OpenLDI/FPD-LINK/LVDS Receiver IP

User Guide

FPGA-IPUG-02116-1.1

December 2023

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 REDUDANCIES, 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.

Contents

Figures

Tables

Acronyms in This Document

A list of acronyms used in this document.

Acronym	Definition
FPD	Flat Panel Display
LVDS	Low Voltage Differential Signaling
LDI	LVDS Display Interface
RTL	Register Transfer Language

1. Introduction

The Lattice Semiconductor OpenLDI/FPD-LINK/LVDS Receiver Interface IP translates video streams from a processor with an OpenLDI/FDP-Link/LVDS interface connection to pixel clock domain. This 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 Pixel-to-Byte Converter and CSI-2/DSI D-PHY Transmitter Submodule IPs.

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, it uses LVDS interface for 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 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 OpenLDI/FPD-LINK/LVDS Receiver Interface 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, and Lattice Avant™ devices and implemented using the Lattice Radiant® software Place and Route tool integrated with the Synplify Pro® synthesis tool.

1.1. Quick Facts

Table 1.1 presents a summary of the OpenLDI/FPD-LINK/LVDS Receiver IP.

Table 1.1. OpenLDI/FPD-LINK/LVDS Receiver IP Quick Facts

IP Requirements	Supported FPGA Families	CrossLink-NX, Certus-NX, CertusPro-NX, MachXO5-NX, and Lattice Avant
Resource Utilization	Targeted Devices	LIFCL-40, LIFCL-17, LFD2NX-40, LIFCL-33, LFCPNX-100, LFMXO5-25, LFMXO5-55T, LFMXO5-100T, LAV-AT-E70, LAV-AT-G70, and LAV-AT-X70
	Supported User Interface	Native LVDS to parallel
	Resources	See Table A.1. Resource Utilization and Table A.2. Resource Utilization using Lattice Avant.
Design Tool Support	Lattice Implementation	IP Core v1.x.x - Lattice Radiant Software 2.1 and later IP Core v1.4.x - Lattice Radiant Software 2023.1
	Synthesis	Lattice Synthesis Engine
		Synopsys® Synplify Pro for Lattice
	Simulation	For a list of supported simulators, see the Lattice Radiant Software 2.1 User Guide.

1.2. Features

The key features of the OpenLDI/FPD-LINK/LVDS 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 3–4 LVDS data lanes per channel
- Supports one or two output pixels per pixel clock
- Supports interfacing up to 7.560 Gb/s

1.3. Conventions

1.3.1. Nomenclature

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

1.3.2. 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

1.3.3. Data Ordering and Data Types

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

1.3.4. Attribute Names

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

2. Functional Descriptions

2.1. Overview

The OpenLDI/FPD-LINK/LVDS Receiver Interface IP converts a standard OpenLDI serial video interface into 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 OpenLDI (LVDS 7:1) interface format. Output interface consists of the RGB control signals, pixel clock, up to two pixel data per pixel clock, and optional debug signals.

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.

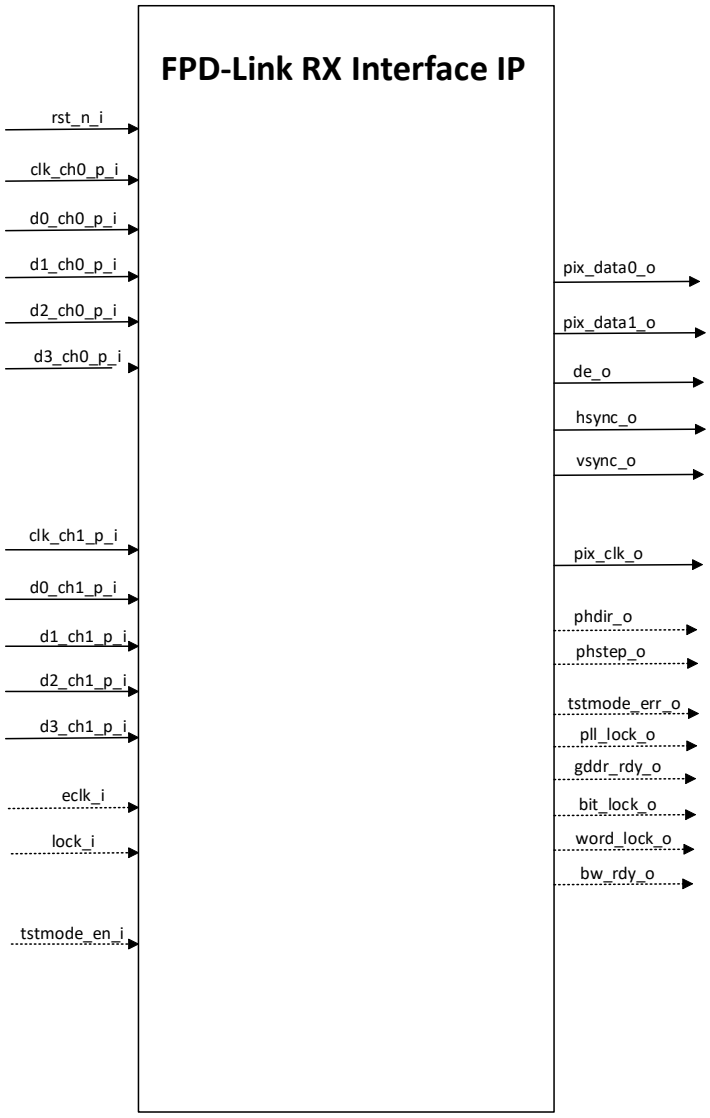

Figure 2.1. Functional Block Diagram

2.2. Signal Description

Table 2.1. OpenLDI/FPD-LINK/LVDS Receiver IP Core Signal Description

Port Name	I/O	Width	Description
Clock and Reset			
rst_n_i	In	1	Asynchronous active low system reset. 0 – System on reset
Pixel Domain Interface			
pix_clk_o	Out	1	Output pixel clock
pix_data0_o	Out	*Number of RX Lanes* * 6	Output pixel data 0 Bus width depends on the data type selected.
pix_data1_o[1,2]	Out	*Number of RX Lanes* * 6	Output pixel data 1 Bus width depends on the data type selected.
de_o	Out	1	Output data enable for parallel interface.
hsync_o	Out	1	Output horizontal sync for parallel interface.
vsync_o	Out	1	Output vertical sync for parallel interface.
OpenLDI/FPD-LINK Interface			
clk_ch0_p_i	In	1	Positive LVDS input clock to LVDS7:1 Rx channel 0.
clk_ch1_p_i	In	1	Positive LVDS input clock to LVDS7:1 Rx channel 1.
d0_ch0_p_i	In	1	Positive LVDS input data lane 0 to LVDS7:1 Rx channel 0.
d1_ch0_p_i	In	1	Positive LVDS input data lane 1 to LVDS7:1 Rx channel 0.
d2_ch0_p_i	In	1	Positive LVDS input data lane 2 to LVDS7:1 Rx channel 0.
d3_ch0_p_i[3]	In	1	Positive LVDS input data lane 3 to LVDS7:1 Rx channel 0.
d0_ch1_p_i[4]	In	1	Positive LVDS input data lane 0 to LVDS7:1 Rx channel 1.
d1_ch1_p_i[4]	In	1	Positive LVDS input data lane 1 to LVDS7:1 Rx channel 1.
d2_ch1_p_i[4]	In	1	Positive LVDS input data lane 2 to LVDS7:1 Rx channel 1.
d3_ch1_p_i[3,4]	In	1	Positive LVDS input data lane 3 to LVDS7:1 Rx channel 1.
PLL Interface[7]			
eclk_i	In	1	Input edge/fast clock from PLL that is 3.5 times faster than the LVDS clock.
lock_i	In	1	PLL lock signal. Once asserted, DDR synchronization process will start.
phdir_o	Out	1	Dynamic phase direction signal. Controls the phase direction of the PLL during bit and word alignment process.
phstep_o	Out	1	Dynamic phase step signal. Controls the phase step of external PLL during bit and word alignment process.
Miscellaneous			
tstmode_en_i[5]	In	1	Enable or disable test mode. 1 – Enable test mode 0 – Disable test mode
tstmode_err_o[5]	Out	1	1 – Indicates that compare mismatch is encountered when test mode is enabled 0 – No error is encountered Automatically set to 0 when test mode is disabled.
pll_lock_o[6,8]	Out	1	General PLL lock signal.
gddr_rdy_o[6]	Out	1	1 – Indicates that DDR synchronization is already done 0 – DDR synchronization is not yet started or still in progress
bit_lock_o[6]	Out	1	1 – Indicates that bit alignment is already done 0 – Bit alignment is not yet started or still in progress
word_lock_o[6]	Out	1	1 – Indicates that word alignment is already done

Port Name	I/O	Width	Description
			0 – Word alignment is not yet started or still in progress
bw_rdy_o[6]	Out	1	1 – Indicates that data training is already done 0 – Data training is not yet started or still in progress

Note:

1. Available only when Number of Output Pixels per Clock is more than one.
2. This pixel data is received from channel 1 when dual channel is selected.
3. Available only for Data Type == RGB888.
4. LVDS channel 1 input ports are not available when single LVDS channel is selected.
5. Available only if Enable Test Mode is checked. This is in beta mode and is not tested in the initial release. See the Debug Mode section for details.
6. Available if Enable Miscellaneous Signals is checked.
7. Available only for Avant devices.
8. Available only for non-Avant devices.

2.3. Attribute Summary

The configurable attributes of the OpenLDI/FPD-LINK/LVDS Receiver IP Core are shown in Table 2.2 and are described in Table 2.3. The attributes can be configured through the IP Catalog's Module/IP wizard of the Lattice Radiant software.

Table 2.2. Attributes Table

Attribute	Selectable Values	Default	Dependency on Other Attributes
Receiver Interface			
Number of RX Channels	1, 2	1	—
RX Interface	LVDS	LVDS	—
Format	Format 1 (JEIDA), Format 2 (VESA)	Format 2 (VESA)	--
Number of RX Lanes	3, 4	4	3 if *Data Type* == RGB666; 4 if *Data Type* == RGB888;
RX Gear	7	7	—
Pixel Interface			
Data Type	RGB888, RGB666	RGB888	—
Number of Output Pixels per Clock	1, 2	1	Calculated based on the formula: *Number of Output Pixels per Clock == Number of RX Channels*
Clock			
RX Total Aggregate Bandwidth (Mbps)	210–7560	3780	Calculated based on the formula: *RX Line Rate per lane * Number of RX Lanes * Number of RX Channels*
RX Line Rate per lane (Mbps)	70–945	945	—
Pixel Clock Frequency (MHz)	10–135	135	Calculated based on the formula: *RX Line Rate per lane / RX Gear*
LVDS Input Clock Frequency (MHz)	10–135	135	Calculated based on the formula: *Pixel Clock Frequency * (RX Gear/7)*
LVDS ECLK Frequency (MHz)	35–472.5	472.5	Calculated based on the formula: *LVDS Input Clock Frequency* * 3.5
Miscellaneous			
Enable Miscellaneous signals	Checked, Unchecked	Checked	—
Enable Test Mode	Checked, Unchecked	Unchecked	—
Test mode expected data in Hex format	21'h000000–28'hFFFFFFF	0	Editable when *Enable Test Mode* is checked. Calculated based on the formula: *Number of RX Lanes* * 7

Table 2.3. Attributes Description

Attribute	Description
Receiver Interface	
Number of RX Channels	Specify how many LVDS links are used
RX Interface	Specify the I/O interface
Format	Specify data format
Number of RX Lanes	Specify number of data lanes per Rx link
RX Gear	Specify what DDR71 gearing is used
Pixel Interface	
Data Type	Specify the data type
Number of Output Pixels per Clock	Specify the number of output pixels per clock
Clock	
RX Total Aggregate Bandwidth (Mbps)	Rx total line rate
RX Line Rate per lane (Mbps)	Target line rate per lane
Pixel Clock Frequency (MHz)	Pixel clock
LVDS Input Clock Frequency (MHz)	LVDS clock
LVDS ECLK Frequency (MHz)	LVDS edge clock
Miscellaneous	
Enable Miscellaneous signals	Enable miscellaneous signals
Enable Test Mode	Enable test mode When enabled, Rx internally compares the actual output of IP and the programmed *Test mode expected data in Hex format*.
Test mode expected data in Hex format	Test data value used when *Enable Test Mode* is checked.

2.4. Interface and Timing Diagrams

Figure 2.2 shows the timing of LVDS 7:1 input interface. There is a 2-bit offset between the rising edge of LVDS clock and the word boundary. Each word is 7-bit 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 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 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.

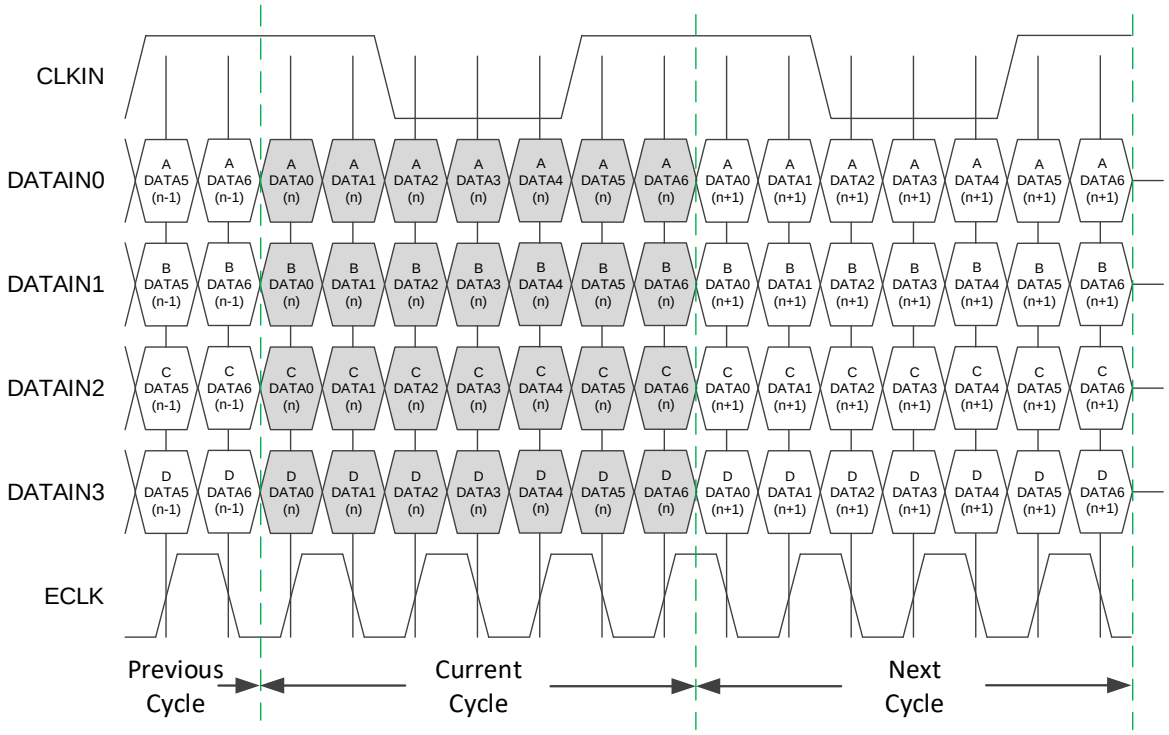

Figure 2.2. OpenLDI/FPD-LINK/LVDS Input Bus Waveform

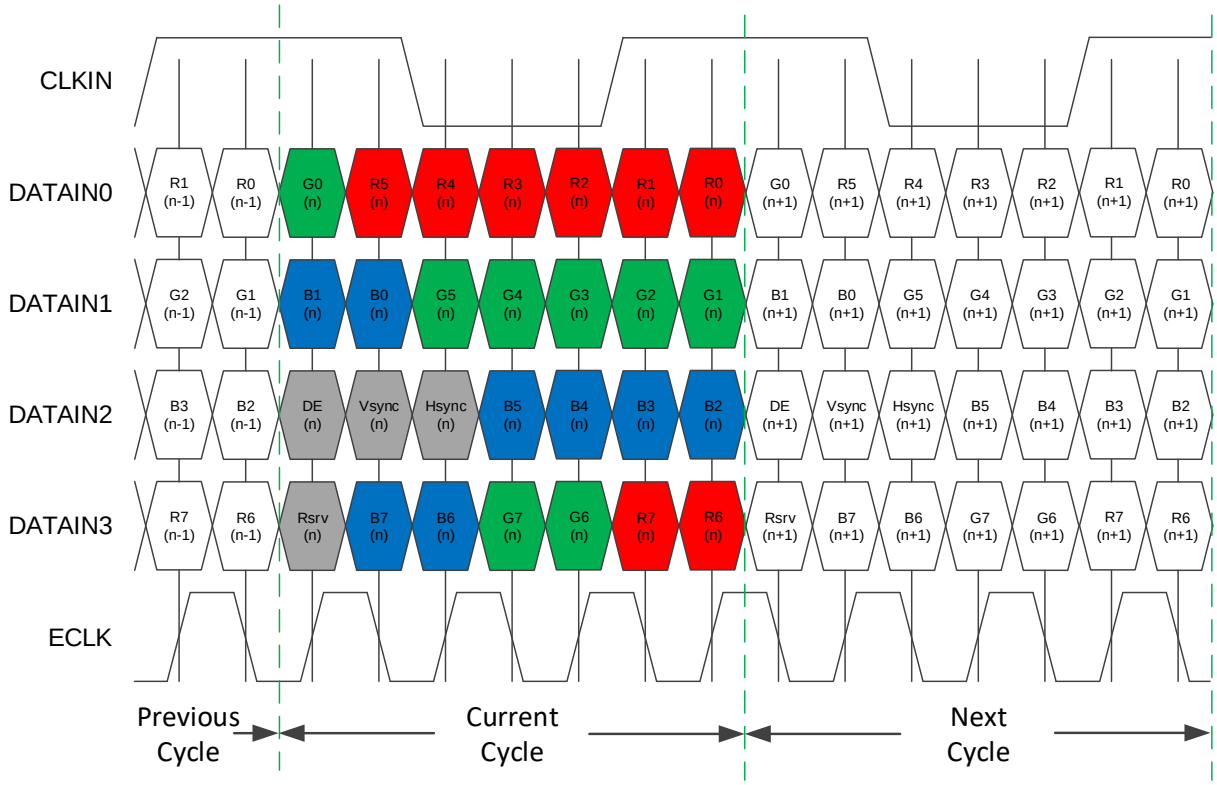

Figure 2.3. Single Channel OpenLDI/FPD-LINK/LVDS Input Bus Waveform for RGB888 Format (VESA)

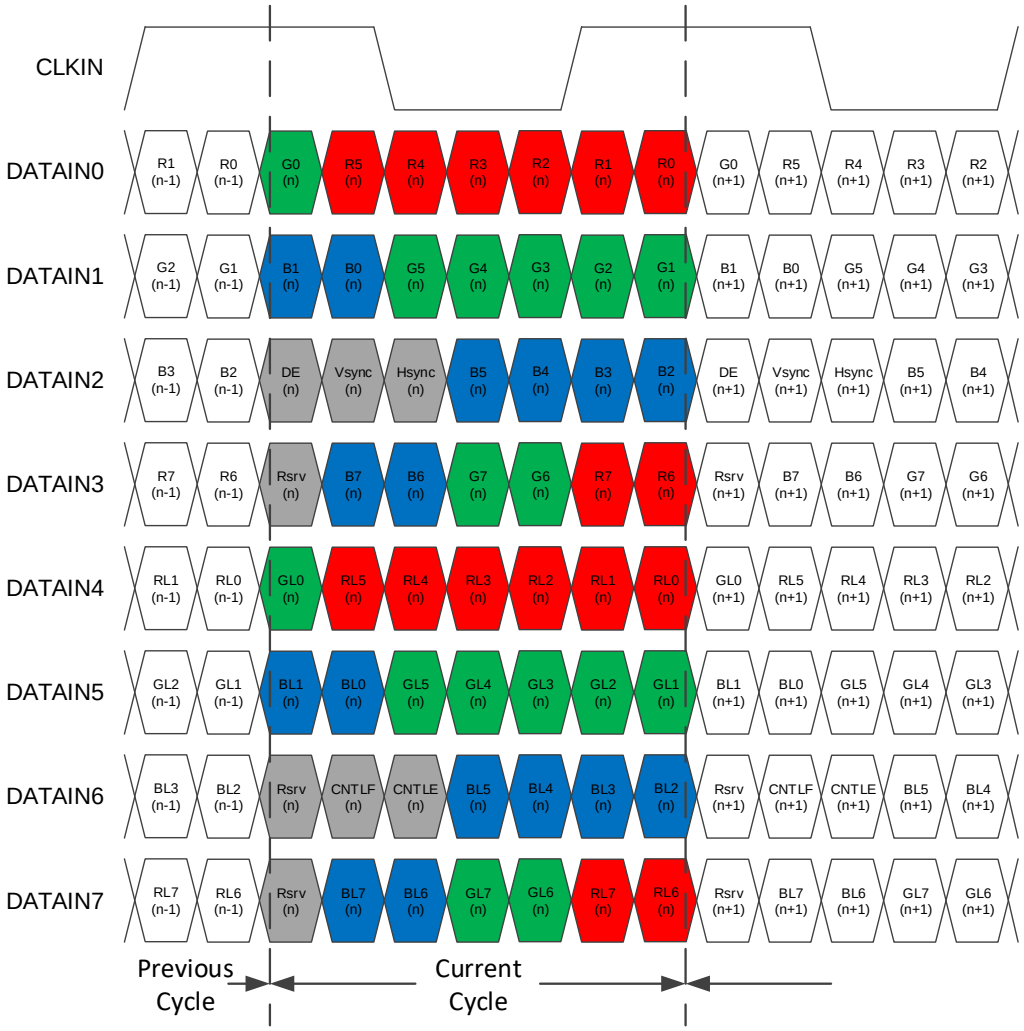

Figure 2.4. Dual Channel OpenLDI/FPD-LINK/LVDS Input Bus Waveform for RGB888 Format (VESA)

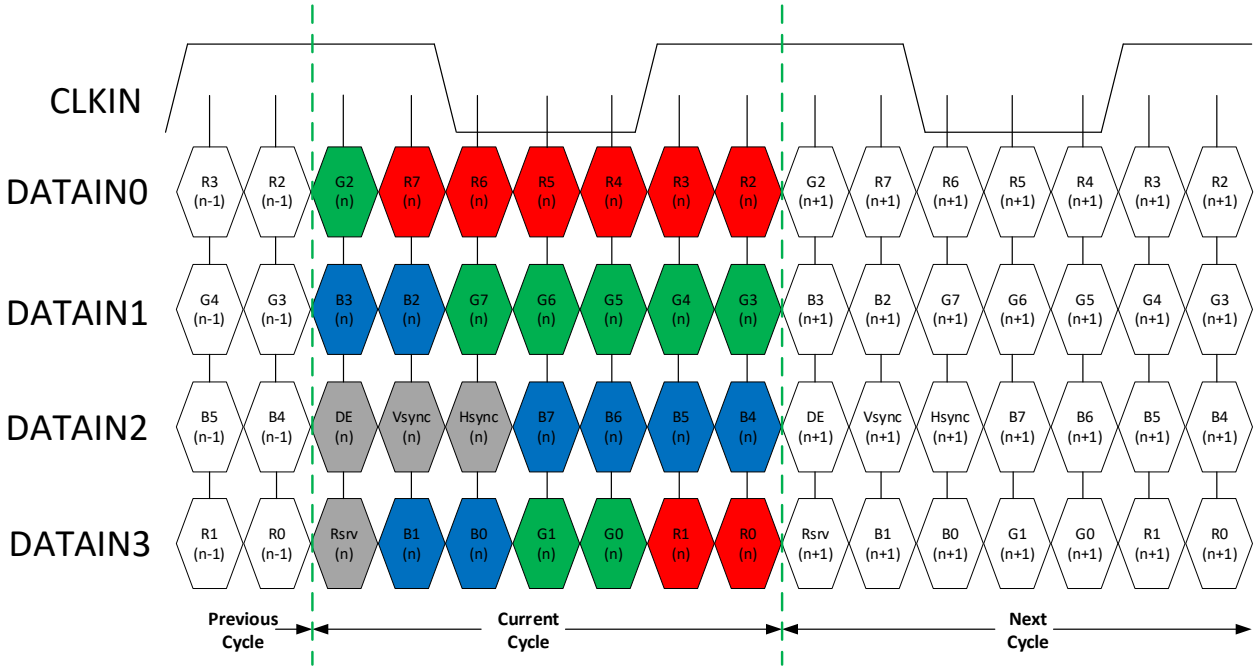

Figure 2.5. Single Channel OpenLDI/FPD-LINK/LVDS Input Bus Waveform for RGB888 Format (JEIDA)

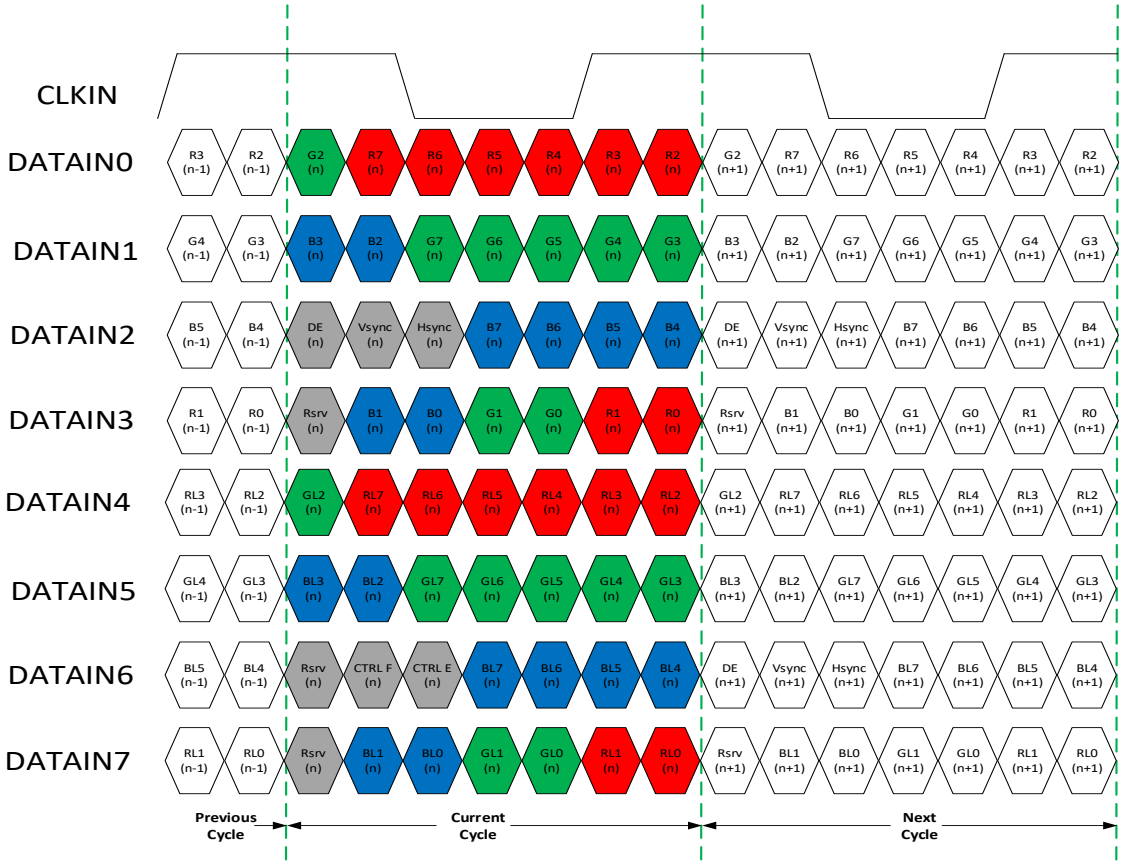

Figure 2.6. Dual Channel OpenLDI/FPD-LINK/LVDS Input Bus Waveform for RGB888 Format (JEIDA)

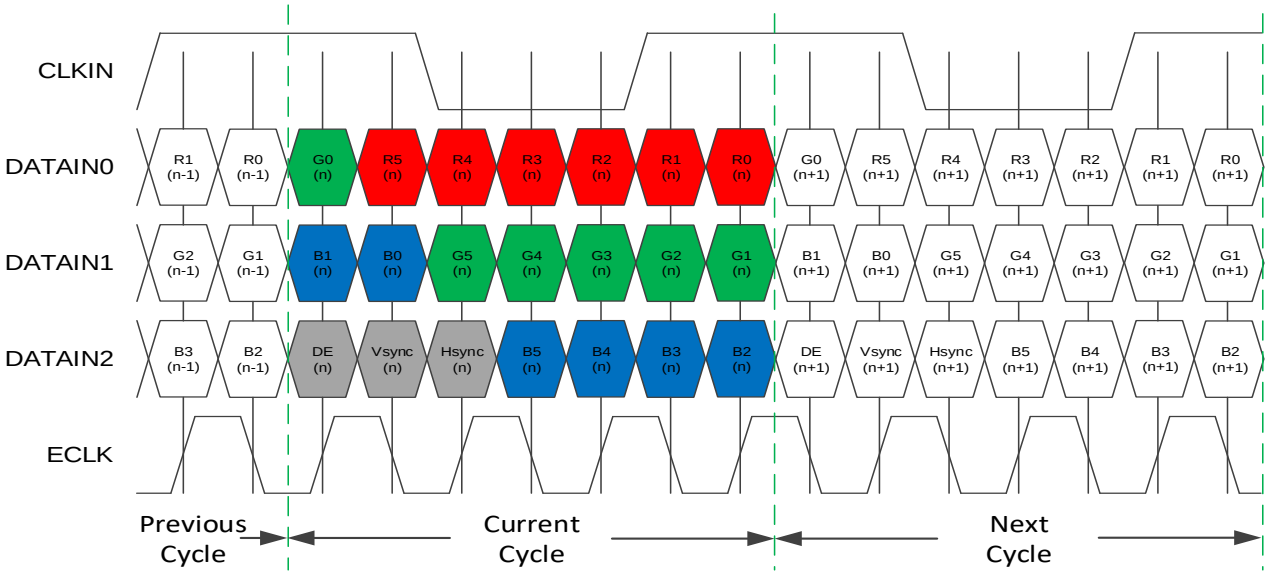

Figure 2.7. Single Channel OpenLDI/FPD-LINK/LVDS Input Bus Waveform for RGB666 Format (JEIDA/VESA)

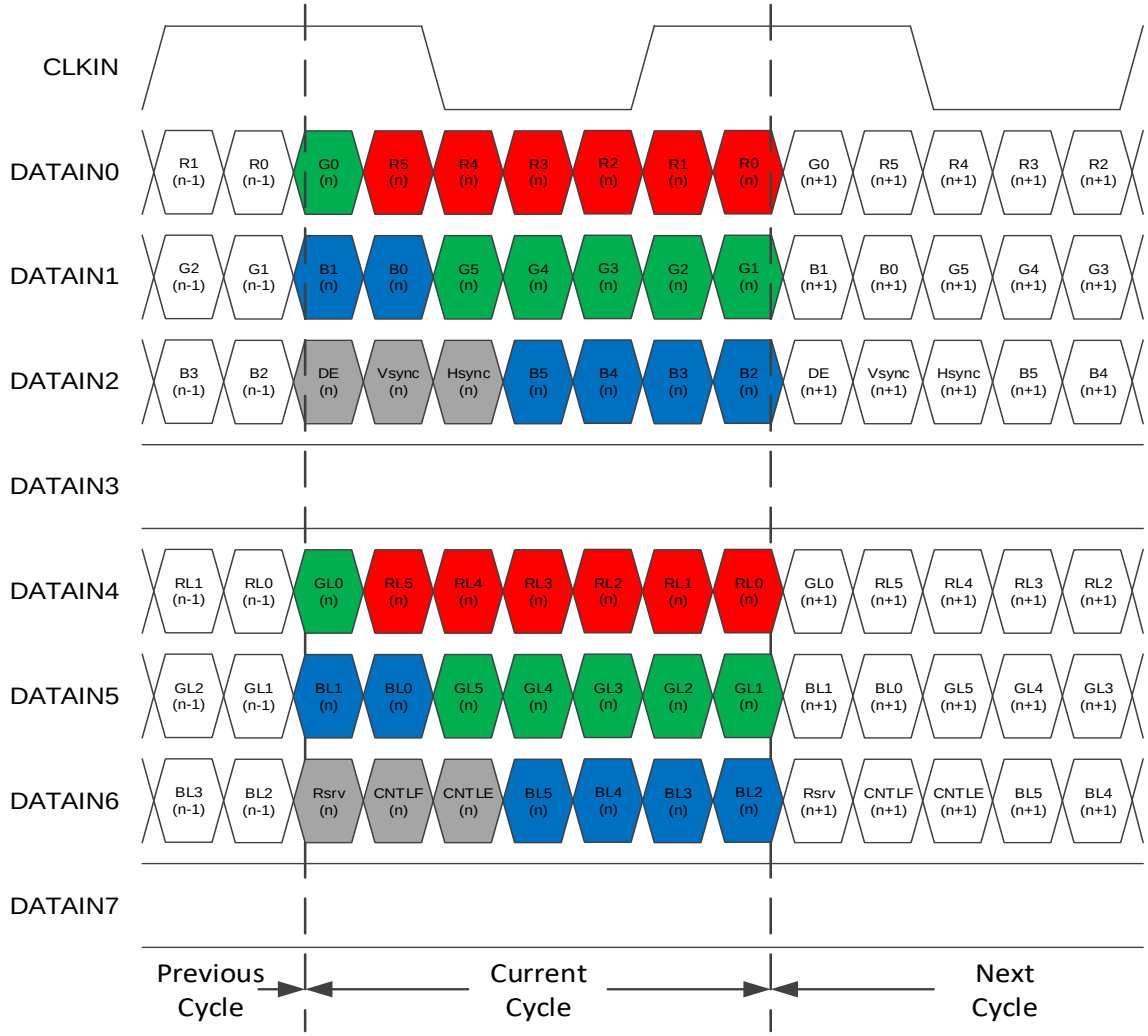

Figure 2.8. Dual Channel OpenLDI/FPD-LINK/LVDS Input Bus Waveform for RGB666 Format (JEIDA/VESA)

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

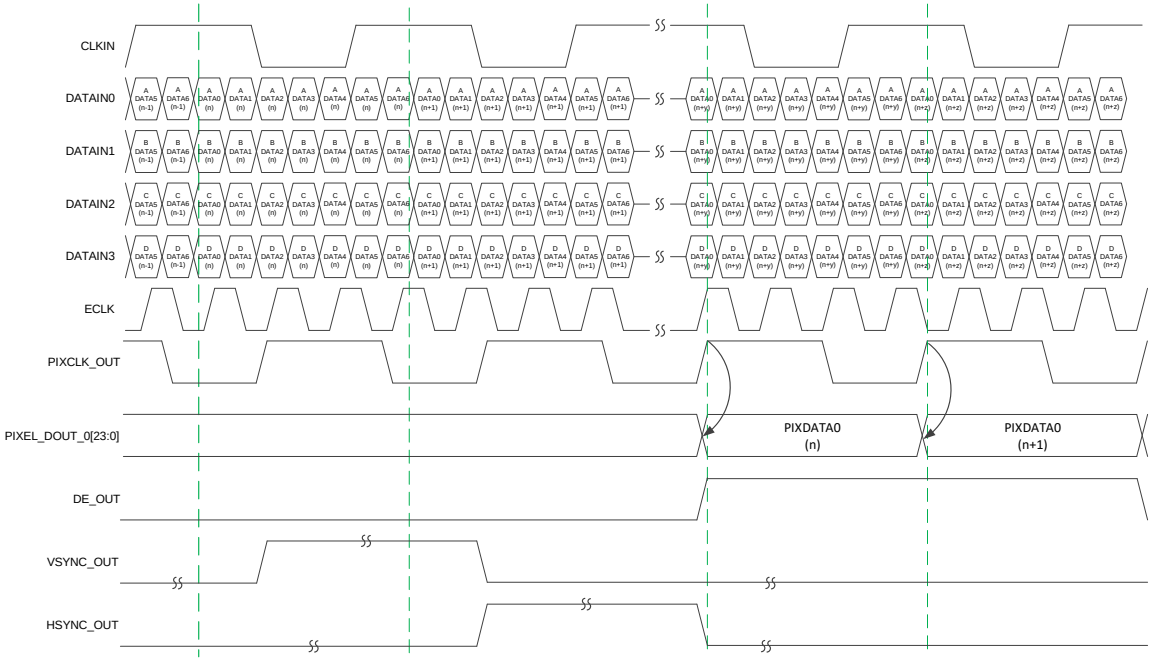

Figure 2.9. Input to Output Waveform for Single Channel OpenLDI/FPD-LINK/LVDS

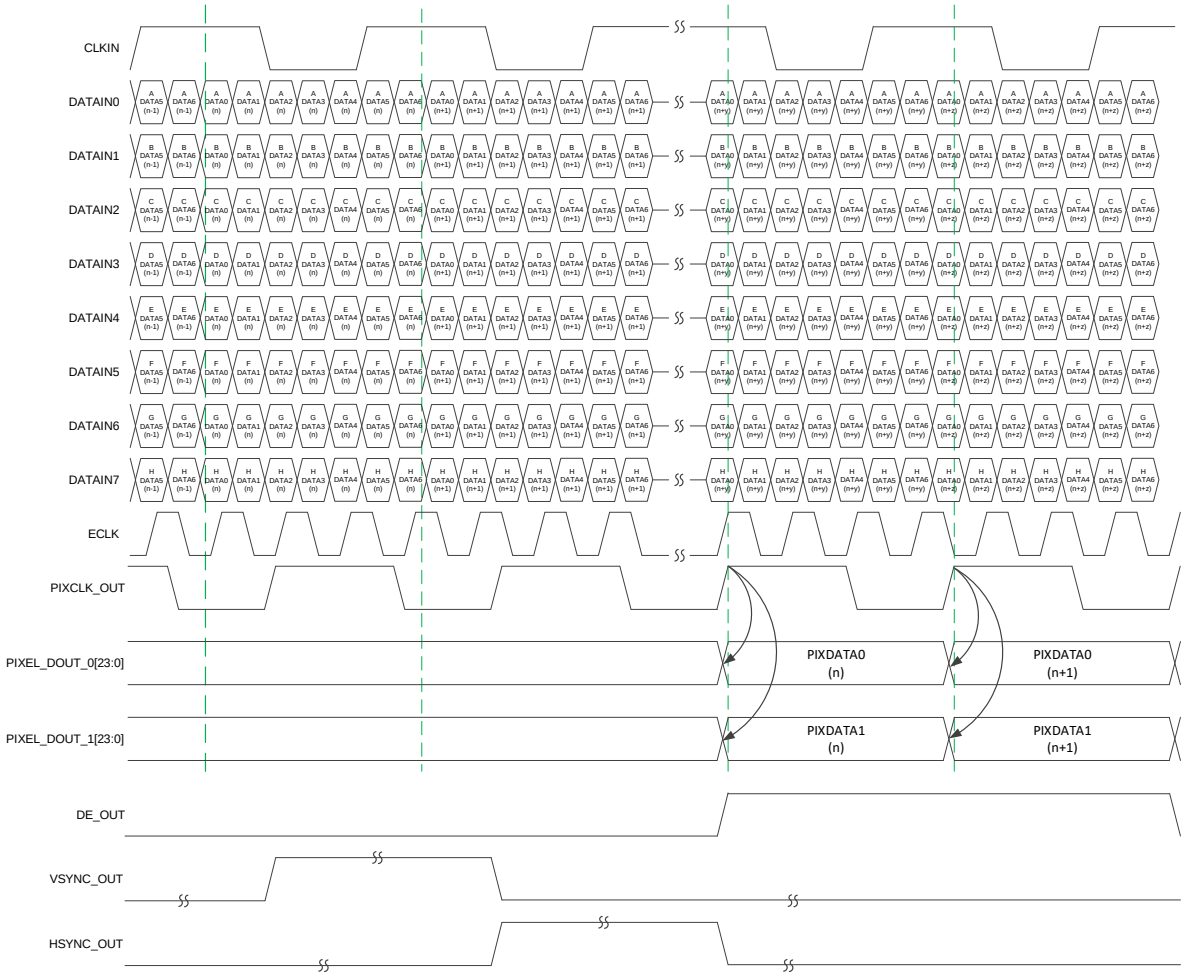

Figure 2.10. Input to Output Waveform for Dual Channel OpenLDI/FPD-LINK/LVDS

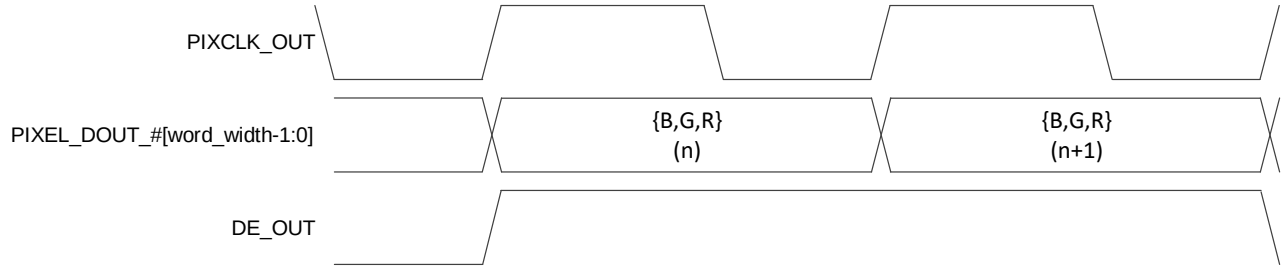

Figure 2.11. Output Pixel Data RGB Arrangement

Table 2.4 Input Pixel Data Summary

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

General arrangement on how pixel data are mapped based on configuration is shown in Figure 2.12.

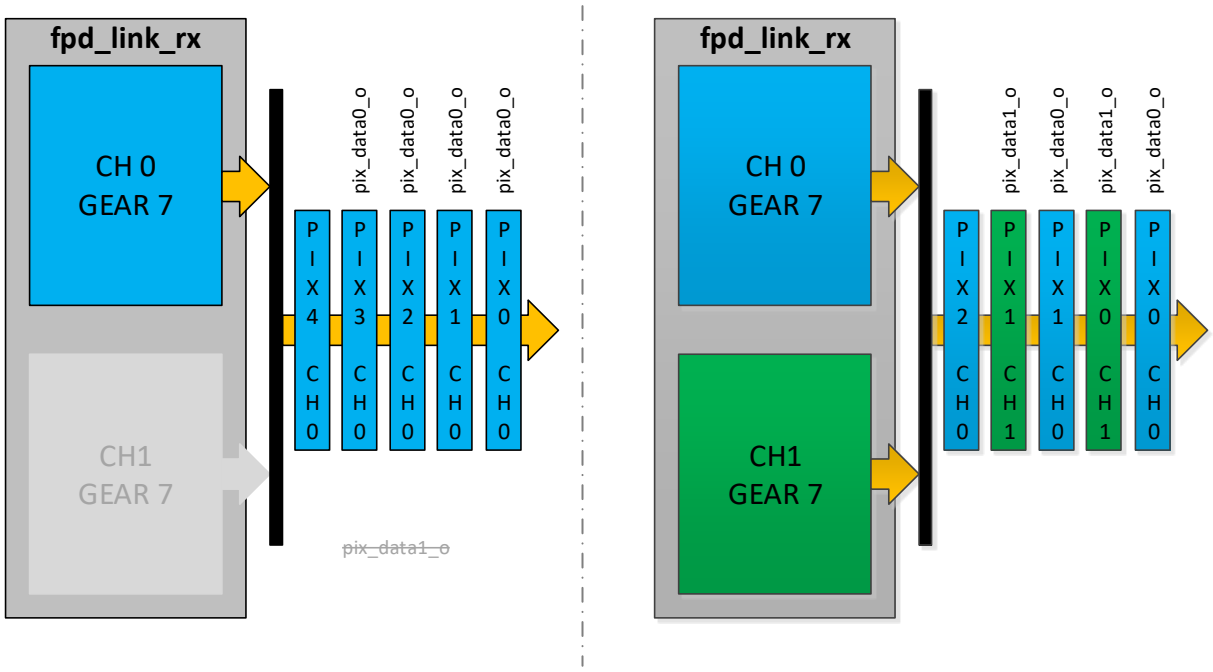

Figure 2.12. Output Pixel Data Arrangement for Single and Dual Channel OpenLDI/FPD-LINK/LVDS

2.5. Clock, Reset, and Initialization

Active low reset is used in the design with synchronous release. This is the system reset input connected to LVDS7:1 Rx module.

Follow this initialization and reset sequence:

1. Assert active low system reset for at least three clock cycle of the slowest clock (pixel clock). It is expected that the input clock is already stable after reset. Clock synchronization is started immediately after release of system reset.
2. 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 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.

If *Enable Miscellaneous signals* is unchecked, wait for some time before sending the valid data to give time for Rx clock synchronization and data training to be finished.

2.6. Clock Domains and Clock Domain Crossing

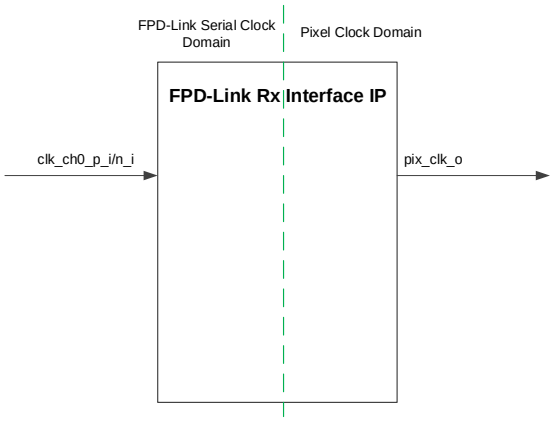

Figure 2.13. Clock Domain Crossing Block Diagram

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

Rx line rate (total) = $RX\ line\ rate\ (per\ lane) * no.of\ RX\ lane * no.of\ Rx\ Channel$

Pixclk (Pixel Clock) = $\frac{RX\ line\ rate\ (per\ lane)}{RX\ gear}$

Rx LVDS Input Clock = $pixclk * \frac{RX\ gear}{7}$

Rx LVDS ECLK = $pixclk * \frac{RX\ gear}{2}$

Number of Pixels per Pixel Clock = $\frac{RX\ gear}{7} * no.of\ Rx\ Channel$

2.7. Module Description

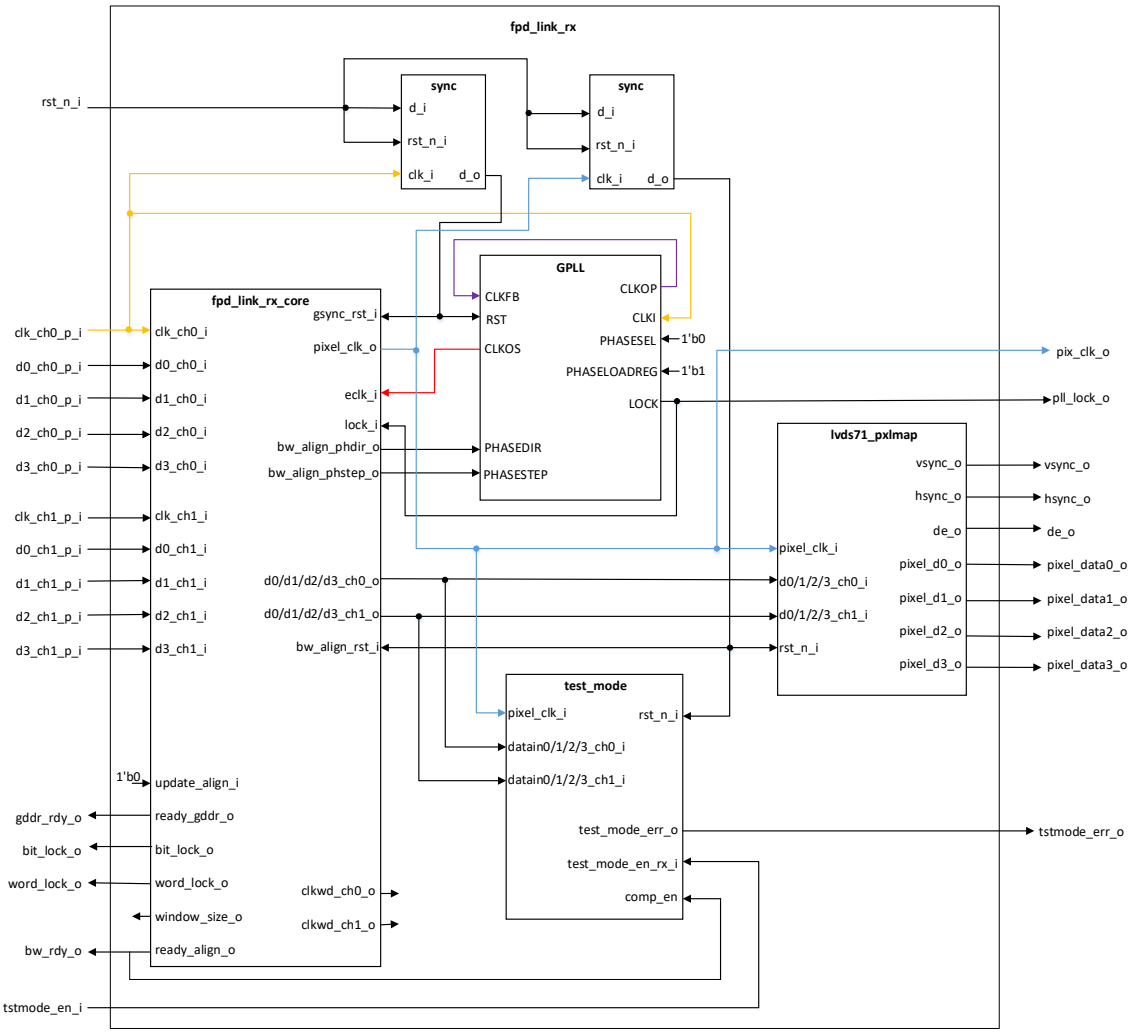

Figure 2.14. FPD-Link Rx Block Diagram

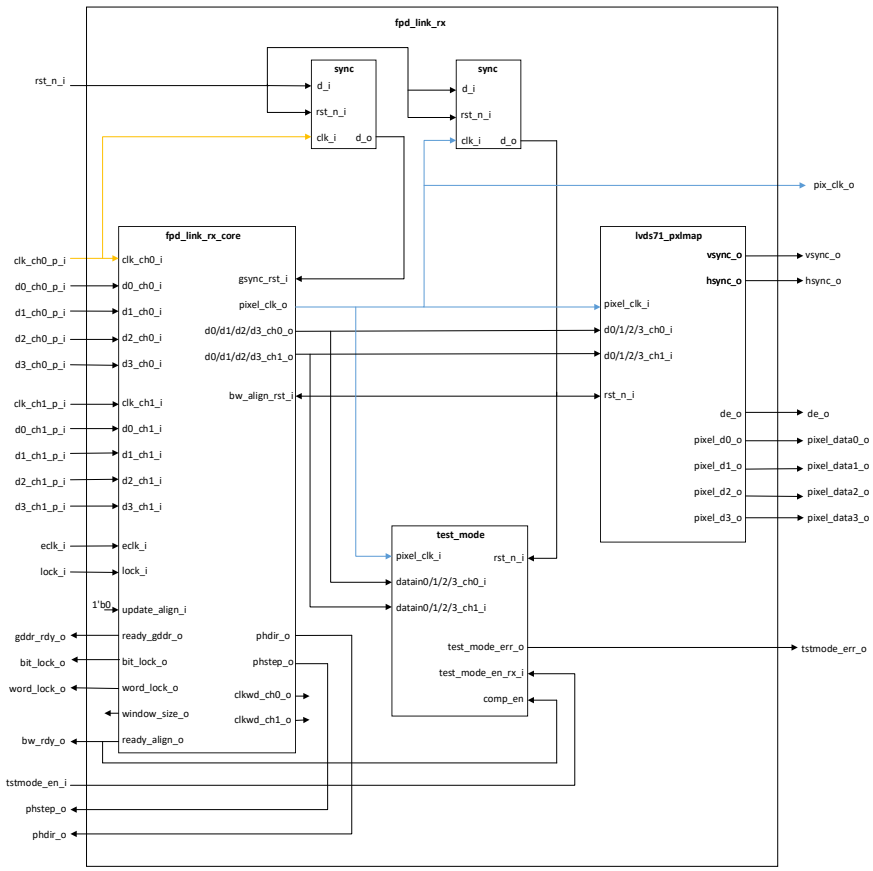

Figure 2.15. FPD-Link Rx Block Diagram without GPLL

The fpd_link_rx.v module instantiates fpd_link_rx_core, test_mode, lvds71_pxlmap,and synchronizer modules. A general purpose PLL (GPLL) is automatically instantiated inside the IP for non-Avant devices as shown in Figure 2.14. 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 is used for internal self-checking. lvds71_pxlmap is used to decode the output parallel data of fpd_link_rx_core and convert them into pixel format. Synchronizers are two-level synchronizers used to sync the system reset into different clock domains before it is used in the system.

2.7.1. FPD-LINK Rx Core

The fpd_link_rx_core module instantiates GDDR_SYNC, BW_ALIGN, several Lattice primitives, and lvds71_ddr_group.v module. 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 LVDS input clock.

LVDS data are fed to the I/O logic IDDR71 register. LVDS clock is also fed to the I/O logic IDDR71 register, as well as to PLL. 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 clock in the middle of the data valid window, alignment IP together with PLL is used to shift ECLK to normally 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 IDDR71 IP.

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

2.7.2. GPLL

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

For non-Avant devices, GPLL is instantiated inside the IP and clock-related operations are automated. For Avant devices, no GPLL is used inside the IP and the user needs to provide the required clock/s.

Below are the general guidelines for setting GPLL:

1. Set reference clock frequency equal to LVDS input clock frequency.
2. Use internal feedback path if possible.
3. Set primary clock (CLKOP) frequency equal to reference clock (x1 reference clock).
4. Set secondary clock (CLKOS) frequency to 3.5x reference clock (x3.5 reference clock). This is equivalent to Rx LVDS ECLK in Clock Domains and Clock Domain Crossing section.
5. Enable Dynamic Phase ports. IP provides control signals for dynamic phase shift to properly align the GPLL output clocks. Refer to Table 2.2 for details.
6. Set phase select to CLKOS.
7. Set Phase Load Reg to SET when applicable.
8. Enable PLL lock signal and reset.

Figure 2.16 shows the connection of the external GPLL to the IP for Avant devices.

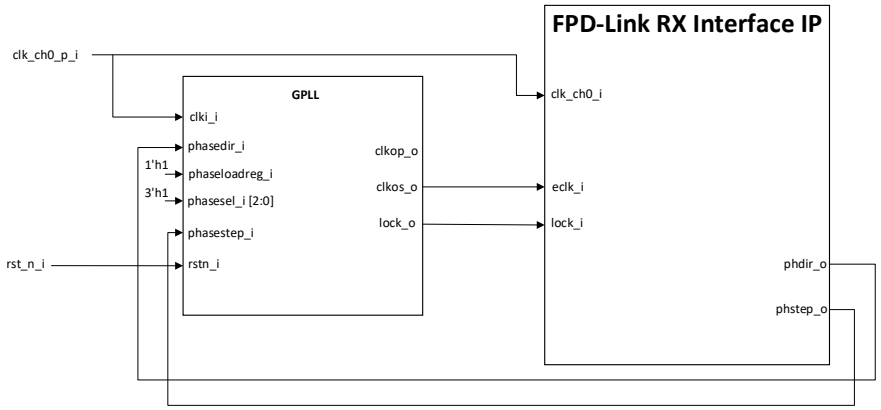

Figure 2.16. GPLL connection for Lattice Avant

2.7.3. LVDS71 Pixel Map Module

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

2.7.4. Test Mode Rx Module

The test_mode module is used to check data from FPD-link Rx before it is decoded into control and pixel data. A predefined 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 fpd_link_rx module.

The comparison of data is enabled 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 chip is powered down.

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

2.7.5. Synchronizer Module

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

2.8. Debug Mode

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

A predefined 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 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 chip 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 2.2)
   - Arranged in 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 sequence as specified in the Clock, Reset, and Initialization 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.

3. IP Generation, Simulation, and Validation

This section provides information on how to generate the OpenLDI/FPD-LINK/LVDS Receiver IP Core using the Lattice Radiant software and how to run simulation and synthesis. For more details on the Lattice Radiant software, please refer to Lattice Radiant Software 2.1 User Guide.

3.1. Generating the IP

The Lattice Radiant software allows the user to customize and generate modules and IPs and integrate them into the device's architecture. The procedure for generating the OpenLDI/FPD-LINK/LVDS Receiver IP Core in Lattice Radiant software is described below.

To generate the OpenLDI/FPD-LINK/LVDS Receiver IP Core:

1. Create a new Lattice Radiant software project or open an existing project.
2. In the **IP Catalog** tab, double-click **FPD-Link Receiver** under the **IP, Audio_Video_Image_Processing** category. The **Module/IP Block Wizard** opens as shown in Figure 3.1.
3. Enter values in the **Instance name** and **Create in** fields. Click **Next**.

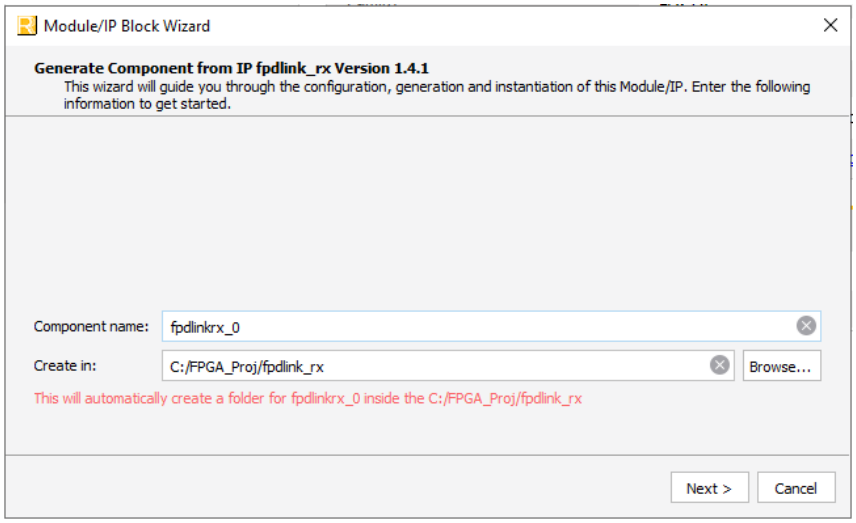

Figure 3.1. Module/IP Block Wizard

4. In the module's dialog box of the **Module/IP Block Wizard** window, customize the selected OpenLDI/FPD-LINK/LVDS Receiver IP Core using drop-down menus and check boxes. As a sample configuration, see Figure 3.2. For configuration options, see the Attribute Summary section.

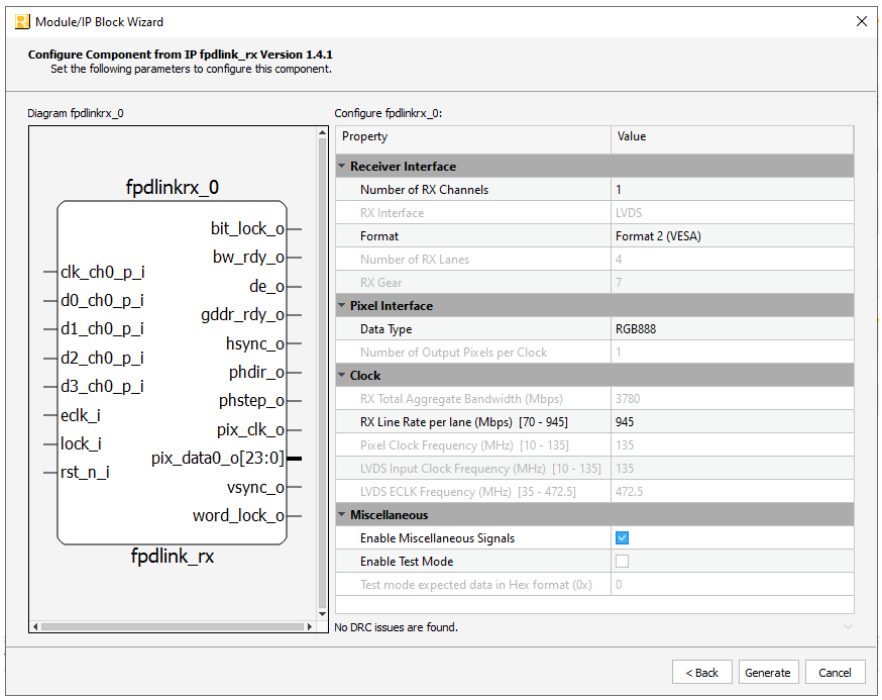

Figure 3.2. Configure User Interface of Selected OpenLDI/FPD-LINK/LVDS Receiver IP Core

5. Click **Generate**. The **Check Generating Result** dialog box opens, showing design block messages and result. See Figure 3.3.

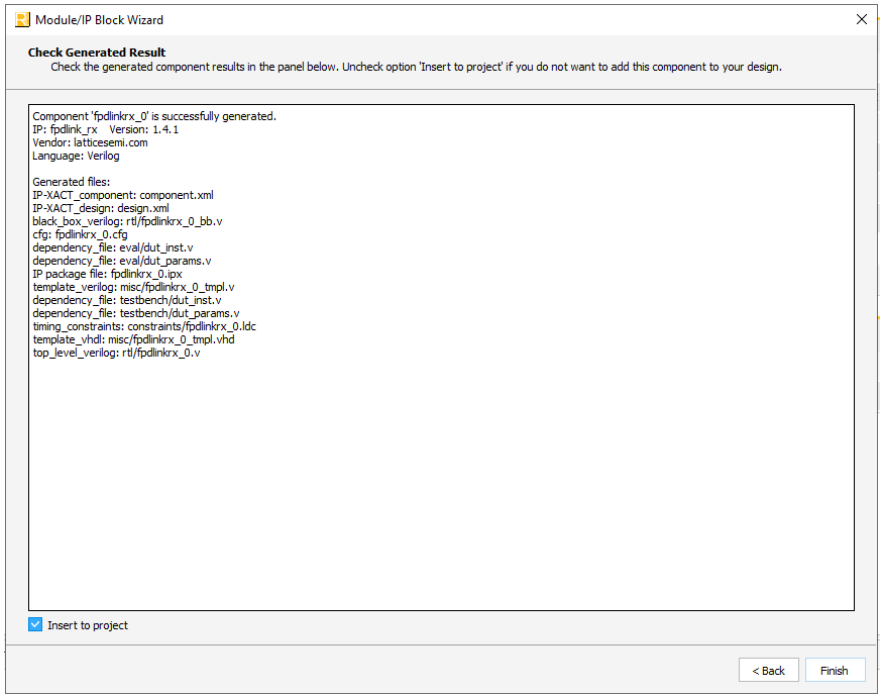

Figure 3.3. Check Generating Result

6. Click the **Finish** button. All the generated files are placed under the directory paths in the **Create in** and **Instance name** fields shown in Figure 3.1.

The generated OpenLDI/FPD-LINK/LVDS Receiver IP Core package includes the black box (<Instance Name>_bb.v) and instance templates (<Instance 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 (<Instance Name>.v) that can be used as an instantiation template for the IP core is also provided. The user can also use this top-level reference as the starting template for the top-level for their complete design. The generated files are listed in Table 3.1.

Table 3.1. Generated File List

Attribute	Description
<Instance Name>.ipx	This file contains the information on the files associated to the generated IP.
<Instance 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/<Instance Name>.v	This file provides an example RTL top file that instantiates the IP core.
rtl/<Instance Name>_bb.v	This file provides the synthesis black box.
misc/<Instance Name>_tmpl.v misc /<Instance Name>_tmpl.vhd	These files provide instance templates for the IP core.
constraints/<Instance Name>.ldc	Contains of constraints to be propagated in the IP.
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.

3.2. Constraining the IP

To ensure that the design meets its desired performance goals on the FPGA, it is the responsibility of the users to provide proper timing and physical design constraints. The content of the following IP constraint file can be added to the user design constraints:

<IP Instance_Path>/<IP Instance_Name>/eval/constraint.pdc

The above constraint file has been verified during IP evaluation with the IP instantiated directly in the top-level module. It can be modified but modifications should be made with thorough understanding of the effect of each constraint. Below are the steps on how to properly constrain the IP:

1. Copy the contents of constraint.pdc to the top-level design constraint for post-synthesis.
2. Modify the period for the create_clock constraints according to the desired value to be used in the IP.
3. Uncomment set_clock_groups constraints only when the clocks to be used in the IP are asynchronous.

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

3.3. Running Functional Simulation

Running functional simulation can be performed after the IP is generated. The following steps can be taken.

1. To run the simulation, add the top level testbench file, tb_top.v in the project as a simulation file. Click the **File** tab and select **Add** in the drop-down menu.
2. Click **Existing Simulation File** and select *<Component name>/testbench/tb_top.v*.
3. Click the  button located on the **Toolbar** to initiate the **Simulation Wizard** shown in Figure 3.4.

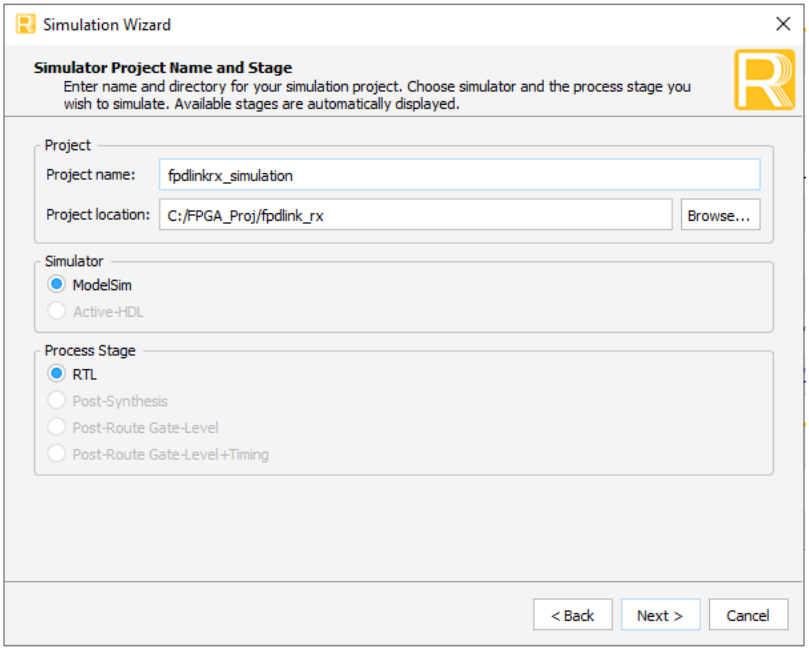

Figure 3.4. Simulation Wizard

4. Click **Next** to open the **Add and Reorder Source** window as shown in Figure 3.5.

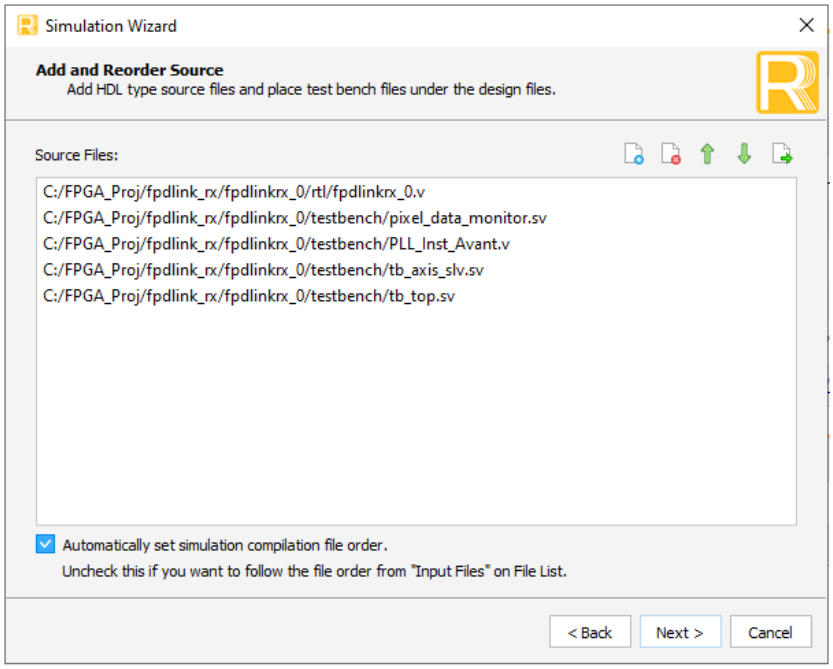

Figure 3.5. Adding and Reordering Source

Table 3.2. Testbench File List

Testbench Files	Description
testbench/tb_top.sv	Top testbench to run loopback test of generated <Instance Name>.sv
testbench/tb_pixel_data_monitor.sv	Testbench to create log files for monitoring of pixel data during transmission.
testbench/PLL_Inst_Avant.v	Testbench to include PLL module in the loopback test for Avant.
testbench/tb_axis_slv.sv	Testbench to run test on AXI4-Stream Transmitter interface.

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

The result of the simulation in our example is provided in Figure 3.6.

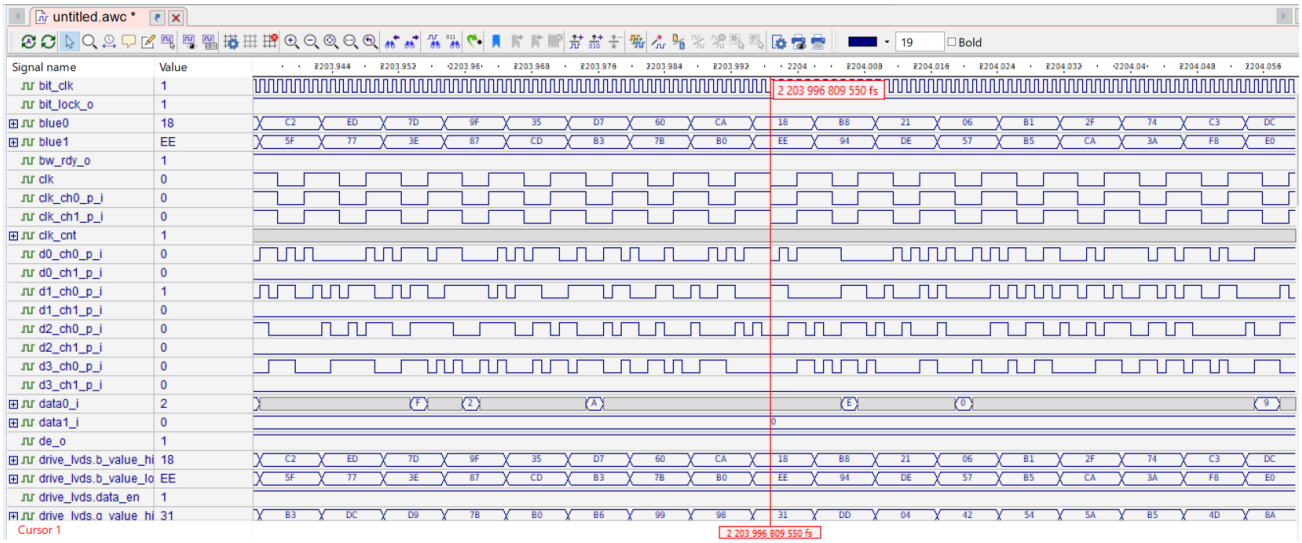

Figure 3.6. Simulation Waveform

Notes:

- It is necessary to follow the procedure above until it is fully automated in the Lattice Radiant software Suite.
- Image resolution of testbench can be configured through **parameter file <Component name>/testbench/tb_params.v**.
- 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.

3.4. IP Evaluation

The OpenLDI/FPD-LINK/LVDS Receiver IP Core supports Lattice's IP hardware evaluation capability when used with Lattice devices. This makes it possible to create versions of the IP core that operate in hardware for a limited period of time (approximately four hours) without requiring the purchase of an IP license. It may also be used to evaluate the core in hardware in user-defined designs. The hardware evaluation capability may be enabled/disabled in the Strategy dialog box. It is enabled by default. To change this setting, go to **Project > Active Strategy > LSE/Synplify Pro Settings**.

4. Ordering Part Number

The Ordering Part Number (OPN) for this IP Core are listed below.

Table 4.1. Ordering Part Numbers

OPN	Device Family	License Type
FPD-RX-CNX-UT	CrossLink-NX	Multi-site Perpetual License
FPD-RX-CNX-US	CrossLink-NX	Single Machine Annual License
FPD-RX-CTNX-UT	Certus-NX	Multi-site Perpetual License
FPD-RX-CTNX-US	Certus-NX	Single Machine Annual License
FPD-RX-CPNX-US	CertusPro-NX	Multi-site Perpetual License
FPD-RX-CPNX-UT	CertusPro-NX	Single Machine Annual License
FPD-RX-XO5-UT	MachXO5-NX	Multi-site Perpetual License
FPD-RX-XO5-US	MachXO5-NX	Single Machine Annual License
FPD-RX-AVE-UT	Avant-AT-E	Multi-site Perpetual License
FPD-RX-AVE-US	Avant-AT-E	Single Machine Annual License
FPD-RX-AVG-UT	Avant-AT-G	Multi-site Perpetual License
FPD-RX-AVG-US	Avant-AT-G	Single Machine Annual License
FPD-RX-AVX-UT	Avant-AT-X	Multi-site Perpetual License
FPD-RX-AVX-US	Avant-AT-X	Single Machine Annual License

Appendix A. Resource Utilization

Table A.1 and Table A.2 show the resource utilization of the OpenLDI/FPD-LINK/LVDS Receiver Core for LIFCL-40-9BG400I device using Lattice Synthesis Engine and LAV-AT-E70-3LFG1156C using Synplify Pro of the Lattice Radiant software respectively. Default configuration is used, and some attributes are changed from the default value to show the effect on the resource utilization.

Table A.1. Resource Utilization

Configuration	Clk Fmax (MHz)[1]	Slice Registers	LUTs[2]	EBRs
Default	196.309	120	214	0
Number of RX Channels = 2, Others = Default	188.111	144	214	0
Data Type = RGB666, Others = Default	194.932	114	216	0
Number of RX Channels = 2, *Data Type* = RGB666, Others = Default	187.864	132	218	0

Notes:

1. Fmax is generated when the FPGA design only contains OpenLDI/FPD-LINK/LVDS Receiver Core and the target frequency is 135 MHz. These values may be reduced when user logic is added to the FPGA design.
2. The *distributed RAM* utilization is accounted for in the total LUT4s utilization. The actual LUT4 utilization is distribution among *logic*, *distributed RAM*, and *ripple logic*.

Table A.2. Resource Utilization using Lattice Avant

Configuration	Clk Fmax (MHz)[1]	Registers	LUTs[2]	EBRs
Default	184.400	129	219	0
Number of RX Channels = 2, Others = Default	200.00	153	228	0
Data Type = RGB666, Others = Default	184.400	123	219	0
Number of RX Channels = 2, *Data Type* = RGB666, Others = Default	200.00	141	228	0

Appendix B. Limitations

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.

References

- Lattice Radiant Timing Constraints Methodology (FPGA-AN-02059)
- CrossLink-NX web page
- Certus-NX web page
- CertusPro-NX web page
- MachXO5-NX web page
- Avant-E web page
- Avant-G web page
- Avant-X web page
- Lattice Radiant Software 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

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 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.

www.latticesemi.com