Byte-to-Pixel Converter IP

IP Version: v1.9.2

User Guide

FPGA-IPUG-02079-2.2

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

Figures

Tables

Abbreviations in This Document

A list of abbreviations used in this document.

Abbreviations	Definition
AXI	Advance eXtensible Interface
BPC	Bits per Color
BPP	Bits per Pixel
CSI-2	Camera Serial Interface-2
CSR	Control and Status Registers
CPP	Colors per Pixel
DSI	Display Serial Interface
EBR	Embedded Block RAM
EOL	End of Line
FIFO	First In, First Out
FPGA	Field-Programmable Gate Array
I/O	Input/Output
IP	Intellectual Property
LUT	Look-Up Table
LSB	Least Significant Bit
MSB	Most Significant Bit
RAM	Random Access Memory
RGB	Red Green Blue
RO	Read Only
RTL	Register Transfer Level
Rx	Receiver
RW	Read/Write
SOF	Start of Frame
Tx	Transmitter
UVSI	Unified Video Streaming Interface
W1C	Write 1 to Clear

1. Introduction

1.1. Overview of the IP

Lattice Semiconductor Byte-to-Pixel Converter IP converts CSI-2/DSI standard-based video payload packets from D-PHY Receiver Module output to pixel format. In addition, Byte-to-Pixel Converter IP generates camera and video control signals in the pixel domain based on CSI-2 or DSI synchronization packets.

Figure 1.1 shows the Byte-to-Pixel Converter IP accepts CSI-2/DSI standard-based video payload packets and generates Pixel Format output.

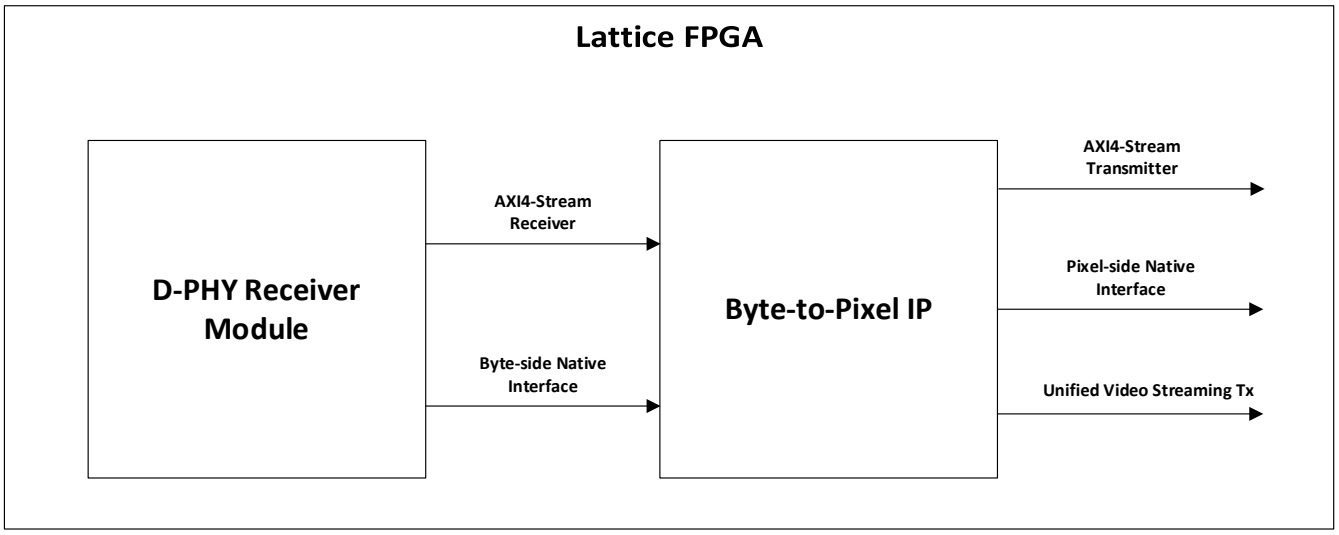

Figure 1.1. Byte-to-Pixel Convertor IP General Diagram

1.2. Quick Facts

Table 1.1. Summary of the Byte-to-Pixel Converter IP

IP Requirements	Supported FPGA Family	CrossLink™-NX, Certus™-NX, CertusPro™-NX, MachXO5™-NX, Lattice Avant™, and Certus-N2
	IP Changes[1]	For a list of changes to the IP, refer to the Byte-to-Pixel Converter IP Release Notes (FPGA-RN-02019).
Resource Utilization	Supported User Interface	Native Interface, AXI4-Stream Interface
	Resources	Refer to Appendix A. Resource Utilization
Design Tool Support	Lattice Implementation	IP Core v1.9.2 – Lattice Radiant Software 2025.2
	Synthesis	Lattice Synthesis Engine (LSE), Synopsys Synplify Pro® for Lattice
	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

The table below lists selected IP configurations. For a full overview of all supported features of this IP, refer to the Features section.

Table 1.2. Byte-to-Pixel Converter IP Support Readiness

Device Family	Data Type	Rx Interface	No. of Rx Lanes	Rx Gear	No. of Output Pixel Lanes	Camera/ Display Control Polarity	Byte Clock Freq. (MHz)	Pixel Clock Freq. (MHz)	Radiant Timing Model	Hardware Validated
Avant, Certus-N2	RAW10	CSI-2	1	8	1	Positive	187.5	150	Preliminary	No
	RGB888	DSI	1	8	1	Negative	187.5	62.5	Preliminary	No
	RAW12	CSI-2	1	8	1	Positive	187.5	125	Preliminary	No
	RAW8	CSI-2	2	8	2	Positive	187.5	187.5	Preliminary	No
	RAW16	CSI-2	4	8	2	Positive	187.5	187.5	Preliminary	No
	RGB565	CSI-2	4	8	2	Positive	187.5	187.5	Preliminary	No
	RGB565	CSI-2	2	16	2	Positive	156.25	156.25	Preliminary	No
	RAW12	CSI-2	4	8	4	Positive	187.5	125	Preliminary	No
	RGB888	DSI	4	16	4	Positive	150	100	Preliminary	No
	RAW14	CSI-2	2	8	2	Positive	175	100	Preliminary	No
	RGB666	DSI	2	8	1	Positive	112.5	100	Preliminary	No
	RAW12	CSI-2	2	16	2	Positive	150	200	Preliminary	No
CrossLink-NX, CertusPro-NX	RAW10	CSI-2	1	8	1	Positive	187.5	150	Final	Yes
	RGB888	DSI	1	8	1	Negative	187.5	62.5	Final	Yes
	RAW12	CSI-2	1	8	1	Positive	187.5	125	Final	Yes
	RAW8	CSI-2	2	8	2	Positive	187.5	187.5	Final	Yes
	RAW16	CSI-2	4	8	2	Positive	187.5	187.5	Final	Yes
	RGB565	CSI-2	4	8	2	Positive	187.5	187.5	Final	Yes
	RGB565	CSI-2	2	16	2	Positive	156.25	156.25	Final	Yes
	RAW12	CSI-2	4	8	4	Positive	187.5	125	Final	Yes
	RGB888	DSI	4	16	4	Positive	150	100	Final	Yes
	RAW14	CSI-2	2	8	2	Positive	175	100	Final	Yes
	RGB666	DSI	2	8	1	Positive	112.5	100	Final	Yes
	RAW12	CSI-2	2	16	2	Positive	150	200	Final	Yes
Certus-NX, MachXO5-NX	RAW10	CSI-2	1	8	1	Positive	187.5	150	Final	No
	RGB888	DSI	1	8	1	Negative	187.5	62.5	Final	No
	RAW12	CSI-2	1	8	1	Positive	187.5	125	Final	No
	RAW8	CSI-2	2	8	2	Positive	187.5	187.5	Final	No
	RAW16	CSI-2	4	8	2	Positive	187.5	187.5	Final	No
	RGB565	CSI-2	4	8	2	Positive	187.5	187.5	Final	No
	RGB565	CSI-2	2	16	2	Positive	156.25	156.25	Final	No
	RAW12	CSI-2	4	8	4	Positive	187.5	125	Final	No
	RGB888	DSI	4	16	4	Positive	150	100	Final	No
	RAW14	CSI-2	2	8	2	Positive	175	100	Final	No
	RGB666	DSI	2	8	1	Positive	112.5	100	Final	No
	RAW12	CSI-2	2	16	2	Positive	150	200	Final	No

1.4. Features

Key features of the Byte-to-Pixel Converter IP include:

- MIPI DSI compatible video formats
- MIPI CSI-2 compatible video formats
- 1-, 2-, or 4-lane inputs
- 8-bit (gear 8) or 16-bit (gear 16) inputs per lane
- 1, 2, or 4 output pixels per pixel clock cycle
- Burst mode, non-burst mode with sync events and non-burst mode with sync pulse
- Supports AXI4-Stream Transmitter and Receiver interfaces
- Supports Unified Video Streaming interface in compliance with the AXI4-Stream protocol
- Supports AXI4-Lite interface for register access.

1.5. Licensing and Ordering Information

The Byte-to-Pixel Converter 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. Resource Utilization for the minimum required resource to instantiate this IP.

1.8. Naming Conventions

1.8.1. Nomenclature

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

1.8.2. Signal Names

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

1.8.3. 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 Byte-to-Pixel Converter IP is used to convert a D-PHY CSI-2/DSI standard-based byte data stream to a standard-pixel data format. It acts as a conversion bridge between the CSI-2/DSI standard-based input byte data stream and the pixel-format output data stream. The optional AXI4-Stream of the Byte Domain Inputs/Outputs is used for receiving video payload packets.

Figure 2.1 shows the general block diagram of the Byte-to-Pixel Converter IP. 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.

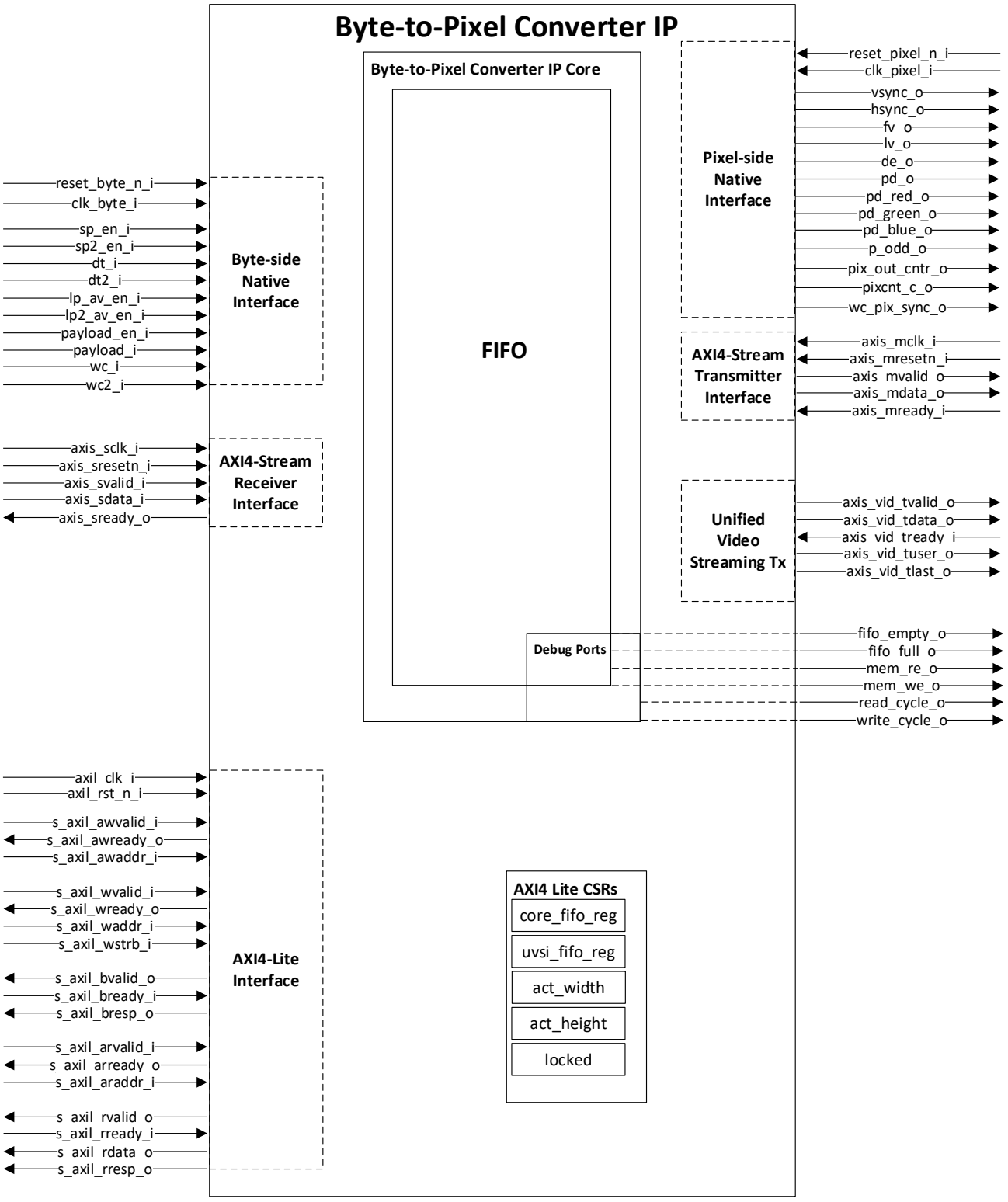

Figure 2.1. Byte-to-Pixel IP Block Diagram

Figure 2.2 shows the functional block diagram of the Byte-to-Pixel Converter IP.

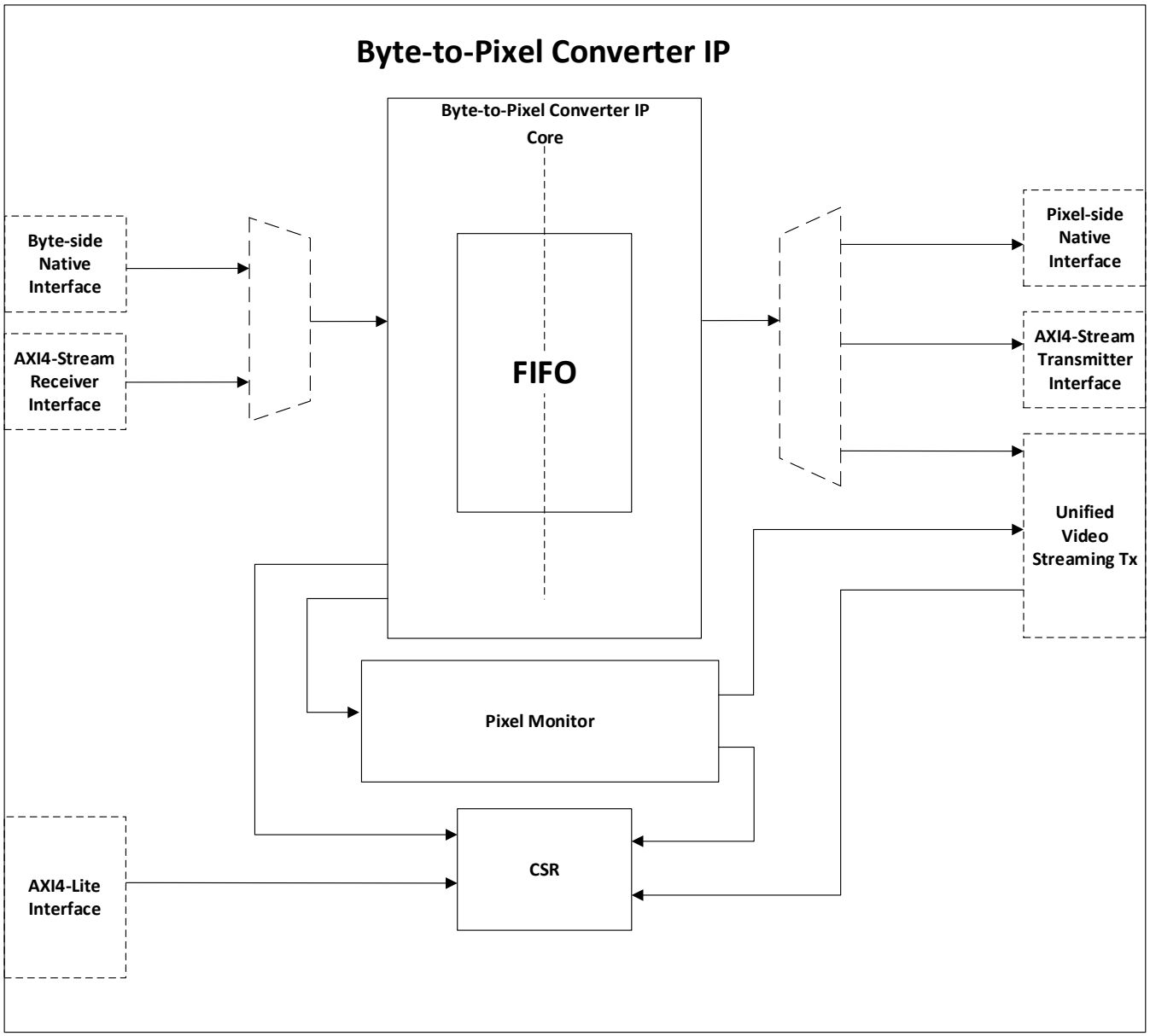

Figure 2.2. Functional Block Diagram

The Byte-to-Pixel Converter IP includes the following layers:

- Byte-to-Pixel Converter IP Core
- Byte-side Native Interface[1] (optional)
- AXI4-Stream Receiver Interface (optional)
- Pixel-side Native Interface[1] (optional)
- AXI4-Stream Transmitter Interface (optional)
- FIFO module
- Debug Interface[2] (optional)
- Unified Video Streaming Tx (optional)
- AXI4-Lite Interface[2] (optional)
- Pixel Monitor Module
- Control and Status Register (CSR) module

Notes:

1. The Native Interface is disabled when its corresponding AXI stream interface is enabled, and vice versa.
2. Debug ports fifo_full_o and fifo_empty_o are remapped as status registers when the AXI4-Lite interface is enabled.

2.2. Clocking

The Byte-to-Pixel Converter IP functions across multiple clock domains, specifically the byte clock domain, pixel clock domain, and AXI4-Lite clock domain. Figure 2.3 shows these different clock domains.

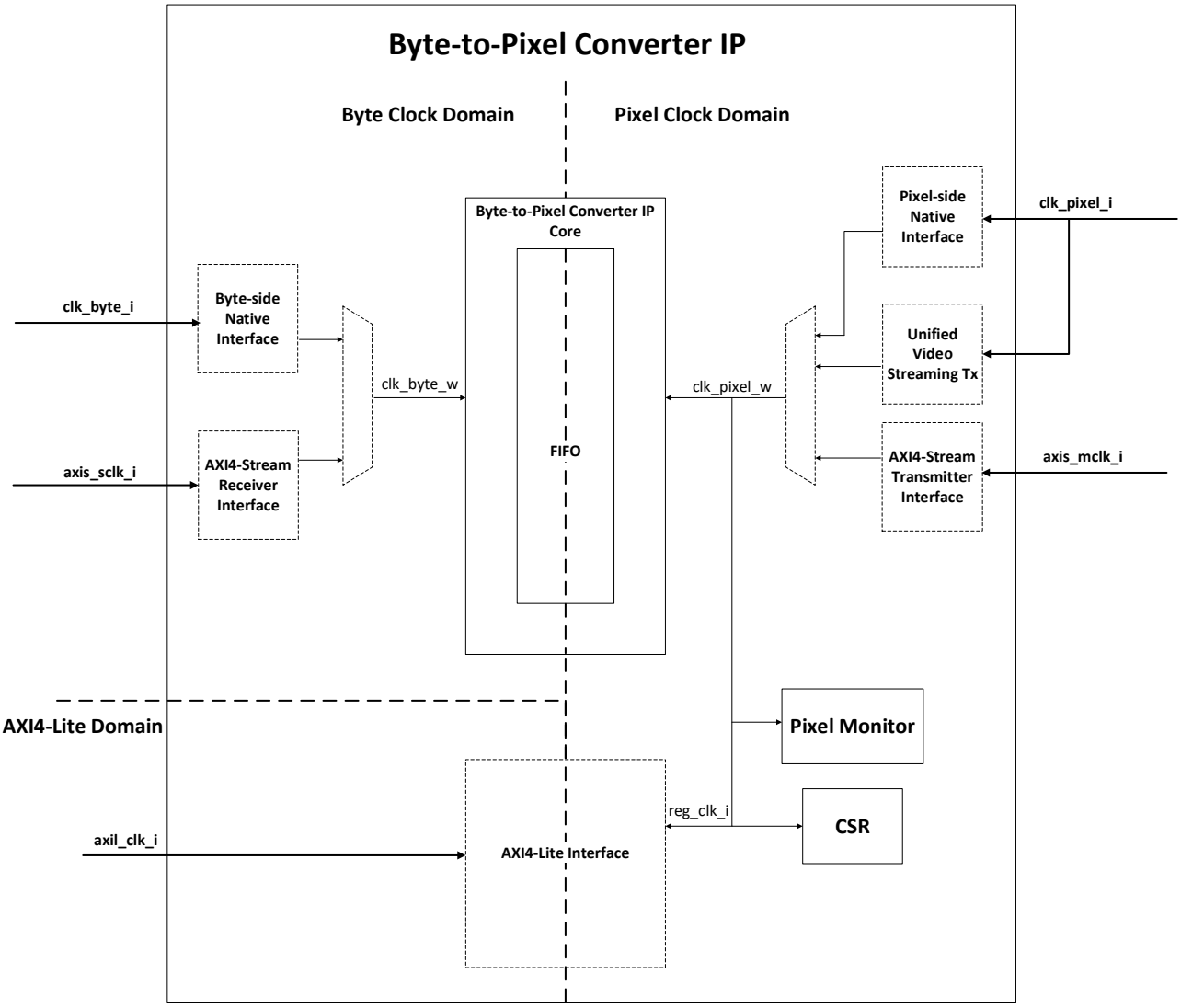

Figure 2.3. Clock Domain Crossing Block Diagram

Refer to the Clock Interface section for more information on the Clock Signal Interface and its requirement.

2.3. Reset

The Byte-to-Pixel Converter has different reset signals for different domains. The active reset per domain depends on the interface selected. Refer to the Reset Interface section for more details. Figure 2.4 shows the reset domain in the Byte-to-Pixel Converter IP.

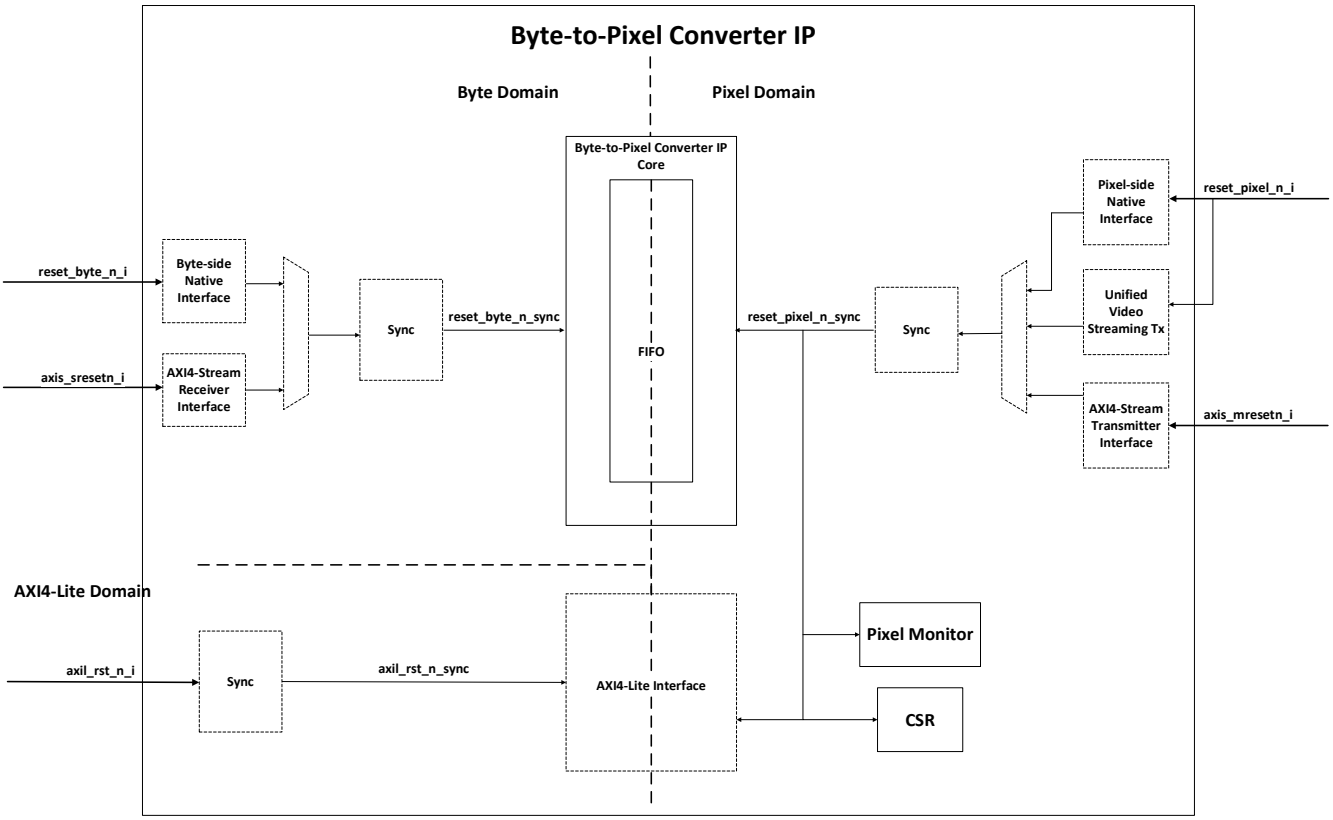

Figure 2.4. Reset Domain Block Diagram

2.3.1. Reset Overview

All reset in the IP are asynchronous active low signal with synchronous release. Synchronizers are used to sync the reset to different clock domains. The active system reset input are either reset_byte_n_i or axis_sresetn_i for Byte Domain and reset_pixel_n_i for Pixel Domain. A separate axil_rst_n_i signal is used to reset the AXI4-Lite Interface. CSR can be reset through the system reset or active reset in the pixel domain.

2.3.2. Initialization and Reset Sequence

Follow the initialization and reset sequence below:

1. Assert an active low system reset for at least three cycles of the slower clock, either clk_byte_i or clk_pixel_i. For AXI4-Lite Interface, assert active low reset signal axil_rst_n_i for at least three clock cycles of the AXI4-Lite Interface Clock. It is expected that the input clock is stable after reset. Clock synchronization starts immediately after the release of system reset.
2. The Byte-to-Pixel Converter is ready to process data after the reset.

2.4. User Interfaces

Table 2.1 lists the available user interface and protocols used on the Byte-to-Pixel Converter IP.

Table 2.1. User Interfaces and Supported Protocols

User Interface	Supported Protocols	Description
Byte/Pixel-side Native Interface	MIPI CSI-2 (Camera Serial Interface 2)	MIPI CSI-2 is a widely adopted, high speed protocol for transmission of still and video images from image sensors to application processors.
	DSI (Display Serial Interface)	DSI protocol is commonly targeted at LCD and similar display technologies. It defines a serial bus communication protocol between the host (source of the image data), and the device which is the destination.
AXI4-Stream Receiver Interface	AXI4-Stream Protocol	Receives the payload data (byte data with data type and word count).
AXI4-Stream Transmitter Interface	AXI4-Stream Protocol	Transmits pixel data. This is a legacy interface retained for backward compatibility. It is recommended to use the Unified Video Streaming Tx Interface.
Unified Video Streaming Tx Interface	AXI4-Stream Protocol	Transmits pixel data.
Control	AXI4-Lite Interface	Allows access to the control and status registers of the Byte-to-Pixel Converter IP.

2.4.1. Native Interfaces

This section contains operational timing diagrams applicable to the Byte-to-Pixel IP Native interfaces. Figure 2.5 shows the timing between input and output for DSI.

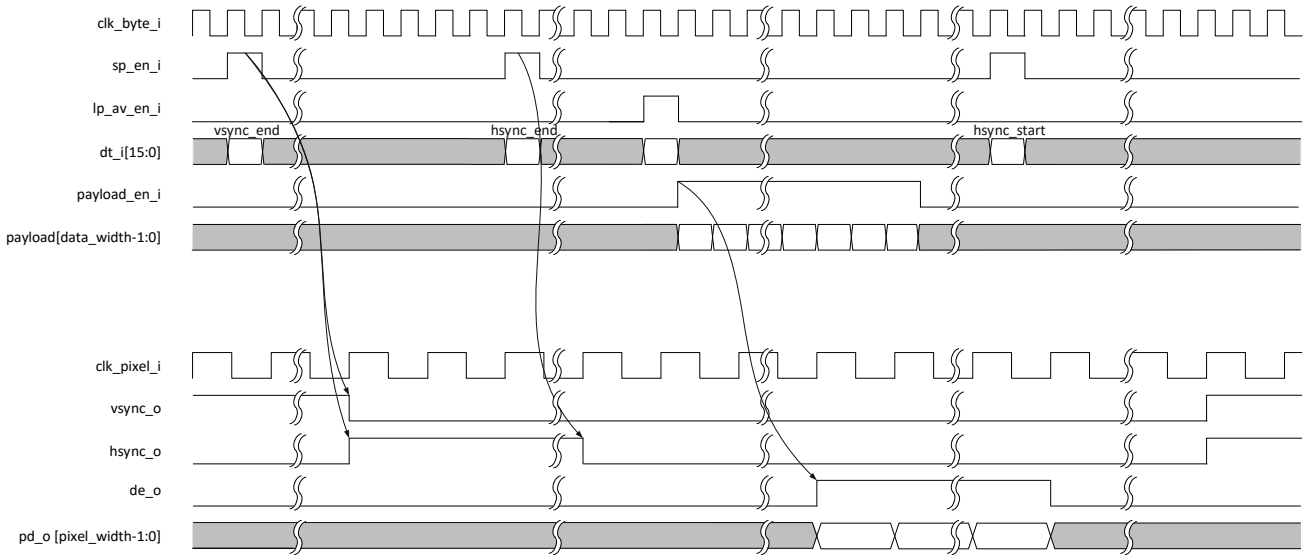

Figure 2.5. Timing Diagram Between Inputs and Outputs for DSI

Reception of a VSYNC start packet triggers the assertion of both hsync_o and vsync_o signals. VSYNC end packets, on the other hand, trigger the deassertion of the vsync_o signal and the assertion of hsync_o signal.

Figure 2.6 shows the timing between input and output for CSI-2.

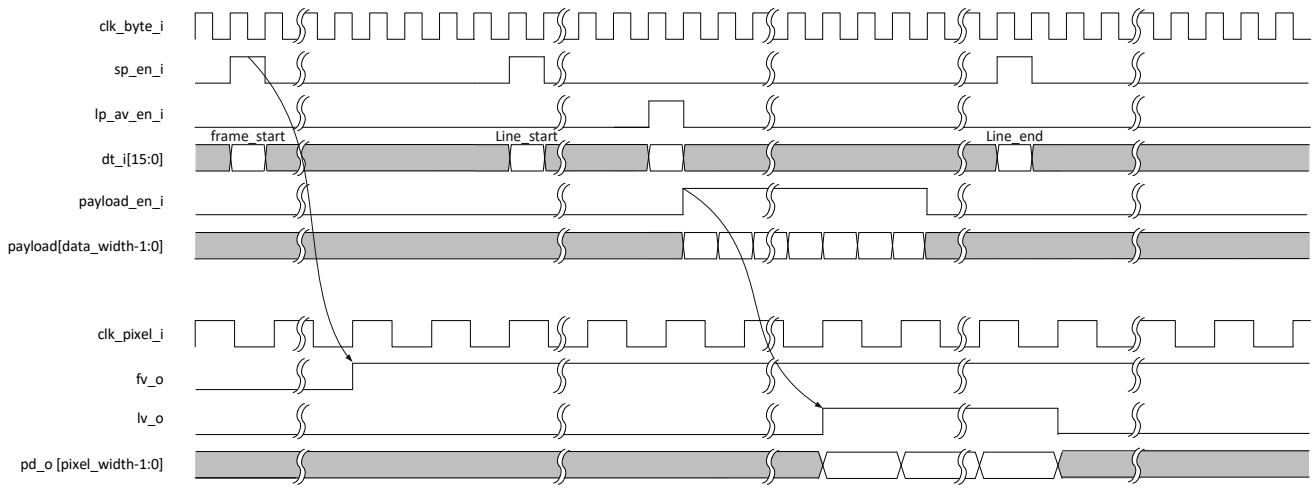

Figure 2.6. Timing Diagram Between Inputs and Outputs for CSI-2

The behavior of the output synchronization signals (frame and line valid for CSI-2, and VSYNC and HSYNC for DSI) depend on the reception of the corresponding short packets. Due to the crossing of clock domains, pulse width and intervals between pulses may vary.

The pixel data is buffered and processed differently than the sync packets. Because of this, the pixel data might come out later that the sync signals. For the DSI protocol, the DSI Sync Packet Delay attribute may be increased to reduce the skew between the sync signals and the pixel data.

Figure 2.7 shows the timing diagram of the interface at the receiver side. Signal lp_av_en_i must be asserted first before receiving payload_en_o. The signals dt_i, vc_i and wc_i must be valid with the assertion of the lp_av_en_i. The assertion of lp_av_en_i should only occur if dt_i matches with the data type configured in the IP, which is managed by you.

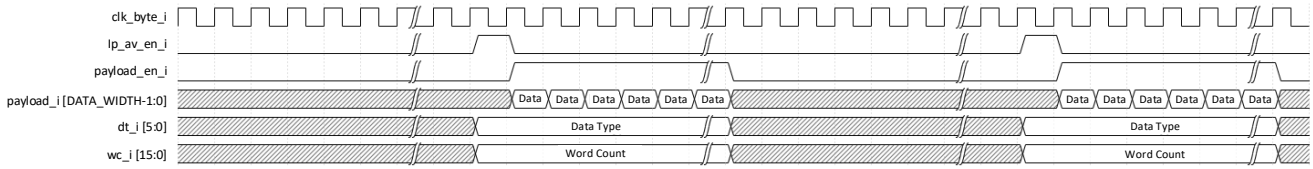

Figure 2.7. Input Timing Diagram for Byte Domain

The output timing from a DSI input is shown in Figure 2.8 while the timing from a CSI-2 input is shown in Figure 2.9.

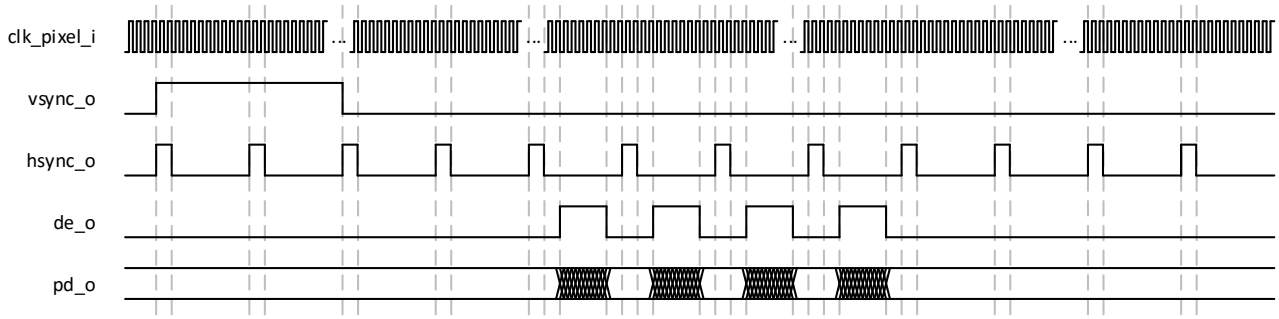

Figure 2.8. Output Timing Diagram from a DSI Input

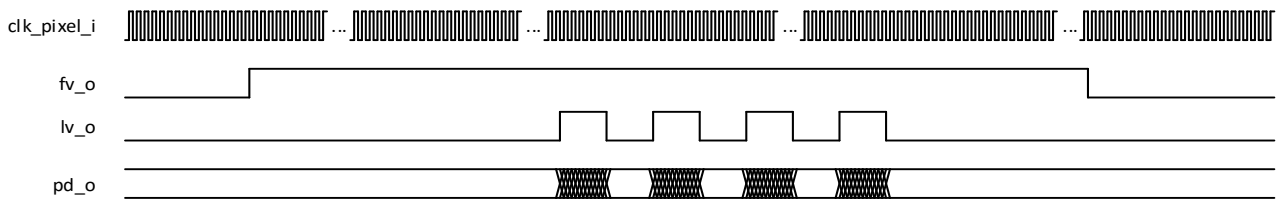

Figure 2.9. Output Timing Diagram from a CSI-2 Input

Table 2.2 shows the pixel data bus width of each data type. Bus width of pd_o is dependent on the following values.

Table 2.2. Pixel Data Bus Width for Supported Data Types

Data Type	Pixel Data Bus Width
RGB888	24
RGB565	16
RGB666	18
RGB666, Loosely packed	18
RAW8	8
RAW10	10
RAW12	12
RAW14	14
RAW16	16
Legacy YUV420 8-bit	8
YUV420 8-bit CSPS	8
YUV420 8-bit	8
YUV420 10-bit CSPS	10
YUV420 10-bit	10
YUV422 8-bit	8
YUV422 10-bit	10
YCbCr422 16-bit	16
YCbCr422 Loosely packed 20-bit	20
YCbCr422 24-bit	24

When the selected data type for either DSI or CSI-2 protocol belongs to RGB, each color component from pd_o is replicated and remapped to their RGB output port counterparts. These are pd_red_o for R, pd_green_o for G, and pd_blue_o for B. These output ports are only available when *Pixel-Side Transmitter Interface == Native Interface*.

Figure 2.10, Figure 2.11, and Figure 2.12 show how each color is placed in their respective output bus for different number of output pixel lanes. Colors from Lane 1 is put in the LSB-placement, followed by colors from other lanes in order of Lane 1, Lane 2, Lane 3, and Lane 4. Each bus width is dependent on the bits per color component multiplied by the number of output pixel lanes configured. The color arrangement of the input byte data must adhere to the CSI-2 and DSI specification. For CSI-2 format, the color arrangement must be in the RGB order with B as the least significant byte. For DSI format, the color arrangement must be in the BGR order with R as the least significant byte.

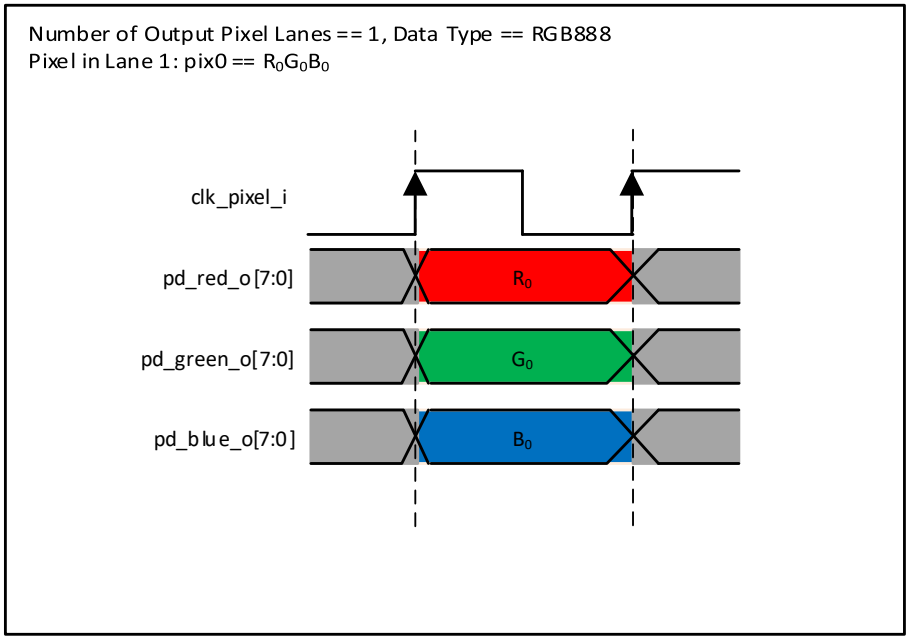

Figure 2.10. RGB88 Color Remapping with *Number of Output Pixel Lanes == 1*

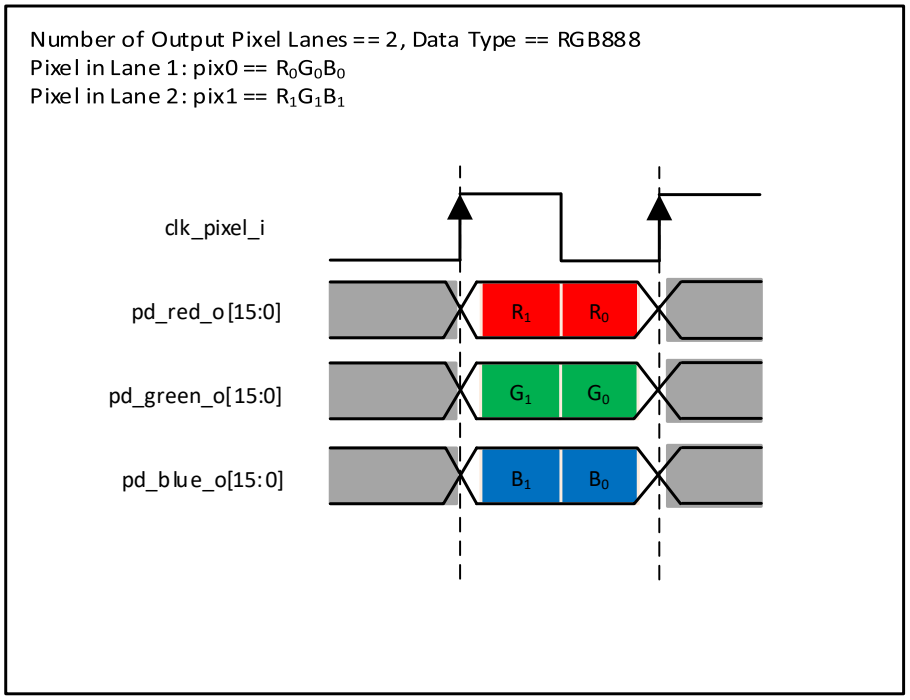

Figure 2.11. RGB888 Color Remapping with *Number of Output Pixel Lanes == 2*

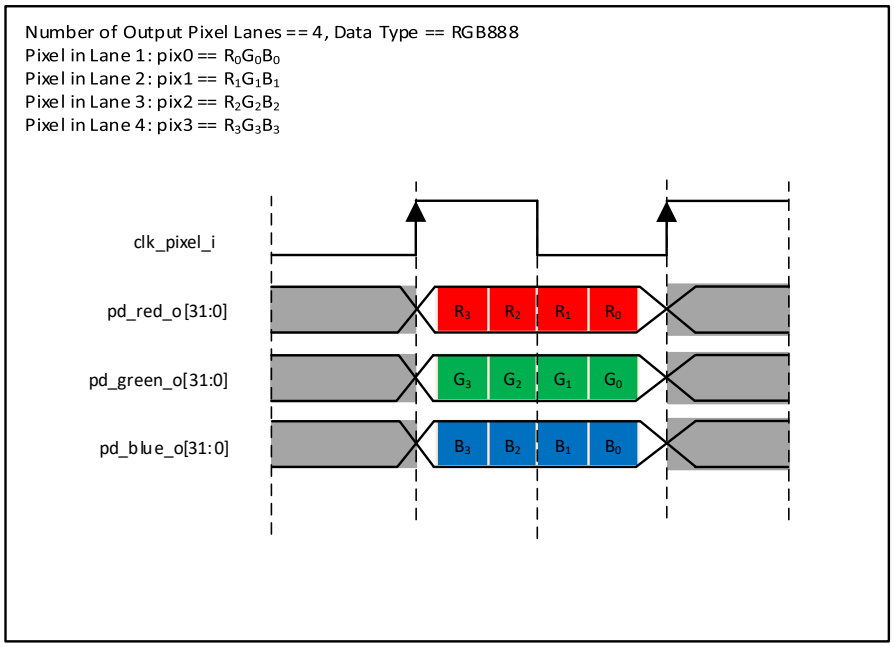

Figure 2.12. RGB888 Color Remaining with *Number of Output Pixel Lanes == 4*

2.4.2. AXI4-Stream Receiver Interface

The AXI4-Stream Receiver provides transmission of payload packets to the Byte-to-Pixel Converter module. The axis_sready_o is always set to *1*. When axis_svalid_i is asserted to *1*, then axis_sdata_i receives the payload video stream. Figure 2.13 shows data format when AXI4-Stream Receiver interface is enabled.

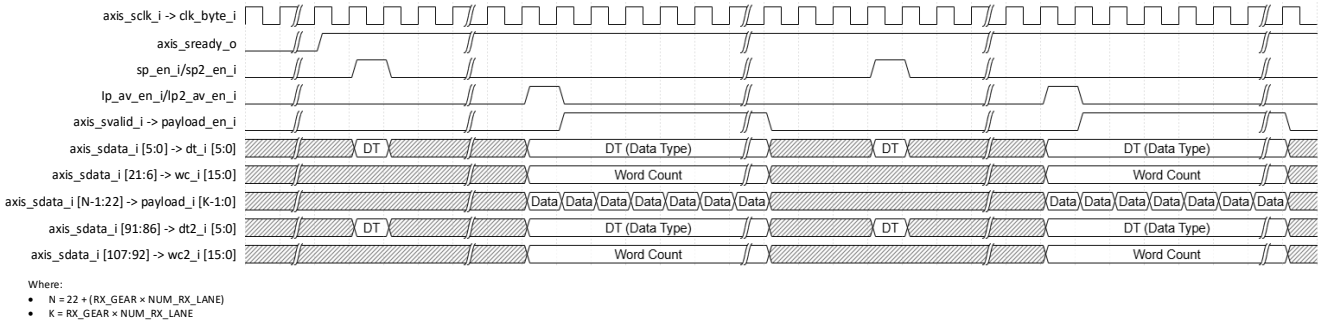

Figure 2.13. AXI4-Stream Receiver Interface Diagram

If the AXI4-Stream Receiver is not enabled, the following signals turn to top-level input signals:

- payload_en_i
- payload_i
- dt_i
- dt2_i
- wc_i
- wc2_i

2.4.3. AXI4-Stream Transmitter Interface

The AXI4-Stream Transmitter provides transmission of data converted to pixel format. Figure 2.14 shows data format when AXI4-Stream Transmitter interface is enabled.

For DSI and CSI-2 modes, the output data is the concatenation of two signals {p_odd_o, pd_o}, respectively, in that order. As the p_odd_o is a two-bit signal, the total width of AXI4 stream data (axis_mdata_o) is equal to 2 + pixel_width × number_of_pixels. Data valid (axis_mvalid_o) becomes active when de_o/lv_o signals become *1*. The signals fv_o for CSI-2 and hsync_o/vsync_o for DSI indicate the frame start and frame end of the output data.

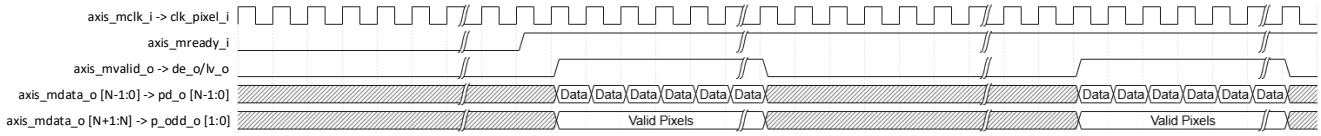

Figure 2.14. AXI4-Stream Transmitter Interface Diagram

If the AXI4-Stream Transmitter is disabled and Native interface is selected, the following signals turn to top-level output signals. These signals are not present when Unified Video Streaming Tx is enabled:

- de_o[1]
- lv_o[2]
- pd_o
- p_odd_o

Notes:

1. Data Enable signal when RX Interface == DSI.
2. Data Enable signal when RX Interface == CSI-2.

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_o/lv_o and pd_o/pd_blue_o, pd_green_o, pd_red_o 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 and has a FIFO depth of 256.

The data mapping of the Unified Video Streaming Tx ports is shown the table below.

Table 2.3. Data Mapping of Unified Video Streaming Tx Ports

Unified Video Streaming Tx Port	Equivalent Signal
axis_vid_tvalid_o	de_o/lv_o
axis_vid_tdata_o	pd_o/pd_blue_o, pd_green_o, pd_red_o (RGB data types)
axis_vid_tuser_o[0]	start_of_frame
axis_vid_tuser_o[1]	Reserved
axis_vid_tlast_o	end_of_line

Figure 2.15 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 on the lock signal.

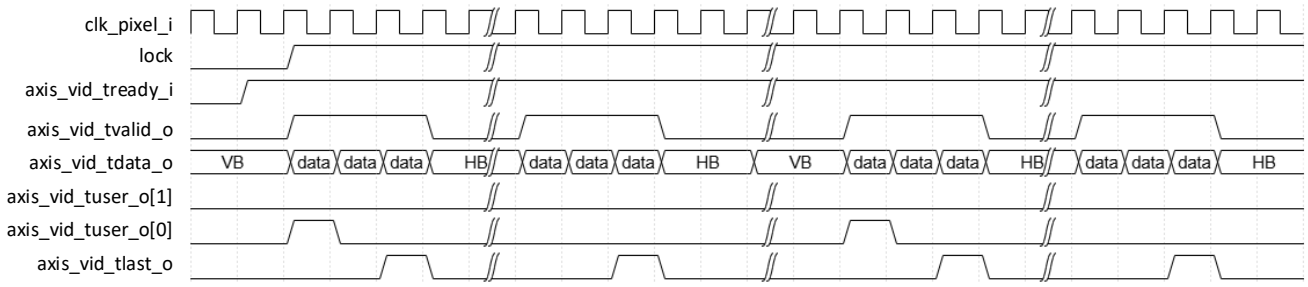

Figure 2.15. Unified Video Streaming Tx Timing Diagram

Figure 2.16 shows the behavior of axis_vid_tdata_o and axis_vid_tuser_o[0] when axis_vid_tready_i is temporarily de-asserted.

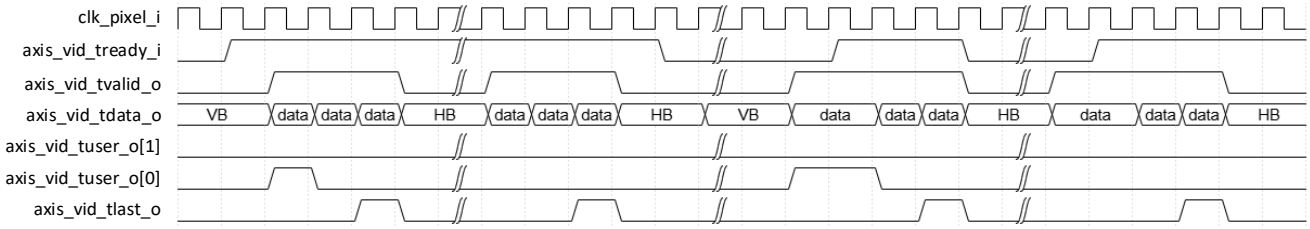

Figure 2.16. Unified Video Streaming Tx Timing Diagram with De-assertion of TREADY

The size of axis_vid_tdata_o varies depending on TD_WD_OUT and *Number of Output Pixel Lanes*:

Width of axis_vid_tdata_o = TD_WD_OUT × *Number of Output Pixel Lanes.*

Where:

- TD_WD_OUT = ceil (BPP / 8) × 8
  TD_WD_OUT is dependent on the following attributes:
  - CPP: Colors per pixel. Supported CPP values are 1, 2, and 3.
    - For Legacy YUV420 8-bit and RGB data types: CPP is 3.
    - For YUV422 8-bit/10-bit and YCbCr data types: CPP is 2.
    - For RAW data types: CPP is 1.
  - BPC: Bits per color, the minimum width of each color component is 1 byte (8 bits).
  - BPP: Bits per pixel, it is calculated as BPP = CPP × BPC.
- *Number of Output Pixel Lanes* can be 1, 2, or 4.

Remapping of pixel data is done through the following rules:

- Pixels comprise between 1, 2, and 3 color components.
- Minimum width of each color component is 1 byte. For data types with color component that is less than 1 byte, their LSBs are padded with 0s until byte alignment is achieved. Refer to Figure 2.18 and Figure 2.19 for RGB888 and RGB666 data type examples.
- Map video packet data across axis_vid_tdata_o with the LSB of the first color component of the first pixel in bit 0.
- Byte align each pixel when mapping multiple pixels in parallel (*Number of Output Pixel Lanes > 1*).
- When a pixel does not perfectly fill a given number of bytes, fill the MSBs with undefined data. Refer to Figure 2.17 for illustration of the MSB padding with RAW10 data type. Figure 2.20 and Figure 2.21 show the tdata mapping of YCbCr and Legacy YUV420 8-bit data types respectively.
- For configurations with number of pixels per active line not multiple of *Number of Output Pixel Lanes*, deduce the exact length of the video packet from the image dimension in the CSR.

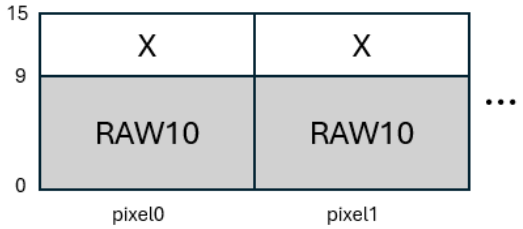

Figure 2.17. Illustration of MSB padding for RAW10 Data Type

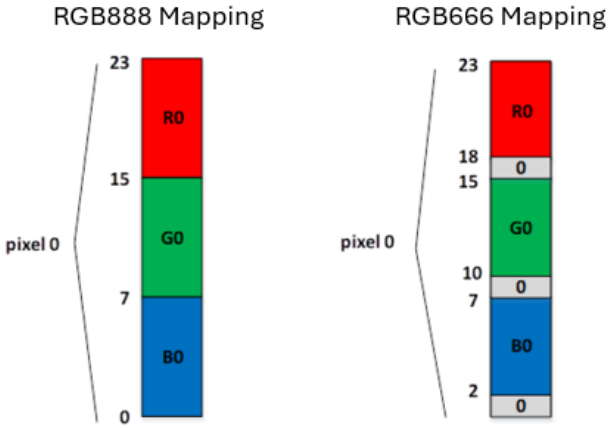

Figure 2.18. Unified Video Streaming Tx TDATA Mapping of RGB888 and RGB666 when Number of Output Pixel Lanes = 1

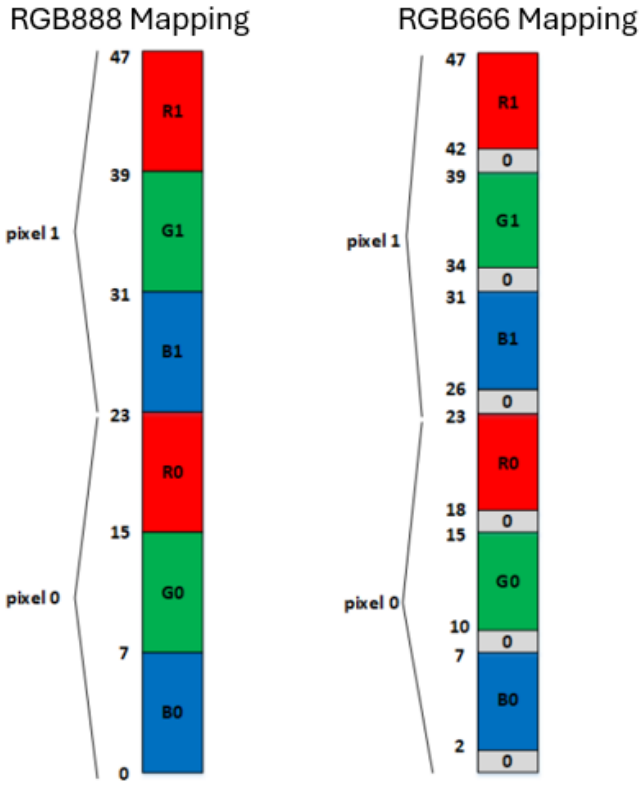

Figure 2.19. Unified Video Streaming Tx TDATA Mapping of RGB888 and RGB666 when Number of Output Pixel Lanes = 2

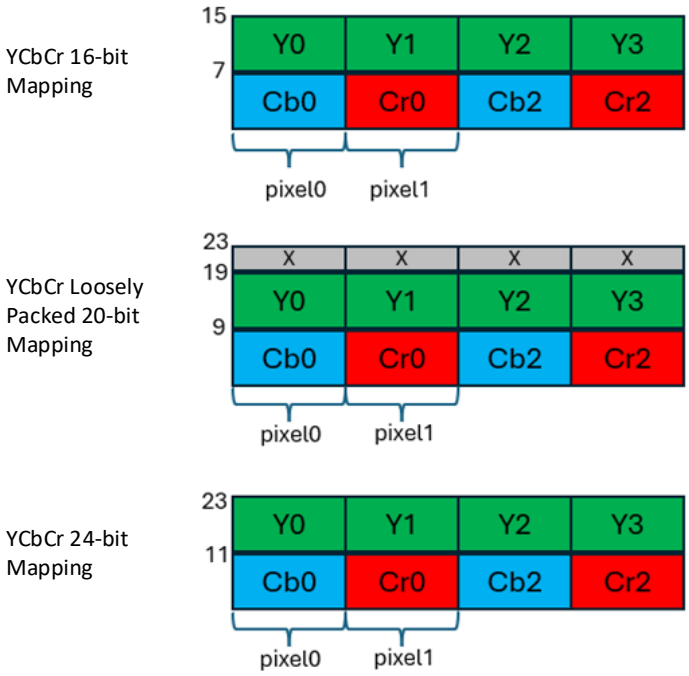

Figure 2.20. Unified Video Streaming Tx TDATA Mapping of YCbCr when Number of Output Pixel Lanes = 1

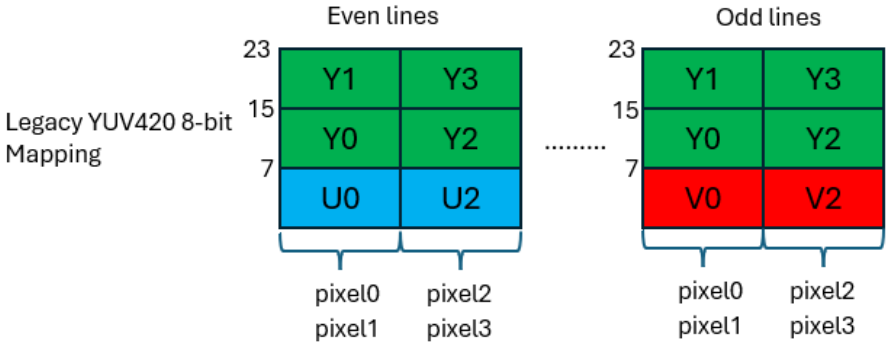

Figure 2.21. Unified Video Streaming Tx TDATA Mapping of Legacy YUV420 8-bit when Number of Output Pixel Lanes = 1

2.4.5. AXI4-Lite Interface

Control and Status registers can be configured and/or accessed when AXI4-Lite Interface is enabled. Both axil_clk_i and clk_pixel_i must be active to ensure proper register access.

Refer to the Register Description section for more information on the registers.

2.5. Other IP Specific Blocks/Layers/Interfaces

2.5.1. Byte-to-Pixel Converter IP Core

Byte-to-Pixel Converter is a FIFO-based block that provides conversion of D-PHY CSI-2 or DSI video payload packets to standard pixel format through generic CSI-2/DSI standard-based input/output signaling.

2.5.2. FIFO Implementation

The FIFO depth is defined based on the design configuration, as shown in the IP Parameter Description section. The width of the FIFO is the lowest multiple of the output pixel data width that is greater than or equal to the input bus width. This determines the number of pixels that are grouped together and written to the same FIFO address. To avoid the FIFO full or empty states, the Data Safe Zone is defined by setting the data Overflow/Underflow Threshold attribute in Table 3.1. The threshold value is used to trigger the start of reading from the FIFO.

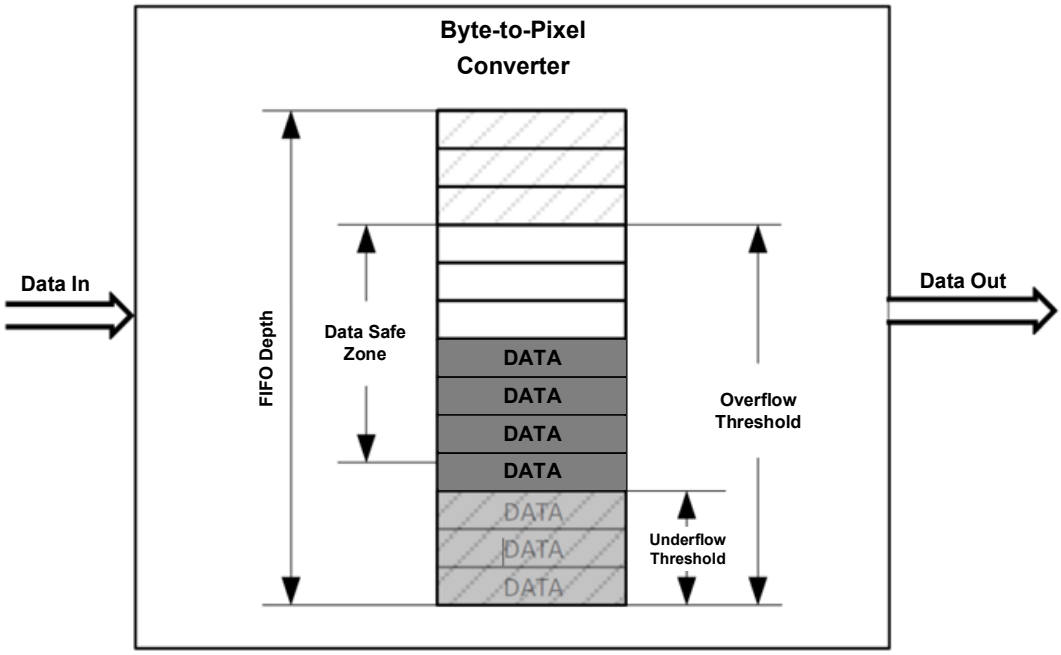

Figure 2.22. Byte-to-Pixel Converter IP FIFO Diagram

The FIFO has the following three attributes:

- FIFO_Width
- FIFO_Depth
- FIFO_Threshold

The FIFO_Threshold is used to synchronize data flows between the byte and pixel sides. Data transfer on the pixel side starts when either the FIFO_Threshold is reached or all valid bytes per line, as qualified by payload_en_i, are received.

The FIFO_Width, FIFO_Depth, and FIFO_Threshold have dependencies on several attributes listed below:

- DATA_TYPE: *Data Type* on the user interface.
- NUM_RX_LANE: *Number of Rx Lanes* on the user interface.
- RX_GEAR: *Rx Gear* on the user interface.
- *Byte Clock Frequency* (clk_byte_i) – It is available on the user interface.
- NUM_TX_CH: *Number of Output Pixel Lanes* on the user interface.
- PD_BUS_WIDTH: Pixel Data Bus Width. It is not available on the user interface. It depends only on the data type that is available on the user interface. For more information, refer to Table 2.2.
- *Pixel Clock Frequency* (clk_pixel_i) – It is available on the user interface.
- WC: Number of bytes in a line transaction. The same as *Word Count* on the user interface.
- Useful Rate: Used for Data Types that do not have *don't care* bits. It is not available on the user interface.

The Useful_Rate depends on DATA_TYPE and is used for easier byte alignment. So,

```
If DATA_TYPE = RGB666_LOOSE
     Useful_Rate = 6.0/8
Else if DATA_TYPE = YCbCr422_20_LOOSE
      Useful_Rate = 40.0/48
Else
      Useful Rate = 1.0
```

Denote Rx_width as the word width on the byte side and Tx_width as the word width on the pixel side. It can be calculated as:

- Rx_width = NUM_Rx_LANE × Rx_GEAR
- Tx_width = PD_BUS_WIDTH × NUM_Tx_CH

The FIFO width depends on DATA_TYPE, Rx_width values, and Tx_width values. It is the minimum multiple of Tx_width, which is not less than Rx_width. When using data types RAW12, RAW14, and RAW16, FIFO Width also depends on the Byte Count Restriction. So,

```
If DATA_TYPE = RAW12
     FIFO_Width = 3.0 * Rx_width
Else if DATA_TYPE = RAW14
     FIFO_Width = 7.0 * Rx_width
Else if DATA_TYPE = RAW16, then
     if Number of RX Lanes==1, RX Gear==8, and Number of Output Pixel Lanes==2, then
           FIFO_Width = 4.0 * Rx_width
     Else,
           FIFO_Width = 2.0 * Rx_width
Else,
     FIFO_Width = TX_width × ceil(RX_width / TX_width)
```

In addition, denote the data flow speed on the byte side as Receiver Data Rate (RX_rate) and the data flow speed on the pixel side as Transmitter Data Rate (TX_rate). The data rates can be calculated as:

- Receiver Data Rate = RX_width × clk_byte_i x Useful_Rate
- Transmitter Data Rate = TX_width × clk_pixel_i

The FIFO Depth (minimum value) and FIFO Threshold depend on the ratio (N_ratio) of the data flow speed on the byte side (RX_rate) and data flow speed on the pixel side (TX_rate). The RX_rate must be less than or equal to TX_rate.

The value of FIFO Depth and FIFO Threshold is doubled when using YUV420_10 and non-legacy YUV420_8 data types. This ensures the even lines in the data transaction fit within the FIFO.

- N_ratio = TX_rate/Rx_rate

FIFO Depth and FIFO Threshold can be calculated by the following formulas:

```
If (TX_rate == RX_rate)
    FIFO_Threshold = 4;
    FIFO_Depth = 16;
Else
    FIFO_Threshold = int(math.ceil(8 × WC × (N_ratio-1)/(N_ratio × FIFO_Width) + 1));
    FIFO_Depth = 2 ** clog2(FIFO_Threshold + 6);
```

This computation will satisfy the needed FIFO_Threshold value to properly process the incoming data according to the configuration when the WC set by the user is not enough for the FIFO to write data.

```
If (FIFO_Threshold > ((8 x WC) / FIFO_Width):
                FIFO_Threshold = int(math.ceil((8 x WC)/FIFO_Width);
```

2.5.3. Debug Interface

See the Debugging section for more information about this interface.

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 start of frame (SOF), end of line (EOL), and lock signals to be used in the Unified Video Streaming Tx interface.

The lock signal goes high in the next frame only when two consecutive previous frames with consistent frames are detected. When two consecutive frames have different active widths and heights, the lock signal goes low in the next frame. The lock signal remains low until two consecutive frames with consistent frame characteristics are detected again.

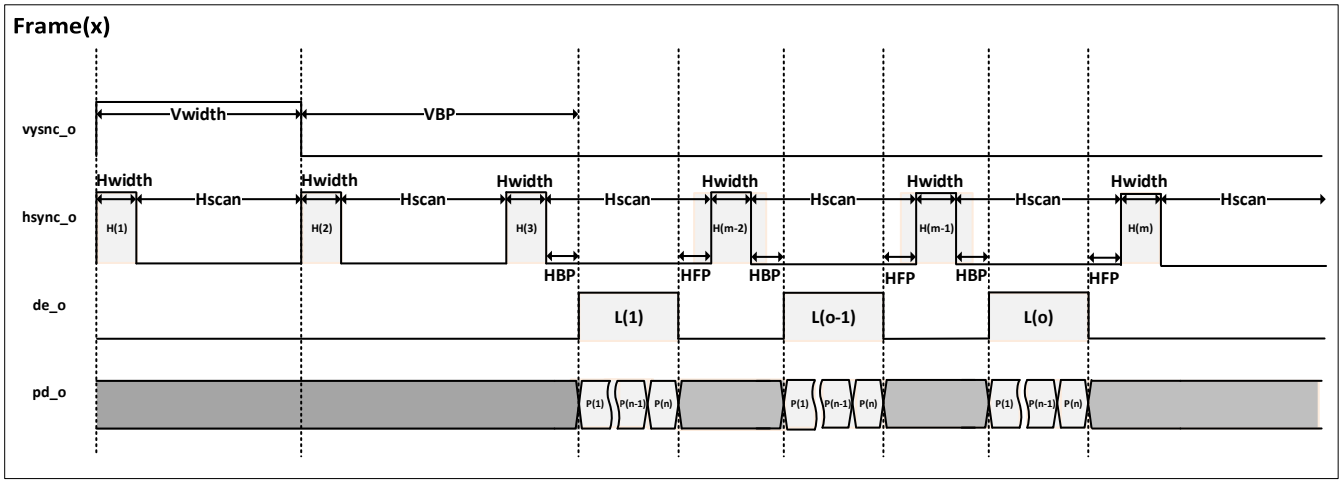

Figure 2.23. DSI Lock Definitions

Figure 2.23 shows the frame definition for DSI format. For lock status, the following criteria must be met for both frame(x) subsequent frame(x+1):

- P(n): Number of pixels per line must be equal for all instances in frame(x) and frame(x+1). This also defines the active width of the frame.
- L(o): Number of de_o assertions must be equal for all instances in frame(x) and frame(x+1). This also defines the active height of the frame.
- Hscan: Horizontal line scan. It must be equal for all instances in frame(x) and frame(x+1).
- HBP: Horizontal Back Porch. The width between hsync_o deassertion and de_o assertion must be equal for all instances in frame(x) and frame(x+1).
- HFP: Horizontal Front Porch. The width between de_o deassertion and hsync_o assertion must be equal for all instances in frame(x) and frame(x+1).
- H(m): Number of hsync_o pulses. It must be equal for all instances in frame(x) and frame(x+1).
- Hwidth: hsync_o pulse width. It must be equal for all instances in frame(x) and frame(x+1).
- Vwidth: vsync_o width. It must be equal for both frame(x) and frame(x+1).
- VBP: Vertical Back Porch. The width between vsync_o deassertion and 1st de_o assertion must be equal for both frame(x) and frame(x+1).

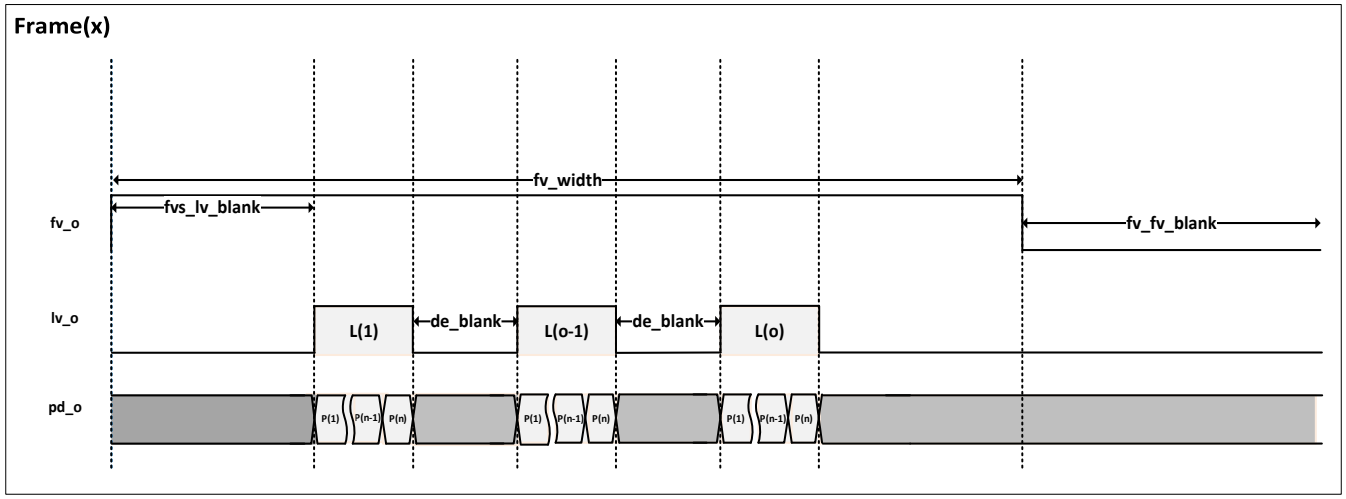

Figure 2.24. CSI-2 Lock Definitions

Figure 2.24 shows the frame definition for CSI-2 format. For lock status, the following criteria must be met:

- P(n): Number of pixels per line must be equal for all instances in frame(x) and frame(x+1). This also defines the active width of the frame.
- L(o): Number of lv_o assertions must be equal for all instances in frame(x) and frame(x+1). This also defines the active height of the frame.
- de_blank: Blanking between each lv_o must be equal for all instances in frame(x) and frame(x+1).
- fvs_lv_blank: Blanking between fv_o start sync event and 1st lv_o assertion must be equal for both frame(x) and frame(x+1).
- fv_width: fv_o width assertion must be equal for both frame(x) and frame(x+1).
- fv_fv_blank: Blanking for each frame must be equal.

3. IP Parameter Description

The configurable attributes of the Byte-to-Converter 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** font.

3.1. General

Table 3.1. General Attributes

Attribute	Selectable Values	Description
General		
Data Type	RGB565, RGB666, RGB666_LOOSE, RGB888, RAW8, **RAW10**, RAW12, RAW14, RAW16, YUV420_8, YUV420_8_CSPS, LEGACY_YUV420_8, YUV420_10, YUV420_10_CSPS, YUV422_8, YUV422_10, YCbCr422_16, YCbCr422_20_LOOSE, YCbCr422_24	Byte-to-Pixel Converter IP supported data types. Refer to the Supported Configurations for DSI and CSI-2 section for configuration options.
Byte Interface		
RX Interface	DSI, **CSI-2**	Byte-to-Pixel Converter IP RX interface.
DSI Mode	**Non-Burst Pulses**, Non-Burst Events, Burst	DSI Modes. Configurable when *RX Interface* == *DSI*.
Number of RX Lanes	**1**, 2, 4	Number of Byte-to-Pixel Converter RX high-speed ports. Refer to the Supported Configurations for DSI and CSI-2 section for configuration options.
RX Gear	**8**, 16	Number of Byte-to-Pixel Converter RX gear. Refer to the Supported Configurations for DSI and CSI-2 section for configuration options.
Byte Clock Frequency (MHz)	10-250, **10** (Avant) 10-200, **10** (Nexus)	Specifies the Byte Clock Frequency.
Enable AXI4-Stream Receiver Interface	Checked, **Not Checked**	Enables AXI4-Stream Receiver interface.
Receiver Data Rate[1]	Calculated	Non-configurable. This is equal to the product of configured Number of RX Lanes, RX Gear, and Byte Clock Frequency.
Pixel Interface		
Number of Output Pixel Lanes	**1**, 2, 4	Byte-to-Pixel Converter IP supported number of output pixel lanes. Refer to the Supported Configurations for DSI and CSI-2 section for configuration options.

Attribute	Selectable Values	Description
Camera/Display Control Polarity[2]	**Positive**, Negative	Shows the polarity of sync signals.
Number of HSYNC Pulses Inside VSYNC Active Region	3–1023, **5**	When the video mode is non-burst with sync events, this is used to determine the deassertion of the vsync_o signal. When non-burst with sync pulses, this is used only by the testbench for simulation; the actual vsync_o deassertion still depends on the reception of the vsync end packet. Configurable when *RX Interface = DSI* and *DSI Mode =Non-Burst Events*, *Burst*.
Number of Pixel Clock Cycles HSYNC is Active	3–1023, **8**	When the video mode is non-burst with sync events, this is used to determine the deassertion of the hsync_o signal. When non-burst with sync pulses, this is used only by the testbench for simulation; the actual hsync_o deassertion still depends on the reception of the hsync end packet. Configurable when *RX Interface = DSI* and *DSI Mode =Non-Burst Events*, *Burst*.
DSI Sync Packet Delay	5–1023, **5**	This is the number of Pixel Clock cycle to delay the assertion of the HSYNC and VSYNC signals. Configurable when *RX Interface = DSI*.
Pixel Clock Frequency (MHz)	10–250, **50** (Avant) 10-200, **50** (Nexus)	Specifies the Pixel Clock Frequency.
Pixel-Side Transmitter Interface	**Native Interface**, AXI4-Stream Transmitter Interface, Unified Video Streaming Tx	Specifies the interface used for transmitting pixel data.
Transmitter Data Rate[1]	Calculated	Non-configurable. This is dependent on the Data Type, Number of Output Pixel Lanes, and Pixel Clock Frequency.
Enable Byte Swap per Pixel[3]	Checked, **Not Checked**	When checked, swaps the MSB and LSB of each RAW16 pixel to be compliant with the MIPI Specifications. Previous versions of the IP have the LSB and MSB bytes swapped.
FIFO		
Manual Adjust[4]	Checked, **Not Checked**	When enabled, this is used to override the computed value of Overflow/Underflow Threshold and FIFO Depth.
Overflow/Underflow Threshold	1–65535, **4**	Configurable when Manual Adjust box is checked. Refer to the FIFO Implementation section for the calculation.
FIFO Depth	8–65536, **16**	Configurable when Manual Adjust box is checked. Refer to the FIFO Implementation section for the calculation.
FIFO Implementation	**EBR**, LUT	—
Word Count	1–65535, **5**	Word Count value depends on the Data Type. This is used for FIFO setting computations. Refer to the FIFO Implementation section for more details. Refer to the *Byte Count Restriction* column in the Pixel and Byte Count Restriction section for additional considerations or requirements when setting the values.
Miscellaneous		
Enable Debug Ports	Checked, **Not Checked**	When enabled, this is used to check the driving and driven signals of core FIFO. See the Debug Ports section for more information. Configurable when *Register Interface == Off* and *Pixel-Side Interface == Native Interface/AXI4-Stream Transmitter*.
Register Interface	**OFF**, AXI4-Lite	Specifies the register interface used.

Notes:

1. Calculated Transmitter Data Rate must be greater than or equal to Receiver Data Rate. See the FIFO Implementation section for more information.
2. Configurable only if *Pixel-Side Transmitter Interface == Native Interface*.
3. Configurable only when selected *Data Type == RAW16* and *Pixel-Side Transmitter Interface == Native Interface/AXI4-Stream Transmitter*. Attribute is added in the Byte-to-Pixel Converter IP v1.9.0.
4. If enabled, computed values of *Overflow/Underflow Threshold* and *FIFO Depth* are replaced with the configured value. This may cause digressions if not properly observed, since values are computed for optimal FIFO operation. It is advised to check *Enable Debug Ports* or access the core FIFO register through AXI4-Lite to debug possible errors that this may cause.

3.2. Supported Configurations for DSI and CSI-2

Table 3.2. Supported Configurations for DSI

Number of Output Pixel Lanes	D-PHY Lanes	RX Gearing	Data Type
1 output pixel	1 lane	8	YCbCr422 – 16 bit[1], YCbCr422 – 24 bit[1], YCbCr422 loosely packed – 20 bit[1], RGB565, RGB666, RGB666 loosely packed, RGB888
		16	YCbCr422 – 16 bit[1], YCbCr422 – 24 bit[1], YCbCr422 loosely packed – 20 bit[1], RGB565, RGB666, RGB666 loosely packed, RGB888
	2 lanes	8	YCbCr422 – 16 bit[1], YCbCr422 – 24 bit[1], YCbCr422 loosely packed – 20 bit[1], RGB565, RGB666, RGB666 loosely packed, RGB888
		16	YCbCr422 – 16 bit[1], YCbCr422 – 24 bit[1], YCbCr422 loosely packed – 20 bit[1], RGB565, RGB666, RGB666 loosely packed, RGB888
	4 lanes	8	YCbCr422 – 16 bit[1], YCbCr422 – 24 bit[1], YCbCr422 loosely packed – 20 bit[1], RGB565, RGB666, RGB666 loosely packed, RGB888
2 output pixel	1 lane	8	YCbCr422 – 16 bit[1], YCbCr422 – 24 bit[1], YCbCr422 loosely packed – 20 bit[1], RGB565

Number of Output Pixel Lanes	D-PHY Lanes	RX Gearing	Data Type
		16	YCbCr422 – 16 bit[1], YCbCr422 – 24 bit[1], YCbCr422 loosely packed – 20 bit[1], RGB565
	2 lanes	8	YCbCr422 – 16 bit[1], YCbCr422 – 24 bit[1], YCbCr422 loosely packed – 20 bit[1], RGB565
		16	YCbCr422 – 16 bit[1], YCbCr422 – 24 bit[1], YCbCr422 loosely packed – 20 bit[1], RGB565, RGB666, RGB666 loosely packed, RGB888
	4 lanes	8	YCbCr422 – 16 bit[1], YCbCr422 – 24 bit[1], YCbCr422 loosely packed – 20 bit[1], RGB565, RGB666, RGB666 loosely packed, RGB888
		16	YCbCr422 – 16 bit[1], YCbCr422 – 24 bit[1], YCbCr422 loosely packed – 20 bit[1], RGB565, RGB666, RGB666 loosely packed, RGB888
4 output pixels	4 lanes	16	YCbCr422 – 16 bit[1], YCbCr422 – 24 bit[1], YCbCr422 loosely packed – 20 bit[1], RGB565, RGB666, RGB666 loosely packed, RGB888

Note:

1. For YCbCr data types, LSB is always transmitted first. The pixel arrangement must be in YCb/YCr, with Cb or Cr as the LSB component.

Table 3.3. Supported Configurations for CSI-2

Number of Output Pixel Lanes	D-PHY Lanes	RX Gearing	Data Type
1 output pixel	1 lane	8	RGB565, Legacy YUV420 8-bit[1], YUV420 8-bit[1,2], YUV420 8-bit CSPS[1,2], YUV422 8-bit[1], YUV420 10-bit[1,2], YUV420 10-bit CSPS[1,2], YUV422 10-bit[1,2], RAW8, RAW10, RAW12, RAW14, RAW16,

Number of Output Pixel Lanes	D-PHY Lanes	RX Gearing	Data Type
			RGB888
		16	RGB565, YUV420 10-bit[1,2], YUV420 10-bit CSPS[1,2], YUV422 10-bit[1,2], RAW10, RAW12, RAW14, RAW16, RGB888
	2 lanes	8	RGB565, Legacy YUV420 8-bit[1], YUV420 8-bit[1,2], YUV420 8-bit CSPS[1,2], YUV422 8-bit[1], YUV420 10-bit[1,2], YUV420 10-bit CSPS[1,2], YUV422 10-bit[1], RAW8, RAW10, RAW12, RAW14, RAW16, RGB888
		16	RGB565, RAW12, RAW14, RAW16, RGB888
	4 lanes	8	RGB565, Legacy YUV420 8-bit[1], YUV420 8-bit[1,2], YUV420 8-bit CSPS[1,2], YUV422 8-bit[1], YUV420 10-bit[1,2], YUV420 10-bit CSPS[1,2], YUV422 10-bit[1], RAW8, RAW10, RAW12, RAW14, RAW16, RGB888
		16	RGB565, RAW12, RAW14, RAW16
2 output pixel	1 lane	8	RGB565, RAW12, RAW14, RAW16
		16	RGB565, Legacy YUV420 8-bit[1,2], YUV420 8-bit[1,2], YUV420 8-bit CSPS[1,2], YUV422 8-bit[1], RAW8, RAW12, RAW14, RAW16
	2 lanes	8	RGB565, Legacy YUV420 8-bit[1,2], YUV420 8-bit[1,2], YUV420 8-bit CSPS[1,2], YUV422 8-bit[1], YUV420 10-bit[1,2], YUV420 10-bit CSPS[1,2], YUV422 10-bit[1,2],

Number of Output Pixel Lanes	D-PHY Lanes	RX Gearing	Data Type
			RAW8, RAW10, RAW12, RAW14, RAW16
		16	RGB565, Legacy YUV420 8-bit[1], YUV420 8-bit[1,2], YUV420 8-bit CSPS[1,2], YUV422 8-bit[1], YUV420 10-bit[1,2], YUV420 10-bit CSPS[1,2], YUV422 10-bit[1,2], RAW8, RAW10, RAW12, RAW14, RAW16, RGB888
	4 lanes	8	RGB565, Legacy YUV420 8-bit[1], YUV420 8-bit[1,2], YUV420 8-bit CSPS[1,2], YUV422 8-bit[1], YUV420 10-bit[1,2], YUV420 10-bit CSPS[1,2], YUV422 10-bit[1,2], RAW8, RAW10, RAW12, RAW14, RAW16, RGB888
		16	RGB565, RAW12, RAW14, RAW16, RGB888
4 output pixel	1 lane	16	RAW12
	2 lanes	8	RAW12
		16	RAW12
	4 lanes	8	RAW12
		16	RAW12, RGB888

Notes:

1. For YUV data types with *Pixel-Side Transmitter Interface == Native Interface/AXI4-Stream Transmitter*, the pixel data bus width refers to the actual bits per data type component instead of the bits per data type pixel. For example, in YUV420 8-bit, selecting *Number of Output Pixel Lanes == 2* means two parallel 8-bit components are received per clk_pixel_i.
2. Configuration is not supported when *Pixel-Side Transmitter Interface == Unified Video Streaming Tx.*

3.3. Pixel and Byte Count Restriction

Table 3.4. Pixel and Byte Count Restriction

Data Type	Pixel Count Restriction	Byte Count Restriction
RGB565	multiple of 1	multiple of 2
RGB666	multiple of 4	multiple of 9
RGB666, Loosely packed	multiple of 1	multiple of 3
RGB888	multiple of 1	multiple of 3
RAW8	multiple of 1	multiple of 1
Legacy YUV420 8-bit	multiple of 2	multiple of 3
YUV420 8-bit CSPS	multiple of 2	multiple of 4
YUV420 8-bit	multiple of 2	multiple of 4

Data Type	Pixel Count Restriction	Byte Count Restriction
YUV422 8-bit	multiple of 2	multiple of 4
RAW10	multiple of 4	multiple of 5
YUV420 10-bit CSPS	multiple of 4	multiple of 10
YUV420 10-bit	multiple of 4	multiple of 10
YUV422 10-bit	multiple of 2	multiple of 5
RAW12	multiple of 2	multiple of 3
YCbCr422 16-bit	multiple of 2	multiple of 4
YcbCr422 24-bit	multiple of 4	multiple of 6
YcbCr422 Loosely packed 20-bit	multiple of 4	multiple of 6
RAW14	multiple of 4	multiple of 7
RAW16	multiple of 1	multiple of 2

4. Signal Description

This section describes the Byte-to-Pixel Converter IP ports.

4.1. Clock Interface

Table 4.1. Clock Interface Signal Descriptions

Port Name	Type	Width	Description
clk_byte_i[1]	Input	1	• Source clock for the Byte Domain Native interface • Active when *Enable AXI4-Stream Receiver Interface == Not Checked*
axis_sclk_i[1]	Input	1	• Source clock for the AXI4-Stream Receiver interface • Active when *Enable AXI4-Stream Receiver Interface == Checked*
clk_pixel_i[1]	Input	1	• Source clock for the Pixel Domain interface • Active when *Pixel-Side Transmitter Interface == Native* Interface */ Unified Video Streaming Tx Interface*
axis_mclk_i[1]	Input	1	• Source clock for the AXI4-Stream Transmitter interface • Active when *Pixel-Side Transmitter Interface == AXI4-Stream Transmitter*
axil_clk_i[1]	Input	1	• Source clock for the AXI4-Lite interface • Active when *Register Interface == AXI4-Lite*

Note:

1. The recommended clock frequency for clock source signal is:
   - For Avant Devices: 10–250 MHz.
   - For Nexus Devices: 10–200 MHz.

4.2. Reset Interface

Table 4.2. Reset Interface Signal Descriptions

Port	Type	Width	Description
reset_byte_n_i	Input	1	• System reset • Active low signal to reset the logic in the Byte-side Native interface when *Enable AXI4-Stream Receiver Interface == Not Checked*
axis_sresetn_i	Input	1	• System reset • Active low signal to reset the logic in the AXI4-Stream Receiver interface when *Enable AXI4-Stream Receiver Interface == Checked*
axis_mresetn_i	Input	1	• Active low signal to reset the logic in the AXI4-Stream Transmitter interface • Active when *Pixel-Side Transmitter Interface == AXI4-Stream Transmitter*
reset_pixel_n_i	Input	1	• Active low signal to reset the logic in the Pixel-side Native interface • Active when *Pixel-Side Transmitter Interface == Native Interface / Unified Video Streaming Tx*
axil_rst_n_i	Input	1	• Active low signal to reset the logic in the AXI4-Lite interface • Active when *Register Interface == AXI4-Lite*

4.3. Byte Domain Interface

Table 4.3. Byte Domain Signal Descriptions

Port	Type	Width	Description
sp_en_i[2]	Input	1	Active high pulse to indicate a valid short packet in the Rx side
lp_av_en_i[2]	Input	1	Active high pulse to indicate an active video long packet in the Rx side. The byte2pixel module prepares for the arrival of the video stream.

Port	Type	Width	Description
dt_i	Input	6	Data type field of the D-PHY Rx header packet
wc_i	Input	16	Word Count field of the D-PHY Rx header packet
payload_i	Input	DATA_WIDTH[4]	This is the active video data stream. The width of the data bus depends on the gearing and the number of D-PHY Rx lanes. Refer to the IP Parameter Description section for possible values for *Number of RX Lanes* and *RX Gear*.
payload_en_i	Input	1	Active high payload valid indicator
sp2_en_i[3]	Input	1	This is valid only for gear 16, 4-lane configuration. Active high pulse to indicate a reception of a second valid short packet in the same byte clock cycle.
lp2_av_en_i[3]	Input	1	This is valid only for gear 16, 4-lane configuration. Active high pulse to indicate a second valid active video long packet in the same byte clock cycle.
dt2_i	Input	6	This is valid only for gear 16, 4-lane configuration. Data type field of the second D-PHY RX header packet.
wc2_i	Input	16	This is valid only for gear 16, 4-lane configuration. Word Count field of the second D-PHY RX header packet.
pixcnt_c_o[1]	Output	19	This is an internal net with critical path. It is ported out so that constraints can still be applied to this signal even on encrypted IP. This port may be left unconnected.
pix_out_cntr_o[1]	Output	16	
wc_pix_sync_o[1]	Output	16	

Notes:

1. Not available if *Data Type == RAW12*, *RAW14*, *RAW16* or if *Pixel-Side Transmitter Interface == Unified Video Streaming Tx* since this interface does not support sideband signal.
2. Port is also available when *Enable AXI4-Stream Receiver Interface == Checked*.
3. Port is also available when *Enable AXI4-Stream Receiver Interface == Checked, Number of RX Lanes = 4,* and *RX Gear = 16*.
4. DATA_WIDTH = *Number of RX Lanes* × *RX Gear*.

4.4. AXI4-Stream Receiver Interface

Table 4.4. AXI4-Stream Receiver Interface Signal Descriptions[1]

Port	Type	Width	Description
axis_svalid_i	Input	1	AXI4-Stream Receiver valid input signal
axis_sready_o	Output	1	AXI4-Stream Receiver ready output signal. Currently, the value of the signal is always set to 1.
axis_sdata_i	Input	RECEIVER_DATA_W[2]	AXI4-Stream Receiver data. RECEIVER_DATA_W is an integer number of bytes mapped to RECEIVER_DATA_T. If RECEIVER_DATA_W is not a multiple of bytes, the MSB of RECEIVER_DATA_W must be padded with 0s to complete the byte.

Notes:

1. Available only if *Enable AXI4-Stream Receiver Interface* is checked.
2. For the case when *Number of RX Lanes* × *RX Gear* ! = 64:
   - RECEIVER_DATA_T = dt_i + wc_i + payload_i = 6 + 16 + (*Number of RX Lanes* × *RX Gear*).
   - RECEIVER_DATA_W = [ceil (RECEIVER_DATA_T/8)] × 8

   For the case when *Number of RX Lanes* × *RX Gear* = 64:
   - RECEIVER_DATA_T = dt_i + wc_i + dt2_i + wc2_i + payload_i = 6 + 16 + 6 + 16 + 64 = 108.
   - RECEIVER_DATA_W = [ceil (RECEIVER_DATA_W/8)] × 8

4.5. Pixel Domain Interface

Table 4.5. Pixel Domain Interface Signal Descriptions

Port Name	Direction	Width	Description
vsync_o[7]	Output	1	VSYNC signal for DSI. Active High if *Camera/Display Control Polarity* attribute is Positive. Otherwise, this is active Low.
hsync_o[7]	Output	1	HSYNC signal for DSI. Active High if *Camera/Display Control Polarity* attribute is Positive. Otherwise, this is active Low.
fv_o[8]	Output	1	Frame Valid signal for CSI-2. Active High if *Camera/Display Control Polarity* attribute is Positive. Otherwise, this is active low.
lv_o	Output	1	Line Valid signal for CSI-2. Active High if *Camera/Display Control Polarity* attribute is Positive. Otherwise, this is active Low.
de_o	Output	1	Data Enable signal for DSI. Active high if *Camera/Display Control Polarity* attribute is Positive. Otherwise, this is active low.
pd_o[5,6]	Output	PD_BUS_WIDTH[1] × *Number of Output Pixel Lanes*	Pixel data output. The pixel_width may be 8, 10, 12, 18, 24, 36, 48, 72, or 96 bits. Refer to the IP Parameter Description section for possible values for PD_BUS_WIDTH (Pixel Data Bus Width) and NUM_TX_CH (Number of Output Pixel Lanes).
pd_red_o[2]	Output	BPC[3] × *Number of Output Pixel Lanes*	Red color component of the RGB pixel data.
pd_green_o[2]	Output	BPC[3] × *Number of Output Pixel Lanes*	Green color component of the RGB pixel data.
pd_blue_o[2]	Output	BPC[3] × *Number of Output Pixel Lanes*	Blue color component of the RGB pixel data.
p_odd_o[4]	Output	2	This signal is used to indicate the valid pixels for the last valid pixel data cycle in case of multiple pixel outputs per pixel clock cycle.

Notes:

1. Refer to Table 2.2 for more information of PD_BUS_WIDTH.
2. Only available for *Data Type == RGB888*, *RGB666*, *RGB666 Loosely packed*, *RGB565*, and *Pixel-Side Transmitter Interface == Native Interface*. Ports are added to the Byte-to-Pixel Converter IP v1.9.0 and are not backward-compatible with IP v1.7.0 or lower.
3. BPC is Bits Per Color component of the selected data type:
   - BPC of Red and Blue color components are 8 for RGB888, 6 for RGB666 and RGB666 Loosely packed, and 5 for RGB555.
   - BPC of Green color component is 8 for RGB888, and 6 for RGB666, RGB666 Loosely packed, and RGB565.
4. This is a single bit signal for a 2-pixel output configuration, or a 2-bit bus for a 4-pixel output configuration.
   - 00 – All pixels are valid
   - 01 – Only the first pixel (LSB) is valid
   - 10 – Only the lower two pixels in the lower bits are valid
   - 11 – The last pixel (MSB) is not valid
5. Output data for pd_o when *Data Type == RAW14, YCbCr 16-bit, YCbCr Loosely packed 20-bit, YCbCr 24-bit* is not backward compatible with Byte-to-Pixel Converter IP v1.7.0 or lower for MIPI specification compliance.
6. Port is not available if pd_red_o, pd_green_o, and pd_blue_o exist as output ports.
7. Port is also available when *Pixel-Side Transmitter Interface == AXI4-Stream Transmitter* and *RX Interface == DSI*.
8. Port is also available when *Pixel-Side Transmitter Interface == AXI4-Stream Transmitter* and *RX Interface == CSI-2*.

4.6. AXI4-Stream Transmitter Interface

Table 4.6. AXI4-Stream Transmitter Interface Signal Descriptions[1]

Port Name	Direction	Width	Description
axis_mvalid_o	Output	1	AXI4-Stream Transmitter valid output signal
axis_mready_i	Input	1	AXI4-Stream Transmitter ready input signal

Port Name	Direction	Width	Description
axis_mdata_o	Output	TRANSMITTER_DATA_W[2]	AXI4-Stream Transmitter data. TRANSMITTER_DATA_W is an integer number of bytes mapped to TRANSMITTER_DATA_T. If TRANSMITTER_DATA_T is not a multiple of bytes, the MSB of TRANSMITTER_DATA_W is padded with 0s to complete the byte.

Notes:

1. Available only if *Pixel-Side Transmitter Interface == AXI4-Stream Transmitter*.
2. TRANSMITTER_DATA_T = p_odd_o + pd_o = 2 + (*PD_BUS_WIDTH* × *Number of Output Pixel Lanes*). And TRANSMITTER_DATA_W = [ceil (TRANSMITTER_DATA_T /8)] × 8. Refer to Table 2.2 for more information on PD_BUS_WIDTH.

4.7. Unified Video Streaming Tx Interface

Table 4.7. Unified Video Streaming Tx Interface Signal Descriptions[1]

Port Name	Direction	Width	Description
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/lv_o of the Pixel-side Native interface
axis_vid_tdata_o	Output	TD_WD_OUT × *Number of Output Pixel Lanes*[2]	Unified Video Streaming Tx interface data mapped from pd_o of the Pixel-side Native interface
axis_vid_tuser_o[3]	Output	2	Unified Video Streaming Tx interface user-defined additional data information output
axis_vid_tlast_o	Output	1	Unified Video Streaming Protocol last pixel of the line output signal

Notes:

1. Available only if *Pixel-Side Transmitter Interface = Unified Video Streaming Tx*.
2. TD_WD_OUT = tdata width per pixel. Refer to the Unified Video Streaming Tx Interface section for more information.
3. axis_vid_tuser_o bit assignment:
   - [0]: Start of Frame
     - 0 – Not the start of new frame
     - 1 – Start of new frame
   - [1]: Unused

4.8. Debug Ports

Optional interfaces serve for debug purpose only. Available only if *Enable Debug Ports* is checked.

Table 4.8. Debug Ports

Port	Type	Width	Description
write_cycle_o[1,2]	Output	4	Payload data write cycle
mem_we_o[2]	Output	1	Payload data Write Enable, active high
mem_re_o[2]	Output	1	Payload data Read Enable, active high
read_cycle_o[2]	Output	2	Pixel data read cycle
fifo_empty_o	Output	1	Indicates FIFO empty condition
fifo_full_o	Output	1	Indicates FIFO full condition

Notes:

1. write_cycle_o is not available when *Data Type == RAW8*, *YUV420 8-bit*, *Legacy YUV420 8-bit*, *YUV420 8-bit CSPS*, or *YUV422 8-bit*.
2. Ports are not available when *Data Type == RAW12*, *RAW14*, or *RAW16.*

4.9. AXI4-Lite Interface

Table 4.9. AXI4-Lite Interface Ports[1]

Port	Type	Width	Description
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 valid signal
s_axil_wdata_i	Input	32	AXI4-Lite data signal
s_axil_wstrb_i	Input	4	AXI4-Lite 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*.

5. Register Description

Byte-to-Pixel Converter IP supports register programming and status reading. All registers listed can be accessed when AXI4-Lite interface is enabled.

The behavior of registers to write and read access is defined by its access type, which is defined in the following table:

Table 5.1. Access Type Definitions

Access Type	Description
RW	Read/Write
RO	Read Only
W1C	Write 1 to Clear

5.1. Register Address Map

The configuration register address map is as shown in the following table:

Table 5.2. Configuration Register Address Map

Address Offset	Name	Description	Access	Default
0x0000	CORE_FIFO_REG	Core FIFO register.	RW	32'h1
0x0004	UVSI_FIFO_REG	Unified Video Streaming Interface FIFO register.	RW	32'h1
0x0200	ACT_WIDTH	Active width register. Valid value of this register will be reflected once LOCKED is 1.	RO	32'h0
0x0204	ACT_HEIGHT	Active height register. Valid value of this register will be reflected once LOCKED is 1.	RO	32'h0
0x0280	LOCKED	Locked status register. The value of this register will change depending on the frame consistency detected by the pixel monitor.	RO	32'h0

5.2. Byte-to-Pixel Converter Configuration Registers

The Byte-to-Pixel Converter configuration registers are shown in the following tables:

Table 5.3. CORE_FIFO_REG Register

Field	Name	Description	Access	Default
[31:2]	rsvd	Reserved	RO	30'h0
[1]	fifo_full	Indicates the condition for full core FIFO.	W1C	1'h0
[0]	fifo_empty	Indicates the condition for empty core FIFO.	RO	1'h1

Table 5.4. UVSI_FIFO_REG Register

Field	Name	Description	Access	Default
[31:2]	rsvd	Reserved	RO	30'h0
[1]	uvsi_fifo_full	Indicates the condition for full UVSI FIFO.	W1C	1'h0
[0]	uvsi_fifo_empty	Indicates the condition for empty UVSI FIFO.	RO	1'h1

Table 5.5. ACT_WIDTH Register

Field	Name	Description	Access	Default
[31:16]	rsvd	Reserved	RO	16'h0

Field	Name	Description	Access	Default
[15:0]	act_width[1]	Active width in units of pixels. Active width of the video stream data extracted from the pixel monitor. Valid values on this register show once LOCKED register is 1. Otherwise, default value is reflected.	RO	16'h0

Note:

1. When *Data Type == Legacy YUV420 8-bit*, the video packet length (per clock cycle) is act_width ÷ 2. This is because axis_vid_tdata_o carries 2 pixels per clock cycle for this data type.

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 the pixel monitor. Valid values on this register show once LOCKED register is 1. Otherwise, default value is reflected.	RO	16'h0

Table 5.7. LOCKED Register

Field	Name	Description	Access	Default
[31:1]	rsvd	Reserved	RO	31'h0
[0]	locked	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]. The value of this register changes if two consistent consecutive frames are detected by the Pixel Monitor module. Otherwise, default value is reflected.	RO	1'h0

Notes:

1. Video stream data is inconsistent if two consecutive frames have different active heights and widths.
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's architecture. The steps below describe how to generate the Byte-to-Pixel Converter IP in the Lattice Radiant software.

To generate the Byte-to-Pixel Converter IP:

1. Create a new Lattice Radiant software project or open an existing project.
2. In the **IP Catalog** tab, double-click **Byte-to-Pixel Converter** under **IP, Audio_Video_and_Image_Processing** category. The **Module/IP Block Wizard** opens as shown in Figure 6.1. Enter values in the **Component name** and the **Create in** fields and click **Next**.

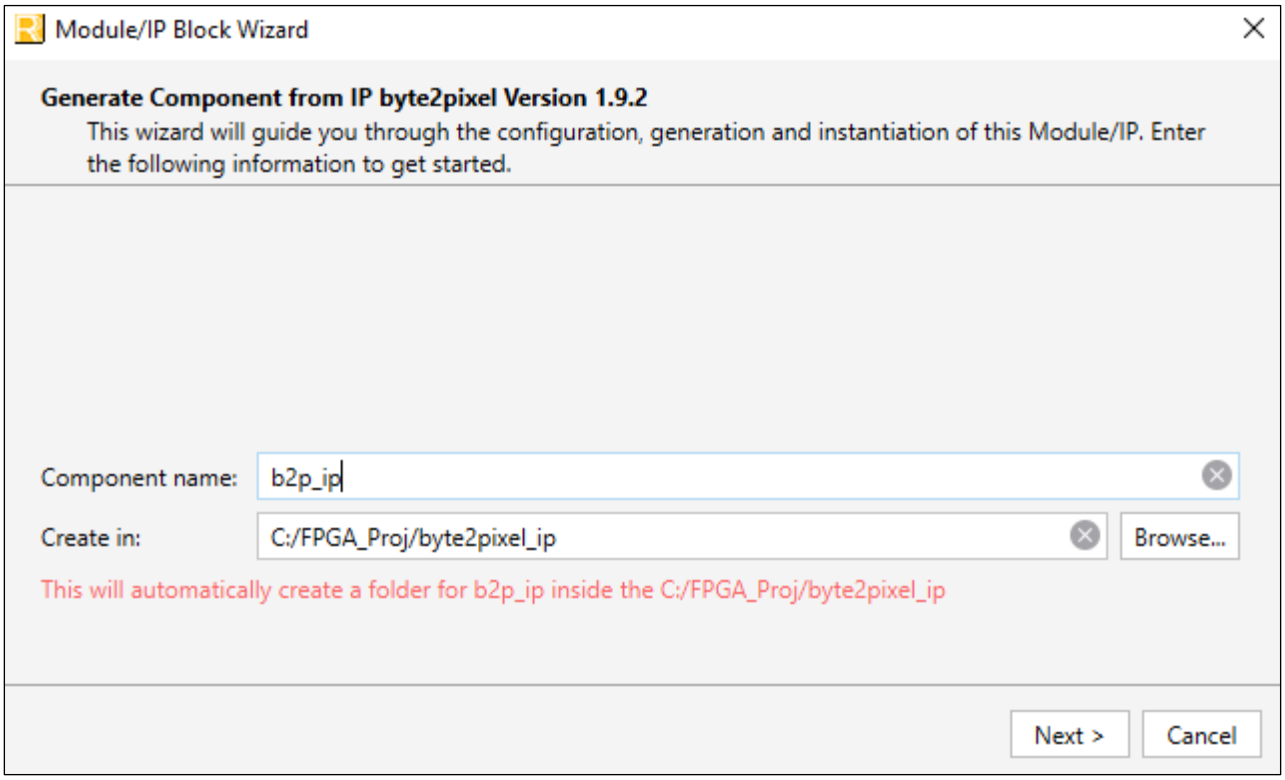

Figure 6.1. Module/IP Block Wizard

3. In the next **Module/IP Block Wizard** window, customize the selected Byte-to-Pixel Converter IP using the drop-down lists and checkboxes. Figure 6.2 shows an example configuration of the Byte-to-Pixel Converter IP. For details on the configuration options, refer to the IP Parameter Description section.

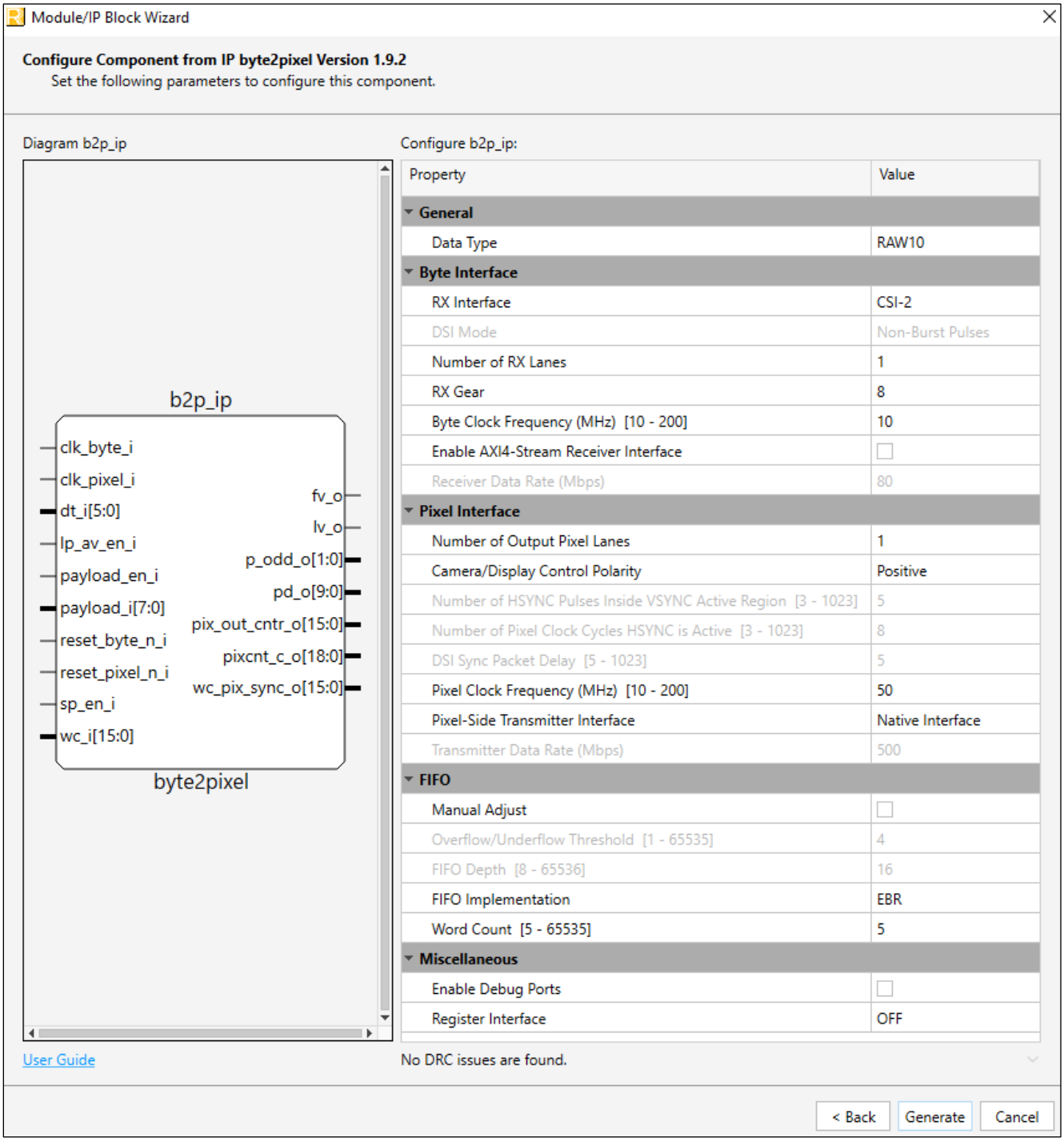

Figure 6.2. IP Configuration

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

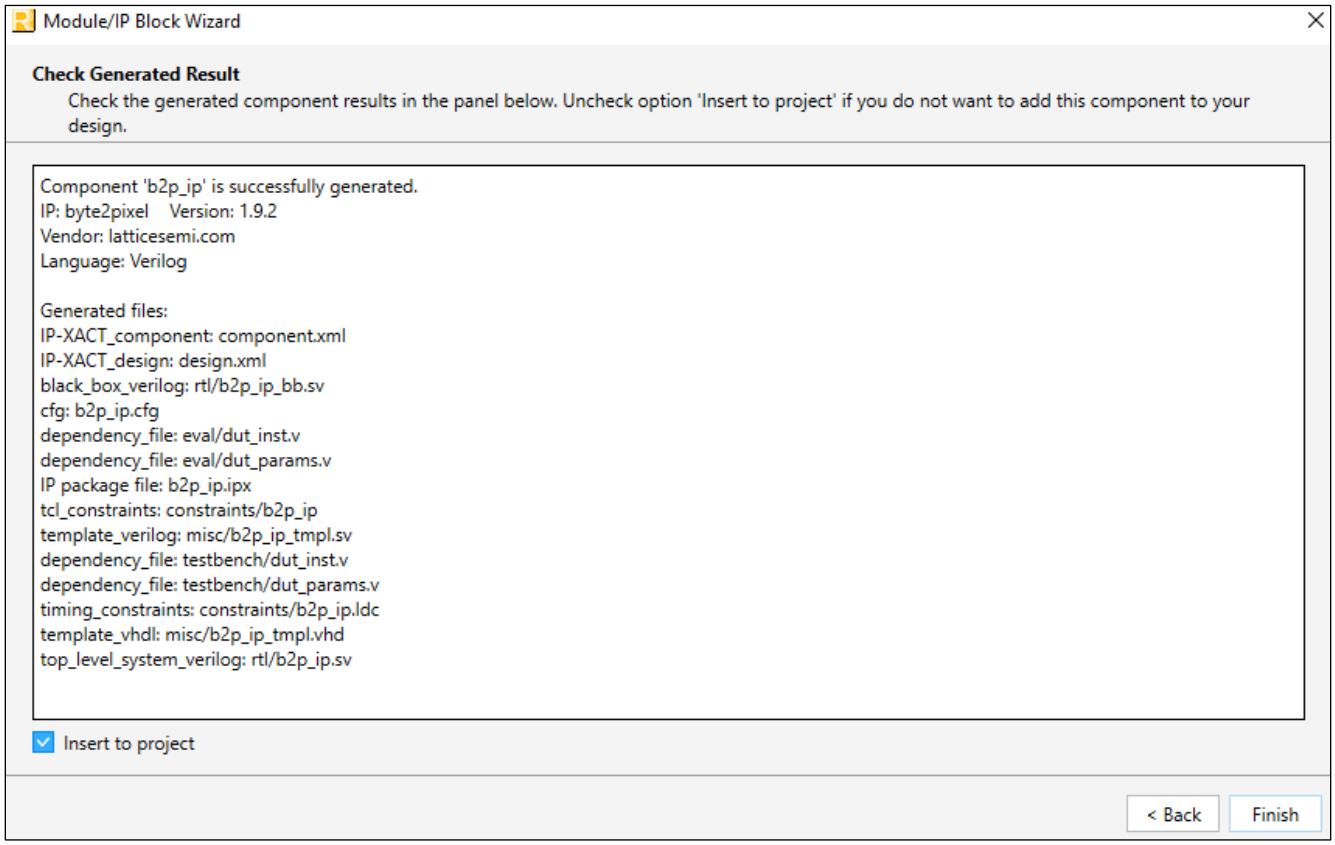

Figure 6.3. Check Generated Result

5. Click **Finish**. All the generated files are placed under the directory paths in the **Create in** and the **Component name** fields shown in Figure 6.1.

6.1.1. Generated Files and File Structure

The generated Byte-to-Pixel Converter IP module package includes the black box (B2P_bb.v) and instance templates (B2P_tmpl.v/vhd) that can be used to instantiate the core in a top-level design. An example RTL top-level reference source file (B2P.v) that can be used as an instantiation template for the module is also provided. You may also use this top-level reference as the starting template for their complete design. The generated files are listed in Table 6.1.

Note: The component name used in this example is *B2P* which is customizable based on your preference <Instance Name>.

Table 6.1. 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>.sv	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/constraint.sdc	This file provides information on how to constrain the IP in your design.
eval/constraint.pdc	This file constrains the clock used in your design. Refer to the Timing Constraints section on how to use this file.

6.2. Design Implementation

Completing your design includes additional steps to specify analog properties, pin assignments, 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 and 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 Byte-to-Pixel Converter IP generates the following constraint files:

- A legacy pre-synthesis constraint file in LDC format *(<ip_instance_path>/constraints/<instance_name>.ldc*), which is automatically used and propagated by the software tool.
- A constraint file in SDC format (*<ip_instance_path>/constraints/constraint.sdc*) that contains post-synthesis IP constraints. These constraints are automatically used and propagated by the software tool for the Lattice Radiant software version 2024.1 and higher.
- An evaluation post-synthesis constraint file in PDC format (*<ip_instance_path>/eval/constraint.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.

```
#-------------------------------------------------------------------------------------------
# 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.
#
#-------------------------------------------------------------------------------------------

#-------------------------------------------------------------------------------------------
# Section 1: SETTINGS (to be used for evaluation)
#-------------------------------------------------------------------------------------------
#-- For AXI4-Lite clock, you must manually constrain this based on actual usage. --#
#-- Sample is currently set to 10ns which is equivalent to 100MHz.               --#
set IP_INST_AXI4_LITE_PRD 10

#-- With IP-Related Parameters
set IP_INST_BYTECLK_PERIOD [expr {double(round(1000000/$IP_INST_BYTE_CLK_FREQ))/1000}]
set IP_INST_PIXELCLK_PERIOD [expr {double(round(1000000/$IP_INST_PIX_CLK_FREQ))/1000}]

#-------------------------------------------------------------------------------------------
# Section 2: EVALUATION
#-------------------------------------------------------------------------------------------
#-- CLOCKS
#   Clocks defined in this sample PDC are for evaluation purposes and only serves as a
#   guidance. All clocks should be defined in the top-level PDC.
#--
# -- AXI4-STREAM RECEIVER CLOCK --#
if {$IP_INST_AXI4_RX=="ON"} {
  create_clock -name {axis_sclk_i} -period $IP_INST_BYTECLK_PERIOD [get_ports axis_sclk_i]
} else {
  create_clock -name {clk_byte_i} -period $IP_INST_BYTECLK_PERIOD [get_ports clk_byte_i]
}
# -- AXI4-STREAM TRANSMITTER CLOCK -- #
if {$IP_INST_AXI4_TX=="AXI4_TX_LEGACY"} {
  create_clock -name {axis_mclk_i} -period $IP_INST_PIXELCLK_PERIOD [get_ports axis_mclk_i]
} else {
  create_clock -name {clk_pixel_i} -period $IP_INST_PIXELCLK_PERIOD [get_ports clk_pixel_i]
}
#-- AXI4-LITE CLOCK -- #
if {$IP_INST_REG_INT=="AXI4_LITE"} {
  create_clock -name {axil_clk_i} -period $IP_INST_AXI4_LITE_PRD [get_ports axil_clk_i]
}
```

Figure 6.4. Clock Constraining in Constraint PDC File

To run the software implementation flow using the provided constraint file after the IP is generated, follow these steps:

1. In the Post-Synthesis Constraint Files section, add *<ip_instance_name>/eval/constraint.pdc* as show in Figure 6.5.

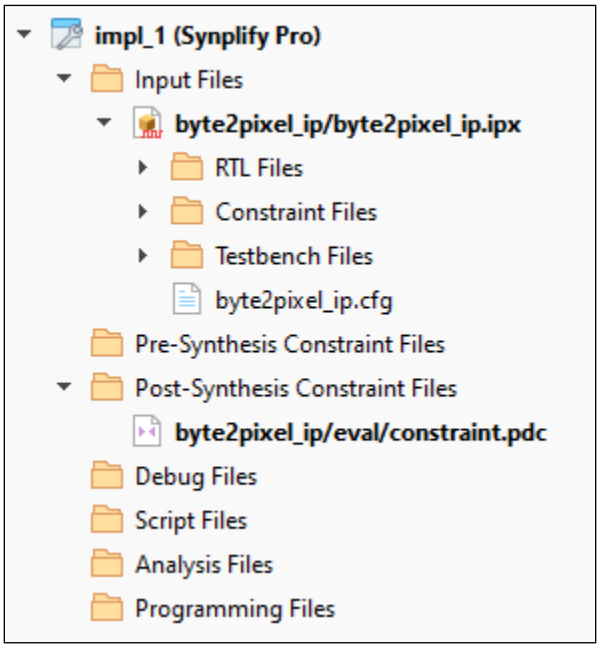

Figure 6.5. Adding Constraint in PDC File

2. Run the implementation flow.

Notes:

- During synthesis, you can ignore clock-related warnings as the evaluation IP does not include clock-related constraints at the pre-synthesis level.
- During post-synthesis, warnings related to dropped constraints may be shown. Since the IP supports many configurations and parameter combinations, some default constraints may now be applicable to the selected configuration.

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

6.4. Specify the Strategy

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

6.5. Running Functional Simulation

You can run functional simulation after the IP is generated. To run functional simulation:

1. Click the  button located on the **Toolbar** to initiate the **Simulation Wizard** shown in Figure 6.6.

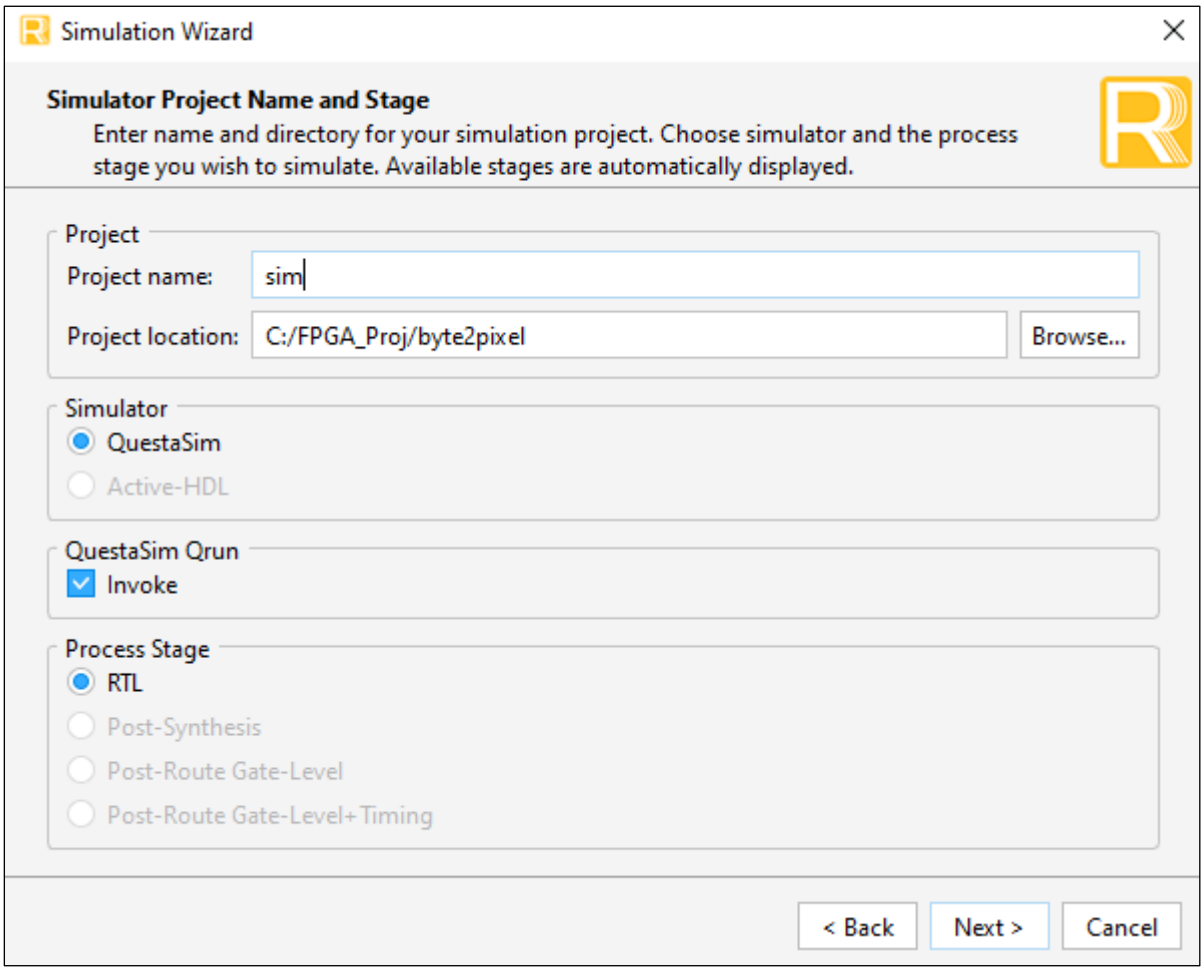

Figure 6.6. Simulation Wizard

2. Click **Next** to open the **Add and Reorder Source** window as shown in Figure 6.7.

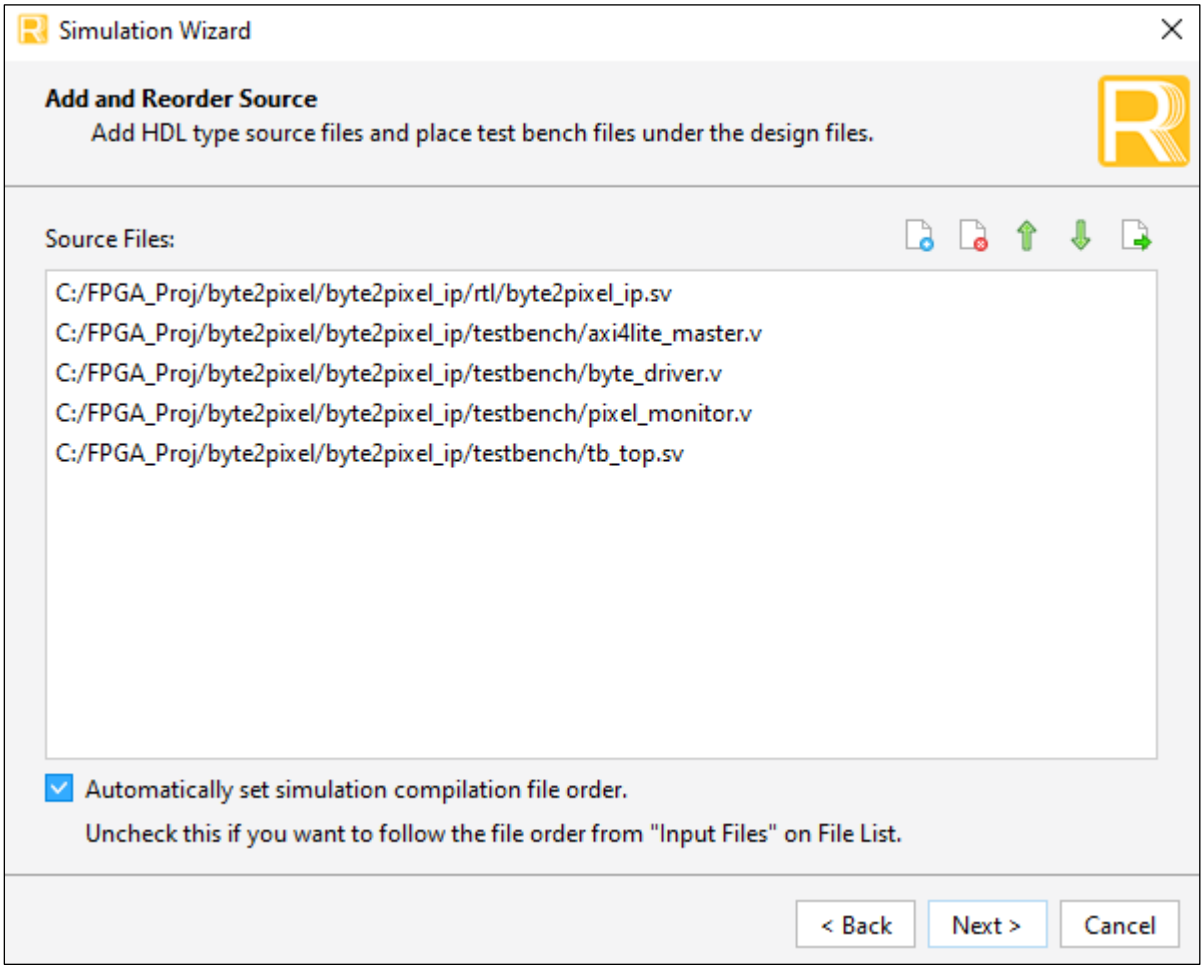

Figure 6.7. Add and Reorder Source

3. Add the tb_top.v file from the testbench directory.

Table 6.2. Testbench File List

Testbench Files	Description
testbench/tb_top.sv	Top testbench to run loopback test of generated <Instance Name>.v
testbench/byte_driver.v	Testbench to create log files for monitoring byte data during transmission.
testbench/pixel_monitor.v	Testbench to create log files for monitoring pixel data during transmission.
testbench/axi4lite_master.v	Testbench to run read and write operation for AXI4-Lite register access

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

The waveform in Figure 6.8 shows an example simulation result.

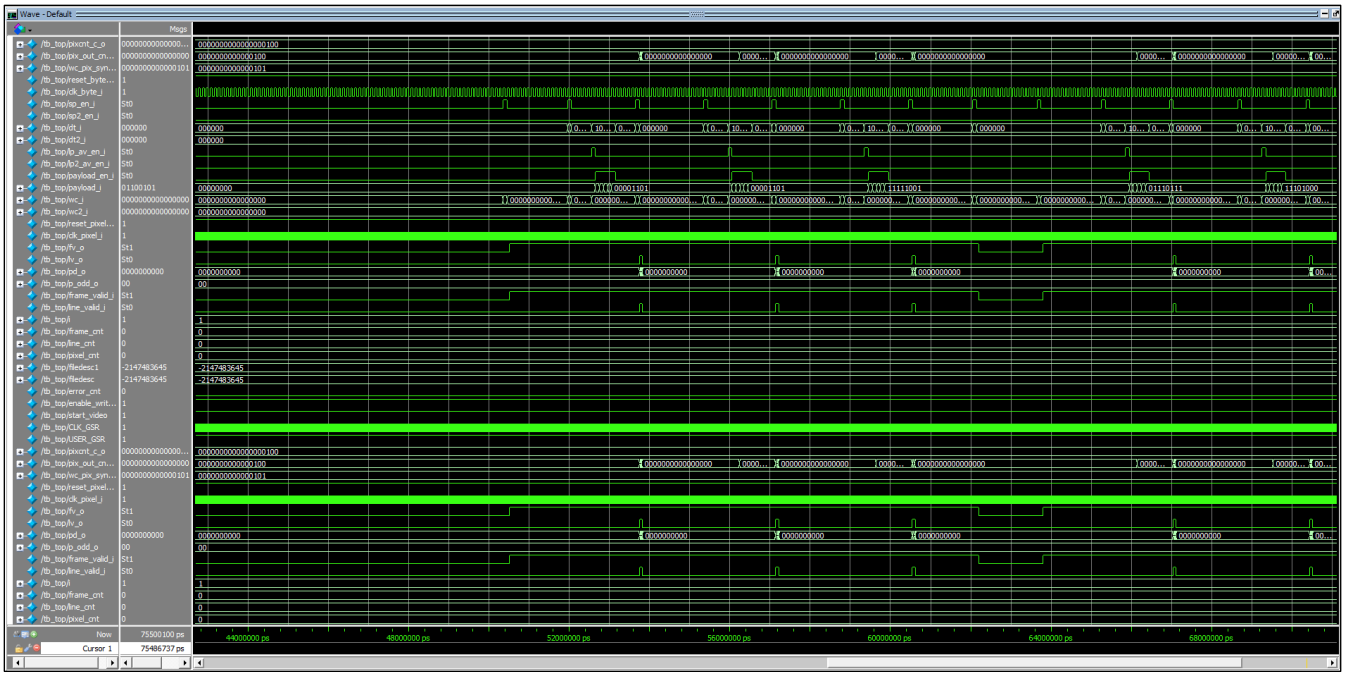

Figure 6.8. Simulation Waveform

6.5.2. Simulation Results

Simulating the IP, the expected result is shown in Figure 6.9.

```
VSIM 4> run -all
# ..... Transmitting Data .....
# number of num_bytes:     1
# number of num_bytes:     1
# number of num_bytes:     1
# number of num_bytes:     1
# number of num_bytes:     1
# number of num_bytes:     1
# ..... Transmit DONE! .....
# ------------------------------------------------
# ------------------------------------------------
##### DATA COMPARING IS STARTED #####
# ------------------------------------------------
# ------------------------------------------------
# ------------------------------------------------
# **** I N F O : Pixel Count is 24
# **** I N F O : NUM_FRAMES=2, NUM_LINES=3, Word Count=5
# ------------------------------------------------
# ------------------------------------------------------
# ----------------- SIMULATION PASSED ------------------
# ------------------------------------------------------
```

Figure 6.9. Simulation Result

7. Debugging

7.1. Debug Interface Ports

Optional interface for debugging purposes only.

Table 7.1. Debug Interface Ports

Port	Type	Description
write_cycle_o[3:0][1,2]	Output	Data write cycle to the FIFO
mem_we_o	Output	Active high data Write Enable to the FIFO
read_cycle_o[1:0][1,3]	Output	Data read cycle to the FIFO
mem_re_o	Output	Active high data Read Enable to the FIFO
fifo_empty_o	Output	Indicates FIFO empty condition
fifo_full_o	Output	Indicates FIFO full condition

Notes:

1. These signals are different per data type.
2. *write_cycle_o* signal is not continuous because the data is accumulated before writing to the FIFO.
3. *read_cycle_o* signal is not continuous because the logic will remap the data first after reading from the FIFO.

7.2. Debugging Using Debug Ports (Sample Configuration)

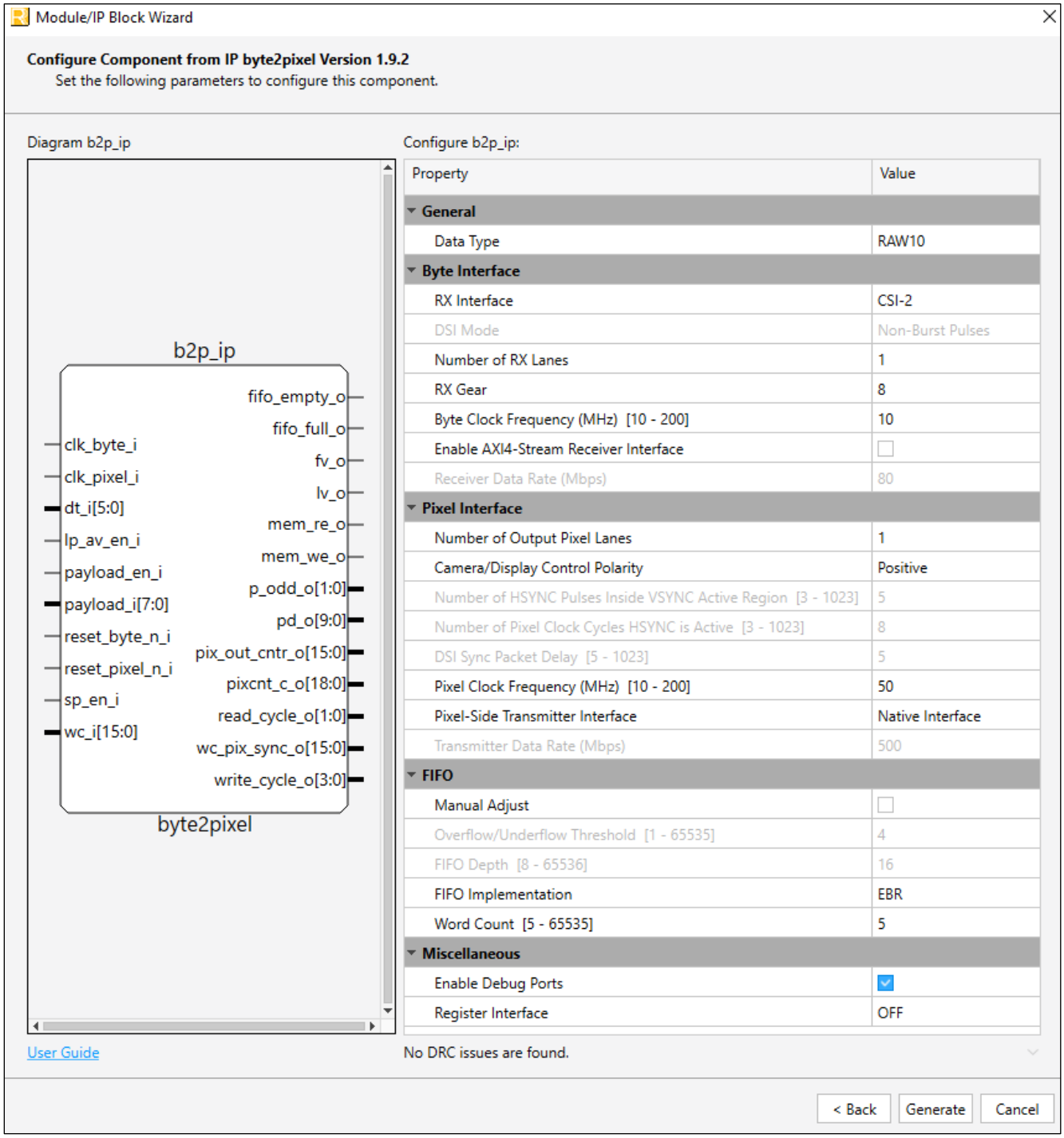

Figure 7.1. Sample Configuration with Enabled Debug Ports

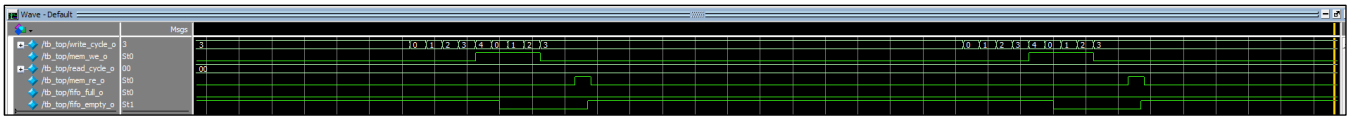

Figure 7.2. Debug Ports Simulation Using the Sample Configuration

***Disclaimer:** Enabled debug ports expose internal debug signals for ease in the debugging process.

8. Design Considerations

8.1. Design Considerations of Byte-to-Pixel Converter as a Standalone IP and/or Connected to Different Video IPs

- Verify the supported MIPI CSI-2-compatible video formats.
- Verify the supported MIPI DSI-compatible video formats.
- Verify the AXI4-Stream write and read transactions.

8.2. Limitations

- AXI4-Stream interface does not support back-pressure.
- The Byte-to-Pixel Converter IP does not support small horizontal blanking for both DSI and CSI-2. Ensure that new byte transmissions do not overlap with the ongoing output pixel stream. This can be achieved by increasing the blanking period in between lines and frames.
- If *Unified Video Streaming Tx* is enabled, data compare function is not supported in your testbench. You can verify the expected and actual output in the Simulation Waveform.
- When *Unified Video Streaming Tx* is enabled, ensure that axis_vid_tready_i is driven such that the pixel write and read operations per frame in the UVSI FIFO do not overlap. You have the option to access the CSR to check if the video packet dimensions are correct or no error flags are asserted.
- Some IP configurations may have slower Fmax when used in devices with a slow speed grade. The following Fmax values are approximate and may vary depending on the system-level design:
  - Lattice Nexus™ devices: 160 MHz
  - Lattice Avant and Nexus 2 devices: 220 MHz

Appendix A. Resource Utilization

Table A.1 and Table A.2 show the maximum frequency and resource utilization of the LFCPNX-100-7LFG672C device for a certain IP configuration.

Table A.1. Device and Tool Tested

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

Table A.2. LFCPNX-100-7LFG672C Device Resource Utilization

Configuration	Byte Interface	Pixel Interface	FMax (MHz)	LUTs	Registers	sysMem EBRs	Programmable I/O
Default	Native	Native	200	531	284	1	62
RGB888, Word Count = 720, Others = Default	AXI4-Stream	AXI4-Stream	200	496	308	1	62
RGB888, Number of RX Lanes = 4, Word Count = 2052, Others = Default	Native	Native	200	517	385	3	62
RGB888, Number of RX Lanes = 4, Word Count = 3600, Others = Default	AXI4-Stream	AXI4-Stream	200	569	380	3	62
DSI, RGB666, Word Count = 2160, Others = Default	Native	Native	188	684	399	2	62
DSI, RGB666, Number of RX Lanes = 4, Word Count = 3006, Others = Default	Native	UVSI Tx	174	1925	1294	3	11

Note: The *distributed RAM* utilization is accounted for in the total LUT4 utilization. The actual LUT4 utilization is distribution among *logic*, *distributed RAM*, and *ripple logic*. Fmax is generated through multiple iterations of place and route.

Table A.3 and Table A.4 show the maximum frequency and resource utilization of the LN2-CT-20ES1-1CBG484C device for a certain IP configuration.

Table A.3. Device and Tool Tested

Test Parameter	Value
Software Version	Lattice Radiant software 2025.2 production build
Device Used	LN2-CT-20ES1-1CBG484C
Performance Grade	1
Synthesis Tool	Synplify Pro, October 2025

Table A.4. LN2-CT-20ES1-1CBG484C Device Resource Utilization

Configuration	Byte Interface	Pixel Interface	FMax (MHz)	LUTs	Registers	sysMem EBRs	Programmable I/O
Default	Native	Native	250	562	331	1	59
RGB888, Word Count = 720, Others = Default	AXI4-Stream	AXI4-Stream	250	537	391	1	59
RGB888, Number of RX Lanes = 4, Word Count = 2052, Others = Default	Native	Native	250	646	475	1	59
RGB888, Number of RX Lanes = 4, Word Count = 3600, Others = Default	AXI4-Stream	AXI4-Stream	250	665	478	1	59
DSI, RGB666, Word Count = 2160, Others = Default	Native	Native	250	747	493	1	59
DSI, RGB666, Number of RX Lanes = 4, Word Count = 3006, Others = Default	Native	UVSI Tx	224	1979	1452	2	8

Note: The *distributed RAM* utilization is accounted for in the total LUT4 utilization. The actual LUT4 utilization is distribution among *logic*, *distributed RAM*, and *ripple logic*. Fmax is generated through multiple iterations of place and route.

References

For more information, refer to:

- Lattice Radiant Timing Constraints Methodology (FPGA-AN-02059)
- Byte-to-Pixel Converter IP Release Notes (FPGA-RN-02019)
- 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
- Lattice Radiant Software web page
- Byte to Pixel Converter IP Core web page
- Lattice Solutions IP Cores web page
- MIPI Camera Serial Interface 2 web page
- MIPI Display Serial Interface web page
- AMBA 4 AXI4-Stream Protocol Specification web page
- Lattice Insights web page for Lattice Semiconductor training courses and learning plans

Technical Support Assistance

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

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

Revision History

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 2.2, IP v1.9.2, December 2025

Section	Change Summary
All	• Updated the IP version on the cover page. • Added a note on the IP version in the *Quick Facts* and *Revision History* sections. • Made editorial fixes.
Introduction	• Updated *Lattice Implementation* in Table 1.1. Summary of the Byte-to-Pixel Converter IP. • Updated the Licensing and Ordering Information section and removed the *Ordering Part Number* section. • Added the Attribute Names section.
Functional Description	Updated the descriptions for *Overflow/Underflow Threshold* and *FIFO Depth* in Table 3.1. General Attributes.
Designing with the IP	• Added a note on screenshots in this section. • Updated Figure 6.1. Module/IP Block Wizard – Figure 6.3. Check Generated Result.
Debugging	Updated Figure 7.1. Sample Configuration with Enabled Debug Ports.
Resource Utilization	• Updated resource utilizations for the Lattice Radiant software version 2025.2. • Updated the notes for Table A.2. LFCPNX-100-7LFG672C Device Resource Utilization and Table A.4. LN2-CT-20ES1-1CBG484C Device Resource Utilization.

Revision 2.1, IP v1.9.1, June 2025

Section	Change Summary
All	• Updated the IP version on the cover page. • Made editorial fixes.
Abbreviations in This Document	Updated the definition for *RTL*.
Introduction	• Updated Table 1.1. Summary of the Byte-to-Pixel Converter IP and Table 1.3. Ordering Part Number. • Added an introductory paragraph to the IP Support Summary section.

Functional Description	• Updated the descriptions in the following sections:
	  • IP Architecture Overview
	  • Clocking
	  • Reset
	  • Reset Overview
	  • Initialization and Reset Sequence
	  • AXI4-Stream Transmitter Interface
	  • Unified Video Streaming Tx Interface
	  • AXI4-Lite Interface
	• Added Figure 2.1. Byte-to-Pixel IP Block Diagram.
	• Updated the following figures:
	  • Figure 2.2. Functional Block Diagram
	  • Figure 2.3. Clock Domain Crossing Block Diagram
	  • Figure 2.4. Reset Domain Block Diagram (including the caption)
	  • Figure 2.7. Input Timing Diagram for Byte Domain (including the description)
	  • Figure 2.13. AXI4-Stream Receiver Interface Diagram
	  • Figure 2.14. AXI4-Stream Transmitter Interface Diagram
	  • Figure 2.15. Unified Video Streaming Tx Timing Diagram
	  • Figure 2.16. Unified Video Streaming Tx Timing Diagram with De-assertion of TREADY
	• Removed the following figures:
	  • *Figure 2.2. Byte-to-Pixel Converter IP Block Diagram with AXI4-Stream Receiver*

Section	Change Summary
	Interface Enabled • *Figure 2.3. Byte-to-Pixel Converter IP Block Diagram with AXI4-Stream Transmitter Interface Enabled* • *Figure 2.4. Byte-to-Pixel Converter IP Block Diagram with Both AXI4-Stream Interfaces Enabled* • *Figure 2.5. Byte-to-Pixel Converter IP Block Diagram with Both Native Interfaces Enabled* • Updated the equivalent signal for *axis_vid_tdata_o* in Table 2.3. Data Mapping of Unified Video Streaming Tx Ports.
IP Parameter Description	Updated the following attributes in Table 3.1. General Attributes: • *Byte Clock Frequency* • *Receiver Data Rate* • *DSI Sync Packet Delay* • *Pixel Clock Frequency* • *Pixel-Side Transmitter Interface* • *Transmitter Data Rate* • *Enable Debug Ports* • *Register Interface*
Signal Description	Updated the following tables: • Table 4.1. Clock Interface Signal Descriptions • Table 4.2. Reset Interface Signal Descriptions • Table 4.3. Byte Domain Signal Descriptions • Table 4.4. AXI4-Stream Receiver Interface Signal Descriptions[1] • Table 4.5. Pixel Domain Interface Signal Descriptions (only the Notes section) • Table 4.6. AXI4-Stream Transmitter Interface Signal Descriptions[1]
Register Description	• Updated the description for *ACT_WIDTH*, *ACT_HEIGHT*, and *LOCKED* in Table 5.2. Configuration Register Address Map. • Updated *Rsvd* to *rsvd* in Table 5.3. CORE_FIFO_REG Register. • Updated the *act_width* description in Table 5.5. ACT_WIDTH Register. • Updated the *act_height* description in Table 5.6. ACT_HEIGHT Register. • Updated the *locked* description in Table 5.7. LOCKED Register.
Designing with the IP	• Updated the following figures: • Figure 6.1. Module/IP Block Wizard • Figure 6.2. IP Configuration • Figure 6.3. Check Generated Result • Figure 6.4. Clock Constraining in Constraint PDC File • Figure 6.5. Adding Constraint in PDC File • Figure 6.6. Simulation Wizard • Figure 6.7. Add and Reorder Source • Updated the description in the Timing Constraints section. • Added the Specify the Strategy section.
Debugging	Updated Figure 7.1. Sample Configuration with Enabled Debug Ports.
Design Considerations	Updated the description on small horizontal blanking in the Limitations section.
Resource Utilization	Updated resource utilizations for the Lattice Radiant software version 2025.1.

Revision 2.0, IP v1.9.0, February 2025

Section	Change Summary
All	• Added the IP version information on the cover page. • Made editorial fixes.
Abbreviations in This Document	• Added *Bits per Color* (*BPC*), *Bits per Pixel* (*BPP*), *Control and Status Registers* (*CSR*), *Colors per Pixel* (*CPP*), *End of Line* (*EOL*), *First In, First Out* (*FIFO*), *Intellectual Property* (*IP*), *Least*

Section	Change Summary
	Significant Bit (*LSB*), *Most Significant Bit* (*MSB*), *Red Green Blue* (*RGB*), *Receiver* (*Rx*), *Start of Frame* (*SOF*), and *Transmitter* (*Tx*), and *Unified Video Streaming Interface* (*UVSI*).

Introduction	• Updated the following items:
	  • Figure 1.1. Byte-to-Pixel Convertor IP General Diagram
	  • Table 1.1. Summary of the Byte-to-Pixel Converter IP
	• Replaced the *IP Validation Summary* section with the IP Support Summary section.
	• Added the *Unified Video Streaming interface*, and *AXI4-Lite interface* feature supports in the Features section.
	• Added the *Certus-N2* OPNs to Table 1.3. Ordering Part Number.
	• Added the Hardware Support section and updated the heading numbers of remaining sections accordingly.

Functional Description	• Updated the IP Architecture Overview section.
	• Updated Figure 2.1. Byte-to-Pixel IP Functional Diagram and Figure 2.6. Clock Domain Crossing Block Diagram.
	• Updated the Reset section and added Figure 2.7. Byte-to-Pixel Converter IP Reset Domain Block Diagram.
	• Added the Reset Overview and Initialization and Reset Sequence sections.
	• Updated the *Byte/Pixel-side Native Interface* and *AXI4-Stream Transmitter Interface* descriptions, and added the *Unified Video Streaming Tx Interface* and *Control* interface to Table 2.1. User Interfaces and Supported Protocols.
	• In the Native Interfaces section:
	  • Added Table 2.2. Pixel Data Bus Width for Supported Data Types.
	  • Added descriptions on the RGB output ports feature and color arrangements.
	  • Added Figure 2.13. RGB88 Color Remapping with Number of Output Pixel Lanes == 1, Figure 2.14. RGB888 Color Remapping with Number of Output Pixel Lanes == 2, and Figure 2.15. RGB888 Color Remaining with Number of Output Pixel Lanes == 4.
	• Added the Unified Video Streaming Tx Interface, AXI4-Lite Interface, and Pixel Monitor Module sections and updated the heading numbers of remaining sections accordingly.
	• Updated the AXI4-Stream Transmitter Interface and FIFO Implementation sections.

IP Parameter Description	• In Table 3.1. General Attributes:
	  • Added *Pixel-Side Transmitter Interface*, *Enable Byte Swap per Pixel*, and *Register Interface*.
	  • Removed *Enable AXI4 Stream Transmitter Interface*.
	  • Updated *Byte Clock Frequency*, *Receiver Data Rate*, *Camera/Display Control Polarity*, *Number of HSYNC Pulses Inside VSYNC Active Region*, *Number of Pixel Clock Cycles HSYNC is Active*, *DSI Sync Packet Delay*, *Pixel Clock Frequency*, *Transmitter Data Rate*, *Manual Adjust*, *Overflow/Underflow Threshold*, *FIFO Depth*, *Word Count*, and *Enable Debug Ports*.
	  • Renamed the *Debug* section to *Miscellaneous*.
	• Added *and CSI-2* to the Supported Configurations for DSI and CSI-2 section title.
	• Removed the Supported Configurations for CSI-2 section title.
	• Added *Note* to Table 3.2. Supported Configurations for DSI.
	• Updated the *Data Type* for *1 output pixel* and *2 output pixel*, and *Notes* to Table 3.3. Supported Configurations for CSI-2.
	• Added *YUV420 8-bit CSPS* and *YUV420 10-bit CSPS* to Table 3.4. Pixel and Byte Count Restriction.

Signal Description	• Updated the following tables:
	  • Table 4.1. Clock Interface Signal Descriptions
	  • Table 4.2. Reset Interface Signal Descriptions
	  • Table 4.3. Byte Domain Signal Descriptions
	  • Table 4.4. AXI4-Stream Receiver Interface Signal Descriptions
	  • Table 4.5. Pixel Domain Interface Signal Descriptions
	  • Table 4.6. AXI4-Stream Transmitter Interface Signal Descriptions

Section	Change Summary
	• Table 4.8. Debug Ports • Added the Unified Video Streaming Tx Interface and AXI4-Lite Interface sections.
Register Description	Added this section.

Designing with the IP	• Updated the following figures:
	  • Figure 6.1. Module/IP Block Wizard
	  • Figure 6.2. IP Configuration
	  • Figure 6.3. Check Generated Result
	  • Figure 6.4. Clock Constraining in Constraint PDC File
	  • Figure 6.6. Simulation Wizard
	  • Figure 6.7. Add and Reorder Source
	• Updated the *rtl/<Instance Name>.sv* attribute in Table 6.1. Generated File List.
	• Updated *testbench/tb_top.sv* and added *testbench/axi4lite_master.v* to Table 6.2. Testbench File List.

Debugging	• Updated Figure 7.1. Sample Configuration with Enabled Debug Ports. • Updated the *Disclaimer* in the Debugging Using Debug Ports (Sample Configuration) section.
Design Considerations	Updated the Limitations section.
Resource Utilization	Updated this section for the Lattice Radiant software version 2024.2 information.
References	• Added the *Certus-N2*, *MIPI Camera Serial Interface 2*, *MIPI Display Serial Interface*, and *AMBA 4 AXI4-Stream Protocol Specification* web pages. • Added *Byte-to-Pixel Converter IP Release Notes (FPGA-RN-02019)*.

Revision 1.9, Lattice Radiant SW version 2024.1, June 2024

Section	Change Summary
All	• Removed *Core* from the document title. • Made editorial fixes.

Abbreviations in This Document	• Replaced *acronyms* with *abbreviations* in this section.
	• Added the following abbreviations:
	  • *Embedded Block RAM (EBR)*
	  • *Input/Output (I/O)*
	  • *Look-Up Table (LUT)*
	  • *Random Access Memory (RAM)*

Introduction	• In Table 1.1. Summary of the Byte-to-Pixel Converter IP:
	  • Updated the description for *IP Core v.1.6.x*.
	  • Added *IP Core v.1.7.x*.
	  • Added *Native Interface*.
	• Added the *AXI4-stream transmitter and receiver interface* in the Features section.
	• Updated Table 1.2. Ordering Part Number and Table 1.3. IP Validation Level.
	• Removed *_io* from the list in the Signal Names section.

Functional Description	• In the IP Architecture Overview section:
	  • Updated the layer descriptions of *Byte-to-Pixel IP* functional block.
	  • Removed the previous section header *2.1.5. Byte-to-Pixel Converter IP Interface Configurations*.
	  • Updated all figures and their captions.
	• Added *clk_pixel_i* and *clk_byte_i* to *Note* in the Reset section.
	• Moved the following previous sections into the User Interfaces and Other IP Specific Blocks/Layers/Interfaces sections:
	  • *2.1.1. Byte-to-Pixel Converter*
	  • *2.1.2. AXI4 Stream Receiver*
	  • *2.1.3. AXI4 Stream Transmitter*
	  • *2.1.4. FIFO Implementation*

Section	Change Summary

	• In Table 2.2. User Interfaces and Supported Protocols:
	  • Removed *AXI-4 Stream Receiver/Transmitter Interface*.
	  • Added *AXI-4 Stream Receiver Interface*.
	  • Added *AXI-4 Stream Transmitter Interface*.
	• Renamed and changed the header number of the previous *2.5. Timing Specifications* section to the 2.4.1. Native Interfaces section.
	• Removed the following previous section headers:
	  • *2.5.1. Input Timing*
	  • *2.5.2. Output Timing*
	• Added the Debug Interface section.

IP Parameter Description	• Updated Table 3.1. General Attributes.
	• In Table 3.3. Supported Configurations for CSI-2:
	  • Removed *Data Type RGB888* from *RX Gearing 16 – 2 lanes – 1 output pixel*.
	  • Added *Note 3*.

Signal Description	• Updated the descriptions for all ports in Table 4.1. Clock Interface Signal Descriptions and Table 4.2. Reset Interface Signal Descriptions. • Added *Signal Descriptions* to all table captions in this section except for Table 4.7. Debug Interface. • Moved *AXI4-Stream Receiver Interface* information from the Byte Domain Interface section to the newly added AXI4-Stream Receiver Interface section. • Moved *AXI4-Stream Transmitter Interface* information from the Pixel Domain Interface section to the newly added AXI4-Stream Transmitter Interface section.

Designing with the IP	• Updated the following figures:
	  • Figure 5.1. Module/IP Block Wizard
	  • Figure 5.2. IP Configuration
	  • Figure 5.3. Check Generated Result
	  • Figure 5.4. Clock Constraining in Constraint PDC file
	• Added the following attributes to Table 5.1. Generated File List:
	  • *constraints/constraint.sdc*
	  • *eval/constraint.pdc*
	• Updated the Design Implementation and Timing Constraints sections.

Design Considerations	• Updated Figure 6.1. Sample Configuration with Enabled Debug Ports. • Added the Limitations section.
Resource Utilization	Updated this section.
References	Added the following references: • *Certus-NX web page* • *CertusPro-NX web page* • *CrossLink-NX web page* • *MachXO5-NX web page* • *Avant-E web page* • *Avant-G web page* • *Avant-X web page* • *Lattice Solutions IP Cores web page*

Revision 1.8, Lattice Radiant SW version 2023.2, December 2023

Section	Change Summary
All	• Updated the document title from Byte-to-Pixel Converter IP Core – Lattice Radiant Software to Byte-to-Pixel Converter IP Core. • Reworked the document structure for clarity by re-arranging section and subsections.
Introduction	• Reworked section contents. • Reworked old Section 4 – Ordering Part Number and converted to Subsection 1.3 Licensing

Section	Change Summary
	and Ordering Information. • Added IP Validation Summary subsection. • Added Minimum Device Requirements subsection. • Reworked old Subsection 1.3 Conventions and renamed to Subsection 1.7 Naming Conventions.
Functional Description	• Added Clocking, Reset, and User Interface subsections. • Reworked old Subsection 2.4 – Modules Description and renamed to Subsection 2.1 IP Architecture.
IP Parameter Description	Reworked old Subsection 2.3 – Attributes Summary, and moved under this main section.
Signal Description	Reworked old Subsection 2.2 – Signal Description, and converted it to this main section.
Designing with the IP	• Reworked old Section 3 – IP Generation, Simulation and Validation, and converted it to this main section. • Reworked old Subsection 3.2 – Running Functional Simulation and moved to this main section. • Reworked old Subsection 3.3 - Constraining the IP and moved to this main section.
Debugging	Added this section.
Design Considerations	Added this section.
Appendix A. Resource Utilization	Reworked section contents.
Appendix B. Limitations	Removed this section.
References	Reworked section contents.

Revision 1.7, Lattice Radiant SW version 2023.1, April 2023

Section	Change Summary
All	Updated for inclusive language.
Inclusive Language	Added this section.

Introduction	• In Table 1.1. Quick Facts:
	  • Added MachXO5-NX in the Supported FPGA Family field.
	  • Added LFMXO5-25 and LIFCL-33 in the Targeted Devices field.

Functional Description	• Updated Table 2.1. Byte-to-Pixel IP Ports. • Updated Table 2.2. Attributes Table.
IP Generation, Simulation, and Validation:	Updated Figure 3.1. Configure Block of Byte-to-Pixel Converter.
Ordering Part Number	Added part numbers for MachXO5-NX.
Appendix A. Resource Utilization	• Added Table A.3. Lattice Avant Device and Tool Tested. • Added Table A.4. Resource Utilization using Lattice Avant.
Technical Support Assistance	Added reference link to the Lattice Answer Database.

Revision 1.6, Lattice Radiant SW version 2022.1, November 2022

Section	Change Summary

Introduction	• In Table 1.1. Quick Facts:
	  • Added Lattice Avant in the Supported FPGA Family field.
	  • Added LAV-AT-500E in the Targeted Devices field.

IP Generation, Simulation, and Validation:	• Revised the title from 'IP Generation and Evaluation' to 'IP Generation, Simulation, and Validation'. • Deleted the section 'Licensing the IP'. • Revised the title of section 3.1 from 'Generation and Synthesis' to 'Generating the IP'. • Added Constraining the IP section. • Added IP Evaluation section.

Section	Change Summary
Ordering Part Number	Added part numbers for Lattice Avant-E

Revision 1.5, Lattice Radiant SW version 3.1, November 2021

Section	Change Summary
Functional Description	• Added DSI Sync Packet Delay to Table 2.2. Attributes Table. • Updated the descriptions of timing diagrams in the Timing Specifications section.

Revision 1.4, Lattice Radiant SW version 3.0, June 2021

Section	Change Summary
Introduction	Updated Table 1.1 to include CertusPro-NX support.
Ordering Part Number	Added part number for CertusPro-NX.

Revision 1.3, Lattice Radiant SW version 2.1, November 2020

Section	Change Summary
Introduction	Updated reference to the Lattice Radiant Software User Guide.
Functional Description	• Added ports to Table 2.1. Byte-to-Pixel IP Ports. • Added RAW14 and RAW16 selectable values to Table 2.2. Attributes Table. • Added RAW12, RAW14, and RAW16 data types to Table 2.4. Supported Configuration for CSI-2. • Added RAW14 and RAW16 data types to Table 2.5. Pixel and Byte Count Restriction. • Updated FIFO Implementation section.
Core Generation, Simulation, and Validation	Updated reference to the Lattice Radiant Software User Guide
References	Updated reference to the Lattice Radiant Software User Guide

Revision 1.2, Lattice Radiant SW version 2.0, August 2020

Section	Change Summary
Acronyms in This Document	Updated content.
Introduction	• Updated Table 1.1. • Updated Features section.
Functional Description	• Updated Table 2.1, Table 2.2, Table 2.3, Table 2.4, and Table 2.5. • Updated FIFO Implementation section.
IP Generation and Evaluation	• Updated Figure 3.1 and Figure 3.4. • Added Required Post-Synthesis Constraints section.
Appendix A. Resource Utilization	Updated section content including Table A.1 and Table A.2.
Appendix B. Limitations	Added this section.

Revision 1.1, Lattice Radiant SW version 2.0, February 2020

Section	Change Summary
Introduction	Updated Table 1.1 to add LIFCL-17 as targeted device.
Functional Description	• Updated descriptions for p_odd_o[1:0], axis_mdata_o[MASTER_DATA_W-1:0], and for axis_sdata_i[SLAVE_DATA_W-1:0] in Table 2.1. Byte-to-Pixel IP Ports. • Updated Transmitter and FIFO attributes in Table 2.2. Attributes Table. • Changed caption to Figure 2.6. Byte-to-Pixel IP FIFO Diagram. • Updated formulas in FIFO Implementation section.

Revision 1.0, Lattice Radiant SW version 2.0, November 2019

Section	Change Summary
All	Changed document status from Preliminary to final.
Introduction	65536axUpdated Table 1.1. Quick Facts.
Functional Description	Updated Receiver and Transmitter attributes in Table 2.2. Attributes Table.
Ordering Part Number	Added this section.
Appendix A. Resource Utilization	Added this section.

Revision 0.80, Lattice Radiant SW version 2.0, October 2019

Section	Change Summary
All	Preliminary release

www.latticesemi.com