MIPI DSI/CSI-2 to OpenLDI LVDS Interface Bridge

Reference Design

FPGA-RD-02060-1.1

September 2019

Disclaimers

Contents

Figures

Tables

Acronyms in This Document

A list of acronyms used in this document.

Acronym	Definition
AP	Application Processor
DSI	Display Serial Interface
CSI-2	Camera Serial Interface 2
DDR	Double Data Rate
FV	Frame Valid
GPLL	General Purpose PLL
HBP	Horizontal Back Porch
HFP	Horizontal Front Porch
HS	High Speed
LP	Low Power
LV	Line Valid
LVDS	Low Voltage Differential Signal
MIPI	Mobile Industry Processor Interface
OpenLDI	Open LVDS Display Interface
PLL	Phase Locked Loop
RD	Reference Design
RX	Receiver
TX	Transmitter
VBP	Vertical Back Porch
VFP	Vertical Front Porch
VCO	Voltage Controlled Oscillator

Supported Device and IP

This reference design supports following devices with IP versions shown below.

Device Family	Part Number	Compatible IP
CrossLink	LIF-MD6000 LIA-MD6000	D-PHY Receiver IP version 1.2 and 1.3 Byte-to-Pixel Converter IP version 1.1 and 1.2 LVDS Transmitter Interface IP version 1.1 and 1.2
CrossLinkPlus	LIF-MDF6000	D-PHY Receiver IP version 1.3 Byte-to-Pixel Converter IP version 1.2 LVDS Transmitter Interface IP version 1.2

CrossLink refers to both CrossLink and CrossLinkPlus in this document unless noted.

1. Introduction

Mobile Industry Processor Interface (MIPI®) Display Serial Interface (DSI) is one of the most popular display interface in the consumer market today. On the other hand, OpenLDI LVDS is still popular in some areas as the main predecessor of display interface. The majority of image sensors and application processors (AP) in the consumer market use MIPI Camera Serial Interface 2 (CSI-2) as a video signal interface. In some cases, the interface and/or format conversion is useful to connect devices which cannot connect directly.

The Lattice Semiconductor MIPI DSI/CSI-2 to OpenLDI LVDS Interface Bridge reference design for CrossLink™ devices takes DSI or CSI-2 MIPI data and converts them to OpenLDI format on LVDS. The MIPI RX module can also be realized by a MIPI hard macro IP or soft macro utilizing general DDR modules (D-PHY Soft IP) while LVDS TX module is realized by soft macro utilizing general DDR modules.

1.1. Features

- Single DSI input (RGB888 or RGB666) to single or dual channel LVDS outputs (RGB888 or RGB666)
- Single CSI-2 input (RGB888, RAW8, RAW10, or RAW12) to single or dual channel RGB888 LVDS outputs (RGB888)
- RX channel can have one, two, or four lanes with the bandwidth up to 1.2 Gbps per lane using RX D-PHY Soft IP. The bandwidth can be up to 1.5 Gbps per lane when the RX Hard D-PHY IP is used.
- Number of TX data lanes is three (RGB666) or four (RGB888) per TX channel
- Maximum TX bandwidth is 1.2 Gbps per lane
- Image cropping option is available in case of CSI-2 input
- Dynamic parameter setting is possible via I^{2}C in case of CSI-2 input

1.2. Block Diagram

Figure 1.1 shows the block level diagram of the MIPI DSI/CSI-2 to OpenLDI LVDS Interface Bridge reference design with single TX channel. It is not recommended to use Hard D-PHY for RX channel due to IP issues until the new version (1.4) of RX D-PHY IP is released unless dual channel TX with RGB888 is required. The current version of RX Hard D-PHY IP might cause simulation issues as well as functional issues. Please contact Lattice if you need to use RX Hard D-PHY IP.

There exist two main clock domains for main video data path: byte clock and pixel clock. GPLL is required to generate the edge clock of LVDS TX module. Also, it might be used to generate the continuous byte clock when RX D-PHY is in HS_LP mode. I^{2}C Slave modules enables parameter change on the fly if necessary for CSI-2 RX.

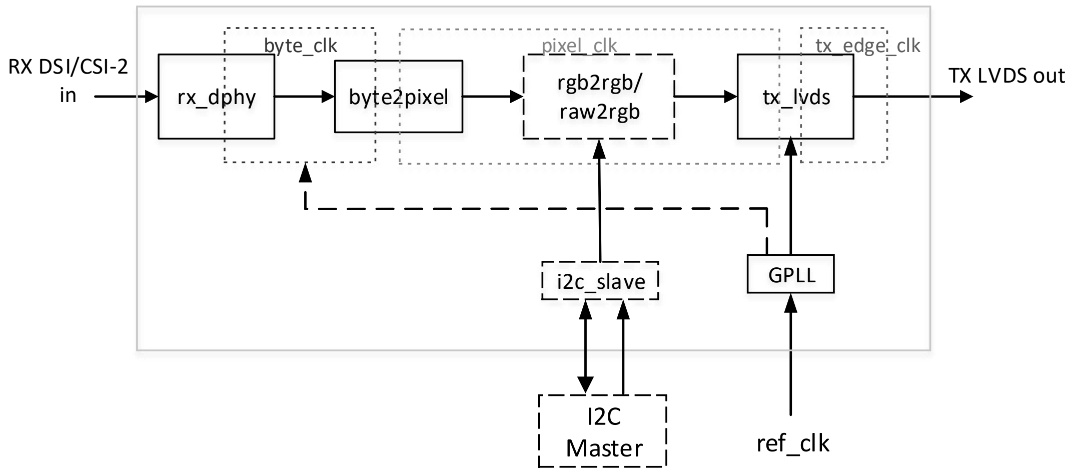


Figure 1.1. DSI/CSI-2 to OpenLDI LVDS Interface Bridge Block Diagram

1.3. RX and TX Permutations

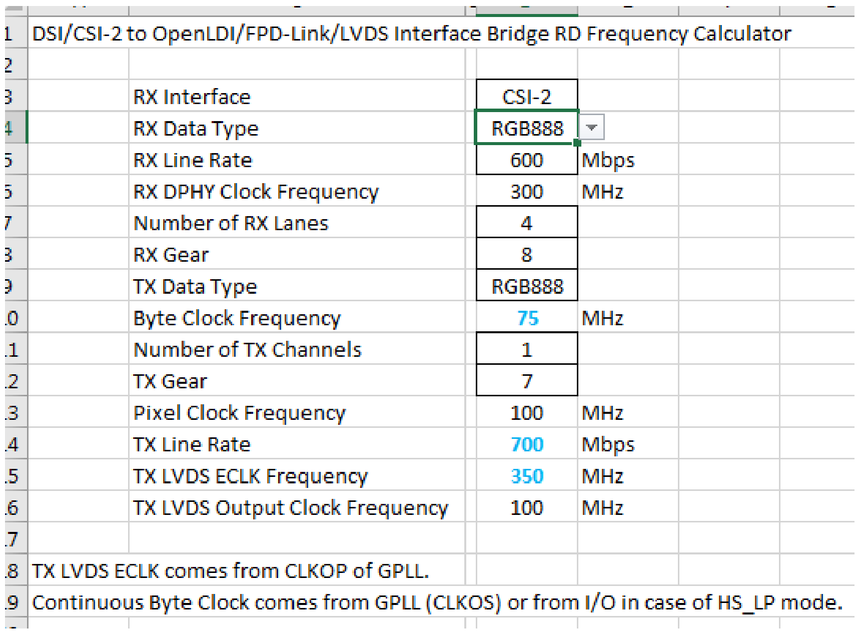
Table 1.1 shows the available permutations of RX and TX configurations. Some permutations exist only for DSI and some only for CSI-2 due to the limitations of the Byte-to-Pixel IP used in this reference design. Note that TX Data Type is the same as the RX Data Type in the case of DSI. The TX Data Type is always RGB888 in the case of CSI-2.

In case of dual channel TX with RGB888, ten LVDS output pairs are required. This means that the two I/O Banks have to be set to 2.5 V since Bank 1 has only 14 I/O and Bank 2 has only 16 I/O (LIF-MD6000-6KMG80I). On the other hand, one of these two Banks has to be set to 1.2 V when using Soft D-PHY on MIPI RX channel. To avoid this, Hard D-PHY IP has to be used on RX channel. Known issues in the current version (1.2) of RX Hard D-PHY IP created by Clarity Designer could introduce problems. Consult with Lattice if you need to use RX Hard D-PHY IP.

Table 1.1. RX and TX Permutation

D-PHY Type	Data Type	Number of RX Lanes	RX Gear	Number of Pixels/ Pixel Clock	TX Gear	Number of TX Channel
DSI	RGB888 or RGB666	1	8	1	7	1
			16	1	7	1
		2	8	1	7	1
			16	1	7	1
				2	7	2
					14	1
		4	8	1	7	1
				2	7	2
					14	1
			16	2	7	2
					14	1
				4	14	2
CSI-2	RGB888, RAW8, RAW10, or RAW12	1	8	1	7	1
			16	1	7	1
				2	7	2
					14	1
		2	8	1	7	1
				2	7	2
					14	1
			16	1	7	1
				2	7	2
					14	1
		4	8	1	7	1
				2	7	2
					14	1
			16	2	7	2
					14	1

The Microsoft Excel sheet (mipi2lvds_clock.xlsx) is provided to get the byte clock, TX edge clock, and others from RX bandwidth, in addition to other information. This sheet is useful in configuring IPs. A sample entry is shown in Figure 1.2. By entering MIPI bandwidth and other information, Byte clock, LVDS bandwidth, and TX ECLK are automatically calculated. The results can be used to configure TX LVDS IP and GPLL.



1	DSI/CSI-2 to OpenLDI/FPD-Link/LVDS Interface Bridge RD Frequency Calculator		
2			
3	RX Interface	CSI-2	
4	RX Data Type	RGB888	
5	RX Line Rate	600	Mbps
5	RX DPHY Clock Frequency	300	MHz
7	Number of RX Lanes	4	
3	RX Gear	8	
9	TX Data Type	RGB888	
.0	Byte Clock Frequency	75	MHz
.1	Number of TX Channels	1	
.2	TX Gear	7	
.3	Pixel Clock Frequency	100	MHz
.4	TX Line Rate	700	Mbps
.5	TX LVDS ECLK Frequency	350	MHz
.6	TX LVDS Output Clock Frequency	100	MHz
.7			
.8	TX LVDS ECLK comes from CLKOP of GPLL.		
.9	Continuous Byte Clock comes from GPLL (CLKOS) or from I/O in case of HS_LP mode.		

Figure 1.2. Bandwidth and Clock Frequency Calculator

2. Parameters and Port List

There are two directive files for this reference design:

- synthesis_directives.v – used for design compilation by Lattice Diamond® and for simulation.
- simulation_directives.v – used for simulation.

You can modify these directives according to your own configuration. The settings in these files must match RX D-PHY IP, Byte-to-Pixel IP, and TX LVDS IP settings created by Clarity Designer.

2.1. Synthesis Directives

Table 2.1 shows the synthesis directives that affect this reference design. These are used for both synthesis and simulation. As shown in Table 2.1 and Table 2.2, some parameter selections are restricted by other parameter settings.

Table 2.1. Synthesis Directives

Category	Directive	Remarks
External reference clock[1]	EXT_REF_CLK	Enable this when the reference clock is fed from a pin.
RX D-PHY Type	RX_TYPE_DSI	Define the D-PHY type on the RX channel. Only one of these directives must be defined.
	RX_TYPE_CSI2	
RX Data Type	RX_RGB888	Define the data type on the RX channel. Only one of these directives must be defined. In case of DSI, only RGB888 and RGB666 are allowed. In case of CSI-2, only RGB888, RAW8, RAW10, and RAW12 are allowed.
	RX_RGB666	
	RX_RAW8	
	RX_RAW10	
	RX_RAW12	
Number of RX Channel Lanes	NUM_RX_LANE_1	Number of lanes in the RX channel. Only one of these three directives must be defined.
	NUM_RX_LANE_2	
	NUM_RX_LANE_4	
RX D-PHY Clock Gear	RX_GEAR_8	Number of RX clock gears on the RX channel. Only one of these directives must be defined.
	RX_GEAR_16	
RX Hard D-PHY	RX_DPHY_HARD	Enable this to use Hard D-PHY. If undefined, Soft D-PHY is used. Not recommended for use until the new version (1.4) of RX D-PHY IP is released.
RX D-PHY Clock Mode[2]	RX_CLK_MODE_HS_ONLY	RX D-PHY clock mode. Only one of these two directives must be defined.
	RX_CLK_MODE_HS_LP	
Hard D-PHY Word Alignment	WORD_ALIGN	Enable word aligner in RX Hard D-PHY IP. Ignored when Hard D-PHY is not used.
Line Buffer Depth	LB_DEPTH_512	Define the depth of the line buffer in case the RX Data Type is RAW8, RAW10, or RAW12. This depth must be equal or greater than the active pixel count per line, irrespective of the data type.
	LB_DEPTH_1024	
	LB_DEPTH_2048	
	LB_DEPTH_4096	
Sync Signal Polarity	SYNC_POLARITY_POS	Define sync signal (VSYNC, HSYNC) polarity. Only one of these two directives must be defined.
	SYNC_POLARITY_NEG	
Data Enable Polarity	DE_POLARITY_POS	Define data enable (DE) polarity. Only one of these two directives must be defined.
	DE_POLARITY_NEG	
TX Data Type[3]	TX_RGB888	Define the data type on TX channel. Only one of these two directives must be defined. RGB666 must be defined in case of RX_RGB666. RGB888 must be defined in all other cases.
	TX_RGB666	
Number of TX Channels	NUM_TX_CH_1	Number of channels on TX. Only one of these two directives must be defined. Two channel is only allowed for DSI input.
	NUM_TX_CH_2	
TX LVDS Gear	TX_GEAR_7	Number of TX clock gears on RX channel. Only one of these directives must be defined.
	TX_GEAR_14	

Category	Directive	Remarks
Parameter set by I^{2}C	USE_I2C	Define this to use I^{2}C I/F to set the parameters on the fly. This is applicable to CSI-2 input only.
I^{2}C Slave Address (MSB) [4]	I2C_SLAVE_ADR_MSB {value}	Define MSB 5bits of I^{2}C Slave Address. Value must be 5'h00 – 5'h1F. Applicable only when USE_I2C is defined. This value overwrites the value set in Clarity when IP is created.
Software Reset Register[5]	SW_RST_N {value}	Default value of the software reset register of I^{2}C Slave module. Value must be 1'b0 or 1'b1 Applicable only when USE_I2C is defined. Active low.
Shift Register Delay[6]	SR_DELAY {value}	Define shift register delay. This value is used as a fixed value when USE_I2C is not defined and used as the default value of the I2C register when USE_I2C is defined. Value must be 8'd2 – 8'd255. Applicable only to CSI-2 input with RGB888.
Bayer Pattern[7]	BAYER_PATTERN {value}	Define the Bayer pattern. This value is used as a fixed value when USE_I2C is not defined and used as the default value of the I^{2}C register when USE_I2C is defined. Value must be 2'b00 – 2'b11. Applicable only to CSI-2 input with RAW8/RAW10/RAW12.
Horizontal Front Porch[8]	HFP {value}	Define the horizontal front porch. This value is used as a fixed value when USE_I2C is not defined and used as the default value of the I^{2}C register when USE_I2C is defined. Value must be 8'd1 – 8'd255. Applicable only to CSI-2 input.
HSYNC Pulse Length[8]	HS_LENGTH {value}	Define HSYNC pulse length. This value is used as a fixed value when USE_I2C is not defined and used as the default value of the I^{2}C register when USE_I2C is defined. Value must be 8'd1 – 8'd255. Applicable only to CSI-2 input.
Vertical Front Porch[9]	VFP {value}	Define the vertical front porch. This value is used as a fixed value when USE_I2C is not defined and used as the default value of the I^{2}C register when USE_I2C is defined. Value must be 6'd1 – 6'd63. Applicable only to CSI-2 input.
VSYNC Pulse Length[9]	VS_LENGTH {value}	Define VSYNC pulse length. This value is used as a fixed value when USE_I2C is not defined and used as the default value of the I^{2}C register when USE_I2C is defined. Value must be 6'd1 – 6'd63. Applicable only to CSI-2 input.
Left Pixel Trimming[10]	LEFT_TRIM {value}	Define the number of pixels to be trimmed before TX. This value is used as a fixed value when USE_I2C is not defined and used as the default value of the I^{2}C register when USE_I2C is defined. Value must be 6'd0 – 6'd63. Applicable only to CSI-2 input.
Horizontal Active Pixels on TX[10]	H_TX_PEL {value}	Define the number of active pixels to send on TX. This value is used as a fixed value when USE_I2C is not defined and used as the default value of the I^{2}C register when USE_I2C is defined. Value must be 12'd1 – 12'd4095. The value must be even in case of TX_GEAR_14 or NUM_TX_CH_2. Applicable only to CSI-2 input.
Top Line Trimming[11]	TOP_TRIM {value}	Define the number of lines to be trimmed before TX. This value is used as a fixed value when USE_I2C is not defined and used as the default value of the I^{2}C register when USE_I2C is defined. Value must be 6'd0 – 6'd63. Applicable only to CSI-2 input.
Vertical Active Lines on TX[11]	V_TX_LINE {value}	Define the number of active lines to be sent on TX. This value is used as a fixed value when USE_I2C is not defined and used as the default value of the I^{2}C register when USE_I2C is defined. Value must be 12'd1 – 12'd4095. Applicable only to CSI-2 input.

Notes:

1. The external clock is required in most cases since the clock ratio between RX side and TX side is usually not simple.
2. HS_LP mode means non-continuous clock mode and HS_ONLY means continuous clock mode. HS_LP mode works only if RX byte clock can be generated internally or directly fed from I/O pin.
3. TX data type must match RX data type in case of RGB input. RGB888 output is used for all RAW input.
4. LSB two bits of I^{2}C slave address is automatically set when I^{2}C IP is created by Clarity.

5. Logical OR between this register and system reset (reset_n_i) is used to reset modules other than I^{2}C slave module.
6. Refer to the Sync Signal Generation section for details.
7. Refer to the RGB Data Creation section for pattern details.
8. (HFP + HS_LENGTH) must be less than the horizontal blanking period after the byte-to-pixel conversion. It is your responsibility to manage this. Small values are recommended if you have no idea about the length of the blanking period.
9. (VFP + VS_LENGTH) must be less than the vertical blanking period after the byte-to-pixel conversion. It is your responsibility to manage this. Small values are recommended if you have no idea about the length of the blanking period.
10. (LEFT_TRIM + H_TX_PEL) cannot exceed the horizontal active pixel count of the incoming RX data. It is your responsibility to manage this.
11. (TOP_TRIM + V_TX_LINE) cannot exceed the vertical active line count of the incoming RX data. It is your responsibility to manage this.

2.2. Simulation Directives

Table 2.2 shows the simulation directives for this reference design.

Table 2.2. Simulation Directives

Category	Directive	Remarks
Simulation setup	SIM	Define to select the right behavioral model in simulation.
Reference clock period	REF_CLK {value}	Reference clock period in ps
RX D-PHY clock period	DPHY_CLK {value}	RX DPHY clock period in ps
Number of frames to run	NUM_FRAMES {value}	Number of video frames fed by testbench
Number of active lines	NUM_LINES {value}	Number of active video lines per frame
Number of active pixels	NUM_PIXELS {value}	Number of active video pixels per line
HSYNC period in DSI	DSI_HSA_PAYLOAD {value}	HSYNC active count in the unit of payload byte. Applicable to DSI RX.
Blanking period in DSI	DSI_BLLP_PAYLOAD {value}	Blanking period in the unit of payload byte. Used in HS_ONLY mode. Applicable to DSI RX.
Horizontal Back Porch in DSI[1]	DSI_HBP_PAYLOAD {value}	Horizontal Back Porch or Low Power period in the unit of payload byte count. Used in HS_LP mode or non-burst sync pulse in HS_ONLY mode. Applicable to DSI RX.
Horizontal Front Porch in DSI[1]	DSI_HFP_PAYLOAD {value}	Horizontal Front Porch or Low Power period in the unit of payload byte count. Used in HS_LP mode or non-burst sync pulse in HS_ONLY mode. Applicable to DSI RX.
VSYNC period in DSI	DSI_VSA_LINES {value}	Number of VSYNC active lines. Applicable to DSI RX.
Vertical Back Porch in DSI	DSI_VBP_LINES {value}	Number of Vertical Back Porch lines. Applicable to DSI RX.
Vertical Front Porch in DSI	DSI_VFP_LINES {value}	Number of Vertical Front Porch lines. Applicable to DSI RX.
EOTP packet on/off in DSI	DSI_ETOP_ENABLE {value}	Enable/Disable EOTP (End of Transmission Packet). Applicable to DSI RX. Value must be 1 (enable) or 0 (disable).
Blanking period in DSI	DSI_LPS_BLLP_DURATION {value}	Blanking (LP) period in ps. Used in HS_LP mode. Applicable to DSI RX.
Horizontal Back Porch period in DSI[1]	DSI_LPS_HBP_DURATION {value}	Horizontal Back Porch period (using LP mode) in ps. Used for non-burst sync events and burst mode in HS_LP mode. Applicable to DSI RX.
Horizontal Front Porch period in DSI[1]	DSI_LPS_HFP_DURATION {value}	Horizontal Front Porch period (using LP mode) in ps. Used for non-burst sync events and burst mode in HS_LP mode. Applicable to DSI RX.
Initial delay on RX channel	READY_DURATION {value}	Initial delay to activate RX channel in ps.
Line Gap period in CSI-2	DPHY_LPS_GAP {value}	Line Gap time in ps. Applicable to CSI-2 RX.
Frame Gap period	FRAME_LPM_DELAY {value}	Frame Gap time in ps. This is LP period between Frame End and Frame Start in case of CSI-2 and LP period after VFP period end and before VSYNC start in case of DSI.

Category	Directive	Remarks
Software Reset Register[2]	I2C_SW_RST_N {value}	Write value to the software reset register of I^{2}C Slave module. Value must be 1'b0 or 1'b1. Applicable only when USE_I2C is defined. Active low.
Shift Register Delay[3]	I2C_SR_DELAY {value}	Write value to the shift register delay register of I^{2}C Slave. Value must be 8'd2 – 5'd255. Applicable only to CSI-2 input with RGB888.
Bayer Pattern[4]	I2C_BAYER_PATTERN {value}	Write value to the Bayer pattern register of I^{2}C Slave. Value must be 2'b00 – 2'b11. Applicable only to CSI-2 input with RAW8/RAW10/RAW12.
Horizontal Front Porch[5]	I2C_HFP {value}	Write value to the horizontal front porch register of I^{2}C Slave. Value must be 8'd1 – 8'd255. Applicable only to CSI-2 input.
HSYNC Pulse Length[5]	I2C_HS_LENGTH {value}	Write value to HSYNC pulse length register of I^{2}C Slave. Value must be 8'd1 – 8'd255. Applicable only to CSI-2 input.
Vertical Front Porch[6]	I2C_VFP {value}	Write value to the vertical front porch register of I^{2}C Slave. Value must be 6'd1 – 6'd63. Applicable only to CSI-2 input.
VSYNC Pulse Length[6]	I2C_VS_LENGTH {value}	Write value to VSYNC pulse length register of I^{2}C Slave. Value must be 6'd1 – 6'd63. Applicable only to CSI-2 input.
Left Pixel Trimming[7]	I2C_LEFT_TRIM {value}	Write Value to the pixel trim register of I^{2}C Slave. Value must be 6'd0 – 6'd63. Applicable only to CSI-2 input.
Horizontal Active Pixels on TX[7]	I2C_H_TX_PEL {value}	Write value to the active pixel register of I^{2}C Slave. Value must be 13'd1 – 13'd8191. The value must be even in case of TX_GEAR_14. Applicable only to CSI-2 input.
Top Line Trimming[8]	I2C_TOP_TRIM {value}	Write value to the line trim register of I^{2}C Slave. Value must be 6'd0 – 6'd63. Applicable only to CSI-2 input.
Vertical Active Lines on TX[8]	I2C_V_TX_LINE {value}	Write value to the active line register of I^{2}C. Value must be 12'd1 – 12'd4095. Applicable only to CSI-2 input.

Notes:

1. Refer to MIPI® Alliance Specification for Display Serial Interface (DSI) Version 1.1 for details.
2. Logical OR between this register and system reset (reset_n_i) is used to reset modules other than I^{2}C slave module.
3. Refer to the Sync Signal Generation section for details.
4. Refer to the RGB Data Creation section for pattern details.
5. (I2C_HFP + I2C_HS_LENGTH) must be less than the horizontal blanking period after the byte-to-pixel conversion. It is your responsibility to manage this. Small values are recommended if you have no idea about the length of the blanking period.
6. (I2C_VFP + I2C_VS_LENGTH) must be less than the vertical blanking period after the byte-to-pixel conversion. It is your responsibility to manage this. Small values are recommended if you have no idea about the length of the blanking period.
7. (I2C_LEFT_TRIM + I2C_H_TX_PEL) cannot exceed the horizontal active pixel count of the incoming RX data. It is your responsibility to manage this.
8. (I2C_TOP_TRIM + I2C_V_TX_LINE) cannot exceed the vertical active line count of the incoming RX data. It is your responsibility to manage this.

2.3. Top-Level I/O

Table 2.3 shows the top level I/O of this reference design. Actual I/O depend on the customer's channel and lane configurations. All necessary I/O ports are automatically declared by compiler directives.

Table 2.3. DSI/CSI-2 to OpenLDI LVDS Interface Bridge Top Level I/O

Port Name	Direction	Description
Clocks and Resets		
ref_clk_i	I	Input reference clock. Used to feed a clock to TX D-PHY PLL directly or indirectly. This port is declared only when EXT_REF_CLK is defined in synthesis_directives.v.
reset_n_i	I	Asynchronous active low system reset
Control Interface (optional for CSI-2 RX)		
scl	I/O	I^2C clock
sda	I/O	I^2C data
DSI/CSI-2 RX Interface		
rx_clk_p_i	I	Positive differential RX D-PHY input clock
rx_clk_n_i	I	Negative differential RX D-PHY input clock
rx_d0_p_i	I	Positive differential RX D-PHY input data 0
rx_d0_n_i	I	Negative differential RX D-PHY input data 0
rx_d1_p_i	I	Positive differential RX D-PHY input data 1 (in case of 2-lane or 4-lane configuration)
rx_d1_n_i	I	Negative differential RX D-PHY input data 1 (in case of 2-lane or 4-lane configuration)
rx_d2_p_i	I	Positive differential RX D-PHY input data 2 (in case of 4-lane configuration)
rx_d2_n_i	I	Negative differential RX D-PHY input data 2 (in case of 4-lane configuration)
rx_d3_p_i	I	Positive differential RX D-PHY input data 3 (in case of 4-lane configuration)
rx_d3_n_i	I	Negative differential RX D-PHY input data 3 (in case of 4-lane configuration)
LVDS TX Interface		
tx0_clk_p_o	O	Positive differential TX LVDS output clock
tx0_clk_n_o	O	Negative differential TX LVDS output clock
tx0_d0_p_o	O	Positive differential TX LVDS output data 0
tx0_d0_n_o	O	Negative differential TX LVDS output data 0
tx0_d1_p_o	O	Positive differential TX LVDS output data 1
tx0_d1_n_o	O	Negative differential TX LVDS output data 1
tx0_d2_p_o	O	Positive differential TX LVDS output data 2
tx0_d2_n_o	O	Negative differential TX LVDS output data 2
tx0_d3_p_o	O	Positive differential TX LVDS output data 3 (in case of TX_RGB888)
tx0_d3_n_o	O	Negative differential TX LVDS output data 3 (in case of TX_RGB888)
tx1_clk_p_o	O	Positive differential TX LVDS output clock on 2nd TX channel (in case of dual TX channel configuration)
tx1_clk_n_o	O	Negative differential TX LVDS output clock on 2nd TX channel (in case of dual TX channel configuration)
tx1_d0_p_o	O	Positive differential TX LVDS output data 0 on 2nd TX channel (in case of dual TX channel configuration)
tx1_d0_n_o	O	Negative differential TX LVDS output data 0 on 2nd TX channel (in case of dual TX channel configuration)
tx1_d1_p_o	O	Positive differential TX LVDS output data 1 on 2nd TX channel (in case of dual TX channel configuration)
tx1_d1_n_o	O	Negative differential TX LVDS output data 1 on 2nd TX channel (in case of dual TX channel configuration)
tx1_d2_p_o	O	Positive differential TX LVDS output data 2 on 2nd TX channel (in case of dual TX channel configuration)

Port Name	Direction	Description
tx1_d2_n_o	O	Negative differential TX LVDS output data 2 on 2nd TX channel (in case of dual TX channel configuration)
tx1_d3_p_o	O	Positive differential TX LVDS output data 3 on 2nd TX channel (in case of dual TX channel configuration with TX_RGB888)
tx1_d3_n_o	O	Negative differential TX LVDS output data 3 on 2nd TX channel (in case of dual TX channel configuration with TX_RGB888)

3. Design and Module Description

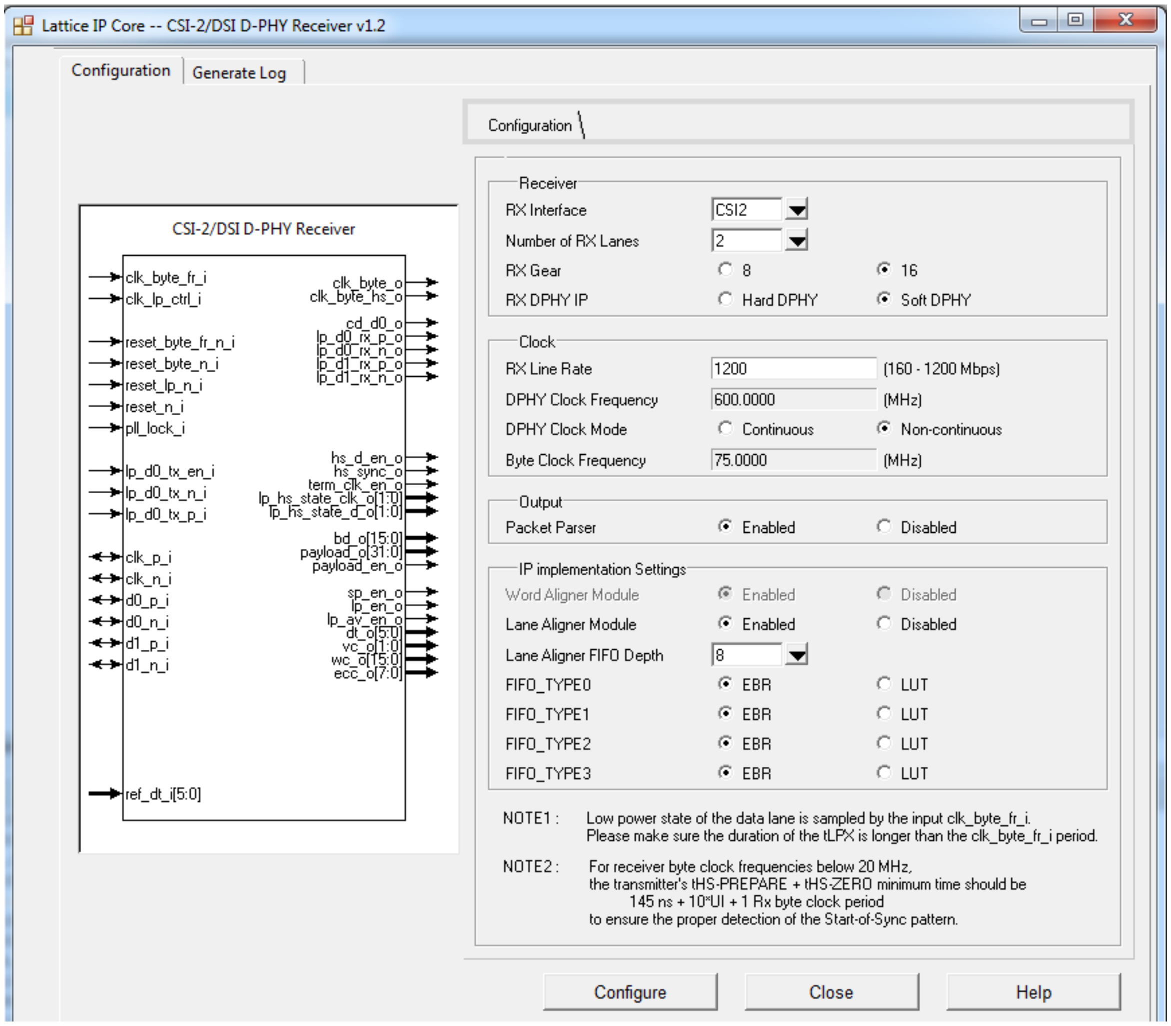
The top-level design (mipi2lvds.v) consists of the following modules:

- rx_dphy
- byte2pixel
- rgb2rgb (used only for CSI-2 RGB888)
- raw2rgb (used only for CSI-2 RAW8/RAW10/RAW12)
- tx_lvds
- int_gpll
- i2c_slave (optional for CSI-2)

The top-level design has a reset synchronization logic.

3.1. rx_dphy

This module must be created for the RX channel according to channel conditions, such as the number of lanes, bandwidth, and others. Figure 3.1 shows an example of IP interface settings in Clarity for the CSI-2/DSI D-PHY Receiver Submodule IP. You can use the .sbx file (*rx/rx.sbx*) included in the sample project and reconfigure according to your needs. Refer to CSI-2/DSI D-PHY Receiver Submodule IP User Guide (FPGA-IPUG-02025) for details.



Receiver

RX Interface	CSI2	
Number of RX Lanes	2	
RX Gear	8	[✓ 16]
RX DPHY IP	Hard DPHY	[✓ Soft DPHY]

Clock

RX Line Rate	1200	(160 - 1200 Mbps)
DPHY Clock Frequency	600.0000	(MHz)
DPHY Clock Mode	Continuous	[✓ Non-continuous]
Byte Clock Frequency	75.0000	(MHz)

Output

Packet Parser	[✓ Enabled]	Disabled

IP implementation Settings

Word Aligner Module	[✓ Enabled]	Disabled
Lane Aligner Module	[✓ Enabled]	Disabled
Lane Aligner FIFO Depth	8	
FIFO_TYPE0	[✓ EBR]	LUT
FIFO_TYPE1	[✓ EBR]	LUT
FIFO_TYPE2	[✓ EBR]	LUT
FIFO_TYPE3	[✓ EBR]	LUT

NOTE1 : Low power state of the data lane is sampled by the input clk_byte_fr_i.
Please make sure the duration of the tLPX is longer than the clk_byte_fr_i period.

NOTE2 : For receiver byte clock frequencies below 20 MHz,
the transmitter's tHS-PREPARE + tHS-ZERO minimum time should be
145 ns + 10*UI + 1 Rx byte clock period
to ensure the proper detection of the Start-of-Sync pattern.

Figure 3.1. rx_dphy IP Creation in Clarity Designer

The following provides the guidelines and parameter settings required for this reference design.

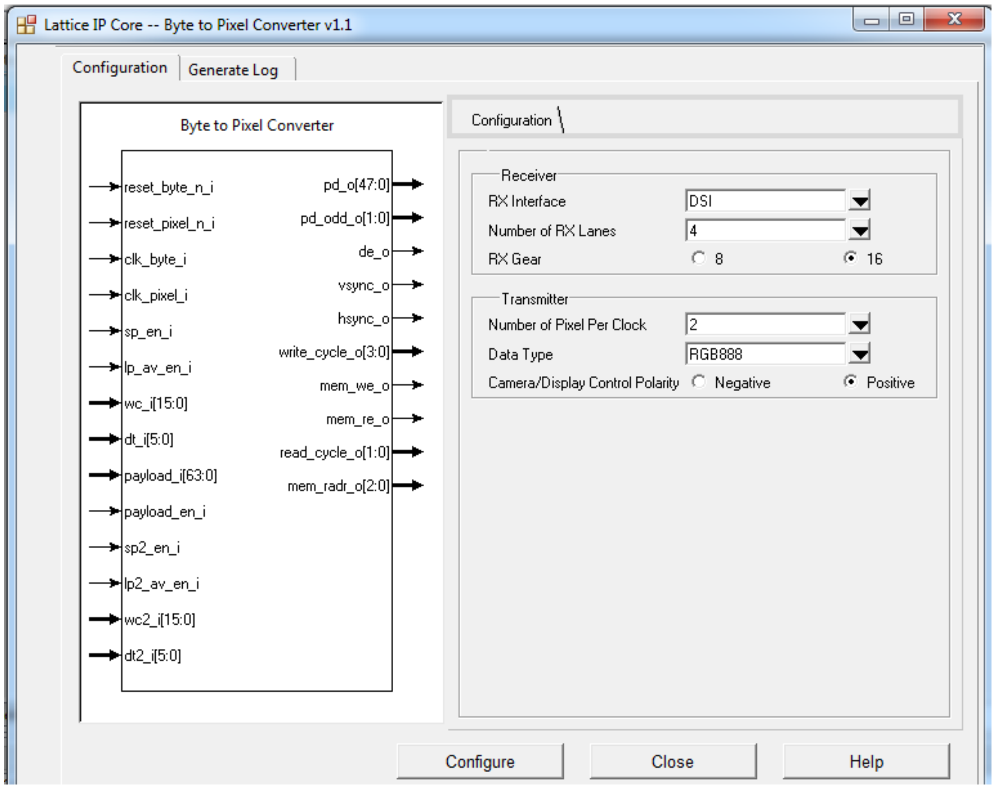
- RX Interface – Select DSI or CSI-2.
- Number of RX Lanes – Set according to channel configuration. The value must match NUM_RX_LANE_* setting.
- RX Gear – Select 8 or 16; 16 is recommended when the RX byte clock speed exceeds 100 MHz with Gear 8.
- RX D-PHY IP – The setting must match RX_D-PHY_HARD setting. Use Soft D-PHY and avoid using Hard D-PHY until the new version (1.4) of RX D-PHY IP is available.
- RX Line Rate – Set according to channel configuration. 800 or below is recommended for 4 lane configuration.
- D-PHY Clock Mode – Select Continuous or Non-continuous. Must match RX*_CLK_MODE_* setting (Continuous = HS_ONLY, Non-continuous = HS_LP).
- Packet Parser – Select Enabled.
- Word Aligner Module –Select Enabled.
- Lane Aligner Module – Select Enabled when 2 or 4 lane configuration. FIFO Depth of 8 and EBR is recommended.

This module takes serial DSI/CSI-2 data and outputs pixel data after de-serialization in DSI/CSI-2 High Speed mode and protocol decoding. If you are creating this IP from scratch, it is recommended to set the design name to *rx* and the module name to *rx_dphy*. This way, you do not need to modify the instance name of this IP in the top-level design, as well as the simulation setup file. Otherwise, you have to modify the names accordingly.

Configuring the CSI-2/DSI D-PHY Receiver Submodule IP as Hard D-PHY may cause compilation and simulation issues. Therefore, it is not recommended to use Hard D-PHY until the new version (1.4) of the IP is available, unless absolutely necessary.

3.2. byte2pixel

This module must be created for the RX channel according to D-PHY type, data type, the number of lanes, RX Gear, and others. Figure 3.2 shows an example of IP interface settings in Clarity for the byte2pixel IP. You can use the .sbx file (*b2p/b2p.sbx*) included in the sample project and reconfigure according to your needs. Refer to Byte-to-Pixel Converter IP User Guide (FPGA-IPUG-02027) for details.



Receiver
RX Interface: DSI
Number of RX Lanes: 4
RX Gear: 8 [✓ 16]

Transmitter
Number of Pixel Per Clock: 2
Data Type: RGB888
Camera/Display Control Polarity: Negative [✓ Positive]

Figure 3.2. byte2pixel IP Creation in Clarity Designer

The following provides the guidelines and parameter settings required for this reference design.

- RX Interface – Select DSI or CSI-2. Set the same type as RX D-PHY IP.
- Number of RX Lanes – Set the same value as RX D-PHY IP.
- RX Gear – Set the same value as RX D_PHY IP.
- Number of Pixel Per Clock – Set 1, 2, or 4. 4 is applicable only to two TX channels with TX Gear 14. 2 is applicable to one TX channel with TX Gear 14 or two TX channels with TX Gear 7. Selection is limited in some configurations (refer to Table 1.1).
- Data Type – Set RGB888 or RGB666 for DSI and RGB888, RAW8, RAW10, or RAW12 for CSI-2. Others are not supported in this reference design.
- Camera/Display Control Polarity – Set Positive. Polarity setting of sync and data enable is done outside of this IP.

If you are creating this IP from scratch, it is recommended to set the design name to *b2p* and the module name to *byte2pix* so that you do not need to modify the instance name of this IP in the top-level design as well as the simulation setup file. Otherwise, you have to modify the names accordingly.

This module receives byte data from RX D-PHY along with packet enable signal, among others, and reorganizes the data to form the desired pixel data via FIFO according to the data type specified. FIFO is used as a data buffer as well as clock domain conversion. Byte data are written to FIFO in byte clock domain and read in pixel clock domain. Pixel clock is provided from TX LVDS module. Depending on the D-PHY type (DSI or CSI-2), different types of sync and data enable signals are generated. Figure 3.3 shows the interface timing diagram for DSI. VSYNC, HSYNC, and DE are generated based on short packet enable and payload enable signals. In case of DSI, these signals are sent to LVDS TX module as is. Depending on the configuration, output data (pd_o) could have one, two, or four pixels per pixel clock. Figure 3.4 shows global output timing for DSI. Due to the clock domain crossing, the pulse length and/or sync signal intervals may vary slightly.

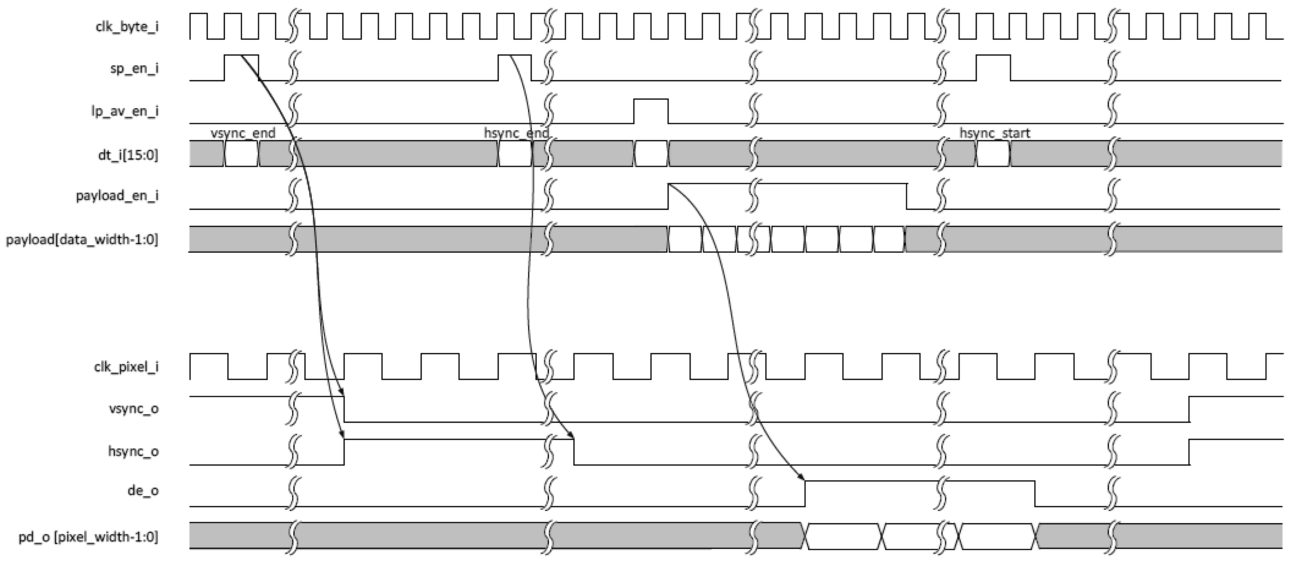


Figure 3.3. Interface Timing Diagram for DSI

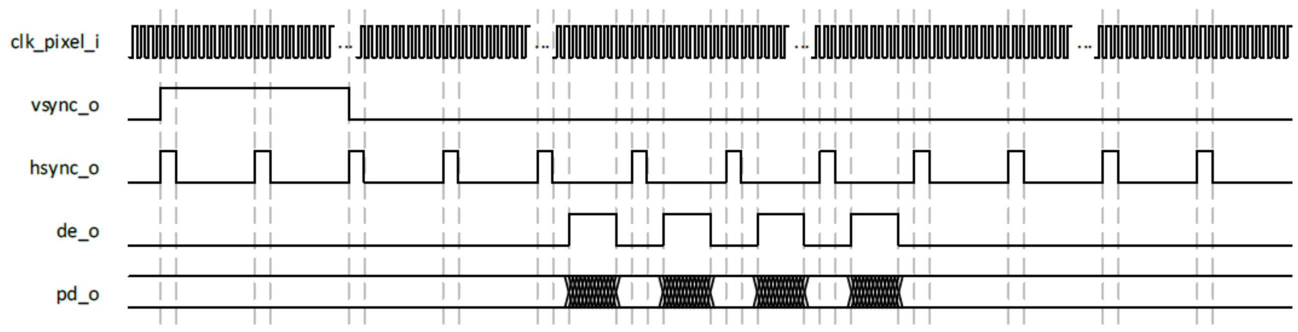


Figure 3.4. Global Output Timing Diagram for DSI

Figure 3.5 shows the interface timing diagram for CSI-2. FV and LV are generated based on short packet enable and payload enable signals. Figure 3.6 shows global output timing for CSI-2. In case of CSI-2, sync signals (VSYNC, HSYNC) have to be created from FV and LV to match the interface format of LVDS. Also in case of RAW8/RAW10/RAW12, RGB pixel data have to be created from RAW data. For these purposes, an additional module (rgb2rgb or raw2rgb) is required for CSI-2 between byte2pixel and tx_lvds. In case of TX Gear 14 or two TX channel outputs, output data (pd_o) has two pixels per pixel clock. Note that TX Gear 14 and two TX channel configuration must not happen together in case of CSI-2 (refer to Table 1.1).

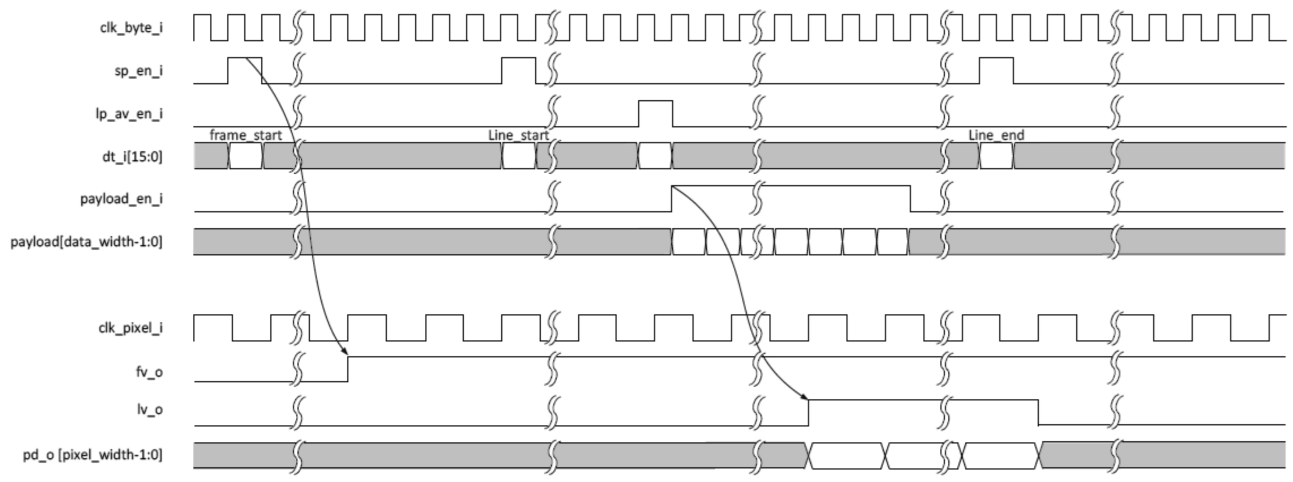


Figure 3.5. Interface Timing Diagram for CSI-2

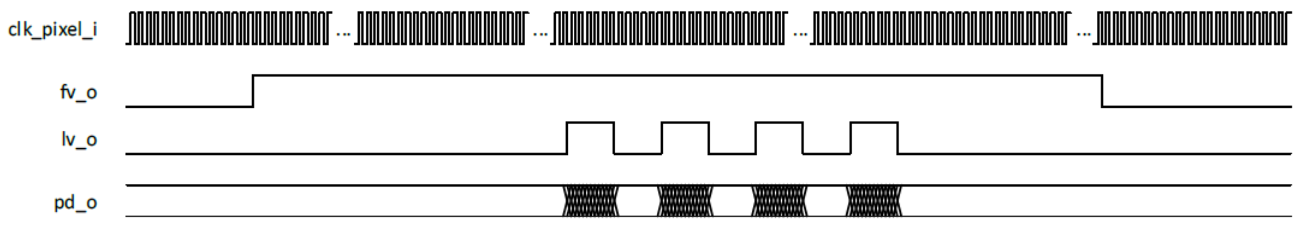


Figure 3.6. Global Output Timing Diagram for CSI-2

3.3. rgb2rgb

This module is instantiated for CSI-2 RX with RGB888 and has two major functions; sync signal generation and video data trimming.

3.3.1. Sync Signal Generation

Since sync signal format is different between CSI-2 and LVDS, VSYNC and HSYNC signals have to be created from FV and LV comes from byte2pixel. For that purpose, the following parameters are prepared:

- HFP (Horizontal Front Porch)
- HS_LENGTH (Horizontal Sync Length)
- VFP (Vertical Front Porch)
- VS_LENGTH (Vertical Sync Length)

The above parameter values can be changed through register writes when USE_I2C is defined. Otherwise, the values specified in synthesis_directives.v are applied. Note that (HFP + HS_LENGTH) must be shorter than the horizontal blanking period. Otherwise, HSYNC active period and DE active period overlap and we cannot expect the anticipated image on the display. If you have no idea about the length of the horizontal blanking period, small values should be set. For example, HFP = 2, HS_LENGTH = 2, and others, as long as the display accepts it. In addition, (VFP + VS_LENGTH) must be smaller than the vertical blanking period for the same reason. HFP and VFP counts begin at the end of non-trimmed active pixel and active line. That means the actual blanking periods are larger than HPF or VFP if trimming happens on the right edge or bottom edge.

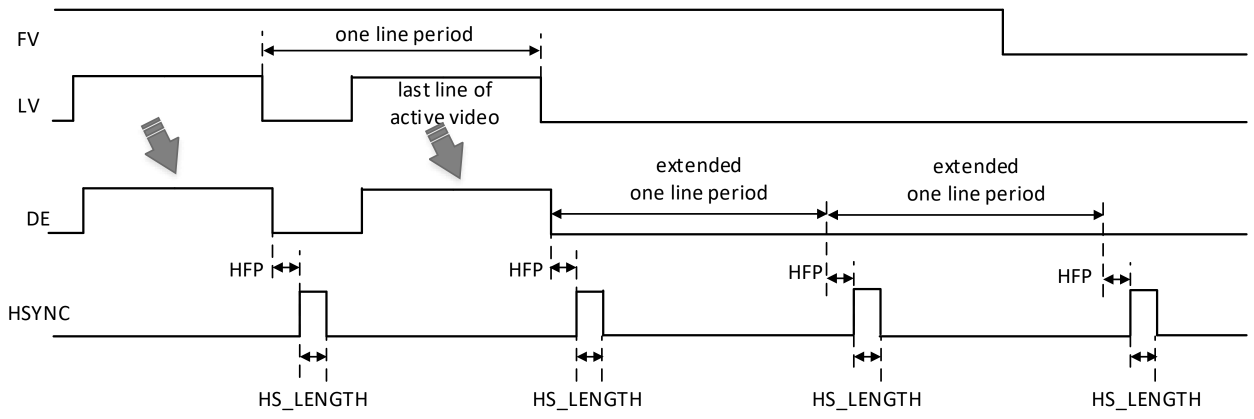


Figure 3.7. HSYNC Generation

During the vertical blanking period, HSYNC is generated by extending the line interval obtained by LV falling edges during the active line period as shown in Figure 3.7. One potential problem, however, is that HSYNC assertion could happen when the first DE of the next frame comes since there is no guarantee that the length of the vertical blanking period is multiple of one horizontal line period. To avoid this situation, a shift register is provided. FV/LV and payload data are all written to the shift register FIFO and comes out after the specified pixel clock cycle delay set by the parameter SR_DELAY. Figure 3.8 shows the masking of HSYNC which overlaps DE period. If the display has the minimum requirement of HBP (Horizontal Blanking Period: period from the end of HSYNC to the beginning of DE), SR_DELAY value should satisfy the following equation:

SR_DELAY > HS_LENGTH + minimum HBP

VSYNC/HSYNC assertions are controlled based on these delayed signals, but HSYNC assertion is disabled when the original LV is active. By changing this parameter value, you can change the HSYNC mask period. This parameter can also be changed through register write when USE_I2C is defined.

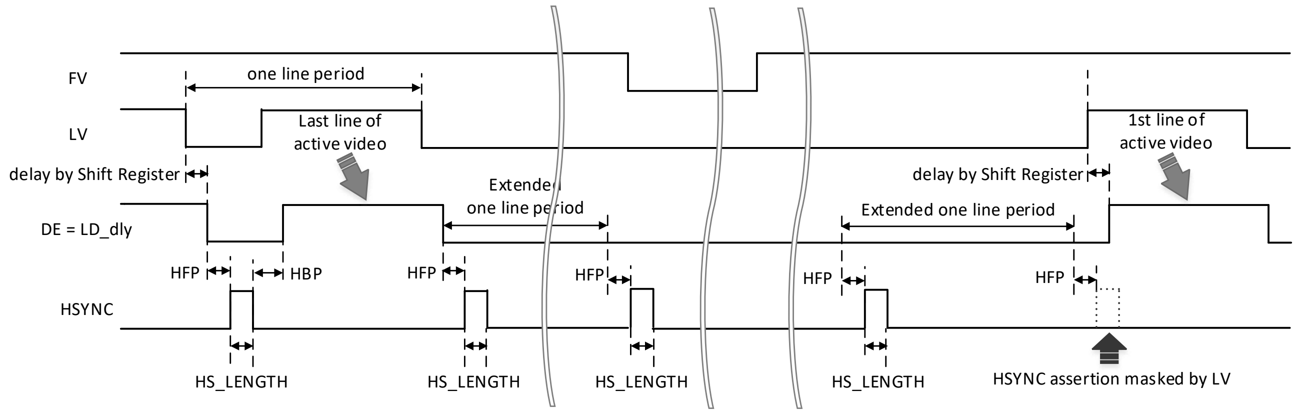


Figure 3.8. HSYNC Masked by LV

Since VSYNC generation is based on the assertion timing of HSYNC, and HSYNC assertion is based on the falling edge of LV, VSYNC of the current outgoing frame requires the previous incoming video timings. Therefore, no VSYNC assertion happens for the first incoming video frame and DE is masked for that frame. Valid VSYNC and video data outputs begin from the second incoming frame.

3.3.2. Active Data Trimming

In some cases, video data from sensor devices have an area that is larger than what is required by the display. For this reason, edge trimming capability is provided using following parameters:

- LEFT_TRIM (trimming pixels from the left edge)
- H_TX_PEL (active pixel count)
- TOP_TRIM (trimming lines from the top edge)
- V_TX_LINE (active line count)

The above parameter values can be changed through register write when USE_I2C is defined. Otherwise, the values specified in synthesis_directives.v are applied. Note that (LEFT_TRIM + H_TX_PEL) must not exceed the original horizontal active pixel count, otherwise output image is corrupted. In addition, (TOP_TRIM + V_TX_LINE) must not exceed the original active line count for the same reason. In case of TX Gear 14, H_TX_PEL must be an even value.

3.4. raw2rgb

This module is instantiated for CSI-2 RX with RAW8/RAW10/RAW12 and has three major functions; RGB data creation, sync signal generation and video data trimming.

3.4.1. RGB Data Creation

In the case of RAW data format, one pixel contains only one color component of RGB. Therefore, missing color components must be created by interpolation to form RGB888. Figure 3.9 shows four scenarios of Bayer patterns of the color components at the top-left pixels come from the image sensor.

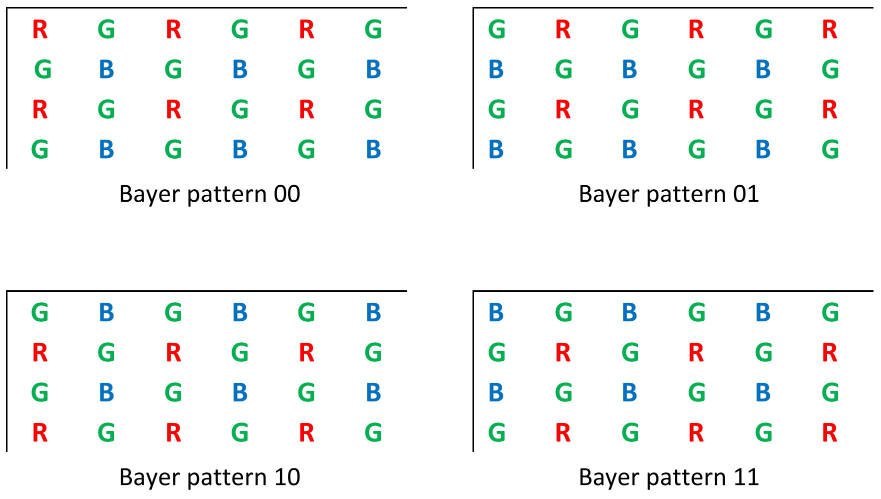
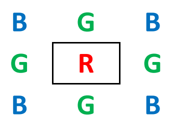
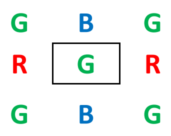


Figure 3.9. Bayer Pattern of RAW Data

Missing color components are created by interpolation using the neighborhood pixels. Figure 3.10 shows examples of interpolations to create all three color components for the center pixel. Either 2-pixel averaging or 4-pixel averaging is applied, depending on the location of the pixel, to create the missing color components.



$$R_{(interpolated)} = R_{(center)}$$
$$G_{(interpolated)} = (G_{(top)} + G_{(bottom)} + G_{(left)} + G_{(right)}) / 4$$
$$B_{(interpolated)} = (B_{(top-left)} + B_{(top-right)} + G_{(bottom-left)} + G_{(bottom-right)}) / 4$$



$$R_{(interpolated)} = (R_{(left)} + R_{(right)}) / 2$$
$$G_{(interpolated)} = G_{(center)}$$
$$B_{(interpolated)} = (B_{(top)} + B_{(bottom)}) / 2$$

Figure 3.10. RGB Data Creation from RAW Data

Due to the necessity of vertical interpolation, this module has two line buffers with the depth specified by the directive LB_DEPTH_*. In case of edge and corner pixel interpolations (top line, bottom line, left edge, right edge), *mirroring* is applied to cover the non-existing pixels. For example, there is no top pixels for the first line pixel interpolation and the bottom (second line) pixels are used instead of the non-existing top pixels. Same method applies to left edge, right edge, and bottom line pixel interpolations. Note that the first line output is roughly aligned with the second incoming line due to the line buffering delay.

3.4.2. Sync Signal Generation

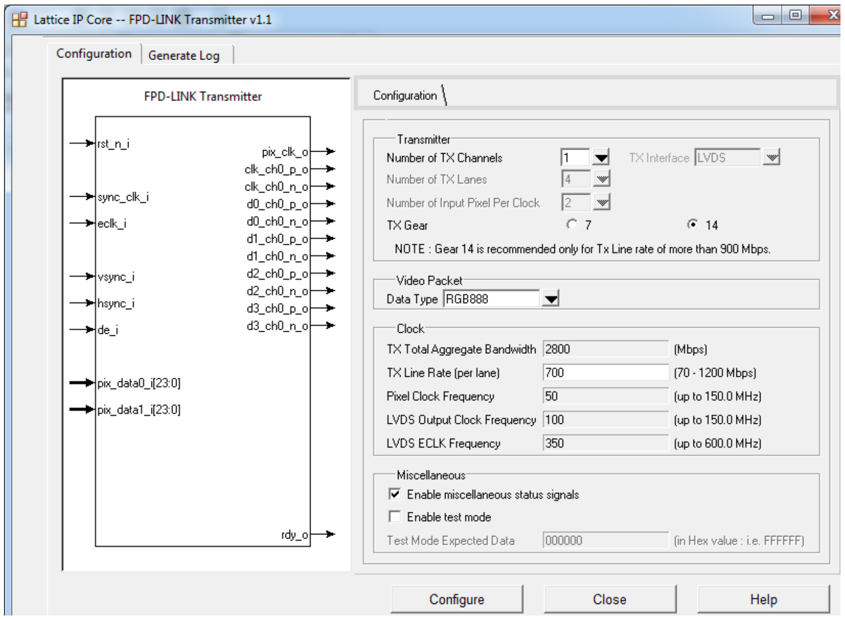
Scheme is same as CSI-2 RGB888 input except for the non-existence of the shift register. In case of RAW data processing, a single line process delay happens due to the line buffer mentioned in the RGB Data Creation section, the HSYNC masking feature is not required using the shift register. Refer to the Sync Signal Generation section for details.

3.4.3. Active Data Trimming

Scheme is same as CSI-2 RGB888 input. Refer to the Active Data Trimming section for details.

3.5. tx_lvds

You must create this module according to channel conditions, such as number of lanes, bandwidth, and others. Figure 3.11 shows an example IP interface setting in Clarity Designer for the OpenLDI/LVDS Transmitter Interface IP. You can use the sbx file (tx/tx.sbx) included in the sample project and re-configure according to your needs. Refer to OpenLDI/FPD-LINK/LVDS Transmitter Interface IP User Guide (FPGA-IPUG-02022) for details.



Transmitter
Number of TX Channels: 1
TX Interface: LVDS
Number of TX Lanes: 4
Number of Input Pixel Per Clock: 2
TX Gear: 7 [✓ 14]
NOTE : Gear 14 is recommended only for Tx Line rate of more than 900 Mbps.

Video Packet
Data Type: RGB888

Clock

TX Total Aggregate Bandwidth	2800	(Mbps)
TX Line Rate (per lane)	700	(70 - 1200 Mbps)
Pixel Clock Frequency	50	(up to 150.0 MHz)
LVDS Output Clock Frequency	100	(up to 150.0 MHz)
LVDS ECLK Frequency	350	(up to 600.0 MHz)

Miscellaneous
☑ Enable miscellaneous status signals
☐ Enable test mode
Test Mode Expected Data: 000000 (in Hex value : i.e. FFFFFF)

Figure 3.11. tx_dphy IP Creation in Clarity Designer

The following provides the guidelines and parameter settings required for this reference design.

- Number of TX Channels – Set according to channel configuration. Must match NUM_TX_CH_* setting. Refer to Table 1.1 for possible TX channel setting.
- TX Gear – Set 7 or 14. Must much TX_GEAR_*.
- Data Type – Select RGB666 in case of RX_RGB666. Otherwise, select RGB888.
- TX Line Rate – Set the value derived from the equation below.
- Enable miscellaneous status signals – Must be enabled (checked).
- Enable test mode – Must be disabled (unchecked).

This module takes the pixel data and outputs LVDS data after 7:1 or 14:1 serialization. If you are creating this IP from scratch, it is recommended to set the design name to tx and module name to tx_lvds so that you do not need to modify the instance name of this IP in the top level design as well as simulation setup file. Otherwise, you need to modify the names accordingly.

TX Line Rate is derived from the following equation:

$$TX\_lane\_bandwidth = \frac{RX\_lane\_bandwidth * number\_of\_RX\_lane * 7}{RX\_pixel\_bus\_width * number\_of\_TX\_channel}$$

Example 1 : CSI-2 RAW10 RX with 4-lane at RX lane bandwidth = 400 Mbps

TX lane bandwidth = (400 x 4 x 7) / (10 x 1) = 1120 Mbps.

Example 2 : DSI RGB888 RX with 4-lane at RX lane bandwidth = 1200 Mbps with dual channel TX outputs

TX lane bandwidth = (1200 x 4 x 7) / (24 x 2) = 700 Mbps.

Figure 3.12 to Figure 3.15 show the output data timings of single and dual channel configuration with RGB888 and RGB666. RGB888 uses four data lanes per channel and RGB666 uses three data lanes per channel.

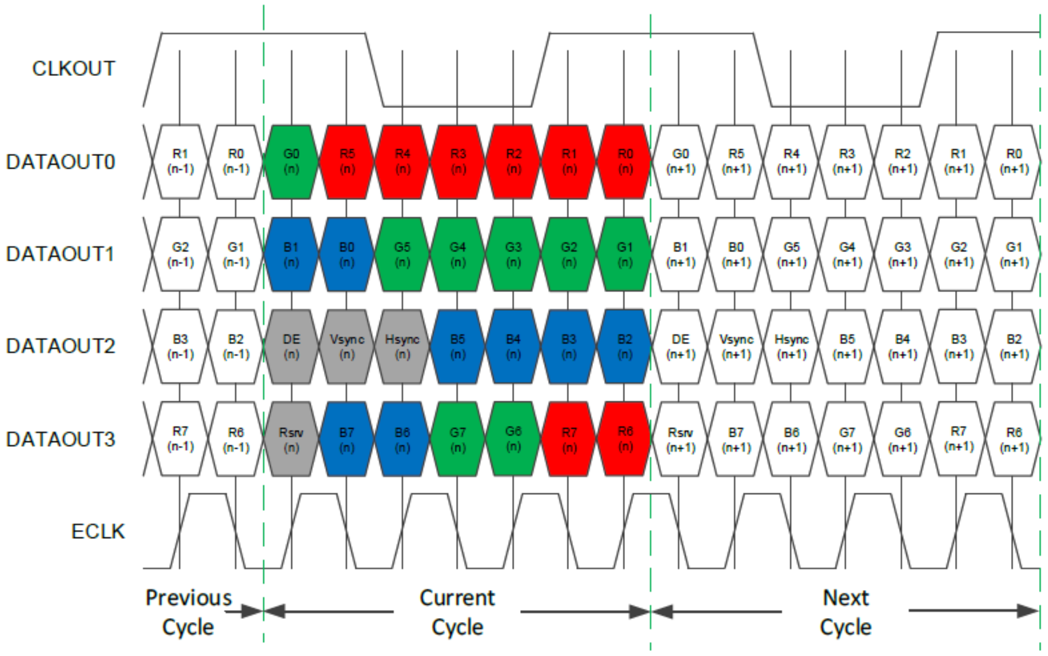


Figure 3.12. Single Channel LVDS Output of RGB888

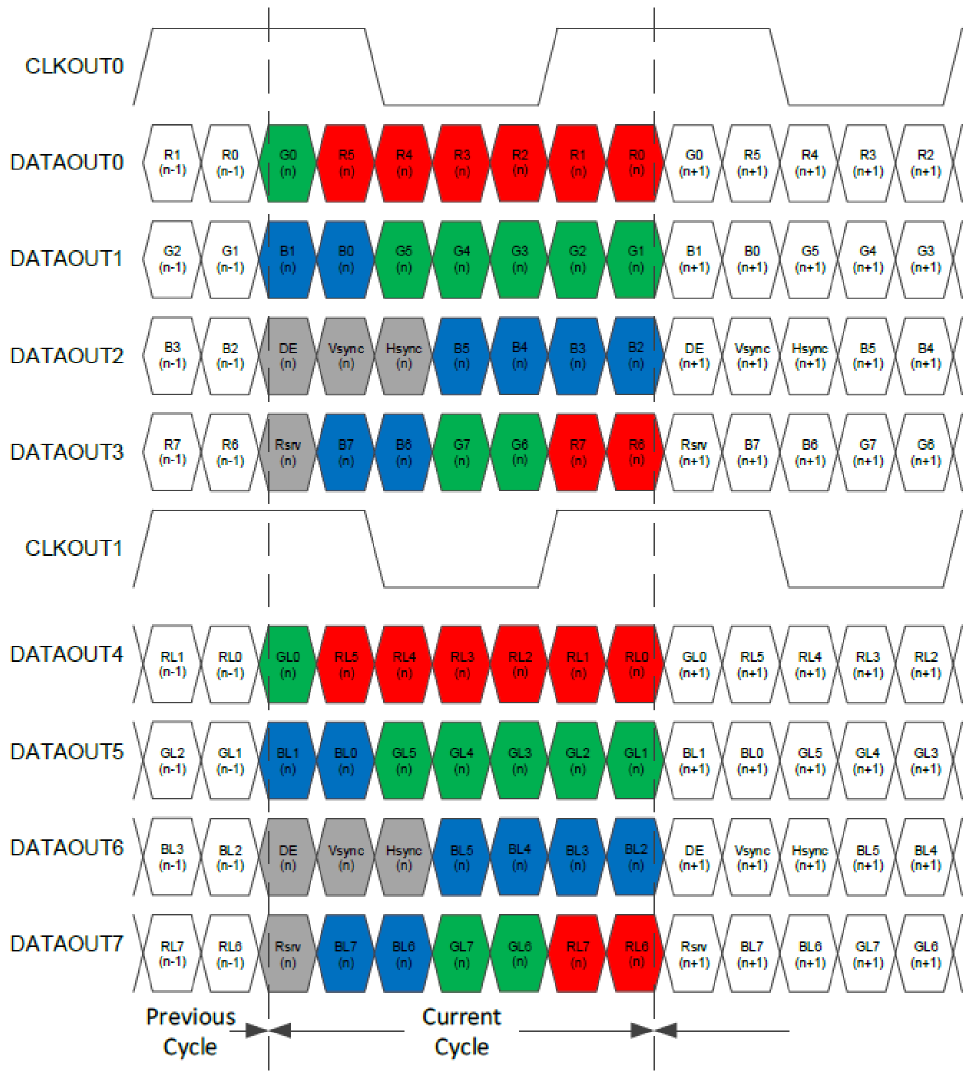

CLKOUT0
DATAOUT0
R1 (n-1) R0 (n-1) G0 (n) R5 (n) R4 (n) R3 (n) R2 (n) R1 (n) R0 (n) G0 (n+1) R5 (n+1) R4 (n+1) R3 (n+1) R2 (n+1)
DATAOUT1
G2 (n-1) G1 (n-1) B1 (n) B0 (n) G5 (n) G4 (n) G3 (n) G2 (n) G1 (n) B1 (n+1) B0 (n+1) G5 (n+1) G4 (n+1) G3 (n+1)
DATAOUT2
B3 (n-1) B2 (n-1) DE (n) Vsync (n) Hsync (n) B5 (n) B4 (n) B3 (n) B2 (n) DE (n+1) Vsync (n+1) Hsync (n+1) B5 (n+1) B4 (n+1)
DATAOUT3
R7 (n-1) R6 (n-1) Rsrv (n) B7 (n) B6 (n) G7 (n) G6 (n) R7 (n) R6 (n) Rsrv (n+1) B7 (n+1) B6 (n+1) G7 (n+1) G6 (n+1)
CLKOUT1
DATAOUT4
RL1 (n-1) RL0 (n-1) GL0 (n) RL5 (n) RL4 (n) RL3 (n) RL2 (n) RL1 (n) RL0 (n) GL0 (n+1) RL5 (n+1) RL4 (n+1) RL3 (n+1) RL2 (n+1)
DATAOUT5
GL2 (n-1) GL1 (n-1) BL1 (n) BL0 (n) GL5 (n) GL4 (n) GL3 (n) GL2 (n) GL1 (n) BL1 (n+1) BL0 (n+1) GL5 (n+1) GL4 (n+1) GL3 (n+1)
DATAOUT6
BL3 (n-1) BL2 (n-1) DE (n) Vsync (n) Hsync (n) BL5 (n) BL4 (n) BL3 (n) BL2 (n) DE (n+1) Vsync (n+1) Hsync (n+1) BL5 (n+1) BL4 (n+1)
DATAOUT7
RL7 (n-1) RL6 (n-1) Rsrv (n) BL7 (n) BL6 (n) GL7 (n) GL6 (n) RL7 (n) RL6 (n) Rsrv (n+1) BL7 (n+1) BL6 (n+1) GL7 (n+1) GL6 (n+1)
Previous Cycle
Current Cycle

Figure 3.13. Dual Channel LVDS Output of RGB888

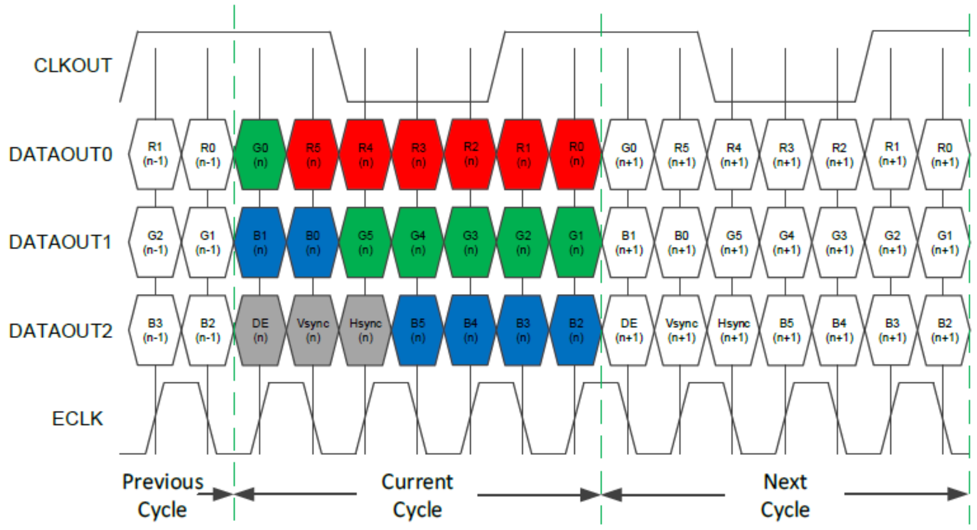

CLKOUT
DATAOUT0
R1 (n-1) R0 (n-1) G0 (n) R5 (n) R4 (n) R3 (n) R2 (n) R1 (n) R0 (n) G0 (n+1) R5 (n+1) R4 (n+1) R3 (n+1) R2 (n+1) R1 (n+1) R0 (n+1)
DATAOUT1
G2 (n-1) G1 (n-1) B1 (n) B0 (n) G5 (n) G4 (n) G3 (n) G2 (n) G1 (n) B1 (n+1) B0 (n+1) G5 (n+1) G4 (n+1) G3 (n+1) G2 (n+1) G1 (n+1)
DATAOUT2
B3 (n-1) B2 (n-1) DE (n) Vsync (n) Hsync (n) B5 (n) B4 (n) B3 (n) B2 (n) DE (n+1) Vsync (n+1) Hsync (n+1) B5 (n+1) B4 (n+1) B3 (n+1) B2 (n+1)
ECLK
Previous Cycle
Current Cycle
Next Cycle

Figure 3.14. Single Channel LVDS Output of RGB666

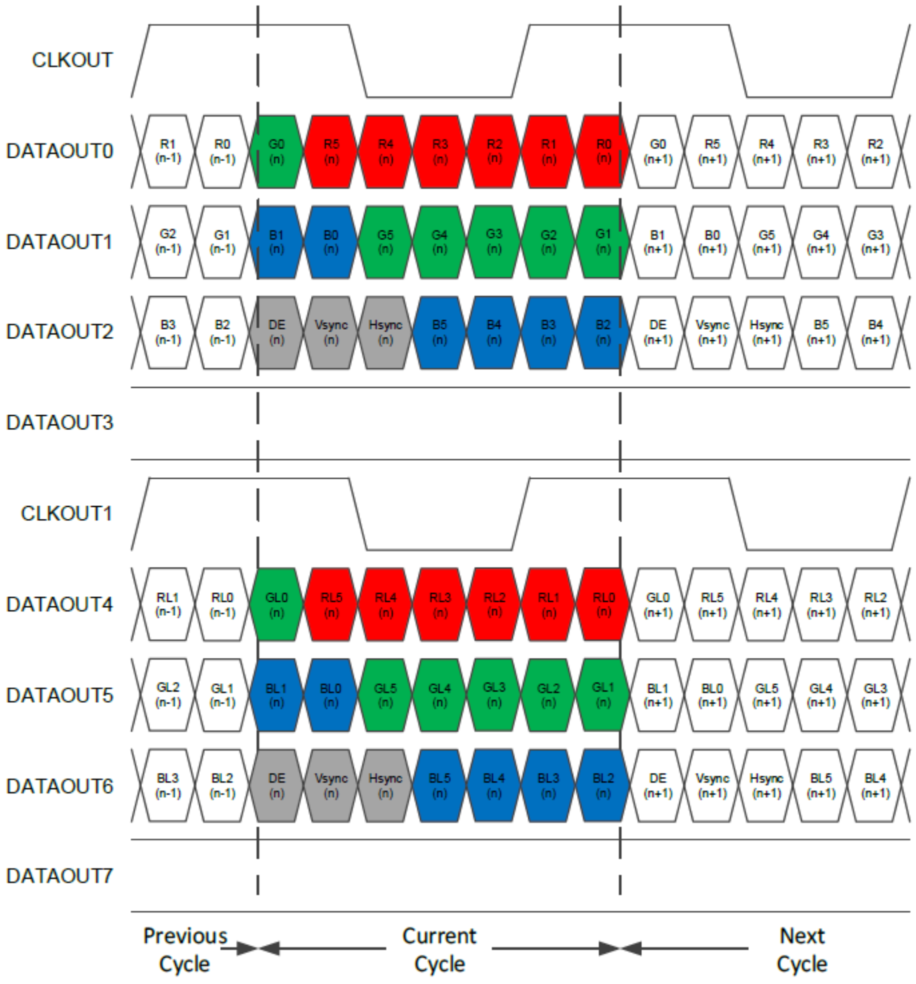
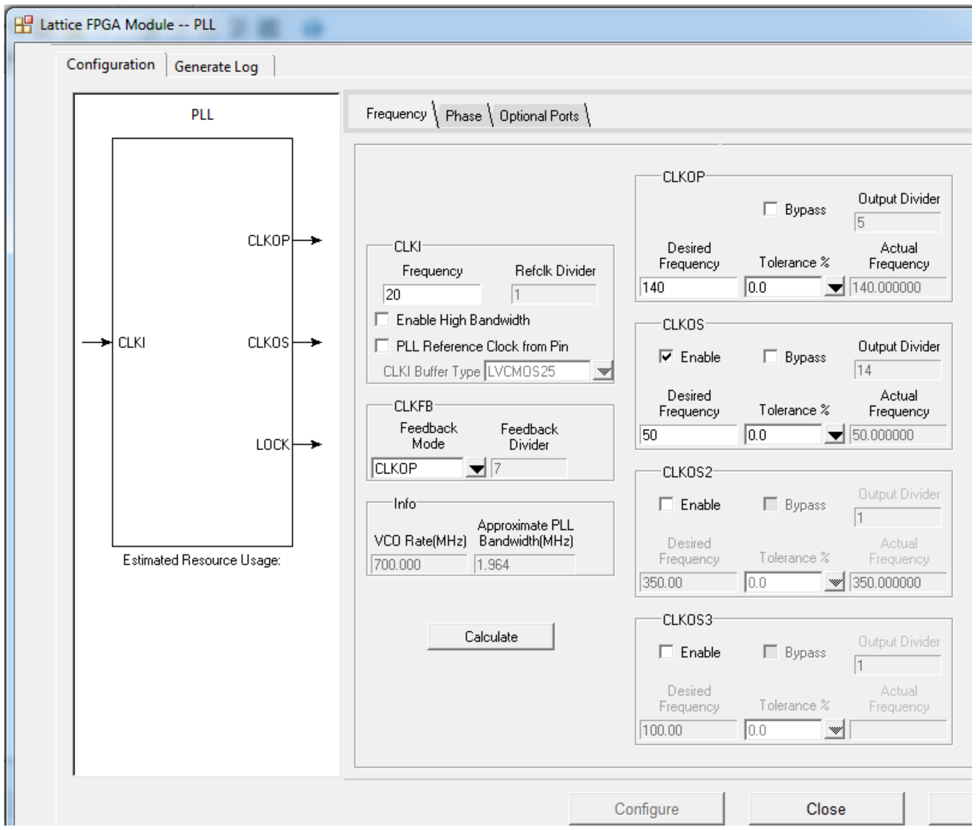


Figure 3.15. Dual Channel LVDS Output of RGB666

3.6. int_gpll

You must create GPLL module to feed the edge clock to tx_lvds. Additionally, the continuous byte clock could be generated when RX D-PHY is in HS_LP mode. You can use the sbx file (int_gpll/int_gpll.sbx) included in the sample project and re-configure according to your needs. In case that you make this IP from scratch, it is recommended to set the design name to *int_gpll* so that you do not need to modify the instance name of this IP in the top level design as well as simulation setup file. Otherwise, you need to modify the names accordingly.



CLKI
Frequency: 20
Refclk Divider: 1
☐ Enable High Bandwidth
☐ PLL Reference Clock from Pin
CLKI Buffer Type: LVCMOS25

CLKFB
Feedback Mode: CLKOP
Feedback Divider: 7

Info
VCO Rate(MHz): 700.000
Approximate PLL Bandwidth(MHz): 1.964

CLKOP
☐ Bypass
Output Divider: 5
Desired Frequency: 140
Tolerance %: 0.0
Actual Frequency: 140.000000

CLKOS
☑ Enable
☐ Bypass
Output Divider: 14
Desired Frequency: 50
Tolerance %: 0.0
Actual Frequency: 50.000000

CLKOS2
☐ Enable
☐ Bypass
Output Divider: 1
Desired Frequency: 350.00
Tolerance %: 0.0
Actual Frequency: 350.000000

CLKOS3
☐ Enable
☐ Bypass
Output Divider: 1
Desired Frequency: 100.00
Tolerance %: 0.0
Actual Frequency:

Figure 3.16. GPLL IP Creation

The TX edge clock frequency is half of the TX lane bandwidth set for tx_lvds. This clock must be assigned to CLKOP. If the byte clock has to be generated, assign it to CLKOS. Note that the VCO clock of GPLL has to be a common multiple of these two clocks. There is a case wherein TX edge clock and byte clock cannot be generated together. In this instance, give up generating the byte clock by GPLL and peform either of the following options:

- Change RX D-PHY to HS_ONLY mode; or
- Feed the continuous byte clock directly from one of the available I/O pins.

If the RX D-PHY is in HS_ONLY mode and GPLL can generate the TX edge clock using the byte clock as the input clock of GPLL (CLKI), the external reference clock is not necessary. int_gpll instantiation based on the configuration is shown below as a code snippet of mipi2lvds.v:

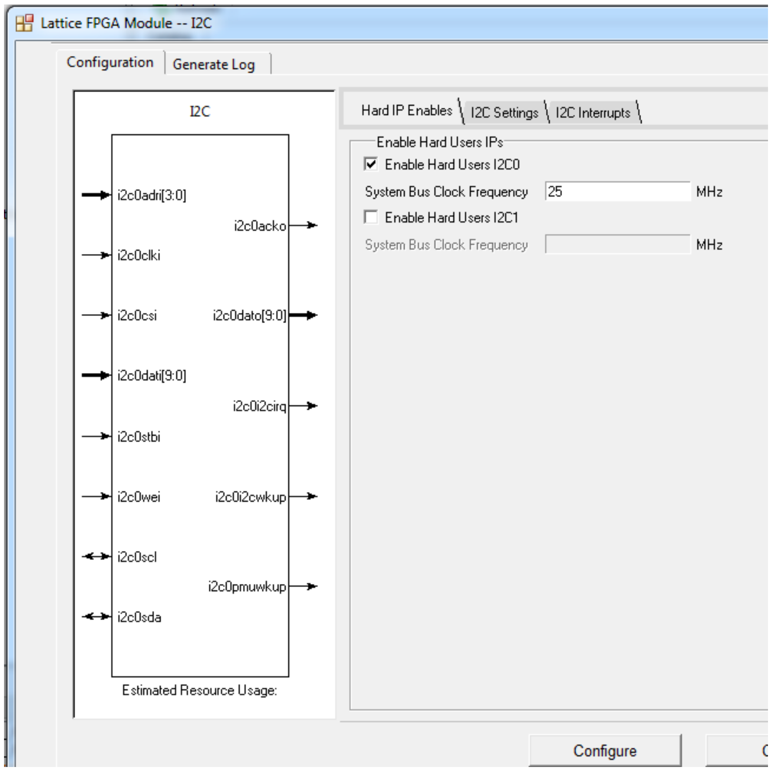
```
//////////////////////////////////////////////////////////////////////////////
/////  GPLL instantiation                                                /////
/////  User has to modify this if necessary                              /////
//////////////////////////////////////////////////////////////////////////////
`ifdef EXT_REF_CLK
       `ifdef RX_CLK_MODE_HS_LP
              int_gpll pll_inst (
                     .CLKI   (ref_clk_i),         // ref clock
                     .CLKOP  (pll_eclk),          // tx edge clock
                     .CLKOS  (pll_byte_clk),      // byte clock
                     .LOCK   (pll_lock)
              );
       `elsif RX_CLK_HS_ONLY
              int_gpll pll_inst (
                     .CLKI   (ref_clk_i),  // cannot create eclk from byte clock
                     .CLKOP  (pll_eclk),
                     .LOCK   (pll_lock)
              );
       `endif
`elsif RX_CLK_HS_ONLY
       int_gpll pll_inst (
              .CLKI   (rx_clk_byte_fr),
              .CLKOP  (pll_eclk),
              .LOCK   (pll_lock)
       );
`endif
//////////////////////////////////////////////////////////////////////////////
```

If you can use ref_clk_i (or simply divided clock of this) as byte_clk in HS_LP mode, you should disable CLKOS and add the following in mipi2lvds.v:

```
assign pll_byte_clk = ref_clk_i;   // ref_clk_i could be simply divided clock of this
```

3.7. i2c_slave

This module is instantiated when USE_I2C is defined and enables you to change CSI-2 related parameters on the fly through I^{2}C connections. I^{2}C Hard IP is instantiated and used as an I^{2}C slave device. You can use the .sbx file (*i2c_s/i2c_s.sbx*) included in the sample project and re-configure. In case that you make this IP from scratch, it is recommended to set the design name to *i2c_s* so that you do not need to modify the instance name of this IP in the top level design as well as simulation setup file. Otherwise, you need to modify the names accordingly. There exist two I^{2}C Hard IP modules in CrossLink and I2C0 is used in this IP as shown in Figure 3.17. You have to change the setting if the other IP (I2C1) is used. Also, System Bus clock frequency must be changed according to the clock fed to this module.



Enable Hard Users IPs

[✓] Enable Hard Users I2C0

System Bus Clock Frequency 25 MHz

[ ] Enable Hard Users I2C1

System Bus Clock Frequency MHz

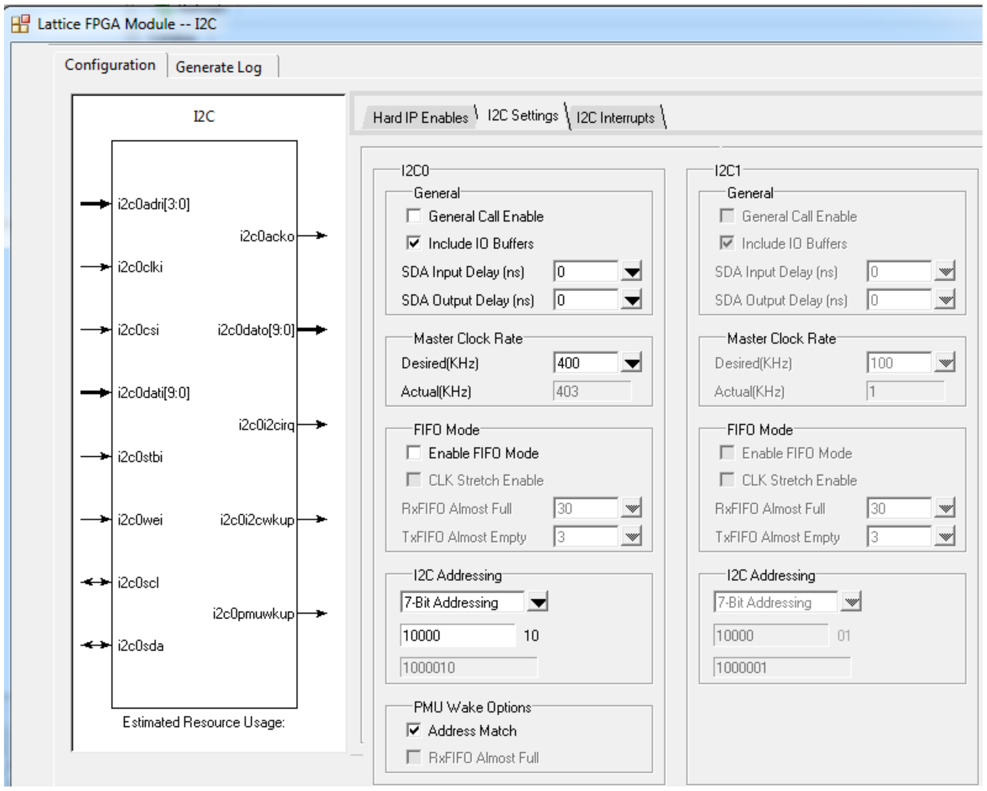
Figure 3.17. I²C IP Creation #1

Figure 3.18 shows basic settings of I²C IP. You can change the settings according to own needs, but the following should be enforced:

- FIFO Mode must be disabled (unchecked)
- Address Match must be enabled (checked)

MSB 5 bits of I²C slave address can be set here. In case of I2C0, LSB 2 bits is fixed to 2'b10.

Figure 3.19 shows interrupt settings. At least Tx/Rx Ready must be enabled (checked).


Lattice FPGA Module -- I2C

Configuration | Generate Log

Hard IP Enables | I2C Settings | I2C Interrupts

I2C0

General
☐ General Call Enable
☑ Include IO Buffers
SDA Input Delay (ns) 0
SDA Output Delay (ns) 0

Master Clock Rate
Desired(KHz) 400
Actual(KHz) 403

FIFO Mode
☐ Enable FIFO Mode
☐ CLK Stretch Enable
RxFIFO Almost Full 30
TxFIFO Almost Empty 3

I2C Addressing
7-Bit Addressing
10000 10
1000010

PMU Wake Options
☑ Address Match
☐ RxFIFO Almost Full

I2C1

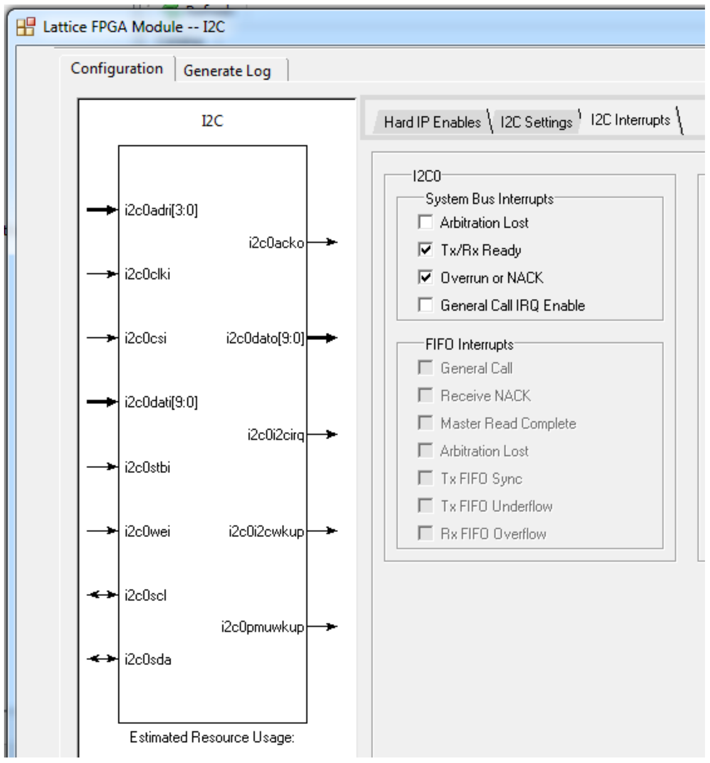
General
☐ General Call Enable
☑ Include IO Buffers
SDA Input Delay (ns) 0
SDA Output Delay (ns) 0

Master Clock Rate
Desired(KHz) 100
Actual(KHz) 1

FIFO Mode
☐ Enable FIFO Mode
☐ CLK Stretch Enable
RxFIFO Almost Full 30
TxFIFO Almost Empty 3

I2C Addressing
7-Bit Addressing
10000 01
1000001

Figure 3.18. I^{2}C IP Creation #2


Lattice FPGA Module -- I2C

Configuration | Generate Log

Hard IP Enables | I2C Settings | I2C Interrupts

I2C0

System Bus Interrupts
☐ Arbitration Lost
☑ Tx/Rx Ready
☑ Overrun or NACK
☐ General Call IRQ Enable

FIFO Interrupts
☐ General Call
☐ Receive NACK
☐ Master Read Complete
☐ Arbitration Lost
☐ Tx FIFO Sync
☐ Tx FIFO Underflow
☐ Rx FIFO Overflow

Figure 3.19. I^{2}C IP Creation #3

This module is equipped with parameter registers of 4-bit address area of I^{2}C sub address shown in Table 3.1.

Table 3.1. I^{2}C Slave Register Map

Sub address	Name	Bits	Description
0	SW Reset	[0]	Software reset register. When this is active, all modules except for i2c_slave are in reset condition. Active low.
1	SR Delay	[7:0]	Shift Register delay register. Sync and data from byte2pixel are delayed before processed in rgb2rgb. The value must be 8’d2 – 8’d255.
2	Bayer Pattern	[1:0]	Bayer Pattern register. The value must be 2’00, 2’b01, 2’b10, or 2’b11.
3	HFP	[7:0]	Horizontal Front Porch register. The value must be 8’d1 – 8’d255.
4	HS_LENGTH	[7:0]	HSYNC Length register. The value must be 8’d1 – 8’d255.
5	VFP	[5:0]	Vertical Front Porch register. The value must be 6’d1 – 6’d63.
6	VS_LENGTH	[5:0]	VSYNC Length register. The value must be 6’d1 – 6’d63.
7	LEFT_TRIM	[5:0]	Left Edge Trim register. The value must be 6’d0 – 6’d63.
8	H_TX_PEL_LSB	[7:0]	Horizontal Active Pixel register. The value must be 12’d1 – 12’d4095.
9	H_TX_PEL_MSB	[11:8]	
10	TOP_TRIM	[5:0]	Top Edge Trim register. The value must be 6’d0 – 6’d63.
11	V_TX_LINE_LSB	[7:0]	Vertical Active Line register. The value must be 12’d1 – 12’d4095.
12	V_TX_LINE_MSB	[11:8]	
13	—	N/A	Reserved
14	—	N/A	Reserved
15	—	N/A	Reserved

Note: All registers less than 8-bit data width are aligned to LSB of 8-bit data area.

All registers are set to the default values specified by corresponding directives defined in synthesis_directives.v.

Software reset works as the system reset (reset_n_i) for all modules other than i2c_slave, therefore you can assert this while updating other I^{2}C registers, then release Software reset upon completing the register update to avoid an unexpected operation during register update. Refer to the rgb2rgb and raw2rgb sections for register details.

4. Design and File Modification

This RD is based on version 1.2/1.3 of the RX D-PHY IP, version 1.1/1.2 of the Byte-to-Pixel Converter IP, and version 1.1/1.2 of the OpenLDI/LVDS Transmitter Interface IP. Due to the limitation of these IPs, some modifications are required depending on user configuration in addition to two directive files (synthesis_directives.v, simulation_directives.v).

4.1. Top Level RTL

In case of HS_LP mode and the external reference clock can be used as byte_clk without using GPLL, you have to modify the related code as described in the int_gpll section.

In addition, instance names of RX D-PHY, Byte-to-Pixel, TX LVDS IP have to be modified if you created these IP with different names.

4.2. RX D-PHY IP

After creating RX D-PHY IP in Clarity Designer with 4-lane, Gear 16 configuration, you must add the following signals as outputs shown below in rx_dphy.v:

```
output wire              sp2_en_o,
output wire              lp2_en_o,
output wire              lp2_av_en_o,
output wire [5:0]        dt2_o,
output wire [1:0]        vc2_o,
output wire [15:0]       wc2_o,
output wire [7:0]        ecc2_o,
```

Since all of these comes with the lower module *rx_dphy_dphy_rx*, similar modifications should be done as shown below in the same file:

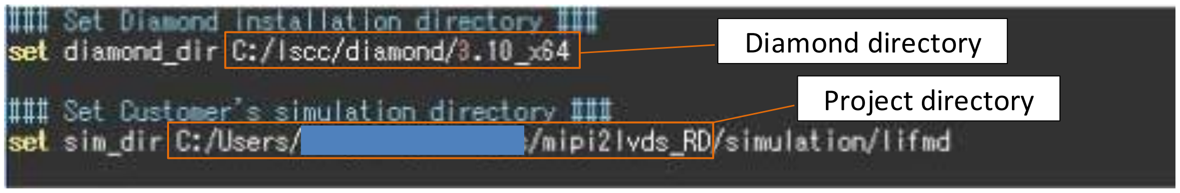
```
 .sp2_en_o               (sp2_en_o),
 .lp2_en_o               (lp2_en_o),
 .lp2_av_en_o            (lp2_av_en_o),
 .dt2_o                  (dt2_o[5:0]),
 .vc2_o                  (vc2_o[1:0]),
 .wc2_o                  (wc2_o[15:0]),
 .ecc2_o                 (ecc2_o[7:0]),
```

The above modifications apply to only 4-lane with Gear 16.

5. Design Simulation

The script file (*mipi2lvds_fsim.do*) and testbench files are provided to run the functional simulation by Active HDL. If you keep the current design names and instance names when RX D-PHY, Byte-to-Pixel, TX LVDS, GPLL, and I^{2}C IPs are created by Clarity Designer, the following are the only changes required in the script file:

- Diamond installation directory path
- User project directory



```
### Set Diamond installation directory ###
set diamond_dir C:/lscc/diamond/3.10_x64

### Set Customer's simulation directory ###
set sim_dir C:/Users/[redacted]/mipi2lvds_RD/simulation/lifmd
```

Figure 5.1. Script File Modification

You need to modify *simulation_directives.v* according to your configuration (refer to the Simulation Directives section for details). By executing the script in Active HDL, compilation and simulation are executed automatically. The testbench takes all data comparison between the expected data and output data from the RD. Note that data comparison begins from the second input frame in case of CSI-2 input due to VSYNC signal creation delay. It shows following statements while running and doing data comparison:

```
# KERNEL:  917327442 DPHY Lane 2 : Driving with data = 67
# KERNEL:  917327442 DPHY Lane 3 : Driving with data = 74
# KERNEL: Frame 1, Line    0, Pixel  142 CH0 Active Data : R = 1c, G = d8, B = 68 matched
# KERNEL: Frame 1, Line    0, Pixel  143 CH0 Active Data : R = e9, G = 8a, B = 5f matched
# KERNEL:  917340786 DPHY Driving Data
# KERNEL:  917340786 DPHY Lane 0 : Driving with data = 5f
# KERNEL:  917340786 DPHY Lane 1 : Driving with data = 5a
# KERNEL:  917340786 DPHY Lane 2 : Driving with data = 2b
# KERNEL:  917340786 DPHY Lane 3 : Driving with data = 2b
# KERNEL: Frame 1, Line    0, Pixel  144 CH0 Active Data : R = 78, G = 9b, B = ce matched
# KERNEL:  917354130 DPHY Driving Data
```

Data comparison is based on the detection of the DE assertion after the deserialization by the testbench. Simulation stops when the data comparison failure happened.

Also, you should find the following statements during the vertical blanking periods since there is no DE assertion:

```
# KERNEL: ##### tx0 VSYNC assertion #####
# KERNEL: ##### tx0 HSYNC assertion #####
# KERNEL: ##### tx0 HSYNC de-assertion #####
# KERNEL: ##### tx0 VSYNC de-assertion #####
# KERNEL: ##### tx0 HSYNC assertion #####
# KERNEL: ##### tx0 HSYNC de-assertion #####
# KERNEL: ##### tx0 HSYNC assertion #####
# KERNEL: ##### tx0 HSYNC de-assertion #####
# KERNEL: ##### tx0 HSYNC assertion #####
# KERNEL: ##### tx0 HSYNC de-assertion #####
# KERNEL: ##### tx0 HSYNC assertion #####
# KERNEL: ##### tx0 HSYNC de-assertion #####
# KERNEL: ##### tx0 HSYNC assertion #####
# KERNEL: ##### tx0 HSYNC de-assertion #####
```

When the simulation is finished, the following statements are displayed (example of CSI-2):

```
# KERNEL: ##### tx0 HSYNC de-assertion #####
# KERNEL: ##### tx1 HSYNC de-assertion #####
# KERNEL: 1910935000 DPHY CSI-2 after frame gap
# KERNEL:
# KERNEL: 1915935000 DPHY CLK CONT : Driving CLK-Trail
# KERNEL: 1915995000 DPHY CLK CONT : Driving CLK-Stop
# KERNEL: Total pixels checked were      38360 = 2 Frames x 1918 x pixels x   10 lines : all matched !!!!!
# KERNEL: Simulation Succeeded !!!!!
# KERNEL: 1916035000 TEST END
```

The above is the result after running three frames. In case of CSI-2, DE is not asserted in the first frame and data comparison occurs in the second frame and after.

Figure 5.2 shows the global timing of DSI RGB888 with single channel LVDS output. In case of DSI, all signals are passed through and de-serialized LVDS signals (tx0_des_*) by the de-serializer model in the testbench shows the same behavior as rx_dphy outputs (rx_vsync/hsync/de) and tx_lvds inputs (tx_vsync/hsync/de). In this case, LP period exists before VSYNC assertion and that leads the un-even interval of HSYNC.

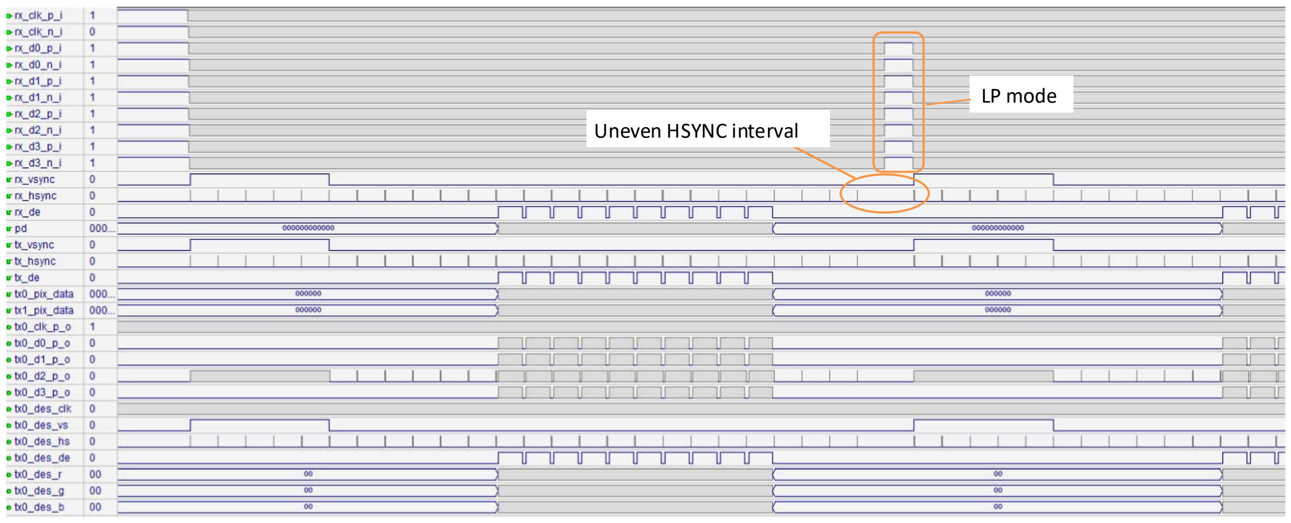


Figure 5.2. Global Timing of DSI RGB888 without Vertical Trimming

Figure 5.3 shows the global timing of CSI-2 RGB888 without vertical trimming. HSYNC is created during the LP mode period between two frames. In this simulation, VFP = 2 and VS_LENGTH = 2 are set.

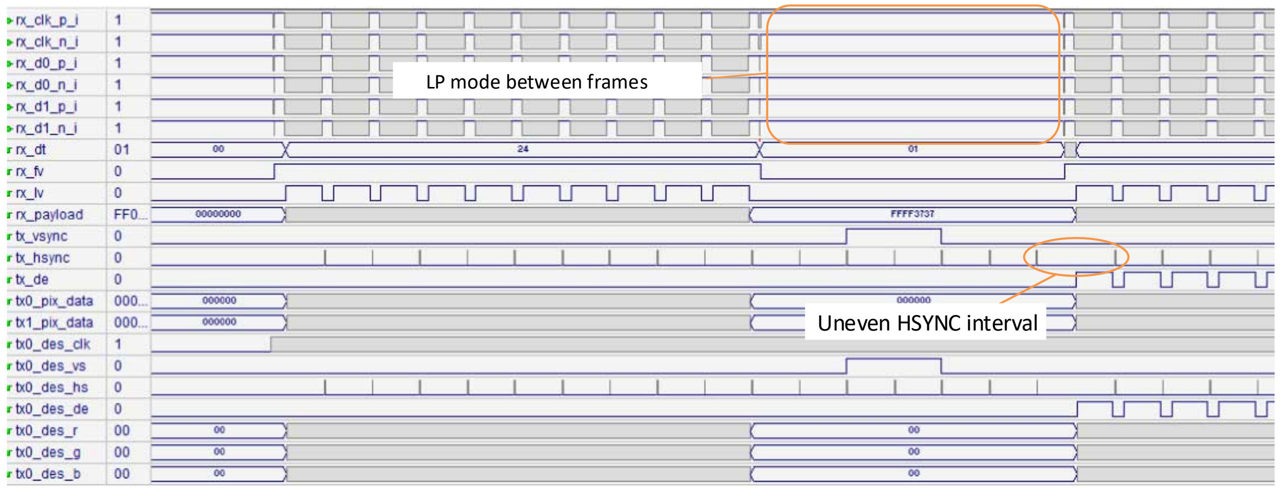


Figure 5.3. Global Timing of CSI-2 RGB888

Figure 5.4 shows the global timing of CSI-2 RAW10 without vertical trimming. DE assertion begins from the second frame after VSYNC creation with VFP = 2 and VS_LENGTH = 2. Due to RAW to RGB conversion, there exists one-line delay between RX data and TX data.

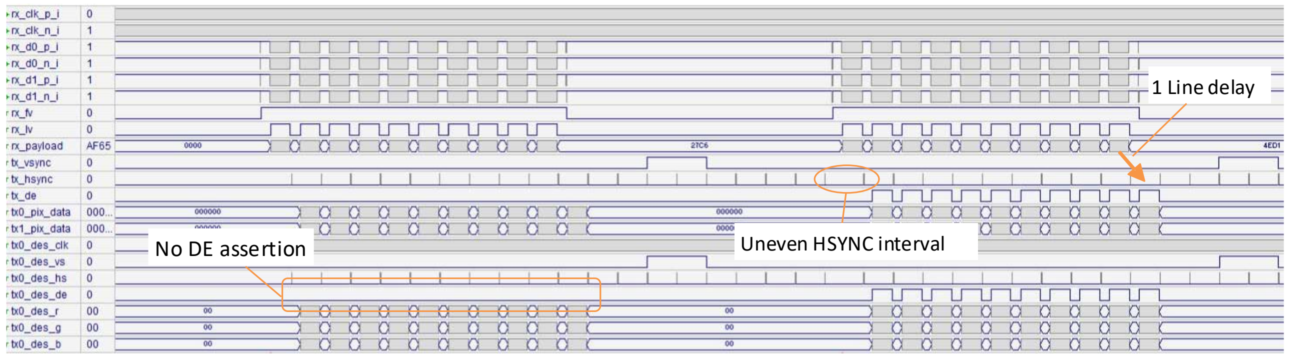


Figure 5.4. Global Timing of CSI-2 RAW10 without Vertical Trimming

Figure 5.5 shows the global timing of CSI-2 RAW10 with vertical trimming with VFP = 2 and VS_LENGTH = 2. The first line and last line are trimmed. The actual VFP is 3 due to the trimming. Before the first frame data are fed, parameter update happens through I^{2}C while software reset (int_rst_n) is asserted. Upon the completion of parameter update, the reset is released and the first frame data input begins.

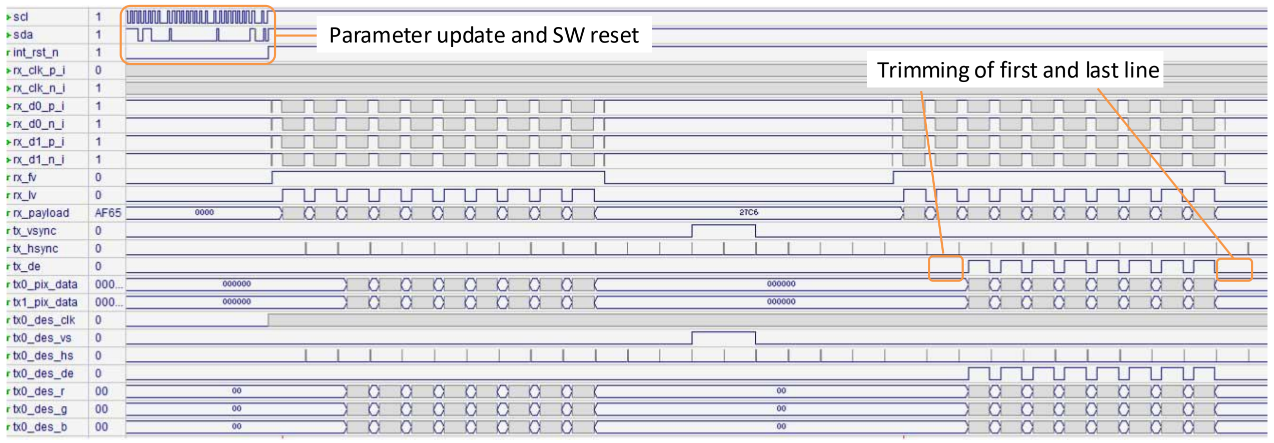


Figure 5.5. Global Timing of CSI-2 RAW10 with Vertical Trimming

6. Known Limitations

The following is a limitation of this reference design:

- Using Hard D-PHY on RX is not recommended (refer to the RX and TX Permutations section) unless dual channel TX RGB888 is required.

7. Design Package and Project Setup

MIPI DSI/CSI-2 to OpenLDI/LVDS Interface Bridge Reference Design for CrossLink is available on www.latticesemi.com. Figure 7.1 shows the directory structure. The design is targeted for LIF_MD6000-6KMG80I. *synthesis_directives.v* and *simulation_directives.v* are set to configure an example shown below:

- RX : RAW10 two lanes with Soft D-PHY, Gear 8 in continuous clock mode, I^{2}C enabled
- TX : RGB888 four lanes with Gear 14

You can modify the directives for own configuration.

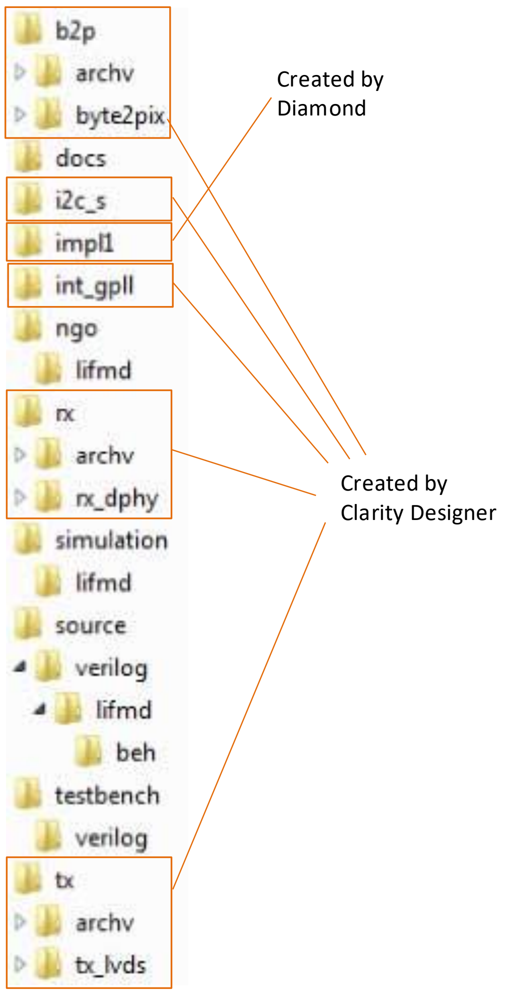
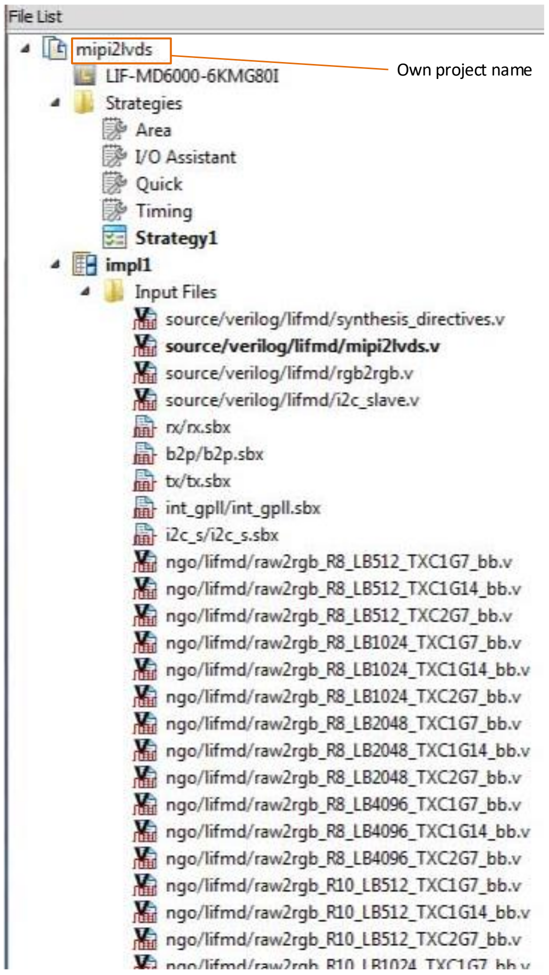


Figure 7.1. Directory Structure

Folders b2p, i2c_s, int_gll, rx, and tx are created by Clarity Designer for corresponding IPs. The ngo\lifmd folder contains mapped netlists of raw2rgb with all possible configurations designated by suffixes. The following provides the descriptions of the suffixes:

- _R* : * RAW data width; 8, 10, or 12
- _LB* : * Line Buffer depth; 512, 1024, 2048, or 4096
- _TXC*G** : Number of TX channels; 1 or 2, ** TX Gear; 7 or 14

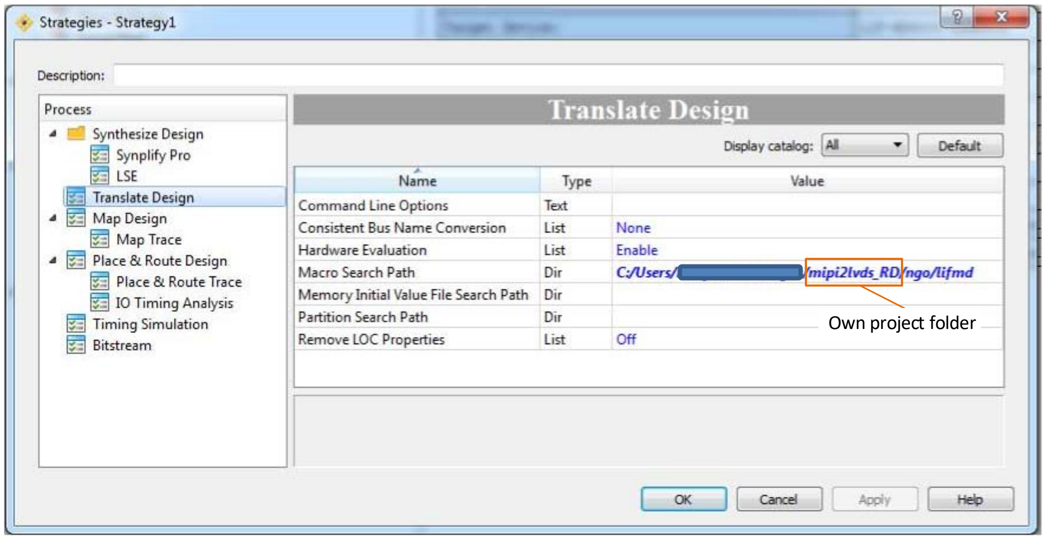
The blackbox files (*_bb.v) are also contained in the ngo\lifmd folder. You can include corresponding _bb.v files as design files in Diamond according to own configuration. Alternatively, it is possible to include all _bb.v files as long as mipi2lvds is specified as the top-level design. Figure 7.2 shows design files used in the Diamond project. Clarity Designer creates five .sbx files. In this example, all 36 _bb.v files are included in the project. By specifying mipi2lvds as a top-level design, all unnecessary files are ignored.


File List

- mipi2lvds
  - LIF-MD6000-6KMG80I
  - Strategies
    - Area
    - I/O Assistant
    - Quick
    - Timing
    - **Strategy1**
  - **impl1**
    - Input Files
      - source/verilog/lifmd/synthesis_directives.v
      - **source/verilog/lifmd/mipi2lvds.v**
      - source/verilog/lifmd/rgb2rgb.v
      - source/verilog/lifmd/i2c_slave.v
      - rx/rx.sbx
      - b2p/b2p.sbx
      - tx/tx.sbx
      - int_gpll/int_gpll.sbx
      - i2c_s/i2c_s.sbx
      - ngo/lifmd/raw2rgb_R8_LB512_TXC1G7_bb.v
      - ngo/lifmd/raw2rgb_R8_LB512_TXC1G14_bb.v
      - ngo/lifmd/raw2rgb_R8_LB512_TXC2G7_bb.v
      - ngo/lifmd/raw2rgb_R8_LB1024_TXC1G7_bb.v
      - ngo/lifmd/raw2rgb_R8_LB1024_TXC1G14_bb.v
      - ngo/lifmd/raw2rgb_R8_LB1024_TXC2G7_bb.v
      - ngo/lifmd/raw2rgb_R8_LB2048_TXC1G7_bb.v
      - ngo/lifmd/raw2rgb_R8_LB2048_TXC1G14_bb.v
      - ngo/lifmd/raw2rgb_R8_LB2048_TXC2G7_bb.v
      - ngo/lifmd/raw2rgb_R8_LB4096_TXC1G7_bb.v
      - ngo/lifmd/raw2rgb_R8_LB4096_TXC1G14_bb.v
      - ngo/lifmd/raw2rgb_R8_LB4096_TXC2G7_bb.v
      - ngo/lifmd/raw2rgb_R10_LB512_TXC1G7_bb.v
      - ngo/lifmd/raw2rgb_R10_LB512_TXC1G14_bb.v
      - ngo/lifmd/raw2rgb_R10_LB512_TXC2G7_bb.v
      - ngo/lifmd/raw2rgb_R10_LB1024_TXC1G7_bb.v

Figure 7.2. Project Files

You need to set the path for .ngo files in the translation stage. This can be done by editing strategy file (*Strategy1*) shown in Figure 7.2. Select Translate Design and set Macro Search Path as shown in Figure 7.3.


Strategies - Strategy1

Description:

Process
- Synthesize Design
  - Synplify Pro
  - LSE
- Translate Design
- Map Design
  - Map Trace
- Place & Route Design
  - Place & Route Trace
  - IO Timing Analysis
- Timing Simulation
- Bitstream

Translate Design

Display catalog: All | Default

Name	Type	Value
Command Line Options	Text	
Consistent Bus Name Conversion	List	None
Hardware Evaluation	List	Enable
Macro Search Path	Dir	C:/Users/[redacted]/mipi2lvds_RD/ngo/lifmd
Memory Initial Value File Search Path	Dir	
Partition Search Path	Dir	
Remove LOC Properties	List	Off

Figure 7.3. Path Setting for .ngo Files

8. Resource Utilization

Resource utilization depends on the configurations. Table 8.1 shows resource utilization examples under certain configurations. LB depth is 2048 in case of CSI-2 RAW8/10/12. Actual usage may vary.

Table 8.1. Resource Utilization Examples

Configuration	LUT %	FF %	EBR	I/O
DSI RGB888 with four lanes, Gear 16 to single channel TX with Gear 14	39	27	10	22
CSI-2 RGB888 with two lanes, Gear 16 to single channel TX with Gear 14 + I^{2}C	29	23	8	20
CSI-2 RAW10 with four lanes Gear 8 to single channel TX with Gear 14 + I^{2}C	51	29	13	24
CSI-2 RAW12 with two lanes Gear 16 to single channel TX with Gear 7 + I^{2}C	47	28	11	20
CSI-2 RAW8 with one lane Gear 8 to single channel TX with Gear 7 + I^{2}C	23	15	4	18

9. References

- MIPI® Alliance Specification for D-PHY Version 1.1
- MIPI® Alliance Specification for Display Serial Interface (DSI) Version 1.1
- MIPI® Alliance Specification for Camera Serial Interface 2 (CSI-2) Version 1.1
- CSI-2/DSI D-PHY Receiver Submodule IP User Guide (FPGA-IPUG-02025)
- Byte-to-Pixel Converter IP User Guide (FPGA-IPUG-02027)
- OpenLDI/FPD-LINK/LVDS Transmitter Interface IP User Guide (FPGA-IPUG-02022)
- Advanced CrossLink I^2C Hardened IP Reference Guide (FPGA-TN-02020)

For more information on the CrossLink FPGA device, visit
http://www.latticesemi.com/Products/FPGAandCPLD/CrossLink.

For complete information on Lattice Diamond Project-Based Environment, Design Flow, Implementation Flow, Tasks, and Simulation Flow, see the Lattice Diamond User Guide.

Technical Support

For assistance, submit a technical support case at www.latticesemi.com/techsupport.

Revision History

Revision 1.1, September 2019

Section	Change Summary
Supported Device and IP	Newly added to support CrossLinkPlus.

Revision 1.0, May 2019

Section	Change Summary
All	Initial release

www.latticesemi.com