OpenLDI/FPD-Link/LVDS Receiver IP

IP Version: v1.9.0

User Guide

FPGA-IPUG-02116-1.5

December 2025

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 7
1. Introduction 8
1.1. Overview of the IP 8
1.2. Quick Facts 8
1.3. IP Support Summary 9
1.4. Features 9
1.5. Licensing and Ordering Information 9
1.6. Hardware Support 9
1.7. Minimum Device Requirements 9
1.8. Conventions 9
1.8.1. Nomenclature 9
1.8.2. Signal Names 9
1.8.3. Data Ordering and Data Types 10
1.8.4. Attribute Names 10
2. Functional Description 11
2.1. IP Architecture Overview 11
2.2. Clocking 13
2.2.1. Clocking Overview 13
2.3. Reset 14
2.3.1. Reset Overview 14
2.3.2. Initialization and Reset Sequence 14
2.4. User Interfaces 15
2.4.1. LVDS Receiver Interface 15
2.4.2. Native Pixel Interface 19
2.4.3. AXI4-Lite Interface 21
2.4.4. Unified Video Streaming Tx Interface 22
2.5. Other IP Specific Blocks/Layers/Interfaces 24
2.5.1. FPD-Link Rx Core Module 24
2.5.2. GPLL 25
2.5.3. LVDS71 Pixel Map Module 25
2.5.4. Pixel Monitor Module 25
2.5.5. Test Mode Rx Module 26
2.5.6. Synchronizer Module 26
2.6. Dynamic Reconfiguration 26
3. IP Parameter Description 27
3.1. General 27
4. Signal Description 29
4.1. FPD-Link RX Interface 29
4.2. Miscellaneous Status Interface 30
4.3. PLL Interface 30
4.4. AXI4-Lite Interface 31
4.5. Unified Video Streaming Tx Interface 31
5. Register Description 32
5.1. Register Address Map 32
5.2. FPD-Link Rx Configuration Registers 32
6. Designing with the IP 34
6.1. Generating and Instantiating the IP 34
6.1.1. Generated Files and File Structure 36
6.2. Design Implementation 36
6.3. Timing Constraints 37
6.4. Specifying the Strategy 37

6.5. Running Functional Simulation 38
6.5.1. Simulation Results 39
7. Debugging 40
7.1. Debug Methods 40
7.1.1. Debug Mode 40
8. Design Considerations 41
Appendix A. Resource Utilization 42
References 44
Technical Support Assistance 45
Revision History 46

Figures

Figure 2.1. FPD-Link Rx IP Block Diagram 11
Figure 2.2. Functional Block Diagram 12
Figure 2.3. FPD-Link Receiver IP Clock Domain Block Diagram 13
Figure 2.4. Reset Domain Block Diagram..14
Figure 2.5. OpenLDI/FPD-Link/LVDS Input Bus Waveform 15
Figure 2.6. Single Channel OpenLDI/FPD-Link/LVDS Input Bus Waveform for RGB888 Format (VESA) 16
Figure 2.7. Dual Channel OpenLDI/FPD-Link/LVDS Input Bus Waveform for RGB888 Format (VESA) 16
Figure 2.8. Single Channel OpenLDI/FPD-Link/LVDS Input Bus Waveform for RGB888 Format (JEIDA) 17
Figure 2.9. Dual Channel OpenLDI/FPD-Link/LVDS Input Bus Waveform for RGB888 Format (JEIDA) 17
Figure 2.10. Single Channel OpenLDI/FPD-Link/LVDS Input Bus Waveform for RGB666 Format (JEIDA/VESA) 18
Figure 2.11. Dual Channel OpenLDI/FPD-Link/LVDS Input Bus Waveform for RGB666 Format (JEIDA/VESA)..................18
Figure 2.12. Input to Output Waveform for Single Channel OpenLDI/FPD-Link/LVDS......................................19
Figure 2.13. Input to Output Waveform for Dual Channel OpenLDI/FPD-Link/LVDS..20
Figure 2.14. Output Pixel Data RGB Arrangement...20
Figure 2.15. Output Pixel Data Arrangement for Single and Dual Channel OpenLDI/FPD-Link/LVDS...............21
Figure 2.16. Unified Video Streaming Tx Timing Diagram 22
Figure 2.17. Unified Video Streaming Tx Timing Diagram with De-assertion of TREADY..................................22
Figure 2.18. Unified Video Streaming Tx TDATA Mapping of RGB888 and RGB666 for PPC = 1 23
Figure 2.19. Unified Video Streaming Tx TDATA Mapping of RGB888 and RGB666 for PPC = 2 23
Figure 2.20. GPLL connection for Avant Devices 25
Figure 6.1. Module/IP Block Wizard 34
Figure 6.2. IP Configuration 35
Figure 6.3. Check Generated Result 35
Figure 6.4. General Evaluation PDC File Header...37
Figure 6.5. Simulation Wizard...38
Figure 6.6. Add and Reorder Source...38
Figure 6.7. Simulation Waveform 39
Figure 6.8. Simulation Results 39

Tables

Table 1.1. Summary of the FPD-Link Receiver IP 8
Table 1.2. FPD-Link Receiver IP Support Readiness 9
Table 2.1. User Interfaces and Supported Protocols 15
Table 2.2. Output Pixel Data Summary 21
Table 2.3. Data Mapping of Unified Video Streaming Tx Interface Ports 22
Table 3.1. General Attributes 27
Table 4.1. FPD-Link Rx Interface Ports 29
Table 4.2. Miscellaneous Status Interface Ports 30
Table 4.3. PLL Interface Ports 30
Table 4.4. AXI4-Lite Interface Ports 31
Table 4.5. Unified Video Streaming Tx Interface Ports 31
Table 5.1. Configuration Register Address Map 32
Table 5.2. CFG_REG Register 32
Table 5.3. TEST_MODE Register 32
Table 5.4. MISC_SIGNALS Register 33
Table 5.5. ACT_WIDTH Register 33
Table 5.6. ACT_HEIGHT Register 33
Table 5.7. LOCKED Register 33
Table 6.1. Generated File List 36
Table A.1. CertusPro-NX Device and Tool Information 42
Table A.2. Resource Utilization on CertusPro-NX Device[1] 42
Table A.3. Avant Device and Tool Information 42
Table A.4. Resource Utilization on Avant Device[1] 42

Abbreviations in This Document

A list of abbreviations used in this document.

Abbreviation	Definition
AXI	Advance eXtensible Interface
CSR	Control and Status Register
FPD	Flat Panel Display
LVDS	Low Voltage Differential Signaling
LDI	LVDS Display Interface
PPC	Pixels per Clock
RTL	Register Transfer Language
UVS	Unified Video Streaming

1. Introduction

The increasing demand for better display technology makes bridging applications popular. FPD-Link is a common application interface. Similar to Channel Link and Camera Link, FPD-Link also uses LVDS interface for the physical layer.

The LVDS standard is commonly used for high-speed differential interface in consumer device, industrial control, medical, and automotive applications. It offers low voltage, low power, and improved signal integrity, which are advantages over single-ended technology.

The 7:1 LVDS interface is a popular standard for source synchronous interfaces which consist of multiple data bits and clocks. Typically, one channel of 7:1 LVDS interface consists of five LVDS pairs (one clock and four data) depending on the data type it supports.

This document describes the use of the FPD-Link Receiver IP and Lattice FPGA technology for LVDS interface applications. The design, which can be applied in multiple configurations, is implemented in Verilog HDL. It can be targeted to CrossLink™-NX, Certus™-NX, CertusPro-NX™, MachXO5™-NX, Lattice Avant™, and Certus-N2 FPGA devices and implemented using the Lattice Radiant™ software Place and Route tool integrated with the Synplify Pro® synthesis tool.

1.1. Overview of the IP

The Lattice Semiconductor FPD-Link Receiver IP translates video streams from a processor with an OpenLDI/FPD-Link/LVDS interface connection to the pixel clock domain. The input interface of the IP consists of a data bus, vertical and horizontal sync flags, a data enable, and a clock in the OpenLDI (LVDS 7:1) interface format. The output interface consists of the RGB control signals, pixel clock, up to four pixel data per pixel clock, and debug signals. This IP can be used to connect with other application interfaces, such as the Mobile Industry Processor Interface (MIPI®) Display Serial Interface (DSI), by integrating it with the Pixel-to-Byte Converter and CSI-2/DSI D-PHY Transmitter IP cores.

1.2. Quick Facts

Table 1.1. Summary of the FPD-Link Receiver IP

IP Requirements	Supported Devices	CrossLink-NX, Certus-NX, CertusPro-NX, MachXO5-NX, Lattice Avant, Certus-N2
	IP Changes[1]	Refer to the OpenLDI/FPD-Link/LVDS Receiver IP Release Notes (FPGA-RN-02013).
Resource Utilization	Supported User Interface	Native LVDS to Parallel Interface, Unified Video Streaming Tx Interface, AXI4-Lite Interface (for register access)
	Resources	Refer to Appendix A. Resource Utilization.
Design Tool Support	Lattice Implementation	IP Core v1.9.0 – Lattice Radiant Software 2025.2
	Synthesis	Synopsys® Synplify Pro® for Lattice, Lattice Synthesis Engine
	Simulation	Refer to the Lattice Radiant Software User Guide for the list of supported simulators.

Note:

1. In some instances, the IP may be updated without changes to the user guide. This user guide may reflect an earlier IP version but remains fully compatible with the later IP version. Refer to the IP Release Notes for the latest updates.

1.3. IP Support Summary

Table 1.2. FPD-Link Receiver IP Support Readiness

Device Family	Rx Channels	Data Mapping Format	Data Type	Video Data Interface	Rx Line Rate (Mb/s)	Register Interface	Radiant Timing Model	Hardware Validated
CertusPro-NX	1, 2	VESA, JEIDA	RGB888, RGB666	Native, Unified Video Streaming Tx	126–945	OFF, AXI4-Lite	Final	No
Avant	1, 2	VESA, JEIDA	RGB888, RGB666	Native, Unified Video Streaming Tx	109.375–1050	OFF, AXI4-Lite	Preliminary	No

1.4. Features

Key features of the FPD-Link Receiver IP include:

- Compliant with Open LVDS Display Interface (OpenLDI) v0.95 specifications.
- Receives in OpenLDI unbalanced operating mode format.
- Supports RGB888 and RGB666 video formats.
- Supports receiving in Dual Channel Flat Panel Display Link protocol (7:1 LVDS).
- Supports three to four LVDS data lanes per channel.
- Supports one or two output pixels per pixel clock.
- Supports interfacing up to 7.560 Gb/s for Nexus devices and 8.4 Gb/s for Avant devices.
- Compliant with the AXI4-Stream protocol through the Unified Video Streaming Tx interface.
- Supports AXI4-Lite interface for register access.
- Supports dynamic reconfiguration in Native Pixel and Unified Video Streaming Tx interfaces.

1.5. Licensing and Ordering Information

The FPD-Link Receiver IP is provided at no additional cost with the Lattice Radiant software.

1.6. Hardware Support

Hardware support is available in a future release.

1.7. Minimum Device Requirements

Refer to Appendix A for the minimum device requirements to instantiate the IP.

1.8. Conventions

1.8.1. Nomenclature

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

1.8.2. Signal Names

Signal names that end with:

- _*n* are active low (asserted when value is logic 0)
- _*i* are input signals
- _*o* are output signals

1.8.3. Data Ordering and Data Types

The most significant bit within the pixel data is the highest index.

1.8.4. Attribute Names

Attribute names in this document are formatted in title case and italicized (*Attribute Name*).

2. Functional Description

2.1. IP Architecture Overview

The FPD-Link Receiver IP converts a standard OpenLDI serial video interface into the pixel clock domain. The input interface for the design consists of a data bus, vertical and horizontal sync flags, a data enable, and a clock in the OpenLDI (FPD-Link) interface format. The output interface consists of the RGB control signals, pixel clock, up to two pixels per clock, and optional debug signals.

Figure 2.1 shows the IP block diagram of the FPD-Link Receiver IP core. The dashed lines in the figure are optional components or signals, which may or may not be available in the IP depending on the attribute and/or device selected.

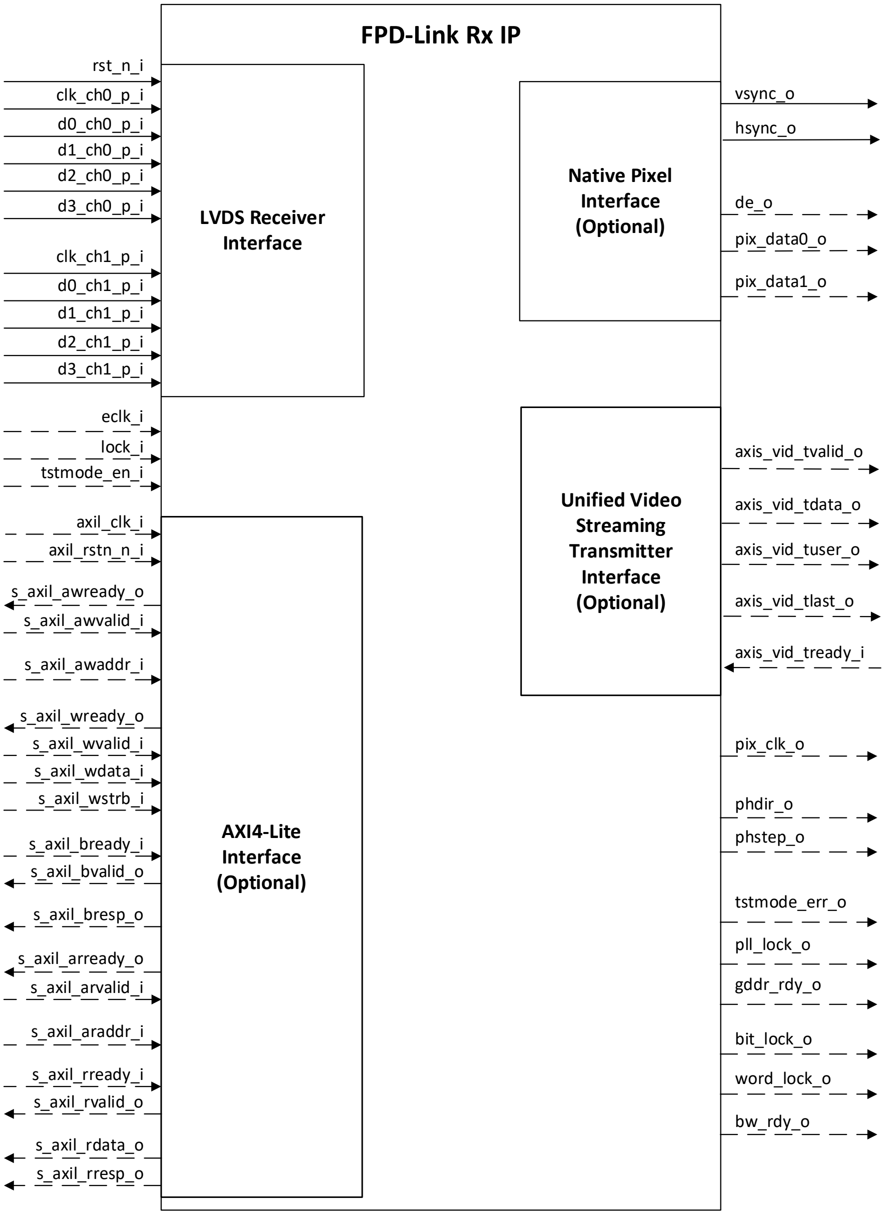


Figure 2.1. FPD-Link Rx IP Block Diagram

Figure 2.2 shows the functional block diagram of the FPD-Link Receiver IP core.

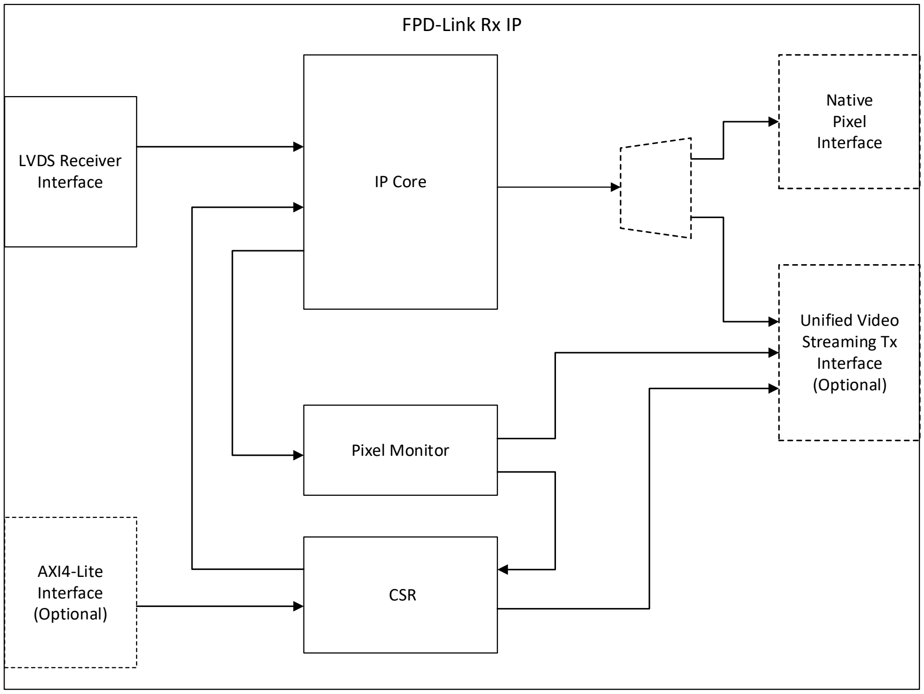


Figure 2.2. Functional Block Diagram

The FPD-Link Receiver IP includes the following layers:

- IP core
- LVDS Receiver interface
- AXI4-Lite interface (optional)
- Native Pixel interface (optional)
- Unified Video Streaming Tx interface (optional)
- Control and Status Register (CSR) module
- Pixel monitor module

2.2. Clocking

The FPD-Link Receiver IP operates on different clock domains which are the FPD-Link serial clock domain, pixel clock domain, and AXI4-Lite clock domain. For Avant devices, an additional ECLK (eclk_i) is required as an input to the IP. Figure 2.3 shows the different clock domains.

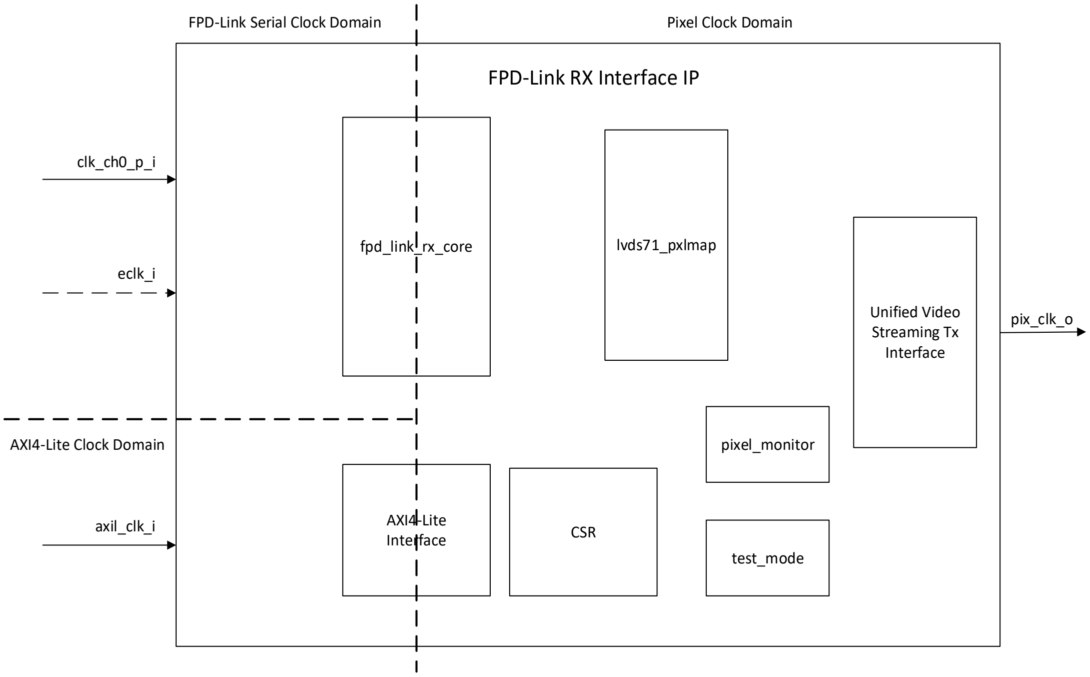


Figure 2.3. FPD-Link Receiver IP Clock Domain Block Diagram

2.2.1. Clocking Overview

The general formula for computing the required clocks of the IP:

Rx Line Rate (Total) $= RX\ Line\ Rate\ (per\ Lane) \times\ No.of\ RX\ Lanes \times No.of\ Rx\ Channels$

Pixclk (Pixel Clock) $= \frac{RX\ Line\ Rate\ (per\ Lane)}{RX\ Gear}$

Rx LVDS Input Clock $= Pixclk \times \frac{RX\ Gear}{7}$

Rx LVDS ECLK $= Pixclk \times \frac{RX\ Gear}{2}$

No. of Pixels per Pixel Clock $= \frac{RX\ Gear}{7} \times No.of\ Rx\ Channels$

2.3. Reset

The FPD-Link Receiver IP has two reset signals. One reset signal is used to reset the system. The other reset signal is used to reset the AXI4-Lite interface. Figure 2.4 shows the different reset domains.

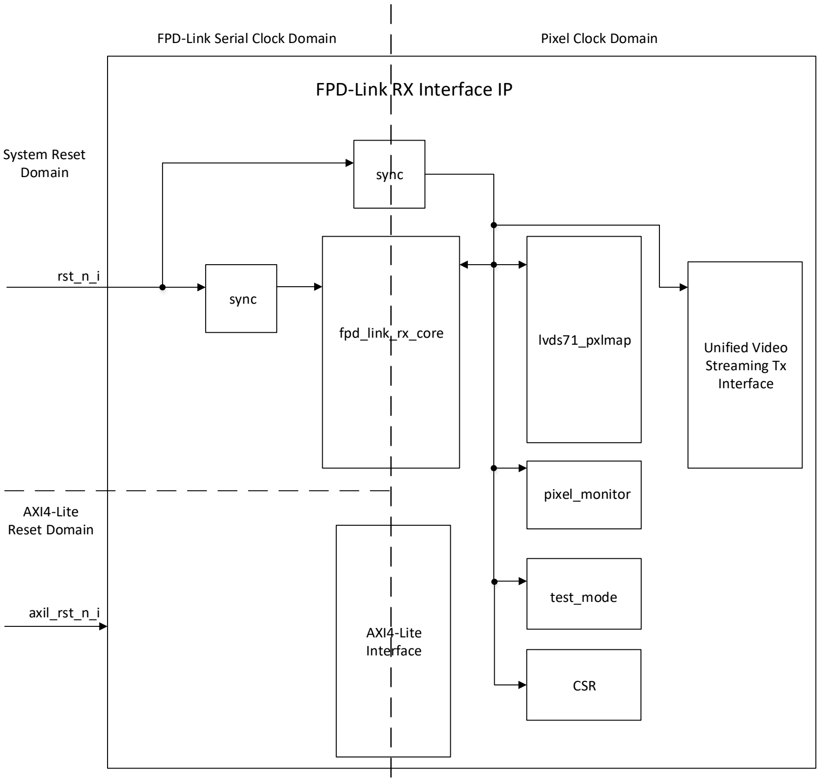


Figure 2.4. Reset Domain Block Diagram

2.3.1. Reset Overview

The rst_n_i signal is an asynchronous active low system reset but with a synchronous release with respect to both the serial clock domain and pixel clock domain. Synchronizers are used to sync the reset to different clock domains in the system. A separate axil_rst_n_i signal is used to reset the AXI4-Lite Interface.

2.3.2. Initialization and Reset Sequence

Follow this initialization and reset sequence:

1. Assert active low system reset for at least three clock cycles of the slowest clock (pixel clock). It is expected that the input clock is already stable immediately after reset. Clock synchronization starts immediately after release of system reset.
   Note: Active low reset is used in the design with synchronous release. This is the system reset (rst_n_i) input connected to the LVDS7:1 Rx module.
2. Wait for GPLL lock to be asserted. GPLL lock is used to indicate that the GPLL output clock is or GPLL output clocks are already stable. For Avant devices, this is indicated by the lock_i signal while for Nexus devices, this is indicated by the pll_lock_o signal. Wait for the bw_rdy_o signal to be asserted. The bw_rdy_o signal is used to indicate that LVDS 7:1 Rx data training is done. Only when bw_rdy_o is asserted can valid data be sampled and correctly transmitted by the FPD-Link Rx IP.

If *Enable Miscellaneous Signals* is checked, wait for bw_rdy_o to be asserted. If *Enable Miscellaneous Signals* is unchecked, wait for pixel clock (pixel_clk_o) to be stable or until the computed pixel clock frequency is achieved before sending the valid data to allow Rx clock synchronization and data training to complete. Refer to the Clocking Overview section for more details on how pixel clock frequency is computed.

2.4. User Interfaces

Table 2.1 lists the available user interfaces and supported protocols of the FPD-Link Receiver IP.

Table 2.1. User Interfaces and Supported Protocols

User Interface	Supported Protocols	Description
LVDS Receiver Interface	OpenLDI/FPD-Link (LVDS 7:1) Serial Video Interface	Receives data in the OpenLDI serial video format to be converted to the standard CSI2/DSI parallel video format.
Native Pixel Interface	CSI2/DSI Parallel Video Interface	Outputs the converted CSI2/DSI parallel pixel data stream.
Control	AXI4-Lite Interface	Allows access to the control and status registers of the FPD-Link Receiver IP.
Unified Video Streaming Tx Interface	AXI4-Stream Interface	When enabled, this interface transmits pixel data being converted by the FPD-Link Receiver IP. This interface is compliant with the AXI4-Stream Interface.

2.4.1. LVDS Receiver Interface

Figure 2.5 shows the timing of the LVDS 7:1/FPD-Link input interface. There is a 2-bit offset between the rising edge of the LVDS clock and the word boundary. Each word is 7 bits long.

DATAIN0, DATAIN1, DATAIN2, and DATAIN3 are the data lanes. CLKIN is the LVDS clock lane. For every 7-bit data packet, LSB is the first input serial data to the receiver.

A processor sends video packet data to the FPGA chip through the OpenLDI/FPD-Link (LVDS 7:1) interface. One channel of the LVDS 7:1 receiver has a maximum of five lanes. Each channel consists of one LVDS clock pair and four LVDS data pairs (RGB888) or three LVDS data pairs (RGB666). A maximum of two LVDS 7:1 channels can be used. When dual channel is selected, additional data lanes are activated. The clock runs at 1/7th of the data rate, as per the standard for the LVDS 7:1 interface. The default mode for the LVDS operating system is unbalanced, as this is commonly used. The maximum supported data rate per lane for LVDS is 945 Mb/s for Nexus devices and 1050 Mb/s for Avant devices.

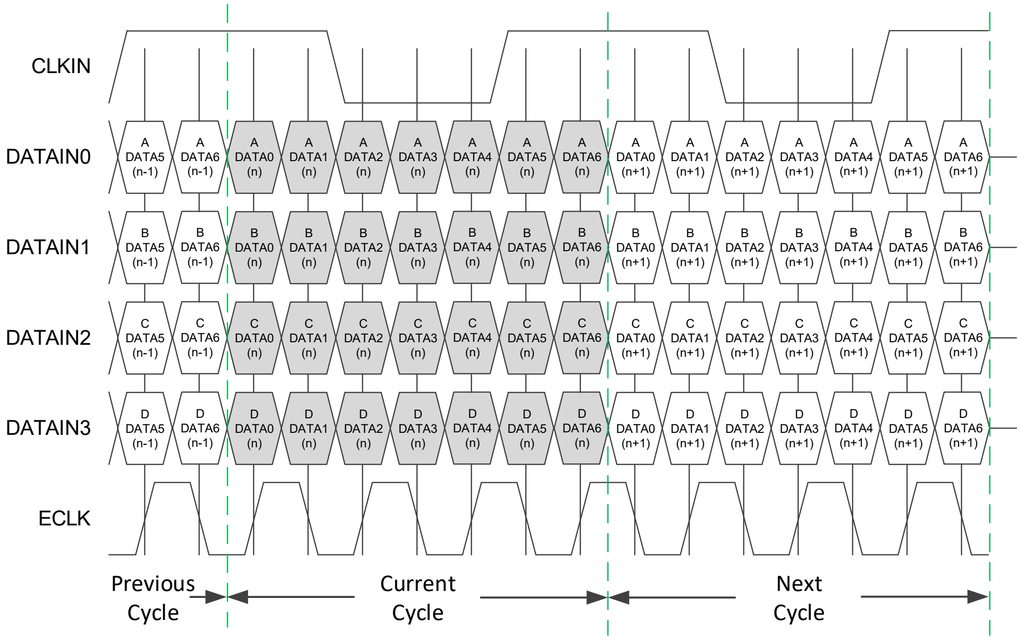


Figure 2.5. OpenLDI/FPD-Link/LVDS Input Bus Waveform

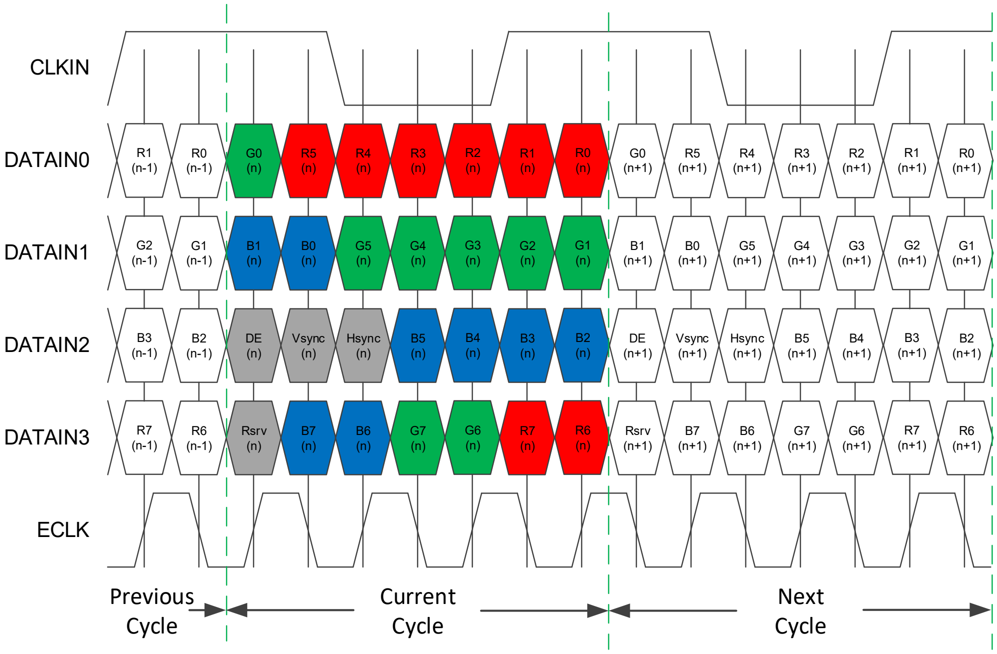

CLKIN
DATAIN0
R1 (n-1) R0 (n-1) G0 (n) R5 (n) R4 (n) R3 (n) R2 (n) R1 (n) R0 (n) G0 (n+1) R5 (n+1) R4 (n+1) R3 (n+1) R2 (n+1) R1 (n+1) R0 (n+1)
DATAIN1
G2 (n-1) G1 (n-1) B1 (n) B0 (n) G5 (n) G4 (n) G3 (n) G2 (n) G1 (n) B1 (n+1) B0 (n+1) G5 (n+1) G4 (n+1) G3 (n+1) G2 (n+1) G1 (n+1)
DATAIN2
B3 (n-1) B2 (n-1) DE (n) Vsync (n) Hsync (n) B5 (n) B4 (n) B3 (n) B2 (n) DE (n+1) Vsync (n+1) Hsync (n+1) B5 (n+1) B4 (n+1) B3 (n+1) B2 (n+1)
DATAIN3
R7 (n-1) R6 (n-1) Rsrv (n) B7 (n) B6 (n) G7 (n) G6 (n) R7 (n) R6 (n) Rsrv (n+1) B7 (n+1) B6 (n+1) G7 (n+1) G6 (n+1) R7 (n+1) R6 (n+1)
ECLK
Previous Cycle
Current Cycle
Next Cycle

Figure 2.6. Single Channel OpenLDI/FPD-Link/LVDS Input Bus Waveform for RGB888 Format (VESA)

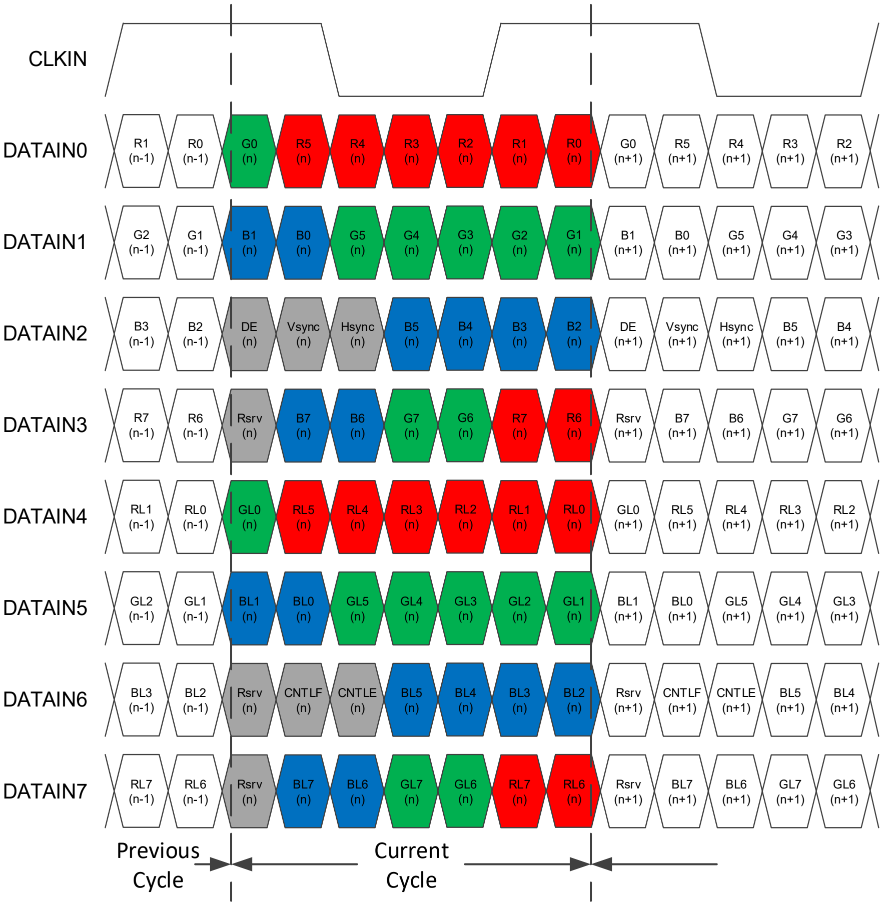

CLKIN
DATAIN0
R1 (n-1) R0 (n-1) G0 (n) R5 (n) R4 (n) R3 (n) R2 (n) R1 (n) R0 (n) G0 (n+1) R5 (n+1) R4 (n+1) R3 (n+1) R2 (n+1)
DATAIN1
G2 (n-1) G1 (n-1) B1 (n) B0 (n) G5 (n) G4 (n) G3 (n) G2 (n) G1 (n) B1 (n+1) B0 (n+1) G5 (n+1) G4 (n+1) G3 (n+1)
DATAIN2
B3 (n-1) B2 (n-1) DE (n) Vsync (n) Hsync (n) B5 (n) B4 (n) B3 (n) B2 (n) DE (n+1) Vsync (n+1) Hsync (n+1) B5 (n+1) B4 (n+1)
DATAIN3
R7 (n-1) R6 (n-1) Rsrv (n) B7 (n) B6 (n) G7 (n) G6 (n) R7 (n) R6 (n) Rsrv (n+1) B7 (n+1) B6 (n+1) G7 (n+1) G6 (n+1)
DATAIN4
RL1 (n-1) RL0 (n-1) GL0 (n) RL5 (n) RL4 (n) RL3 (n) RL2 (n) RL1 (n) RL0 (n) GL0 (n+1) RL5 (n+1) RL4 (n+1) RL3 (n+1) RL2 (n+1)
DATAIN5
GL2 (n-1) GL1 (n-1) BL1 (n) BL0 (n) GL5 (n) GL4 (n) GL3 (n) GL2 (n) GL1 (n) BL1 (n+1) BL0 (n+1) GL5 (n+1) GL4 (n+1) GL3 (n+1)
DATAIN6
BL3 (n-1) BL2 (n-1) Rsrv (n) CNTLF (n) CNTLE (n) BL5 (n) BL4 (n) BL3 (n) BL2 (n) Rsrv (n+1) CNTLF (n+1) CNTLE (n+1) BL5 (n+1) BL4 (n+1)
DATAIN7
RL7 (n-1) RL6 (n-1) Rsrv (n) BL7 (n) BL6 (n) GL7 (n) GL6 (n) RL7 (n) RL6 (n) Rsrv (n+1) BL7 (n+1) BL6 (n+1) GL7 (n+1) GL6 (n+1)
Previous Cycle
Current Cycle

Figure 2.7. Dual Channel OpenLDI/FPD-Link/LVDS Input Bus Waveform for RGB888 Format (VESA)

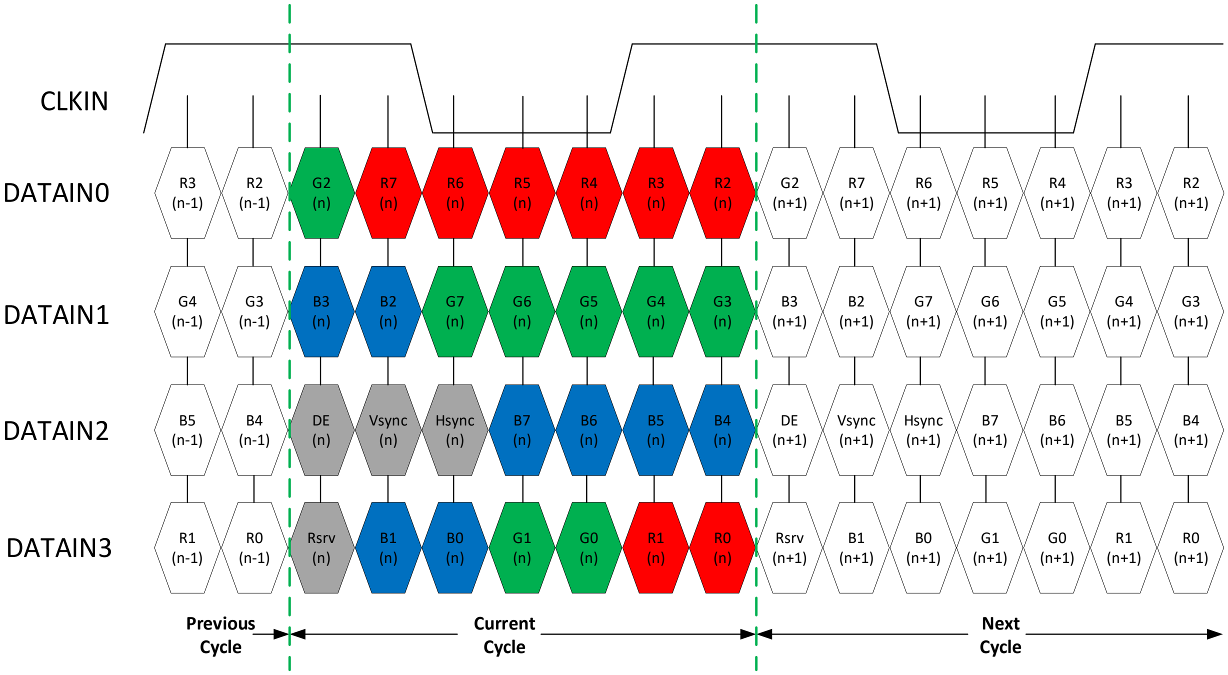

CLKIN
DATAIN0: R3 (n-1), R2 (n-1), G2 (n), R7 (n), R6 (n), R5 (n), R4 (n), R3 (n), R2 (n), G2 (n+1), R7 (n+1), R6 (n+1), R5 (n+1), R4 (n+1), R3 (n+1), R2 (n+1)
DATAIN1: G4 (n-1), G3 (n-1), B3 (n), B2 (n), G7 (n), G6 (n), G5 (n), G4 (n), G3 (n), B3 (n+1), B2 (n+1), G7 (n+1), G6 (n+1), G5 (n+1), G4 (n+1), G3 (n+1)
DATAIN2: B5 (n-1), B4 (n-1), DE (n), Vsync (n), Hsync (n), B7 (n), B6 (n), B5 (n), B4 (n), DE (n+1), Vsync (n+1), Hsync (n+1), B7 (n+1), B6 (n+1), B5 (n+1), B4 (n+1)
DATAIN3: R1 (n-1), R0 (n-1), Rsrv (n), B1 (n), B0 (n), G1 (n), G0 (n), R1 (n), R0 (n), Rsrv (n+1), B1 (n+1), B0 (n+1), G1 (n+1), G0 (n+1), R1 (n+1), R0 (n+1)
Previous Cycle
Current Cycle
Next Cycle

Figure 2.8. Single Channel OpenLDI/FPD-Link/LVDS Input Bus Waveform for RGB888 Format (JEIDA)

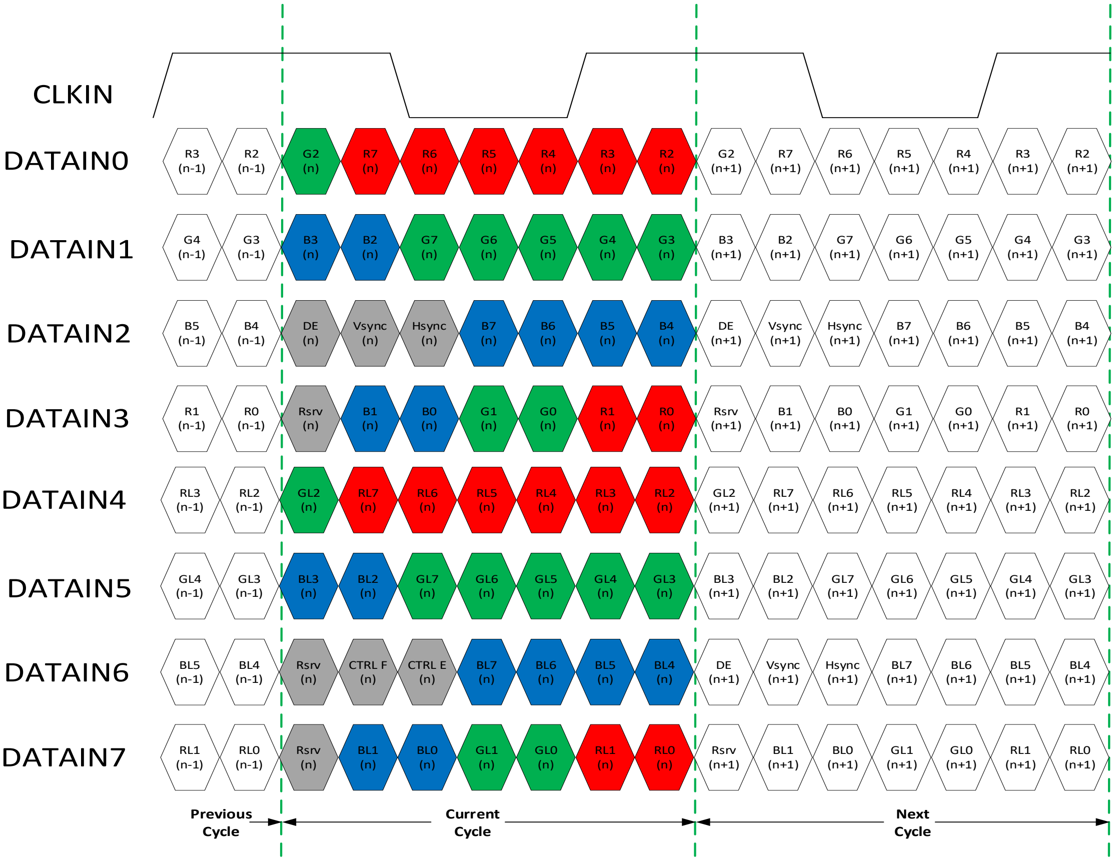

CLKIN
DATAIN0: R3 (n-1), R2 (n-1), G2 (n), R7 (n), R6 (n), R5 (n), R4 (n), R3 (n), R2 (n), G2 (n+1), R7 (n+1), R6 (n+1), R5 (n+1), R4 (n+1), R3 (n+1), R2 (n+1)
DATAIN1: G4 (n-1), G3 (n-1), B3 (n), B2 (n), G7 (n), G6 (n), G5 (n), G4 (n), G3 (n), B3 (n+1), B2 (n+1), G7 (n+1), G6 (n+1), G5 (n+1), G4 (n+1), G3 (n+1)
DATAIN2: B5 (n-1), B4 (n-1), DE (n), Vsync (n), Hsync (n), B7 (n), B6 (n), B5 (n), B4 (n), DE (n+1), Vsync (n+1), Hsync (n+1), B7 (n+1), B6 (n+1), B5 (n+1), B4 (n+1)
DATAIN3: R1 (n-1), R0 (n-1), Rsrv (n), B1 (n), B0 (n), G1 (n), G0 (n), R1 (n), R0 (n), Rsrv (n+1), B1 (n+1), B0 (n+1), G1 (n+1), G0 (n+1), R1 (n+1), R0 (n+1)
DATAIN4: RL3 (n-1), RL2 (n-1), GL2 (n), RL7 (n), RL6 (n), RL5 (n), RL4 (n), RL3 (n), RL2 (n), GL2 (n+1), RL7 (n+1), RL6 (n+1), RL5 (n+1), RL4 (n+1), RL3 (n+1), RL2 (n+1)
DATAIN5: GL4 (n-1), GL3 (n-1), BL3 (n), BL2 (n), GL7 (n), GL6 (n), GL5 (n), GL4 (n), GL3 (n), BL3 (n+1), BL2 (n+1), GL7 (n+1), GL6 (n+1), GL5 (n+1), GL4 (n+1), GL3 (n+1)
DATAIN6: BL5 (n-1), BL4 (n-1), Rsrv (n), CTRL F (n), CTRL E (n), BL7 (n), BL6 (n), BL5 (n), BL4 (n), DE (n+1), Vsync (n+1), Hsync (n+1), BL7 (n+1), BL6 (n+1), BL5 (n+1), BL4 (n+1)
DATAIN7: RL1 (n-1), RL0 (n-1), Rsrv (n), BL1 (n), BL0 (n), GL1 (n), GL0 (n), RL1 (n), RL0 (n), Rsrv (n+1), BL1 (n+1), BL0 (n+1), GL1 (n+1), GL0 (n+1), RL1 (n+1), RL0 (n+1)
Previous Cycle
Current Cycle
Next Cycle

Figure 2.9. Dual Channel OpenLDI/FPD-Link/LVDS Input Bus Waveform for RGB888 Format (JEIDA)

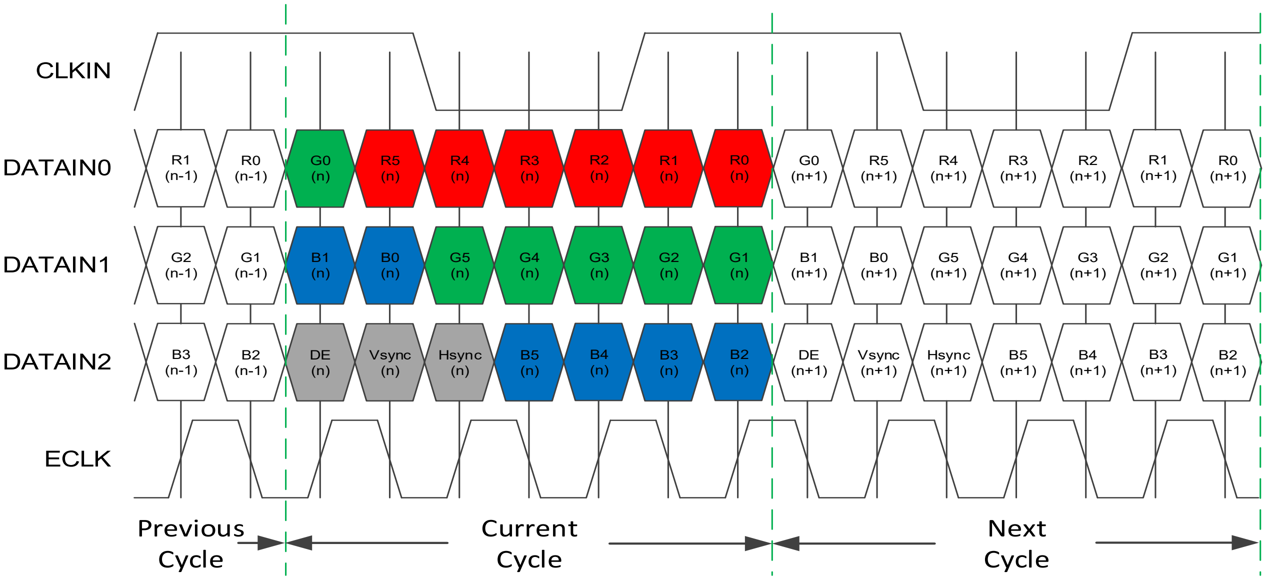


Figure 2.10. Single Channel OpenLDI/FPD-Link/LVDS Input Bus Waveform for RGB666 Format (JEIDA/VESA)

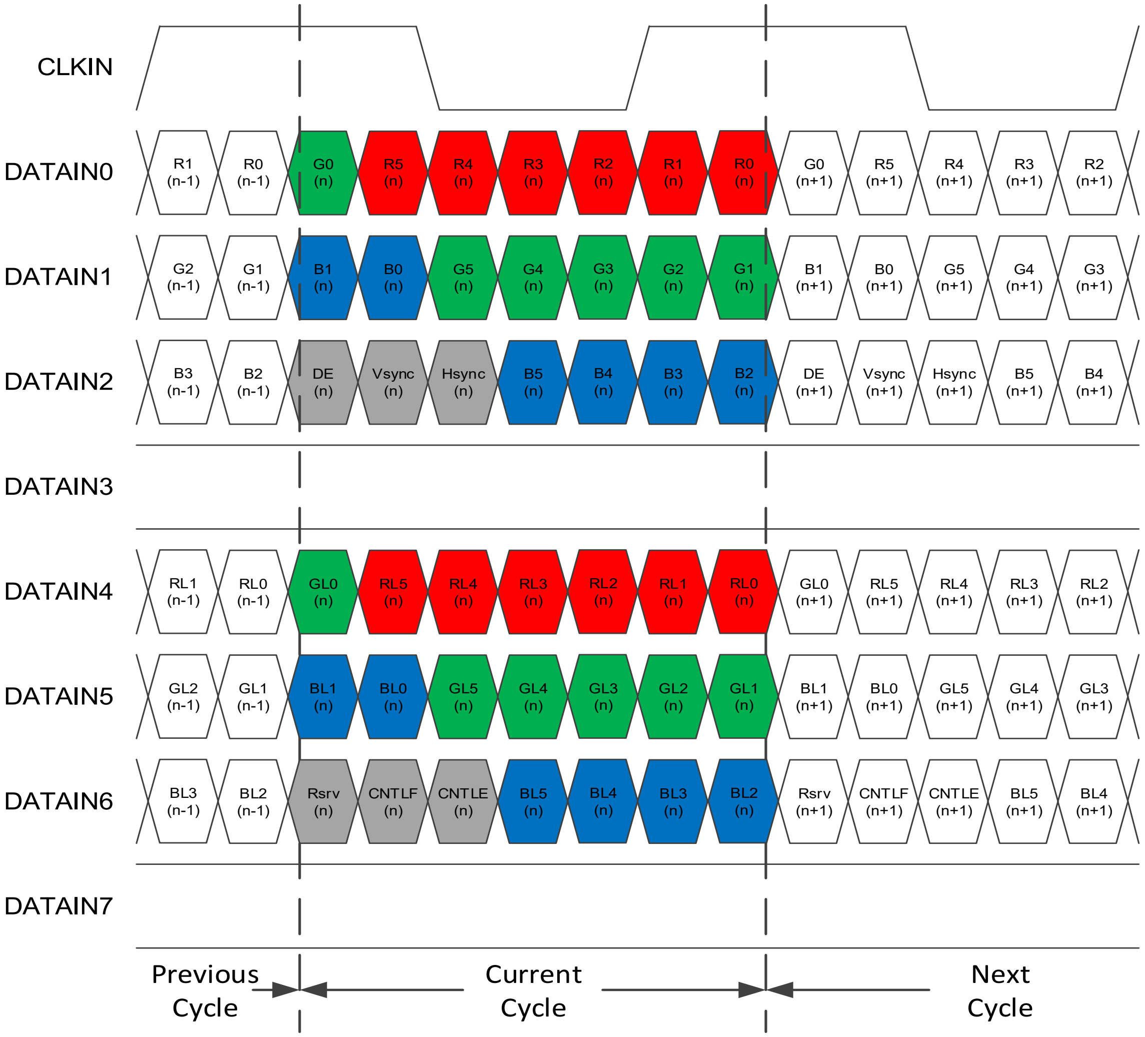


Figure 2.11. Dual Channel OpenLDI/FPD-Link/LVDS Input Bus Waveform for RGB666 Format (JEIDA/VESA)

2.4.2. Native Pixel Interface

From the processor, data and clock are transmitted serially to the LVDS 7:1/FPD-Link receiver. The FPGA device performs data processing such as converting the received LVDS serial data to pixel format.

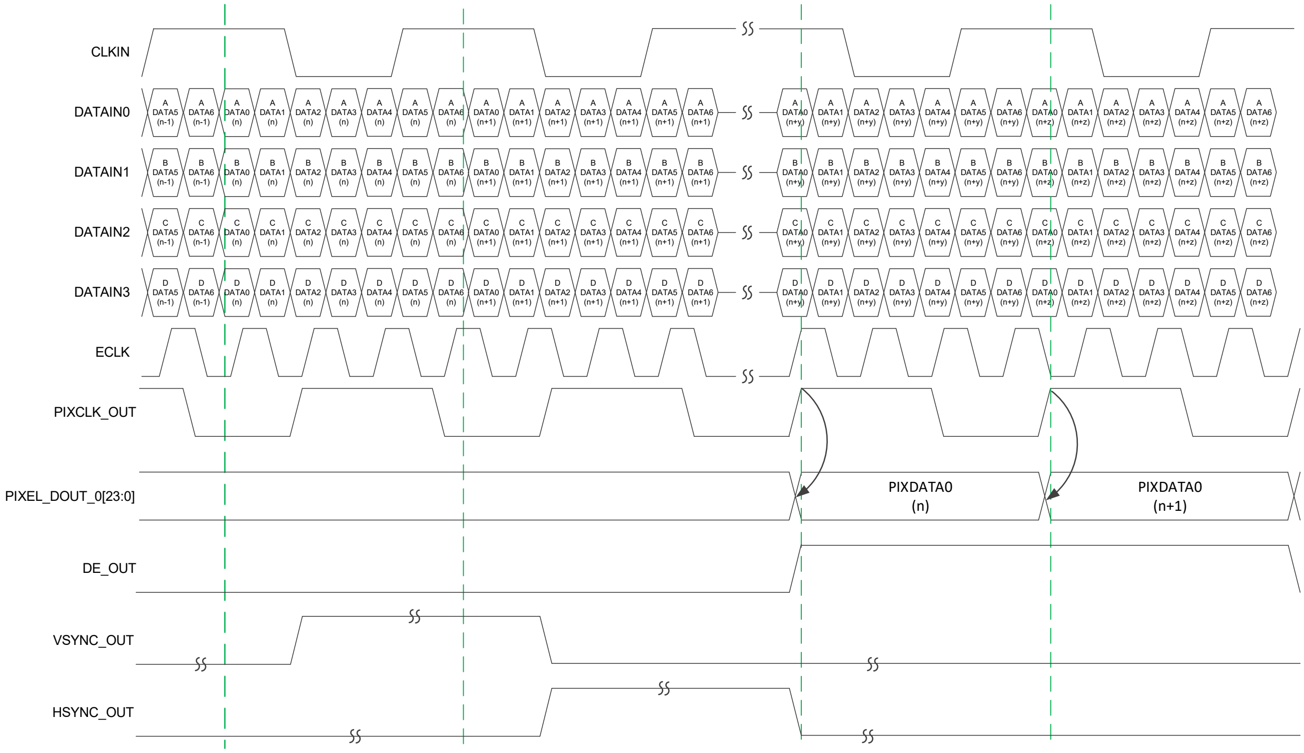


Figure 2.12. Input to Output Waveform for Single Channel OpenLDI/FPD-Link/LVDS

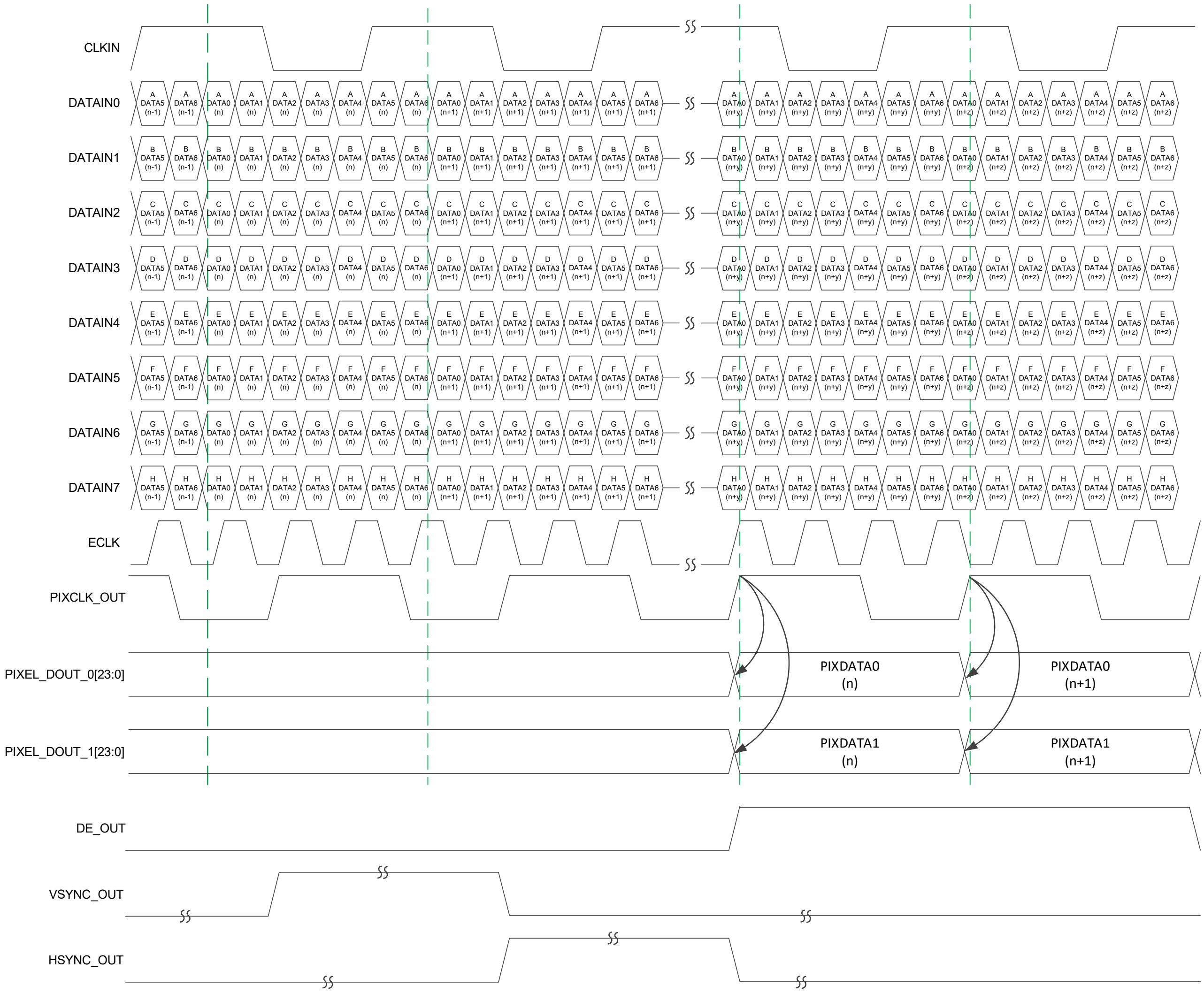


Figure 2.13. Input to Output Waveform for Dual Channel OpenLDI/FPD-Link/LVDS

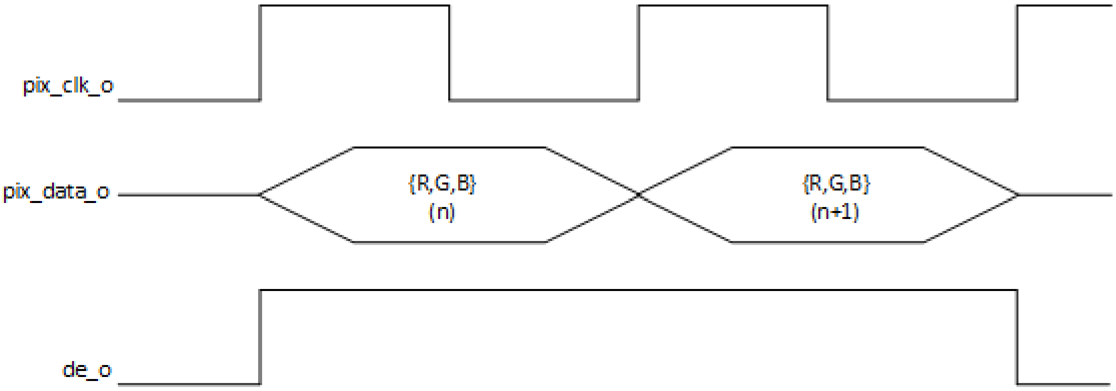


Figure 2.14. Output Pixel Data RGB Arrangement

Table 2.2. Output Pixel Data Summary

Number of FPD-Link Channels	Gear	No. of Output Pixel Data	Pixel Clock
1	7	1	pix_clk_o
2	7	2	pix_clk_o

Figure 2.15 shows the general arrangement and pixel data mapping based on device configuration.

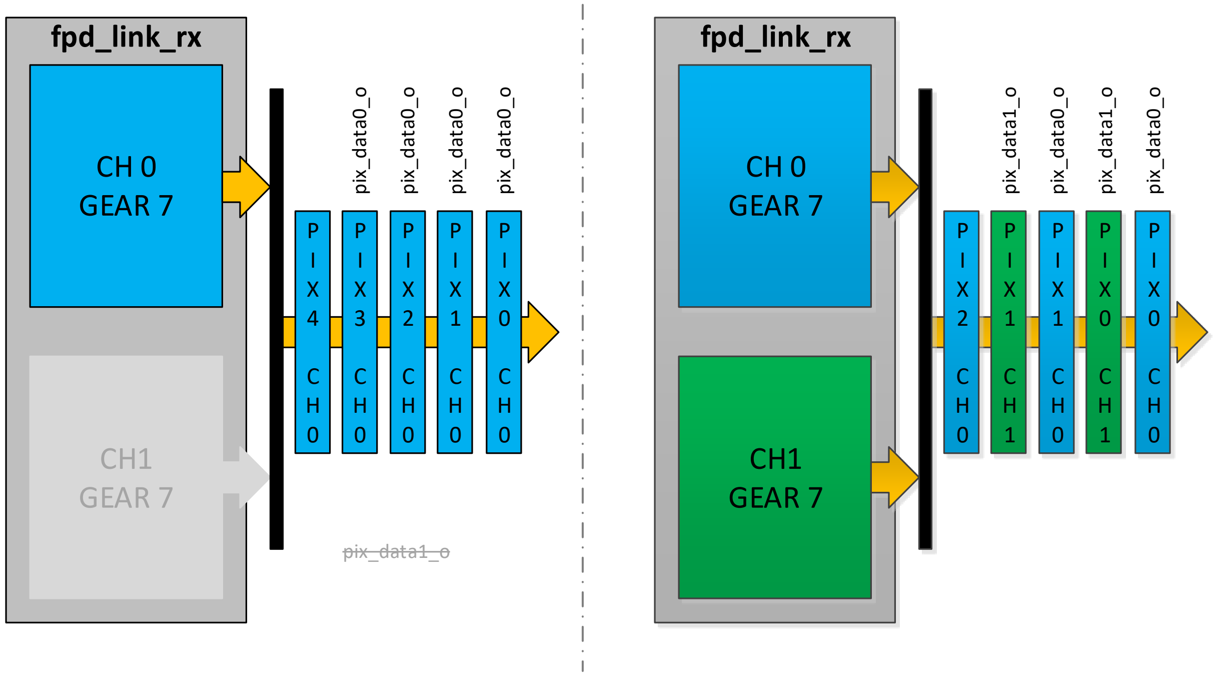


Figure 2.15. Output Pixel Data Arrangement for Single and Dual Channel OpenLDI/FPD-Link/LVDS

2.4.3. AXI4-Lite Interface

When the AXI4-Lite Interface is enabled, registers in the CSR can be configured and accessed by the user. Not all registers are configurable, and access varies depending on the register. It is recommended that registers be accessed after Rx clock synchronization and data training are done (bw_rdy_o = 1 or pixel_clk_o = pixel clock frequency) to obtain accurate values from the registers. Refer to the Register Description section for more information on the registers.

2.4.4. Unified Video Streaming Tx Interface

When the Unified Video Streaming Tx interface is enabled, pixel data is being transmitted through this interface. axis_vid_tvalid_o and axis_vid_tdata_o signals are mapped from de and pixel_data of the Native Pixel interface. Transmission of pixel data through the Unified Video Streaming Tx interface will only proceed when the lock signal from the pixel_monitor is high. Transfer of pixel data occurs and is considered a valid transaction when both axis_vid_tvalid_o and axis_vid_tready_i are asserted. FIFO is implemented in this interface to support back pressure.

Table 2.3. Data Mapping of Unified Video Streaming Tx Interface Ports

Unified Video Streaming Protocol Interface	Equivalent Signal
axis_vid_tvalid_o	de_o
axis_vid_tdata_o	pixel_data0/1_o
axis_vid_tuser_o[0]	start_of_frame
axis_vid_tuser_o[1]	Reserved
axis_vid_tlast_o	end_of_line

Figure 2.16 shows the timing diagram for the Unified Video Streaming Tx interface. This interface only starts transmitting data when the lock signal (from the pixel monitor) is high. Refer to the Pixel Monitor Module section for additional details of the lock signal.

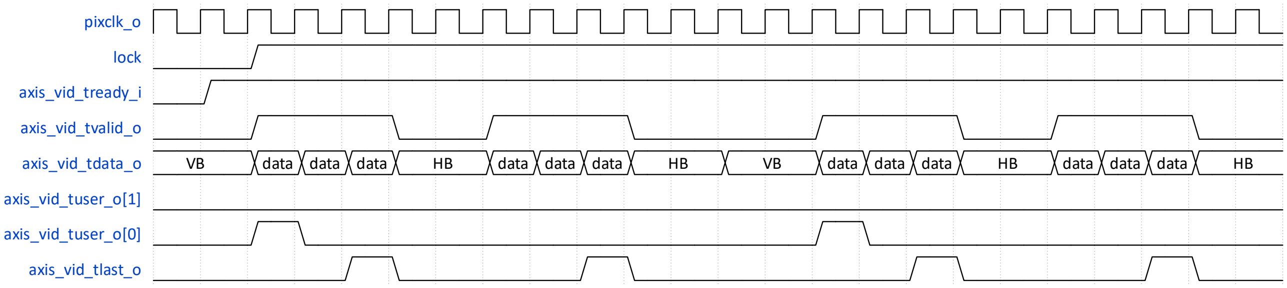


Figure 2.16. Unified Video Streaming Tx Timing Diagram

Figure 2.17 shows the behavior of axis_vid_tdata_o and axis_vid_tuser_o[0] when axis_vid_tready_i is not asserted.

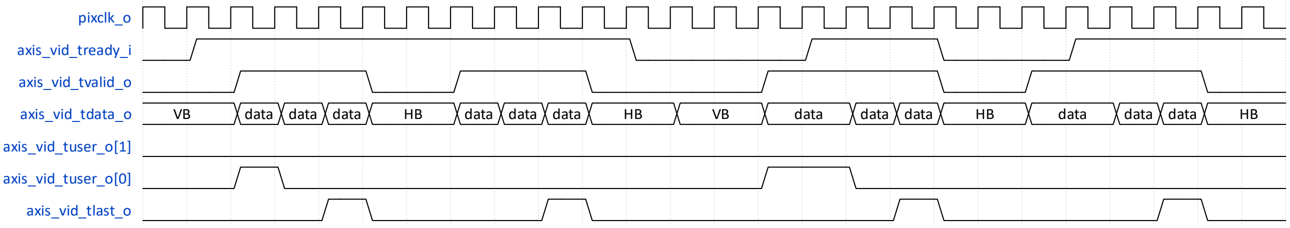


Figure 2.17. Unified Video Streaming Tx Timing Diagram with De-assertion of TREADY

The size of axis_vid_tdata_o varies depending on tdata width per pixel (TD_WD) and pixels per clock (PPC). PPC can be 1 or 2. TD_WD is dependent on the following attributes:

- CPP – Colors per pixel. Three colors per pixel are supported namely red, blue, and green.
- BPC – Bits per color. Minimum width of each color component is 1 byte (8 bits). RGB888 and RGB666 use 8 bits and 6 bits per color, respectively. These bits are known as active bits.
- BPP – Bits per pixel. Calculated as BPP = CPP × BPC.

TD_WD = ceil(BPP/8) × 8 and TDATA = TD_WD × PPC.

When the active bits per color is less than 8 bits, the additional bits or least significant bits (LSBs) of each color are padded with 0s. For RGB666, with six active bits per color, the two LSBs of each color component are padded with 0s. Figure 2.18 and Figure 2.19 show sample TDATA mapping for both the RGB888 and RGB666 formats.

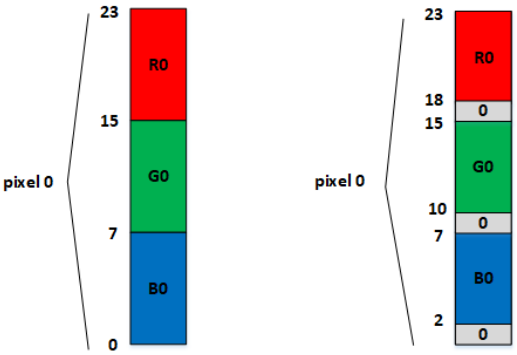


Figure 2.18. Unified Video Streaming Tx TDATA Mapping of RGB888 and RGB666 for PPC = 1

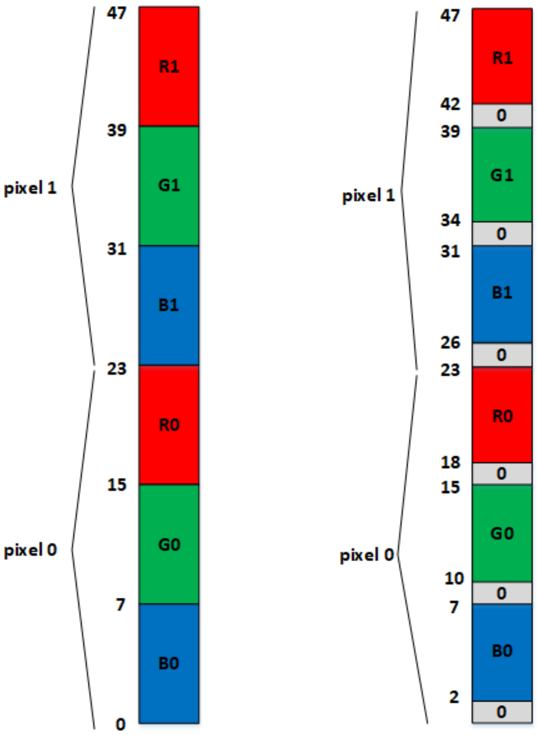


Figure 2.19. Unified Video Streaming Tx TDATA Mapping of RGB888 and RGB666 for PPC = 2

2.5. Other IP Specific Blocks/Layers/Interfaces

The fpd_link_rx.v module instantiates the fpd_link_rx_core, lvds71_pxlmap, pixel_monitor, test_mode, and synchronizer modules. A general purpose PLL (GPLL) is automatically instantiated inside the IP for Nexus devices. The fpd_link_rx_core module is the core module which performs the data training and serial to parallel conversion. GPLL is used for fast clock generation. The test_mode module is used for internal self-checking. The lvds71_pxlmap module is used to decode the output parallel data of fpd_link_rx_core and convert the data into the pixel format. It can also perform dynamic reconfiguration. The synchronizer module is a two-level synchronizer used to sync the system reset into different clock domains before it is used in the system.

2.5.1. FPD-Link Rx Core Module

The fpd_link_rx_core module instantiates GDDR_SYNC, BW_ALIGN, several Lattice primitives, and the lvds71_ddr_group.v module. The GDDR_SYNC module is required to initialize and synchronize DDR clock and tolerate the large skew between stop and reset of the DDR components. To properly sample the received data, BW_ALIGN is used to do data training that includes bit and word alignment with respect to the LVDS input clock.

LVDS data are fed to the I/O logic IDDR71 register. The LVDS clock is also fed to the I/O logic IDDR71 register, as well as to the PLL. The PLL is used to generate the sampling clock (ECLK) which is 3.5 times faster than the LVDS clock. To allow the placing of the edge of the clock in the middle of the data valid window, an alignment IP, together with the PLL, is used to shift the ECLK typically by 90 degrees. CLKDIV is used to generate a slower clock (SCLK), which is 3.5 times slower than ECLK. SCLK is used as the output clock of the IDDR71 IP.

After ECLK and SCLK are generated, the LVDS 7:1 soft IP performs bit and word alignment. Bit alignment places ECLK in the middle of the data training window, and word alignment obtains the correct training sequence after bit alignment. The training pattern is 7’b1100011 and alignment is done with respect to the LVDS clock. It is expected that the LVDS clock and data are edge-aligned with each other. When alignment is done, the LVDS data lanes are aligned with respect to the LVDS clock. Dual LVDS 7:1 Rx can only be supported if the two channels share the same clock.

2.5.2. GPLL

GPLL is a general purpose PLL. In the FPD-Link Rx interface, a PLL is required to generate the ECLK and start the initialization, synchronization, and data alignment processes.

For Nexus devices, GPLL is instantiated inside the IP and clock-related operations are automated. For Avant devices, no GPLL is instantiated inside the IP so the user needs to provide the required clocks.

The following is a general guide for setting the GPLL:

1. Set reference clock frequency equal to LVDS input clock frequency (use clk_ch0_p_i signal as the reference clock).
2. Use internal feedback path if possible.
3. Set primary clock (CLKOP) frequency equal to the reference clock.
4. Set secondary clock (CLKOS) frequency to 3.5 times the reference clock. This is equivalent to Rx LVDS ECLK in the Clocking Overview section.
5. Enable Dynamic Phase ports. The IP provides control signals for dynamic phase shift to properly align the GPLL output clocks. Refer to the PLL Interface section for details.
6. Set phase select to CLKOS.
7. Set Phase Load Reg to SET when applicable.
8. Enable PLL lock signal and reset.

Note: Refer to the Lattice PLL module specifications for the phasesel_i value.

Figure 2.20 shows the connection of the external GPLL to the IP for Avant devices.

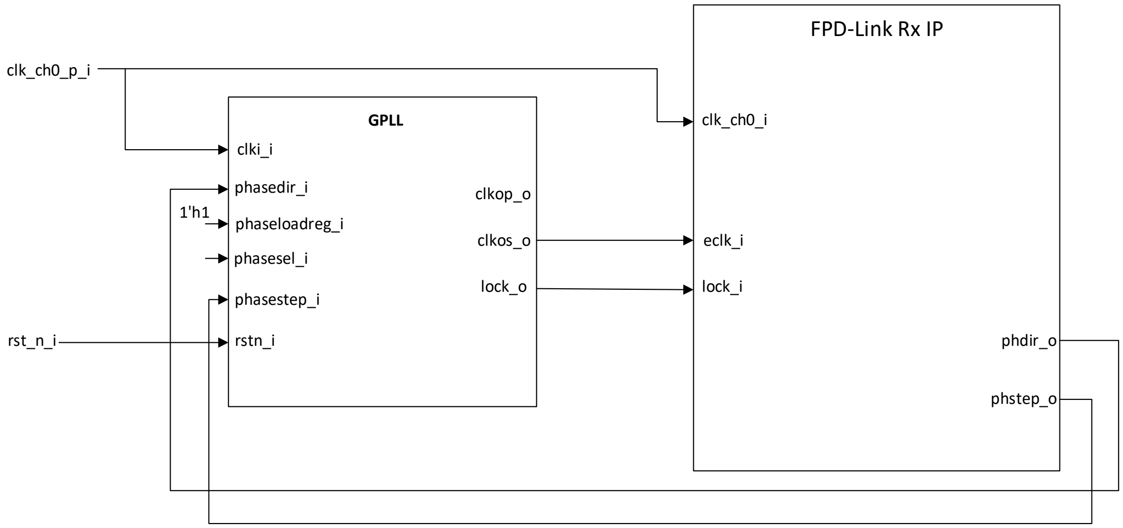


Figure 2.20. GPLL connection for Avant Devices

2.5.3. LVDS71 Pixel Map Module

The lvds71_pxlmap module is used to decode the output parallel data of fpd_link_rx_core and convert the data into pixel format. Up to two valid output pixel data is supported depending on the design configuration.

2.5.4. Pixel Monitor Module

The pixel_monitor module continuously monitors the active height and active width of an incoming video stream. This module is used only when the Unified Video Streaming Tx Interface is enabled. The module generates the SOF (start of frame), EOL (end of line), and lock signals to be used in the Unified Video Streaming Tx interface.

The lock signal is high only when at least two consecutive frames with consistent frames are detected. When two consecutive frames have different active width and height, the lock signal is low. The lock signal remains low until two consecutive frames with consistent frames are detected once again.

2.5.5. Test Mode Rx Module

The test_mode module is used to check data from the FPD-link Rx before the data is decoded into control and pixel data. A predefined data set (specified through *Test Mode Expected Data in Hex Format* and driven as LVDS input data) is compared internally to the actual output of the fpd_link_rx module.

Data comparison starts only after bit and word alignment is completed. If data mismatch is encountered, test_mode_err_o is set to high until reset is asserted or the device is powered down.

Refer to the Debug Mode section for details on how to enable test mode.

2.5.6. Synchronizer Module

The synchronizer module is a two-level synchronizer used to sync the input data into different clock domains. In the design, this is used to synchronize the system reset into different clock domains before it is used in the system.

2.6. Dynamic Reconfiguration

This IP supports dynamic reconfiguration for bits per color. This process can be done by programming the CFG_REG through the AXI4-Lite Interface. Writing 1 in the register CFG_REG[0] will switch decoding of the output parallel data of fpd_link_rx_core from the RGB888 format (JEIDA/VESA) to the RGB666 format.

Writing 1 in the register CFG_REG[0] can be performed at any time but the change takes effect only on the next stable frame for both the Native Pixel and Unified Video Streaming Tx Interfaces. The new output pixel data is similar with the TDATA mapping in the Unified Video Streaming Tx Interface section.

Dynamic reconfiguration of data type in either direction will only be supported if the initial *Data Type* is set to RGB888.

3. IP Parameter Description

The configurable attributes of the FPD-Link Receiver IP are shown in the following tables. 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
Receiver Interface		
Number of RX Channels	**1**, 2	Specifies how many LVDS links are used
RX Interface	**LVDS**	Specifies the I/O interface. This attribute is not configurable.
Data Mapping Format	Format 1 (JEIDA), **Format 2 (VESA)**	Specifies the data mapping format of the input.
Number of RX Lanes	3, **4**	Specifies the number of data lanes per LVDS Rx link. This attribute is not configurable. 3 *Data Type* = RGB666 4 *Data Type* = RGB888
RX Gear	**7**	Specifies the DDR71 gearing used. This attribute is not configurable.
Pixel Interface		
Data Type	**RGB888**, RGB666	Specifies the data type of the output pixel.
Number of Output Pixels per Clock	**1**, 2	Specifies the number of output pixels per clock. *Number of Output Pixels per Clock = Number of RX Channels*
Video Data Interface	**Native**, Unified Video Streaming Tx	Specifies the interface to be used in the pixel domain.
Clock		
RX Total Aggregate Bandwidth (Mbps)	378–7560[1], **3780** 328–8400[2], **3780**	Rx total line rate Calculated based on the formula: *Rx Line Rate per Lane × Number of Rx Lanes × Number of Rx Channels* This attribute is not configurable.
RX Line Rate per lane (Mbps)	126–**945**[1] 109.375–1050[2], **945**	Target line rate per lane
Pixel Clock Frequency (MHz)	18–**135**[1] 15.625–150[2], **135**	Pixel clock Calculated based on the formula: *Rx Line Rate per Lane / Rx Gear* This attribute is not configurable.
LVDS Input Clock Frequency (MHz)	18–**135**[1] 15.625–150[2], **135**	LVDS clock Calculated based on the formula: *Pixel Clock Frequency × (Rx Gear/7)* This attribute is not configurable.
LVDS ECLK Frequency (MHz)	63–**472.5**[1] 46.875–525[2], **472.5**	LVDS edge clock Calculated based on the formula: *LVDS Input Clock Frequency × (Rx Gear/2)* This attribute is not configurable.

Attribute	Selectable Values	Description
Miscellaneous		
Register Interface	**OFF**, AXI4-Lite	Specifies the register interface used.
Enable Miscellaneous Signals	**Checked**, Unchecked	Enables miscellaneous signals. Configurable if *Register Interface* is set to *OFF.*
Enable Test Mode	Checked, **Unchecked**	Enable test mode. When enabled, Rx internally compares the actual output of the IP with the programmed *Test Mode Expected Data in Hex Format*. Configurable if *Register Interface* is set to *OFF.*
Test Mode Expected Data in Hex Format	21’h000000–28’hFFFFFFF, **28’h0000000**	Test data value used when *Enable Test Mode* is checked. Configurable if *Register Interface* is set to *OFF.* Calculated based on the formula: *Number of Rx Lanes* × 7

Notes:

1. Supported range for Nexus devices.
2. Supported range for Avant devices.

4. Signal Description

This section describes the FPD-Link Receiver IP ports.

4.1. FPD-Link RX Interface

Table 4.1. FPD-Link Rx Interface Ports

Port	Type	Width	Description
Clock and Reset			
rst_n_i	Input	1	Asynchronous active low system reset.
axil_rst_n_i[1]	Input	1	Asynchronous active low reset for AXI4-Lite Interface.
clk_ch0_p_i	Input	1	Positive LVDS input clock to FPD-Link Rx channel 0. Supported range is 10 to 135 MHz for Nexus devices and 15.625 to 150 MHz for Avant devices.
clk_ch1_p_i	Input	1	Positive LVDS input clock to FPD-Link Rx channel 1. Supported range is 10 to 135 MHz for Nexus devices and 15.625 to 150 MHz for Avant devices.
eclk_i[2]	Input	1	Input edge/fast clock from PLL that is 3.5 times faster than the LVDS clock. Supported range is 46.875 to 525 MHz for Avant devices.
axil_clk_i[1]	Input	1	AXI4-Lite input clock.
pix_clk_o	Output	1	Output pixel clock.
Native Pixel Interface			
pix_data0_o	Output	*Number of RX Lanes* × 6	Output pixel data 0. Bus width depends on the data type selected.
pix_data1_o[3,4]	Output	*Number of RX Lanes* × 6	Output pixel data 1. Bus width depends on the data type selected.
de_o	Output	1	Output data enable for parallel interface.
hsync_o	Output	1	Output horizontal sync for parallel interface.
vsync_o	Output	1	Output vertical sync for parallel interface.
OpenLDI/FPD-Link Interface			
d0_ch0_p_i	Input	1	Positive LVDS input data lane 0 to FPD-Link Rx channel 0.
d1_ch0_p_i	Input	1	Positive LVDS input data lane 1 to FPD-Link Rx channel 0.
d2_ch0_p_i	Input	1	Positive LVDS input data lane 2 to FPD-Link Rx channel 0.
d3_ch0_p_i[5]	Input	1	Positive LVDS input data lane 3 to FPD-Link Rx channel 0.
d0_ch1_p_i[6]	Input	1	Positive LVDS input data lane 0 to FPD-Link Rx channel 1.
d1_ch1_p_i[6]	Input	1	Positive LVDS input data lane 1 to FPD-Link Rx channel 1.
d2_ch1_p_i[6]	Input	1	Positive LVDS input data lane 2 to FPD-Link Rx channel 1.
d3_ch1_p_i[5,6]	Input	1	Positive LVDS input data lane 3 to FPD-Link Rx channel 1.

Notes:

1. Available only if *Register Interface* = *AXI4-Lite*.
2. Available only for Avant devices.
3. Available only when *Number of Output Pixels per Clock* is more than one.
4. This pixel data is received from channel 1 when dual channel is selected.
5. Available only for *Data Type* = *RGB888*.
6. LVDS channel 1 input ports are not available when single LVDS channel is selected.

4.2. Miscellaneous Status Interface

Table 4.2. Miscellaneous Status Interface Ports

Port	Type	Width	Description
Miscellaneous			
tstmode_en_i[1]	Input	1	Enable or disable test mode. 0 – Disable test mode 1 – Enable test mode
tstmode_err_o[1]	Output	1	0 – No error is encountered 1 – Indicates that compare mismatch is encountered when test mode is enabled Automatically set to 0 when test mode is disabled.
pll_lock_o[2,3]	Output	1	General PLL lock signal.
gddr_rdy_o[2]	Output	1	0 – DDR synchronization is not yet started or still in progress 1 – Indicates that DDR synchronization is already done
bit_lock_o[2]	Output	1	0 – Bit alignment is not yet started or still in progress 1 – Indicates that bit alignment is already done
word_lock_o[2]	Output	1	0 – Word alignment is not yet started or still in progress 1 – Indicates that word alignment is already done
bw_rdy_o[2]	Output	1	0 – Data training is not yet started or still in progress 1 – Indicates that data training is already done

Notes:

1. Available only if both *Register Interface* = *OFF* and *Enable Test Mode* is checked. Refer to the Debug Mode section for details.
2. Available only if both *Register Interface* == *OFF* and *Enable Miscellaneous Signals* is checked.
3. Available only for Nexus devices.

4.3. PLL Interface

Table 4.3. PLL Interface Ports

Port	Type	Width	Description
PLL Interface[1]			
lock_i	Input	1	PLL lock signal. Once asserted, DDR synchronization process starts.
phdir_o	Output	1	Dynamic phase direction signal. Controls the phase direction of the PLL during the bit and word alignment process.
phstep_o	Output	1	Dynamic phase step signal. Controls the phase step of the external PLL during the bit and word alignment process.

Note:

1. Available only for Avant devices.

4.4. AXI4-Lite Interface

Table 4.4. AXI4-Lite Interface Ports

Port	Type	Width	Description
AXI4-Lite Interface[1]			
s_axil_awvalid_i	Input	1	AXI4-Lite write valid address signal.
s_axil_awaddr_i	Input	32	AXI4-Lite write address signal.
s_axil_wvalid_i	Input	1	AXI4-Lite write valid signal.
s_axil_wdata_i	Input	32	AXI4-Lite write data signal.
s_axil_wstrb_i	Input	4	AXI4-Lite write strobe signal.
s_axil_bready_i	Input	1	AXI4-Lite write response ready signal. Indicates that transfer on the write response channel is accepted.
s_axil_arvalid_i	Input	1	AXI4-Lite read valid address signal.
s_axil_araddr_i	Input	32	AXI4-Lite read address signal.
s_axil_rready_i	Input	1	AXI4-Lite read ready signal. Indicates that transfer on the read data channel is accepted.
s_axil_awready_o	Output	1	AXI4-Lite write address ready output signal. Indicates that the write valid address signal is asserted, input write address is valid, and transfer is accepted on write address channel.
s_axil_wready_o	Output	1	AXI4-Lite write ready output signal. Indicates that the data written in write data signal is valid.
s_axil_bvalid_o	Output	1	AXI4-Lite write response valid signal.
s_axil_bresp_o	Output	2	AXI4-Lite write response signal. Indicates the status of a write transaction.
s_axil_arready_o	Output	1	AXI4-Lite read address ready signal.
s_axil_rvalid_o	Output	1	AXI4-Lite read valid signal.
s_axil_rdata_o	Output	32	AXI4-Lite read data signal.
s_axil_rresp_o	Output	2	AXI4-Lite read response signal.

Note:

1. Available only if *Register Interface* = *AXI4-Lite*.

4.5. Unified Video Streaming Tx Interface

Table 4.5. Unified Video Streaming Tx Interface Ports

Port	Type	Width	Description
Unified Video Streaming Tx Interface[1]			
axis_vid_tready_i	Input	1	Unified Video Streaming Tx interface ready input signal.
axis_vid_tvalid_o	Output	1	Unified Video Streaming Tx interface valid data signal mapped from de_o of the Native Pixel interface.
axis_vid_tdata_o	Output	*TD_WD × PPC*[2]	Unified Video Streaming Tx interface data mapped from pix_data0_o or pix_data1_o of the Native Pixel interface.
axis_vid_tuser_o	Output	2	Unified Video Streaming Tx interface user-defined additional data information output. [0]: Start of Frame 0 – Not the start of new frame 1 – Start of new frame [1]: Unused
axis_vid_tlast_o	Output	1	Unified Video Streaming Protocol last pixel of the line output signal.

Notes:

1. Available only if *Video Data Interface* = *Unified Video Streaming Tx*.
2. TD_WD = tdata width per pixel; PPC = pixel per clock. Refer to the Unified Video Streaming Tx Interface section for more information.

5. Register Description

The FPD-Link Receiver IP supports register programming and status reading. All registers listed are accessible through the AXI4-Lite interface.

5.1. Register Address Map

Table 5.1. Configuration Register Address Map

Address Offset	Name	Description	Access	Default
0x0000	CFG_REG	Configuration register	RW	32’h0
0x0004	TEST_MODE	Test mode register	RW	32’h0
0x000C	MISC_SIGNALS	Miscellaneous signals register	RO	32’h0
0x0200	ACT_WIDTH	Active width register	RO	32’h0
0x0204	ACT_HEIGHT	Active height register	RO	32’h0
0x0280	LOCKED	Locked status register	RO	32’h0

5.2. FPD-Link Rx Configuration Registers

Table 5.2. CFG_REG Register

Field	Name	Description	Access	Default
[31:9]	rsvd	Reserved	RO	23’h0
[8]	data_frmt	Data mapping format register bit. 0 – VESA 1 – JEIDA	RO	Depends on the *Data Mapping Format* attribute. 1’b0 – VESA 1’b1 – JEIDA
[7:1]	rsvd	Reserved	RO	7’h0
[0]	act_bpc	Dynamic reconfiguration bit for bits per color (BPC).[1] 0 – RGB888 1 – RGB666	RW	Depends on the *Data Type* attribute. 1’b0 – RGB888 1’b1 – RGB666

Note:

1. Dynamic reconfiguration is only supported when *Data Type* is set to *RGB888*.

Table 5.3. TEST_MODE Register

Field	Name	Description	Access	Default
[31:29]	rsvd	Reserved	RO	3’h0
[28]	testmode_en	Enable or disable test mode. 0 – Disable test mode 1 – Enable test mode	RW	Depends on the *Enable Test Mode* attribute.
[27:0]	testmode_data	Pre-defined data to be used in checking the data before being decoded into pixel data. This pre-defined data should be equal to the actual input. 28 bits and 21 bits ([20:0]) are used for RGB888 and RGB666, respectively.	RW	Depends on the *Test Mode Expected Data in Hex Format* attribute.

Table 5.4. MISC_SIGNALS Register

Field	Name	Description	Access	Default
[31:6]	rsvd	Reserved	RO	25’h0
[5]	testmode_err	Test mode status bit. 0 – No mismatch is encountered. 1 – Compare data mismatch when test mode is enabled. When test mode is disabled, this bit is always set to 0.	RO	1’h0
[4]	bw_rdy_stat	1-bit data training status. 0 – Data training is still in progress or not yet started. 1 – Data training is done.	RO	1’h0
[3]	word_lock_stat	1-bit word alignment status. 0 – Word alignment is still in progress or not yet started. 1 – Word alignment is done.	RO	1’h0
[2]	bit_lock_stat	1-bit bit alignment status. 0 – Bit alignment is still in progress or not yet started. 1 – Bit alignment is done.	RO	1’h0
[1]	gddr_rdy_stat	1-bit DDR synchronization status. 0 – DDR synchronization is still in progress or not yet started. 1 – DDR synchronization is done.	RO	1’h0
[0]	pll_lock_stat	1-bit PLL lock status. 0 – PLL lock status not achieved. 1 – PLL lock status achieved. This bit is used only for Nexus devices.	RO	1’h0

Table 5.5. ACT_WIDTH Register

Field	Name	Description	Access	Default
[31:16]	rsvd	Reserved	RO	16’h0
[15:0]	act_width	Active width in units of pixels. Active width of the video stream data extracted from pixel monitor.	RO	16’h0

Table 5.6. ACT_HEIGHT Register

Field	Name	Description	Access	Default
[31:16]	rsvd	Reserved	RO	16’h0
[15:0]	act_height	Active height in units of lines. Active height status of the video stream data extracted from pixel monitor.	RO	16’h0

Table 5.7. LOCKED Register

Field	Name	Description	Access	Default
[31:1]	rsvd	Reserved	RO	31’h0
[0]	locked_stat	1-bit locked status of the pixel monitor. 0 – Input video stream data has inconsistent frames[1]. 1 – Input video stream data has consistent frames[2].	RO	1’h0

Notes:

1. Video stream data is inconsistent if two consecutive frames have different active height and width.
2. Video stream data is consistent if at least two consecutive frames have the same active height and width.

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.

Note: The screenshots provided are for reference only. Details may vary depending on the version of the IP or software being used. If there have been no significant changes to the GUI, a screenshot may reflect an earlier version of the IP.

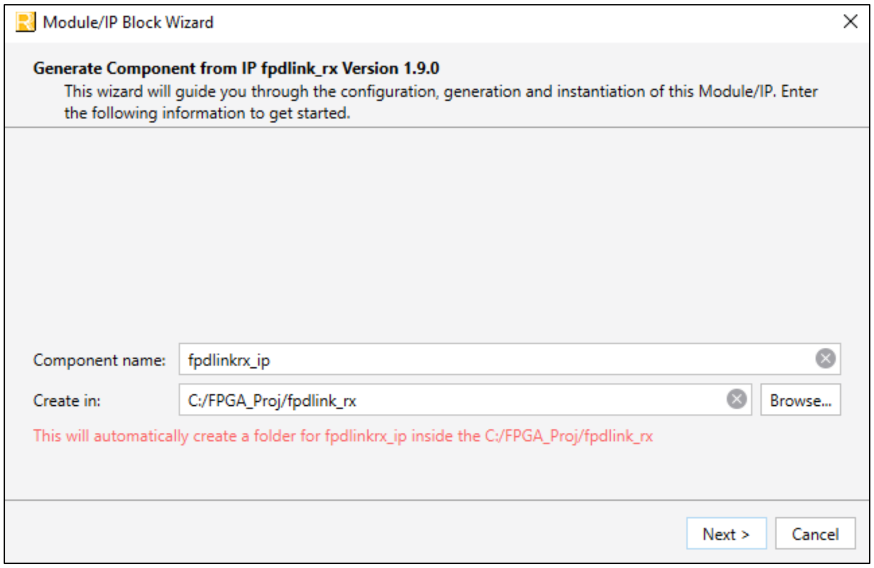
6.1. Generating and Instantiating the IP

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

To generate the FPD-Link Receiver IP:

1. Create a new Lattice Radiant software project or open an existing project.
2. Click the **IP Catalog** button to view the **IP Catalog** pane.
3. On the **IP on Local** tab, double-click **FPD-Link Receiver** under the **IP/…/Audio_Video_Image_Processing** category. The **Module/IP Block Wizard** opens.

   Note: If the IP is not available on the **IP on Local** tab, download the IP from the **IP on Server** tab.



Module/IP Block Wizard

Generate Component from IP fpdlink_rx Version 1.9.0

This wizard will guide you through the configuration, generation and instantiation of this Module/IP. Enter the following information to get started.

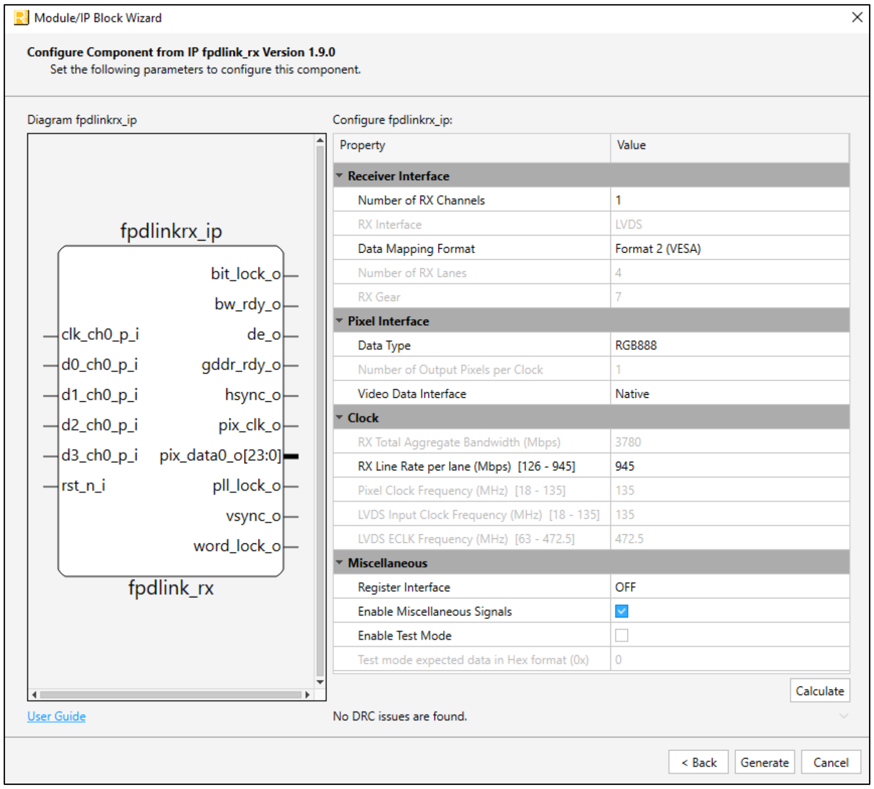
Component name: fpdlinkrx_ip

Create in: C:/FPGA_Proj/fpdlink_rx

This will automatically create a folder for fpdlinkrx_ip inside the C:/FPGA_Proj/fpdlink_rx

Figure 6.1. Module/IP Block Wizard

4. Enter values in the **Component name** and **Create in** fields, then click **Next**.
5. Customize the selected FPD-Link Receiver IP using drop-down lists and check boxes. Figure 6.2 shows an example configuration of the FPD-Link Receiver IP. For details on the configuration options, refer to the IP Parameter Description section.



Module/IP Block Wizard

Configure Component from IP fpdlink_rx Version 1.9.0

Set the following parameters to configure this component.

Diagram fpdlinkrx_ip

Configure fpdlinkrx_ip:

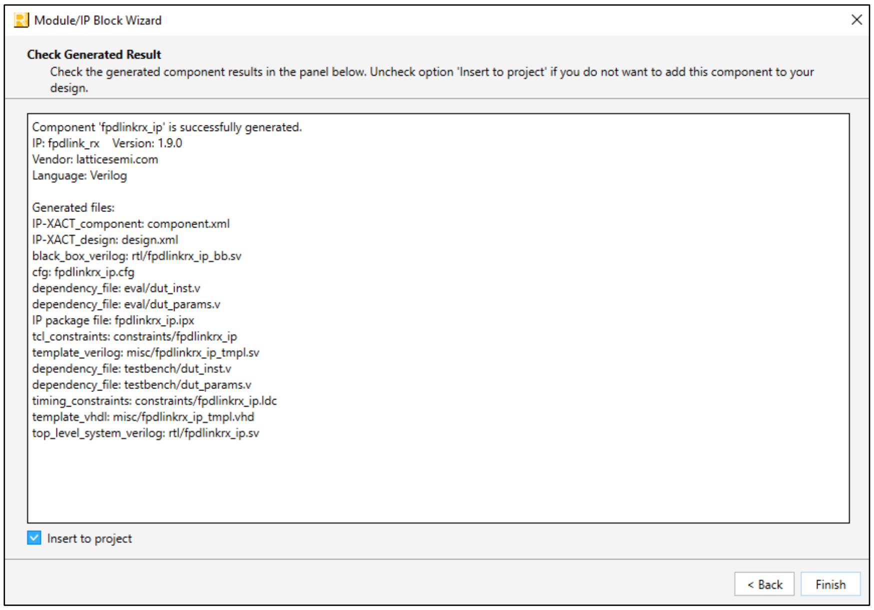
Property	Value
Receiver Interface	
Number of RX Channels	1
RX Interface	LVDS
Data Mapping Format	Format 2 (VESA)
Number of RX Lanes	4
RX Gear	7
Pixel Interface	
Data Type	RGB888
Number of Output Pixels per Clock	1
Video Data Interface	Native
Clock	
RX Total Aggregate Bandwidth (Mbps)	3780
RX Line Rate per lane (Mbps) [126 - 945]	945
Pixel Clock Frequency (MHz) [18 - 135]	135
LVDS Input Clock Frequency (MHz) [18 - 135]	135
LVDS ECLK Frequency (MHz) [63 - 472.5]	472.5
Miscellaneous	
Register Interface	OFF
Enable Miscellaneous Signals	☑
Enable Test Mode	□
Test mode expected data in Hex format (0x)	0

Calculate

No DRC issues are found.

Figure 6.2. IP Configuration

6. Click **Generate**. The **Check Generated Result** window opens. This window shows design block messages and results.



Module/IP Block Wizard

Check Generated Result

Check the generated component results in the panel below. Uncheck option 'Insert to project' if you do not want to add this component to your design.

```
Component 'fpdlinkrx_ip' is successfully generated.
IP: fpdlink_rx    Version: 1.9.0
Vendor: latticesemi.com
Language: Verilog

Generated files:
IP-XACT_component: component.xml
IP-XACT_design: design.xml
black_box_verilog: rtl/fpdlinkrx_ip_bb.sv
cfg: fpdlinkrx_ip.cfg
dependency_file: eval/dut_inst.v
dependency_file: eval/dut_params.v
IP package file: fpdlinkrx_ip.ipx
tcl_constraints: constraints/fpdlinkrx_ip
template_verilog: misc/fpdlinkrx_ip_tmpl.sv
dependency_file: testbench/dut_inst.v
dependency_file: testbench/dut_params.v
timing_constraints: constraints/fpdlinkrx_ip.ldc
template_vhdl: misc/fpdlinkrx_ip_tmpl.vhd
top_level_system_verilog: rtl/fpdlinkrx_ip.sv
```

☑ Insert to project

Figure 6.3. Check Generated Result

7. Click **Finish**. All generated files are placed in the directory specified by the **Component name** and **Create in** fields shown in Figure 6.1.

6.1.1. Generated Files and File Structure

The generated FPD-Link Receiver IP Core 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 IP core is also provided. You may also use this example as the starting template for your top-level design. The generated files are listed in Table 6.1.

Table 6.1. Generated File List

Attribute	Description
<Component Name>.ipx	This file contains information on the files associated with the generated IP.
<Component Name>.cfg	This file contains the parameter values used in IP configuration.
component.xml	This file contains the ipxact:component information of the IP.
design.xml	This file documents the configuration parameters of the IP in IP-XACT 2014 format.
rtl/<Component Name>.sv	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.
eval/dut_inst.v testbench/dut_inst.v	These files provide the list of ports used in the IP configuration.
eval/dut_inst.v testbench/dut_params.v	These files provide the list of parameters used in the IP configuration.

6.2. Design Implementation

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

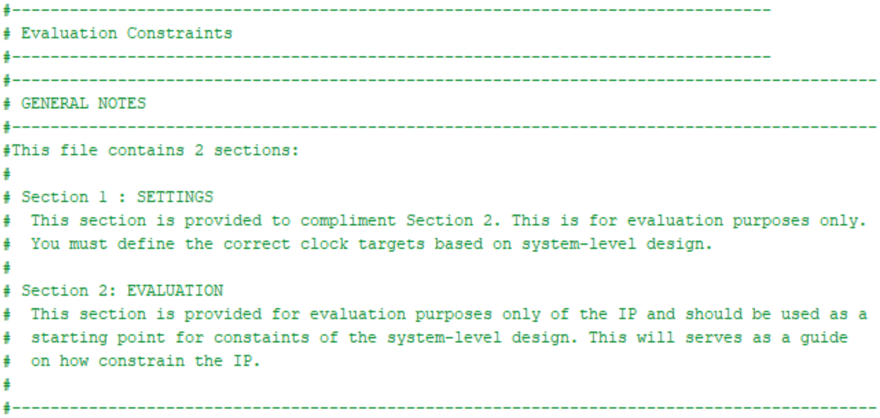
Post-Synthesis constraint files (.pdc) contain both timing and non-timing constraint.pdc source files for storing logical timing/physical constraints. Constraints that are added using the Device Constraint Editor are saved to the active .pdc file. The active post-synthesis design constraint file is then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

The FPD-Link Receiver IP generates the following constraint files:

- A legacy pre-synthesis constraint file in LDC format (*<ip_instance_path>/constraints/<instance_name>.ldc*) automatically used and propagated by the software tool.
- A constraint file in SDC format (*<ip_instance_path>/constraints/constraint.sdc*) that contains both pre-synthesis and post-synthesis IP constraints. These constraints are automatically used and propagated by the software tool starting from the Lattice Radiant software version 2024.1.
- An evaluation post-synthesis constraint file in PDC format (*<ip_instance_path>eval/constraint_eval.pdc*). In this constraint file, sections 1 and 2 are for evaluation purposes and can be used as a starting point for constraints. Clocks must be defined according to your design.



```
#-------------------------------------------------------------------------------
# Evaluation Constraints
#-------------------------------------------------------------------------------
#------------------------------------------------------------------------------------------
# GENERAL NOTES
#------------------------------------------------------------------------------------------
#This file contains 2 sections:
#
# Section 1 : SETTINGS
#  This section is provided to compliment Section 2. This is for evaluation purposes only.
#  You must define the correct clock targets based on system-level design.
#
# Section 2: EVALUATION
#  This section is provided for evaluation purposes only of the IP and should be used as a
#  starting point for constaints of the system-level design. This will serves as a guide
#  on how constrain the IP.
#
#------------------------------------------------------------------------------------------
```

Figure 6.4. General Evaluation PDC File Header

Note: During synthesis, you can ignore clock related warnings as the evaluation IP does not include clock-related constraints at pre-synthesis level.

Note: During post-synthesis, warnings related to dropped constraints may be shown. As the IP supports many configurations and parameter combinations, some default constraints may not be applicable to the selected configuration.

Refer to Lattice Radiant Timing Constraints Methodology (FPGA-AN-02059) for details on how to constrain your design.

6.4. Specifying the Strategy

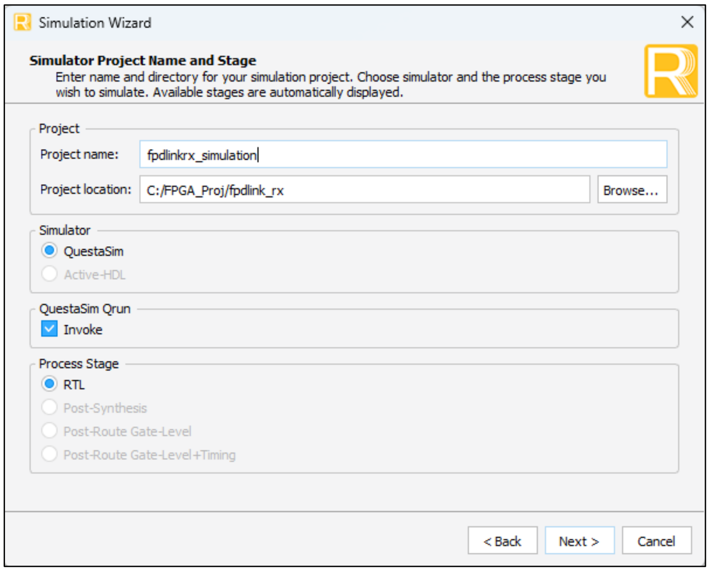
The Radiant software provides two predefined strategies: area and timing. It also enables you to create customized strategies. For details on how to create a new strategy, refer to the Strategies section of the Lattice Radiant Software User Guide.

6.5. Running Functional Simulation

An example simulation environment is provided after you generate the IP. You can find the files in *<ip_instance_path>/testbench/.* This example environment supports limited testing of features as the primary intent is to provide a starting point for checking the functionality of the IP. Official IP verification is done through the Universal Verification Methodology (UVM).

To run functional simulation:

1. Click the button located on the **Toolbar** to initiate the **Simulation Wizard** shown in Figure 6.5.



Simulation Wizard

Simulator Project Name and Stage
Enter name and directory for your simulation project. Choose simulator and the process stage you wish to simulate. Available stages are automatically displayed.

Project
Project name: fpdlinkrx_simulation
Project location: C:/FPGA_Proj/fpdlink_rx

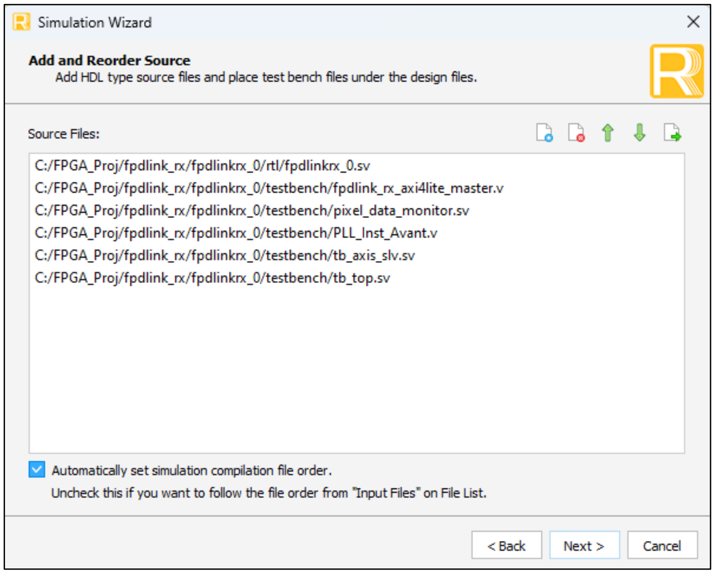
Simulator
● QuestaSim
○ Active-HDL

QuestaSim Qrun
☑ Invoke

Process Stage
● RTL
○ Post-Synthesis
○ Post-Route Gate-Level
○ Post-Route Gate-Level+Timing

Figure 6.5. Simulation Wizard

2. Click **Next** to open the **Add and Reorder Source** window as shown in Figure 6.6.



Simulation Wizard

Add and Reorder Source
Add HDL type source files and place test bench files under the design files.

Source Files:

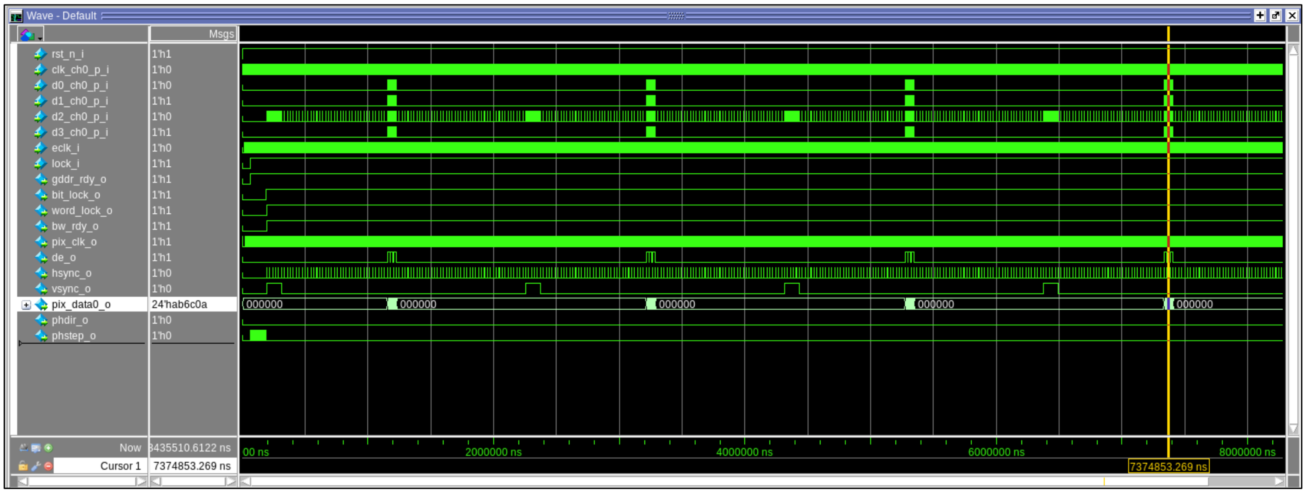
```
C:/FPGA_Proj/fpdlink_rx/fpdlinkrx_0/rtl/fpdlinkrx_0.sv
C:/FPGA_Proj/fpdlink_rx/fpdlinkrx_0/testbench/fpdlink_rx_axi4lite_master.v
C:/FPGA_Proj/fpdlink_rx/fpdlinkrx_0/testbench/pixel_data_monitor.sv
C:/FPGA_Proj/fpdlink_rx/fpdlinkrx_0/testbench/PLL_Inst_Avant.v
C:/FPGA_Proj/fpdlink_rx/fpdlinkrx_0/testbench/tb_axis_slv.sv
C:/FPGA_Proj/fpdlink_rx/fpdlinkrx_0/testbench/tb_top.sv
```

☑ Automatically set simulation compilation file order.
Uncheck this if you want to follow the file order from "Input Files" on File List.

Figure 6.6. Add and Reorder Source

3. Click **Next**. The **Summary** window is shown.
4. Click **Finish** to run the simulation.

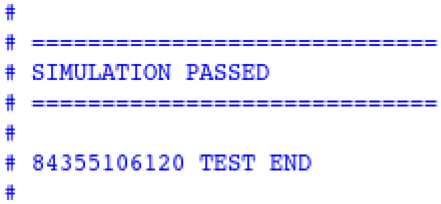
The waveform in Figure 6.7 shows an example simulation result.



	Msgs
rst_n_i	1'h1
clk_ch0_p_i	1'h0
d0_ch0_p_i	1'h0
d1_ch0_p_i	1'h1
d2_ch0_p_i	1'h0
d3_ch0_p_i	1'h1
eclk_i	1'h0
lock_i	1'h1
gddr_rdy_o	1'h1
bit_lock_o	1'h1
word_lock_o	1'h1
bw_rdy_o	1'h1
pix_clk_o	1'h1
de_o	1'h1
hsync_o	1'h0
vsync_o	1'h0
pix_data0_o	24'hab6c0a
phdir_o	1'h0
phstep_o	1'h0
Now	8435510.6122 ns
Cursor 1	7374853.269 ns

Figure 6.7. Simulation Waveform

6.5.1. Simulation Results



```
#
# ==============================
# SIMULATION PASSED
# ==============================
#
# 84355106120 TEST END
#
```

Figure 6.8. Simulation Results

- Follow the procedure in the Running Functional Simulation section to properly simulate your project in the Lattice Radiant software.
- When simulation finishes successfully, it generates log files for input and output data (input_data0.log and pixel_out0.log for Rx channel 0 and input_data1.log and pixel_out1.log for Rx channel 1, respectively) and "Simulation Pass/Fail" is displayed in the terminal.

7. Debugging

This section lists possible issues and suggested troubleshooting steps that you can follow.

7.1. Debug Methods

7.1.1. Debug Mode

This debug feature is used to check data from FPD-Link Rx before it is decoded to control and pixel data.

A pre-defined set of data (set in *Test Mode Expected Data in Hex Format* and driven as LVDS input data) is compared internally to the actual output of the fpd_link_rx_core module. The comparison of data is enabled only after bit and word alignment is completed. If data mismatch is encountered, tstmode_err_o is set to high until reset is asserted or the device is powered down. For dual channel configuration, the same *Test Mode Expected Data in Hex Format* is used.

To enable test mode:

1. Make sure *Enable Test Mode* is checked and *Test Mode Expected Data in Hex Format* is configured during IP generation. Pre-defined data is processed as below:
   - 21-bit/28-bit data is divided into three or four data lanes depending on Data Type selected (see Figure 2.10).
   - Arranged in the following order: MSB = DATAIN4[6] and LSB = DATAIN0[0] with TESTDATA = {DATAIN3[6:0], DATAIN2[6:0], DATAIN1[6:0], DATAIN0[6:0]}.
   - LSB of each lane is the first data input.
2. Drive tstmode_en_i to 1'b1.
3. Continuously drive the 28-bit/21-bit data (should be the same as configured in *Test Mode Expected Data in Hex Format*) per input clock per channel.
   - CLKIN and DATAIN should maintain the same relationship throughout the test.
4. Perform the sequence as specified in the Initialization and Reset Sequence section.
5. Wait for bw_rdy_o to be asserted. Comparison is enabled only after this is asserted.
6. tstmode_err_o is asserted if data mismatch is encountered.

8. Design Considerations

The following are the known limitations:

- For dual channel configuration, odd multiple number of pixels is not supported.
- For dual channel configuration, only the clock from channel 0 is used.
- AXI4-Lite Response signals (BRESP, RRESP) only supports 2’b00 (OKAY).
- Resetting the AXI4-Lite interface (axil_rst_n_i) separately from system reset (rst_n_i) causes replay read in the AXI4-Lite interface. s_axil_bvalid_o is asserted until all data are flashed out. If this is not intended, always toggle axil_rst_n_i and rst_n_i together.
- For Nexus devices, the clk_ch0_p_i pin of the FPD-Link Rx IP is only supported in the bottom banks, which are Banks 3, 4, and 5. Therefore, the use of PLL from the top to drive Banks 3, 4, and 5 at the bottom is not allowed.
- Only the PCLK pin is permitted to drive both the LVDS 7:1 and the PLL simultaneously when configuring this IP using Nexus devices.

Appendix A. Resource Utilization

Table A.1 and Table A.2 show the device information, tool information, and resource utilization of the FPD-Link Receiver IP core on a CertusPro-NX device. Table A.3 and Table A.4 show the device information, tool information, and resource utilization of the FPD-Link Receiver IP core on a Lattice Avant device. The default configuration is used, and some attributes are changed from the default value to show the effect on resource utilization.

Table A.1. CertusPro-NX Device and Tool Information

Software Version	Lattice Radiant Software 2025.2
Device Used	LFCPNX-100-7LFG672C
Performance Grade	7_High-Performance_1.0V
Synthesis Tool	Synplify Pro, October 2025

Table A.2. Resource Utilization on CertusPro-NX Device[1]

Rx Channels	Data Mapping Format	Data Type	Video Data Interface	Rx Line Rate (Mb/s)	Register Interface	f_{MAX} (MHz)[2]	Registers	LUTs[3]	EBRs	High Speed I/O Resources
1	VESA	RGB888	Native Pixel	945	OFF	200	121	214	0	5 x IDDRX71 1 x ECLKSYNC 1 x ECLKDIV 1 x PLL
2	JEIDA	RGB666	Unified Video Streaming TX	945	AXI4-Lite	200	1432	1402	1	4 x IDDRX71 1 x ECLKSYNC 1 x ECLKDIV 1 x PLL

Notes:

1. All other settings are default.
2. f_{MAX} is generated using multiple iterations of place and route.
3. The distributed RAM utilization is accounted for in the total LUT4 utilization. The actual LUT4 utilization is distributed among logic, distributed RAM, and ripple logic.

Table A.3. Avant Device and Tool Information

Software Version	Lattice Radiant Software 2025.2
Device Used	LAV-AT-E70ES1-1LFG1156C
Performance Grade	1
Synthesis Tool	Synplify Pro, October 2025

Table A.4. Resource Utilization on Avant Device[1]

Rx Channels	Data Mapping Format	Data Type	Video Data Interface	Rx Line Rate (Mb/s)	Register Interface	f_{MAX} (MHz)[2]	Registers	LUTs[3]	EBRs	High Speed I/O Resources
1	VESA	RGB888	Native Pixel	1050	OFF	250	121	207	0	5 x IDDRX71 1 x ECLKSYNC 1 x ELCKDIV
2	JEIDA	RGB666	Unified Video Streaming TX	1050	AXI4-Lite	250	1432	1366	1	4 x IDDRX71 1 x ECLKSYNC 1 x ELCKDIV

Notes:

1. All other settings are default.
2. f_{MAX} is generated using multiple iterations of place and route.
3. The distributed RAM utilization is accounted for in the total LUT4 utilization. The actual LUT4 utilization is distributed among logic, distributed RAM, and ripple logic.

For more information regarding a specific configuration, generate the IP, run synthesis and MAP, and check the MAP reports for resource utilization.

References

- OpenLDI/FPD-Link/LVDS Receiver IP Release Notes (FPGA-RN-02013)
- Lattice Radiant Timing Constraints Methodology (FPGA-AN-02059)
- Avant-E web page
- Avant-G web page
- Avant-X web page
- Certus-N2 web page
- Certus-NX web page
- CertusPro-NX web page
- CrossLink-NX web page
- MachXO5-NX web page
- OpenLDI/FPD-Link/LVDS Receiver IP Core web page
- Lattice Radiant Software web page
- Lattice Solutions IP Cores web page
- Lattice Solutions Reference Designs web page
- Lattice Insights 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

Note: In some instances, the IP may be updated without changes to the user guide. The user guide may reflect an earlier IP version but remains fully compatible with the later IP version. Refer to the IP Release Notes for the latest updates.

Revision 1.5, IP v1.9.0, December 2025

Section	Change Summary
All	• Added a note on the IP version in the Quick Facts and Revision History sections. • Made minor editorial fixes.
Introduction	• In Table 1.1. Summary of the FPD-Link Receiver IP: • Updated IP core version and software version in *Lattice Implementation*. • In the Licensing and Ordering Information section: • Updated the licensing and ordering information. • Removed the Ordering Part Number section.
Designing with the IP	• Added note on IP version in GUI. • In the Generating and Instantiating the IP section: • Split Step 2 into Steps 2 and 3. • Updated Steps 2 through 7. • Updated Figure 6.1. Module/IP Block Wizard, Figure 6.2. IP Configuration, and Figure 6.3. Check Generated Result.
Appendix A. Resource Utilization	• In Table A.1. CertusPro-NX Device and Tool Information and Table A.3. Avant Device and Tool Information: • Updated software version. • Updated synthesis tool date. • In Table A.2. Resource Utilization on CertusPro-NX Device: • Updated f_{MAX} and LUTs for Rx Channels = 1. • Updated f_{MAX}, Registers, and LUTs for Rx Channels = 2. • Added note on f_{MAX} in relation to place and route. • In Table A.4. Resource Utilization on Avant Device: • Updated Registers and LUTs. • Added note on f_{MAX} in relation to place and route.

Revision 1.4, IP v1.8.0, July 2025

Section	Change Summary
Introduction	• Added descriptions on input and output interfaces of the IP in the Overview of the IP section. • In Table 1.1. Summary of the FPD-Link Receiver IP: • Renamed *Supported FPGA Family* to *Supported Devices* and incorporated *Targeted Devices* information into row. • Removed *Targeted Devices* row. • Added IP core version to *Lattice Implementation*. • In Table 1.3. Ordering Part Number: • Updated header name from *Multi-Site Perpetual* to *Single Seat Perpetual*.
Designing with the IP	Updated Figure 6.1. Module/IP Block Wizard, Figure 6.2. IP Configuration, and Figure 6.3. Check Generated Result.
Design Considerations	Added design considerations regarding clk_ch0_p_i pin and PCLK.
Appendix A. Resource Utilization	• In Table A.1. CertusPro-NX Device and Tool Information and Table A.3. Avant Device and Tool Information: • Updated software version. • Updated synthesis tool date. • In Table A.2. Resource Utilization on CertusPro-NX Device and Table A.4. Resource

Section	Change Summary
	Utilization on Avant Device: - Updated f_{MAX} for Rx Channels = 1. - Updated f_{MAX}, Registers, LUTs, and High Speed I/O Resources for Rx Channels = 2.
References	- Updated listing name to Lattice Radiant Software web page. - Added OpenLDI/FPD-Link/LVDS Receiver IP Core web page.

Revision 1.3, IP v1.7.0, December 2024

Section	Change Summary
Cover	Added IP version.
Abbreviations in This Document	Updated section title, description, and table header.
Introduction	- Added Certus-N2 to list of targeted devices in description. - In Table 1.1. Summary of the FPD-Link Receiver IP: - Added Certus-N2 to *Supported FPGA Family*. - Added IP Changes row. - Added Lattice Implementation row and moved IP version information into *Lattice Implementation*. - Added IP core v1.7.0 to *Lattice Implementation*. - Removed IP Version row. - Added LFD2NX-9, LFD2NX-17, LFD2NX-28, and LN2-CT-20 to *Targeted Devices*. - Added the IP Support Summary section. - Removed the IP Validation Summary section. - In Table 1.3. Ordering Part Number: - Updated column header to *Single Seat Annual*. - Added part numbers for Certus-N2. - Added the Hardware Support section.
Signal Description	Updated eclk_i description.
Designing with the IP	Updated Figure 6.1. Module/IP Block Wizard, Figure 6.2. IP Configuration, and Figure 6.3. Check Generated Result.
Appendix A	- Updated software version and synthesis tool version in Table A.1. CertusPro-NX Device and Tool Information. - Updated f_{MAX}, registers, and LUTs information in Table A.2. Resource Utilization on CertusPro-NX Device and Table A.4. Resource Utilization on Avant Device. - Updated software version and synthesis tool version in Table A.3. Avant Device and Tool Information.
References	- Added OpenLDI/FPD-Link/LVDS Receiver IP Release Notes and Certus-N2 web page. - Removed Lattice Radiant Software User Guide.

Revision 1.2, June 2024

Section	Change Summary
All	Made editorial changes.
Acronyms and Abbreviations in This Document	- Renamed section. - Added items.
Introduction	- Reworked section content and added the following subsections: - 1.1 Overview of the IP - 1.4 Licensing and Ordering Information - 1.5 IP Validation Summary - 1.6 Minimum Device Requirements - Reworked subsection IP Evaluation and section Ordering Part Number into subsection 1.4 Licensing and Ordering Information. - Table 1.1. Summary of the FPD-Link Receiver IP in subsection 1.2 Quick Facts:

Section	Change Summary
	• Added IP Version. • Updated Supported User Interface and Lattice Implementation. • Updated key features in subsection 1.3 Features.
Functional Description	• Updated and reworked section content and added the following subsections: • 2.1 IP Architecture Overview • 2.2 Clocking • 2.3 Reset • 2.4 User Interfaces • 2.5 Other IP Specific Blocks/Layers/Interfaces
IP Parameter Description	• Added this section. • Reworked subsection Attribute Summary and added into this section.
Signal Description	• Added this section. • Reworked subsection Signal Description and added into this section.
Register Description	• Added this section.
Designing with the IP	• Added this section. • Reworked section IP Generation, Simulation, and Validation and added into this section.
Debugging	• Added this section. • Reworked subsection Debug Mode and added into this section.
Design Considerations	• Added this section. • Moved limitations into this section. • Added limitations.
Appendix A	Reworked and updated this section.

Revision 1.1, December 2023

Section	Change Summary
All	Renamed the document from *OpenLDI/FPD-LINK/LVDS Receiver IP Core - Lattice Radiant Software* to *OpenLDI/FPD-LINK/LVDS Receiver IP.*
Disclaimers	Updated this section.
Introduction	Updated below details in Table 1.1. OpenLDI/FPD-LINK/LVDS Receiver IP Quick Facts: • Added *MachXO5-NX, Lattice Avant*, *CertusPro-NX* to Supported FPGA Families. • Added *LIFCL-33, LFCPNX-100, LFMXO5-25, LFMXO5-55T, LFMXO5-100T, LAV-AT-E70, LAV-AT-G70, LAV-AT-X70* to Targeted Devices. Replaced *IP Core v1.0.0 – Lattice Radiant software 2.1* with *IP Core v1.x.x - Lattice Radiant Software 2.1 and later* and *IP Core v1.4.x - Lattice Radiant Software 2023.1*.
Functional Descriptions	• Replaced paragraph from *The functional block diagram of OpenLDI/FPD-LINK/LVDS Receiver IP Core is shown in Figure 2.1. The dashed lines in the figure are optional components/signals, which means they may not be available in the IP when disabled in the attribute* to *The functional block diagram of OpenLDI/FPD-LINK/LVDS Receiver IP Core is shown in Figure 2.1. The dashed lines in the figure are optional components/signals and may not be available in the IP depending on the attribute and/or device selected* in Overview section. • Updated Figure 2.1. Functional Block Diagram, Figure 2.14. FPD-Link Rx Block Diagram, and added Figure 2.15. FPD-Link Rx Block Diagram without GPLL, Figure 2.16. GPLL connection for Lattice Avant. • Updated Table 2.1. OpenLDI/FPD-LINK/LVDS Receiver IP Core Signal Description, Table 2.2. Attributes Table, and Table 2.3. Attributes Description. • Replaced the step 2 from *If Enable Miscellaneous signals is checked, wait for GPLL lock to be asserted. The pll_lock_o is used to indicate that output clock/s of GPLL is/are already stable. Wait for bw_rdy_o to be asserted. The bw_rdy_o is used to indicate LVDS 7:1 Rx data training is done. Only when bw_rdy_o is asserted, valid data can be sampled and correctly transmitted by FPD-Link IP* to *Wait for GPLL lock to be asserted. GPLL lock is used to indicate that output clock/s of GPLL is/are already stable. For Avant*

Section	Change Summary
	devices, this is indicated by lock_i input signal while for non-Avant devices, this is indicated by pll_lock_o output signal. Wait for bw_rdy_o to be asserted. The bw_rdy_o is used to indicate LVDS 7:1 Rx data training is done. Only when bw_rdy_o is asserted, valid data can be sampled and correctly transmitted by FPD-Link IP in Clock, Reset, and Initialization section. • Deleted *general purpose PLL (GPLL)* and added sentence *A general purpose PLL (GPLL) is automatically instantiated inside the IP for non-Avant devices as shown in Figure 2.12* in Module Description section. • Added the general guidelines for setting GPLL in GPLL section. • Updated the title of below figures: • Figure 2.3. Single Channel OpenLDI/FPD-LINK/LVDS Input Bus Waveform for RGB888 Format (VESA) • Figure 2.4. Dual Channel OpenLDI/FPD-LINK/LVDS Input Bus Waveform for RGB888 Format (VESA) • Figure 2.7. Single Channel OpenLDI/FPD-LINK/LVDS Input Bus Waveform for RGB666 Format (JEIDA/VESA) • Figure 2.8. Dual Channel OpenLDI/FPD-LINK/LVDS Input Bus Waveform for RGB666 Format (JEIDA/VESA) • Added Figure 2.5. Single Channel OpenLDI/FPD-LINK/LVDS Input Bus Waveform for RGB888 Format (JEIDA) and Figure 2.6. Dual Channel OpenLDI/FPD-LINK/LVDS Input Bus Waveform for RGB888 Format (JEIDA).
IP Generation, Simulation, and Validation	• Updated the title of this section from *IP Generation and Evaluation* to *IP Generation, Simulation, and Validation*. • Deleted Licensing the IP section. • Updated the title of the section from *Generation and Synthesis* to Generating the IP. • Updated Figure 3.1. Module/IP Block Wizard, Figure 3.2. Configure User Interface of Selected OpenLDI/FPD-LINK/LVDS Receiver IP Core, Figure 3.3. Check Generating Result, and Figure 3.5. Adding and Reordering Source. • Added attribute *eval/eval.pdc* and description *This file provides the information on how to constrain the clocks* in Table 3.1. Generated File List. • Added *testbench/tb_axis_slv.sv attribute* and its description in Table 3.2. Testbench File List. • Replaced the title from *3.2.1. Required Post-Synthesis Constraints* to Constraining the IP and updated the section as per the actual clock. • Changed the bullet information from *When simulation finishes successfully, it generates log files for input and output data (input_data0.log and pixel_out0.log for Rx channel 0 and input_data1.log and pixel_out1.log for Rx channel 1, respectively). Compare these log files to check if simulation passes* to *When simulation finishes successfully, it generates log files for input and output data (input_data0.log and pixel_out0.log for Rx channel 0 and input_data1.log and pixel_out1.log for Rx channel 1, respectively) and "Simulation Pass/Fail" is displayed in the terminal* in Running Functional Simulation section. • Updated the title from *Hardware Evaluation* to IP Evaluation. • Replaced *LIFCL and LFD2NX devices* with *Lattice* devices in IP Evaluation section.
Ordering Part Number	Added OPNs for Lattice Avant and CertusPro-NX in Table 4.1. Ordering Part Numbers.
Appendix A. Resource Utilization	Replaced *LAV-AT-500E-3LFG1156C* with *LAV-AT-E70-3LFG1156C*.
References	Added links for CrossLink-NX, Certus-NX, CertusPro-NX, MachXO5-NX, Lattice Avant, Lattice Radiant, and Lattice Radiant Timing Constraints Methodology (FPGA-AN-02059) web pages.
Technical Support Assistance	Added web link for Lattice FAQ data base in Technical Support Assistance section.

Revision 1.0, August 2020

Section	Change Summary
All	Initial release.