use this top-level reference as the starting template for the top-level of the complete design. The generated files are listed in [Table 6.1](#).

Table 6.1. Generated File List

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

6.2. Design Implementation

Completing your design includes additional steps to specify analog properties, pin assignments, and timing and physical design constraints. You can add and edit the constraints using the Device Constraint Editor or by manually creating a PDC File.

Post-Synthesis constraint files (.pdc) contain both timing and non-timing constraint .pdc source files for storing logical timing/physical design constraints. Constraints that are added using the Device Constraint Editor are saved to the active .pdc file. The active post-synthesis design constraint file is then used as input for post-synthesis processes.

Refer to the relevant sections in the Lattice Radiant Software User Guide for more information on how to create or edit constraints and how to use the Device Constraint Editor.

6.3. Timing and Physical Design Constraints

You need to provide proper timing and physical design constraints to ensure that your design meets the desired performance goals on the FPGA device. Refer to the file below when crafting the .pdc file for the project:

```
<IP_Instance_Path>/<IP_Instance_Name>/constraints/<IP_Instance_Name>.ldc.
```

The constraint file has been verified during IP evaluation with the IP instantiated directly in the top-level module. You can modify the constraints in this file with thorough understanding of the effect of each constraint.

To use this constraint file, copy the content of *constraints.pdc* to the top-level design constraint for post-synthesis.

Refer to the [Lattice Radiant Timing Constraints Methodology \(FPGA-AN-02059\)](#) for details on how to constraint your design.


6.4. Specifying the Strategy

The Radiant software provides two predefined strategies: Area and Timing. It also enables you to create customized strategies. For details on how to create a new strategy, refer to the *Strategies* section of the Lattice Radiant Software user guide.

6.5. Running Functional Simulation

You can run functional simulation after the IP is generated.

To run functional simulation:

1. Click the  button located on the **Toolbar** to initiate the **Simulation Wizard** shown in [Figure 6.4](#).

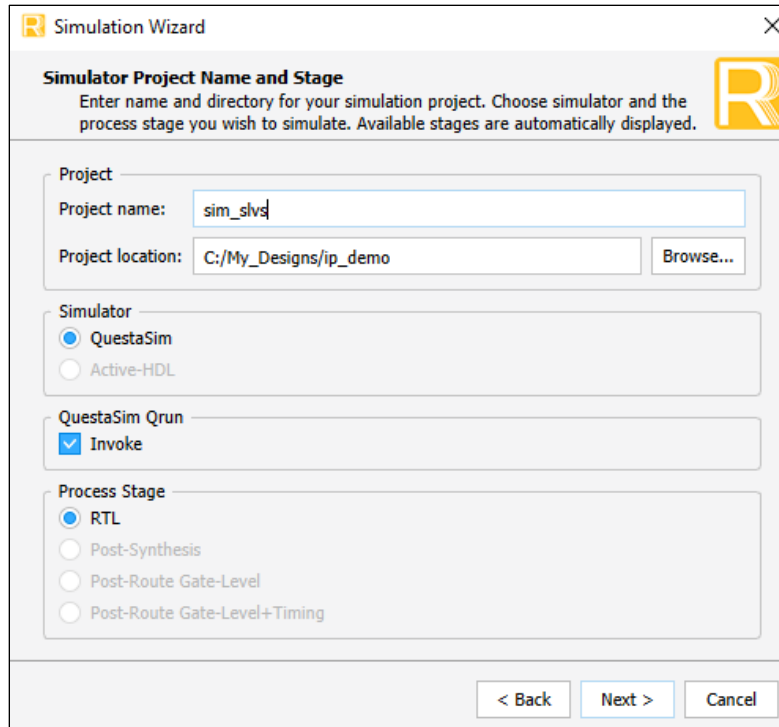


Figure 6.4. Simulation Wizard

2. Click **Next** to open the **Add and Reorder Source** window as shown in Figure 6.5.

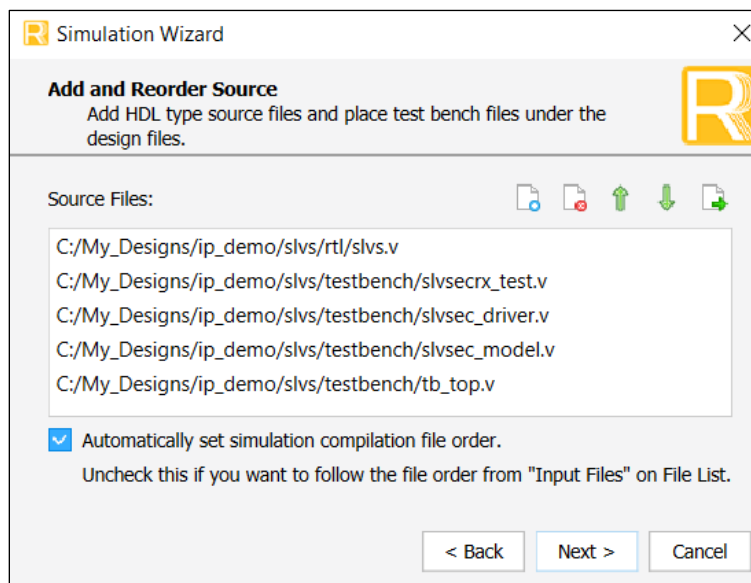


Figure 6.5. Add and Reorder Source

3. Click **Next**. The **Parse HDL files for simulation** window is shown as below. Ensure the **Simulation Top Module** is set to *tb_top*.

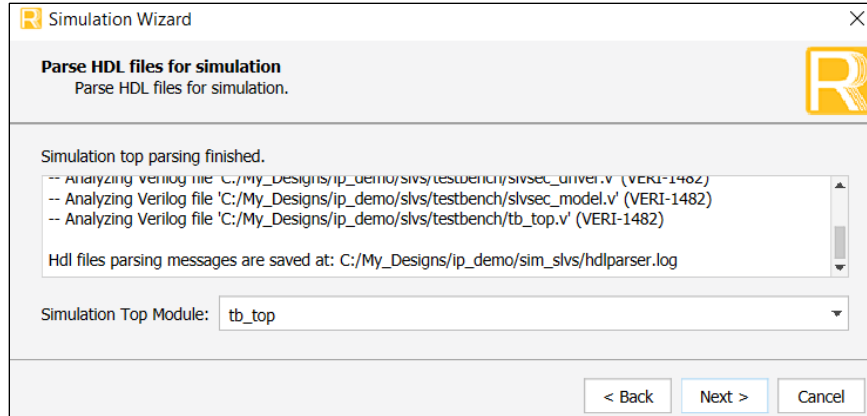


Figure 6.6. Parse HDL files for simulation

4. Click **Next**. The Summary window is shown. Set Default Run to **0 ns** to ensure the simulation is complete.

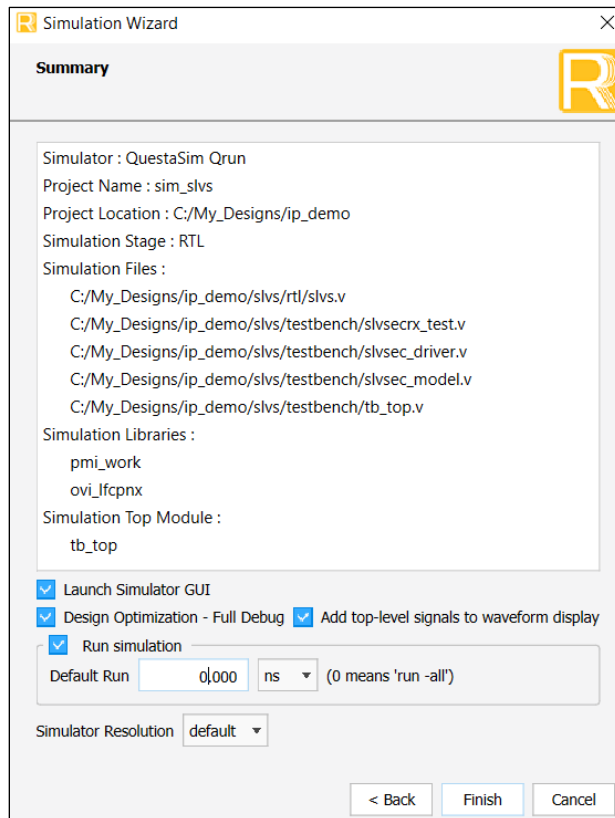


Figure 6.7. Summary

5. Click **Finish** to run the simulation.

The waveform in Figure 6.8 shows an example simulation result.

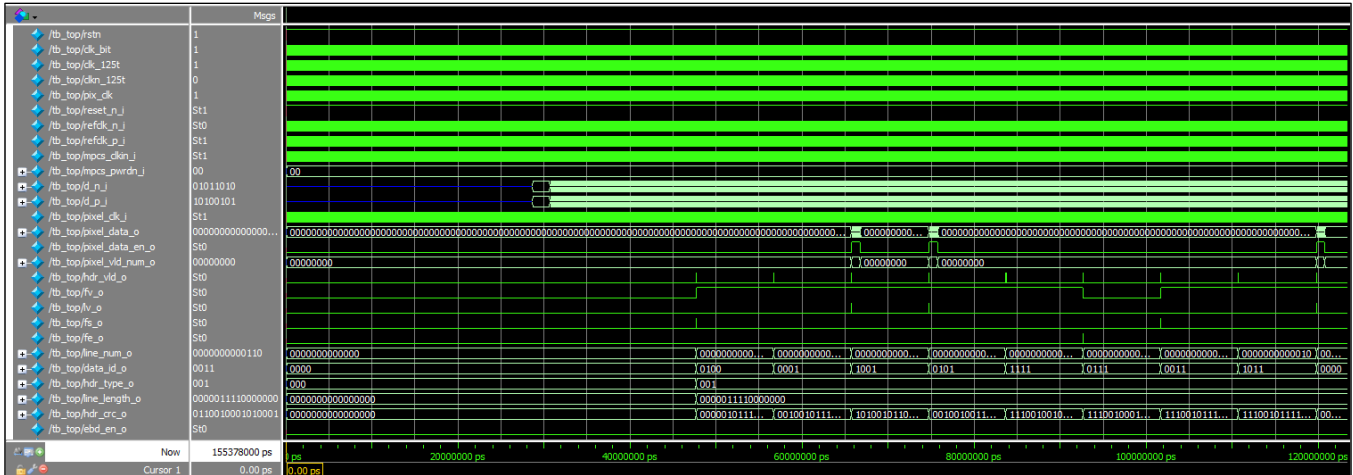


Figure 6.8. Simulation Waveform

After the simulation has been completed, the log message with the *Simulation Done* message will be printed as shown in Figure 6.9 below. Note that the IP testbench does not include data comparator or checker. The testbench is only being set up to send SLVS-EC compliant stimulus to the IP interface ports.

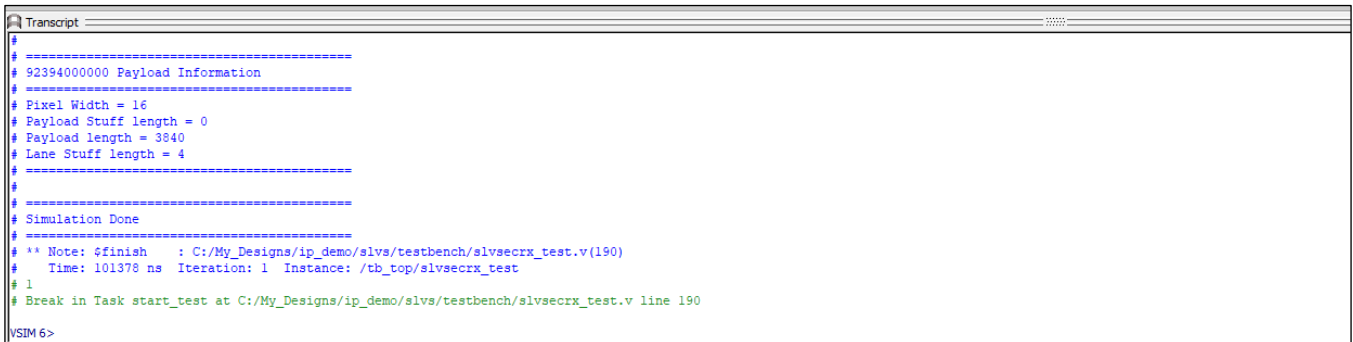


Figure 6.9. Simulation Log on a Complete Run

7. Debugging

This section lists the suggested tools that you can use for your debug.

7.1. Debug Tools

You can use various tools to debug SLVS-EC Receiver IP design issues.

7.1.1. Reveal Analyzer

The Reveal Analyzer continuously monitors signals within the FPGA device for specific conditions that range from simple to complex conditions. When the trigger condition occurs, the Reveal Analyzer saves signal values preceding, during, and following the event for analysis, including a waveform presentation. The data can be saved in the following formats:

- Value change dump file (.vcd) that can be used with tools such as QuestaSim.
- ASCII tabular format that can be used with tools such as Microsoft® Excel.

Before running the Reveal Analyzer, use the Reveal Inserter to add Reveal modules to your design. In these modules, specify the signals to monitor, define the trigger conditions, and set other preferred options. The Reveal Analyzer supports multiple logic analyzer cores using hard/soft JTAG interface. You can have up to 15 modules, typically one for each clock region of interest. When the modules are set up, regenerate the bitstream data file to program the FPGA device. During debug cycles, this tool uses a divide and conquer method to narrow down to problem areas into many small functional blocks to control and monitor the status of each block.

Refer to the *Reveal User Guide for Radiant Software* for details on how to use the Reveal Analyzer.

7.1.2. QuestaSim

The Siemens EDA QuestaSim tool is an OEM simulation tool that is closely linked to the Radiant software environment, and it can be used to perform functional verification of your design and IP. A proper testbench needs to be written to provide input stimulus to the Device Under Test (DUT) and observe the output signals via the QuestaSim Waveform Viewer to verify the correctness of the IP or design. Lattice provides IP testbench and other simulation files when you generate the IP in the Radiant software. You can use this to verify the behavior of the IP and as a reference during your debug activity. To run the IP simulation in QuestaSim, refer to the [Running Functional Simulation](#) section.

Appendix A. Resource Utilization

Table A.1 shows a sample resource utilization of the SLVS-EC Receiver IP Core on the LFCPNX-100-9BBG484C device using Lattice Synthesis Engine of Lattice Radiant software 2024.1.0.

Table A.1. Resource Utilization

Configuration	Clk Fmax (MHz) ¹	Registers	LUTs	Programmable I/O	DSP	sysMEM EBRS
4 Lanes RAW10 Baud Grade 3 FIFO Depth = 16 Native Interface	pixel_clk = 103.466 MHz mpcs_clkln = 162.022MHz	9940	29030	10	6	24

Note:

1. Fmax is generated when the FPGA design only contains SLVS-EC Receiver IP and the target Frequency is 125 MHz. These values may be reduced when a user logic is added to the FPGA design.

Appendix B. Limitation

The unsupported features of the SLVS-EC Receiver IP are listed below:

- The IP does not support payload data Error Correction Code (ECC).
- The IP does not support the following system operations:
 - Standby
 - Mode Change with Standby
 - Interrupt
 - Embedded data
 - Lane deskew during normal operation (that is after packet data transfer)
- The IP does not support the following Frame Synchronization schemes:
 - Digital Signal Processor (DSP) controller
 - CMOS Image Sensor (CIS) controller without shared clock
- The IP does not support extraction of pad codes with only 1 to 3 MPCS Rx clock cycles of active payload in between:
 - Number of bytes per MPCS Rx clock cycle is dependent on the *Baud Grade* selected. Refer to the *MPCS Output Data Width* description in the [Byte-to-Pixel Converter](#) section.

References

- [CertusPro-NX SERDES/PCS Usage Guide \(FPGA-TN-02245\)](#)
- [Lattice Memory Mapped Interface and Lattice Interrupt Interface User Guide \(FPGA-UG-02039\)](#)
- [Lattice Radiant Timing Constraints Methodology \(FPGA-AN-02059\)](#)
- [MPCS Module User Guide \(FPGA-IPUG-02118\)](#)
- [CertusPro-NX web page](#)
- [Lattice Radiant Software web page](#)
- [Lattice Solutions IP Cores web page](#)
- [Lattice Insights web page](#) for Lattice Semiconductor training courses and learning plans

Technical Support Assistance

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

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

Revision History

Revision 2.0, August 2024

Section	Change Summary
All	<ul style="list-style-type: none"> Made editorial fixes. Updated <i>SerDes</i> to <i>SERDES</i>.
Abbreviations in This Document	<ul style="list-style-type: none"> Replaced <i>acronyms</i> with <i>abbreviations</i> in this section. Added the following abbreviations: <ul style="list-style-type: none"> <i>Complementary Metal-Oxide Semiconductor (CMOS)</i> <i>Device Under Test (DUT)</i> <i>Intellectual Property (IP)</i> <i>Least Significant Bit (LSB)</i> <i>Multi-Protocol PCS (MPCS)</i> <i>Receiver (Rx)</i> <i>Serializer Deserializer (SERDES)</i> <i>Transmitter (Tx)</i>
Introduction	<ul style="list-style-type: none"> Moved introductory paragraph in the 1. Introduction section to the 1.1. Overview of the IP section and updated the heading numbers of remaining sections accordingly. Updated Table 1.1. Summary of the SLVS-EC Receiver IP. Removed <i>Not Supported</i> features in the 1.3. Features section. Moved the content from previous 4.1. Licensing the IP and 4.4. Hardware Validation sections to the 1.4. Licensing and Ordering Information section and updated its content. Moved the content from previous 5. Ordering Part Number section to the 1.4.1. Ordering Part Number section and updated its content. Added the 1.5. IP Validation Summary and 1.6. Minimum Device Requirements sections, and updated the heading numbers of remaining sections accordingly. Renamed the previous 1.3. Conventions section to 1.7. Naming Conventions and removed the <i>Attribute</i> information.
Functional Description	<ul style="list-style-type: none"> Moved the content from previous 2.1. General Description and 2.2. Block Diagram sections to the 2.1. IP Architecture Overview section and updated its content. Moved the content from previous 2.2.5. Clock, Reset and Initialization section to the 2.2. Clocking and 2.3. Reset sections and updated their contents. Added the 2.4. User Interfaces section and updated the heading numbers of remaining sections accordingly. Removed the previous 2.2.3. AXI4-Stream Device Transmitter section. Moved the previous 3.3. Timing Descriptions section to the 2.8. Timing Descriptions section.
IP Parameter Description	Moved the content from previous 3.1. Attributes Summary section to this section and updated its content.
Signal Description	Moved the content from previous 3.0. Signal Description section to this section and updated its content.
Register Description	Moved the content from previous 3.2. Register Description section to this section and updated its content.
Designing with the IP	<ul style="list-style-type: none"> Moved the content from previous 4.2. Generation and Synthesis section to the 6.1. Generating and Instantiating the IP section and updated its contents including all the figures. Added the following subsections and updated the heading numbers of remaining sections accordingly: <ul style="list-style-type: none"> 6.2. Design Implementation 6.3. Timing and Physical Design Constraints 6.4. Specifying the Strategy Updated the contents including all the figures in the 6.5. Running Functional Simulation section.
Debugging	Added this section.

Section	Change Summary
Appendix A. Resource Utilization	Updated this section.
Appendix B. Limitation	Moved the previous 1.2.2. <i>Not Supported</i> section to this section and updated its content.
References	<ul style="list-style-type: none"> Added the following references: <ul style="list-style-type: none"> Lattice Radiant Software web page Lattice Solutions IP Cores web page Removed the following references: <ul style="list-style-type: none"> Certus-NX web page CrossLink-NX web page

Revision 1.9, January 2024

Section	Change Summary
All	<ul style="list-style-type: none"> Renamed the document from <i>SLVS-EC Receiver IP Core - Lattice Radiant Software</i> to <i>SLVS-EC Receiver IP</i>. Made editorial fixes.
Disclaimers	Updated boilerplate.
Introduction	<ul style="list-style-type: none"> Moved previous key features in Features section into the Supported section. Added Not Supported section.
Functional Description	<p>Added the following paragraph in the Rx Protocol Management FSM section:</p> <p><i>Current IP only supports lane deskew during initialization/training phase. In cases when lane skew occurs during packet data transfer, you can perform the following procedure:</i></p> <ol style="list-style-type: none"> <i>In the SLVS-EC Rx soft IP, add a custom deskew logic after the MPCS IP instance that will search for the START code from packet data of each lane.</i> <i>Add single clock FIFO that will store these packets after START is detected.</i> <i>Read out from FIFO after START is detected in all lanes.</i>
Signal Description	<p>Updated the Selectable and Default values in Table 3.2. Attributes Table of the following attributes:</p> <ul style="list-style-type: none"> <i>Native Interface</i> <i>AXI4-Stream Transmitter Interface</i> <i>Manual Adjust</i> <i>Pixel Count</i> <i>FIFO Depth</i> <i>FIFO Delay</i> <i>Header Analysis</i> <i>Miscellaneous Signals</i>
Ordering Part Number	Updated this section.
References	<ul style="list-style-type: none"> Updated the names of existing references. Added references to <i>Lattice Radiant Timing Constraints Methodology (FPGA-AN-02059)</i> and <i>Lattice Insights</i> web page.

Revision 1.8, February 2023

Section	Change Summary
Functional Description	<ul style="list-style-type: none"> Updated Figure 2.7. Byte-to-Pixel Converter Implementation to revise 'MPCS clock domain' with 'MPCS Rx clock domain'. Updated Figure 2.2. Clock Domain Crossing Diagram to replace 'mpcs_clk_i' with 'mpcs_clk_in_i'. Replaced 'MPCS Clock' with 'MPCS Rx Clock'. Replaced 'AXI4-Stream Master' with 'AXI4-Stream Transmitter'. Updated Table 2.1. Pre-Configured MPCS PLL Setting for SLVS-EC Rx to add the new column 'MPCS Rx Clock (MHz)'.

Section	Change Summary
Signal Description	<ul style="list-style-type: none"> Updated Table 3.2. Attributes Table to replace 'MPCS Clock Frequency' to 'Reference Clock Frequency'. Replaced 'AXI4-Stream Master' with 'AXI4-Stream Transmitter'.
IP Generation, Simulation, and Validation	Updated Figure 4.2. Configure User Interface of SLVS-EC Rx IP.

Revision 1.7, September 2022

Section	Change Summary
Functional Description	<ul style="list-style-type: none"> Updated the calculating formulas and added sample computations to the SLVS-EC Rx Core section. Changed the description to <i>The MPCS clock used in Table 3.2 is the default 125 MHz</i> in the Clock Domain section. Changed the description to <i>The default MPCS clock displayed in the IP user interface is 125 MHz</i> in the Adjusting Reference Clock for Different Baud Rate section.
Signal Description	<ul style="list-style-type: none"> Updated <code>mpcs_rx_clk_o</code> signal description removing 125 MHz in Table 3.1. SLVS-EC Rx IP Signal Description. Updated the <i>Data Rate</i> group of data in Table 3.2. Attributes Table and Table 3.3. Attributes Descriptions.
Ordering Part Number	Newly added the SLVS-EC-RX-CPNX-US part.

Revision 1.6, May 2022

Section	Change Summary
Introduction	Revised feature to <i>Supports PHY layer functions using the embedded PMA/PCS in CertusPro-NX FPGA device.</i>

Revision 1.5, January 2022

Section	Change Summary
Functional Description	<ul style="list-style-type: none"> Updated MPCS Clock information in Clock Domain. Added Adjusting Reference Clock for Different Baud Rate section.
References	Added reference link to CertusPro-NX SerDes/PCS Usage Guide.

Revision 1.4, November 2021

Section	Change Summary
Introduction	Updated Table 1.1 to change IP version to v1.1.x and Radiant version to 3.0 or later.
Functional Description	Updated FIFO equation in Byte-to-Pixel Converter.
Appendix A. Resource Utilization	Updated Register and LUT values in Table A.1.

Revision 1.3, June 2021

Section	Change Summary
All	<ul style="list-style-type: none"> Minor adjustments in formatting. Added CertusPro-NX across the document,
Introduction	Updated Table 1.1.
Signal Description	<ul style="list-style-type: none"> Updated Table 3.1 to remove LINTR row. Updated Table 3.4 to remove Interrupt Status and Status Register rows and two table notes.
Ordering Part Number	Added this section.

Section	Change Summary
Appendix A. Resource Utilization	Updated Radiant version.

Revision 1.2, April 2021

Section	Change Summary
All	Minor adjustments in formatting.
Acronyms in This Document	Added this section.
Signal Description	<ul style="list-style-type: none"> Updated Table 3.1, Table 3.2, and Table 3.3 to remove Frame Synchronization Signals sections. Updated Table 3.4 to add table note 4 and 5.
IP Generation, Simulation, and Validation	Updated Figure 4.2.
References	Updated section content.

Revision 1.1, March 2021

Section	Change Summary
Introduction	Updated Table 1.1 to change IP v1.0.0 to v1.0.x.
IP Generation, Simulation, and Validation	Updated Running Functional Simulation section content.

Revision 1.0, December 2020

Section	Change Summary
All	Initial release.



www.latticesemi.com