MIPI to Parallel with CrossLink

Reference Design

FPGA-RD-02285-1.0

January 2024

Disclaimers

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

Contents

Figures

Tables

Acronyms in This Document

A list of acronyms used in this document.

Acronym	Definition
CSI-2	Camera Serial Interface 2
DE	Data Enable
DSI	Display Serial Interface
EBR	Embedded Block RAM
FIFO	First In, First Out
GPLL	General Purpose PLL
HS	High Speed
HSYNC	Horizontal Sync
LP	Low Power
LUT	Look Up Table
MIPI	Mobile Industry Processor Interface
PLL	Phase Locked Loop
RX	Receiver
STA	Static Timing Analysis
TX	Transmitter
VSYNC	Vertical Sync

Supported Device and IP

This reference design supports the following devices with IP versions.

Device Family	Part Number	Compatible IP
CrossLink	LIF-MD6000 LIA-MD6000	CSI-2/DSI D-PHY Receiver IP version 1.6 Byte to Pixel Converter IP version 1.3

The IPs above are supported by Lattice Diamond™ software version 3.12 or later.

1. Introduction

The Mobile Industry Processor Interface (MIPI®) D-PHY was developed primarily to support camera and display interconnections in mobile devices, and it has become the industry's primary high-speed PHY solution for these applications in smartphones. It is typically used in conjunction with MIPI Camera Serial Interface-2 (CSI-2) and MIPI Display Serial Interface (DSI) protocol specifications. It meets the demanding requirements of low power, low noise generation, and high noise immunity that mobile phone designs demand.

MIPI D-PHY is a practical PHY for typical camera and display applications. It is designed to replace traditional parallel bus based on LVCMOS or LVDS. However, many processors and displays/cameras still use RGB, CMOS, or MIPI Display Pixel Interface (DPI) as interface.

The MIPI to Parallel reference design allows the quick interface for a processor with a MIPI DSI interface to a display with an RGB interface or a camera with a MIPI CSI-2 interface to a processor with Parallel interface. The Lattice Semiconductor MIPI to Parallel Reference Design provides this conversion for Lattice Semiconductor CrossLink™ devices. This is useful for wearable, tablet, human machine interfacing, medical equipment, and many other applications.

1.1. Features List

The Key features of the MIPI to Parallel Reference Design are:

- Compliant with MIPI D-PHY v1.1, MIPI DSI v1.1, and MIPI CSI-2 v1.1 specifications
- Supports MIPI D-PHY interfacing from 80 Mb/s up to 1.5 Gb/s
- Supports 1, 2, or 4 data lanes and one clock lane
- Supports continuous and non-continuous MIPI D-PHY clock
- Supports common MIPI DSI compatible video formats (RGB888, RGB666, RGB666_LOOSE)
- Supports common MIPI CSI-2 compatible video formats (RGB888, RAW8, RAW10, RAW12, YUV_420_8, YUV_420_8_CSPS, LEGACY_YUV_420_8, YUV_420_10, YUV_420_10_CSPS, YUV_422_8, YUV_422_10)
- Supports MIPI DSI Video Mode operation of Non-Burst Mode with Sync Pulses and Non-Burst Mode with Sync Events
- Supports dedicated End of Transmission short packet (EoTp)

1.2. Block Diagram

Figure 1.1 shows the block level diagram of the MIPI to Parallel Reference Design with Continuous D-PHY Clock Mode.

Figure 1.2 shows the block level diagram of the MIPI to Parallel Reference Design with Non-Continuous D-PHY Clock Mode.

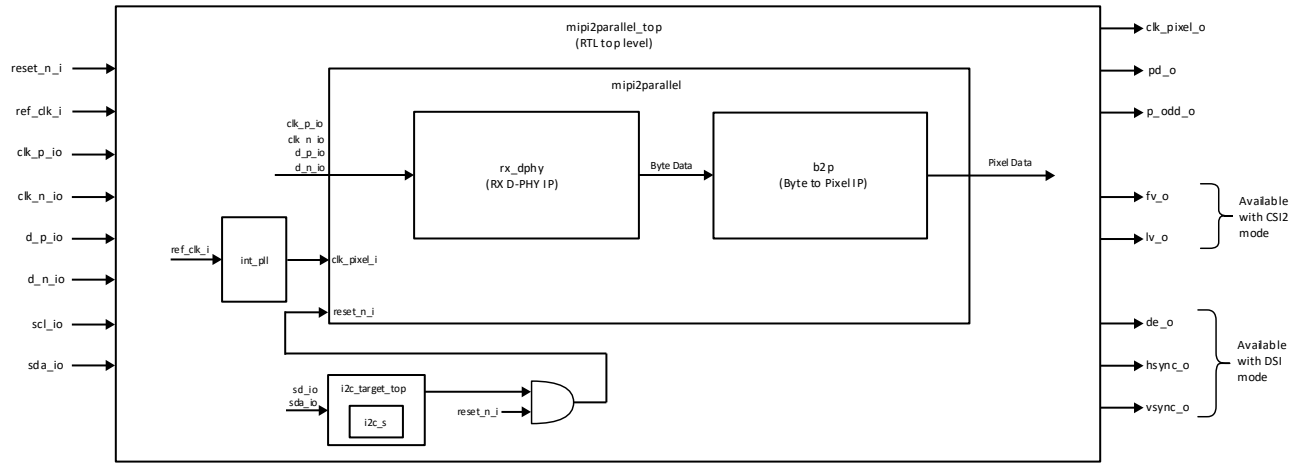

Figure 1.1. MIPI to Parallel Reference Design Block Diagram with Continuous D-PHY Clock Mode

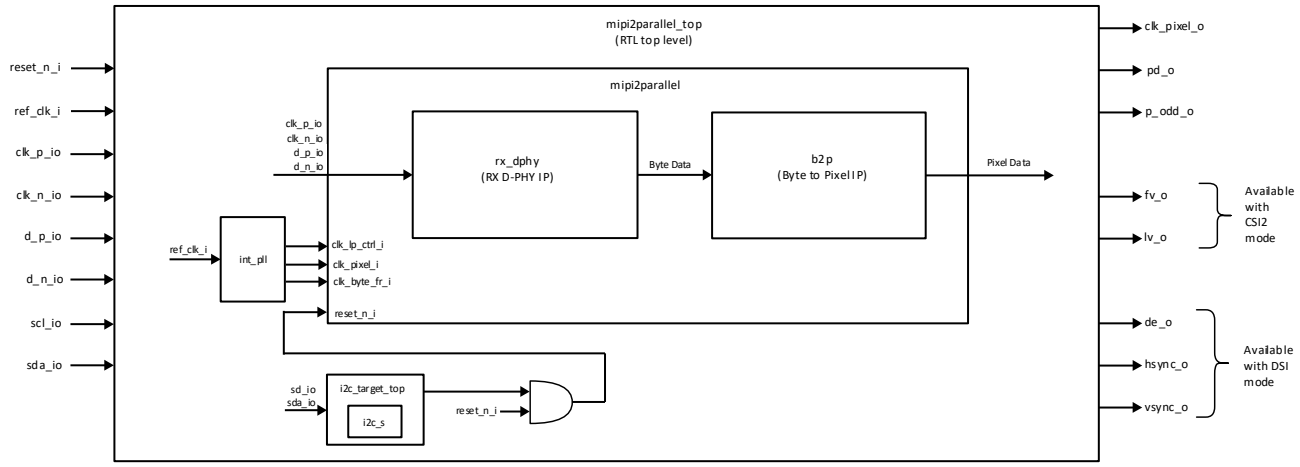

Figure 1.2. MIPI to Parallel Reference Design Block Diagram with Non-Continuous D-PHY Clock Mode

As shown in the Figure 1.1 and Figure 1.2, the block level diagram of the MIPI to Parallel reference design mainly consists of the RX D-PHY and Byte to Pixel IPs. There are two primary clocks for the main video data path: byte clock and pixel clock. Generally, a PLL is used to generate the pixel clock. The same PLL can also be used to generate the continuous byte clock (clk_byte_fr_i) as well as low power control clock (clk_lp_ctrl_i) when the RX D-PHY is in non-continuous clock mode. The input to the PLL is the external reference clock (ref_clk_i) in both continuous and non-continuous clock mode.

An I^2C target module (i2c_target_top) consists of Hardened I^2C IP to allow control over the sub functional blocks. In this example design, users can issue an I^2C command to reset just the mipi2parallel block without resetting any other blocks such as the PLL.

1.3. Functional Description

The MIPI D-PHY receive interface has one clock lane and configurable number of data lanes. The clock lane is center-aligned to the data lanes. The MIPI D-PHY clock can either be continuous (high speed only) or non-continuous.

When the MIPI D-PHY clock is non-continuous, proper transition from low power (LP) to high speed (HS) mode of clock lane is required. The data lanes also require proper transition from LP to HS modes. In HS mode, data stream from each data lane is deserialized to byte data. The deserialization is done with 1:8 gearing. The byte data is word-aligned based on the SoT Sync sequence defined in the MIPI D-PHY Specification version 1.1.

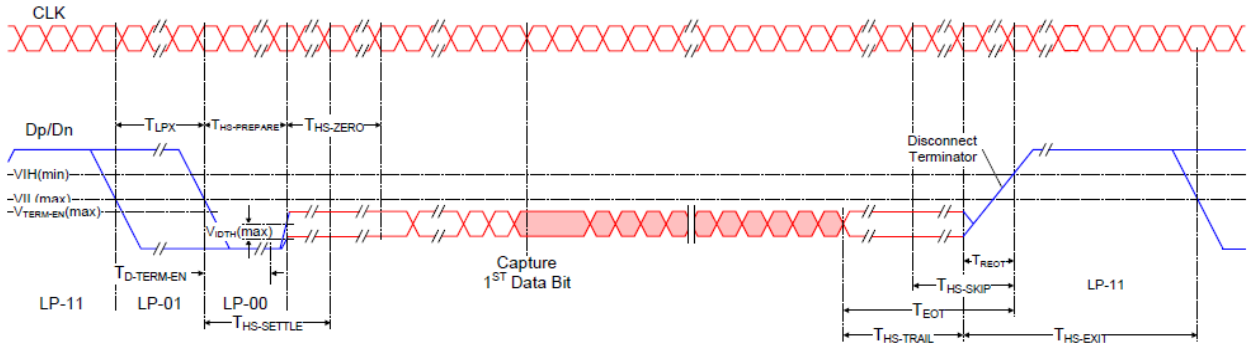

Figure 1.3. High Speed Data Transmission

The parallel transmit interface consists of clock, pixel data, and control signals. The pixel data width is configurable depending on the data type. The control signals are either data enable (DE), vertical and horizontal sync flags (VSYNC and HSYNC) for MIPI DSI applications or frame valid and line valid for MIPI CSI-2 applications.

The clock is edge-aligned against data and control signals. All signal transitions happen in sync with the rising edge of pixel clock as shown in Figure 1.4 and Figure 1.5.

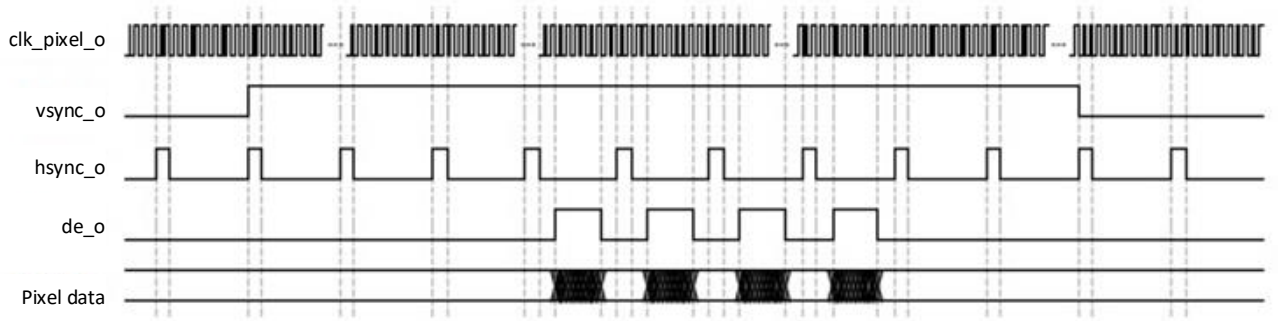

Figure 1.4. Parallel Transmit Interface Timing Diagram (DSI)

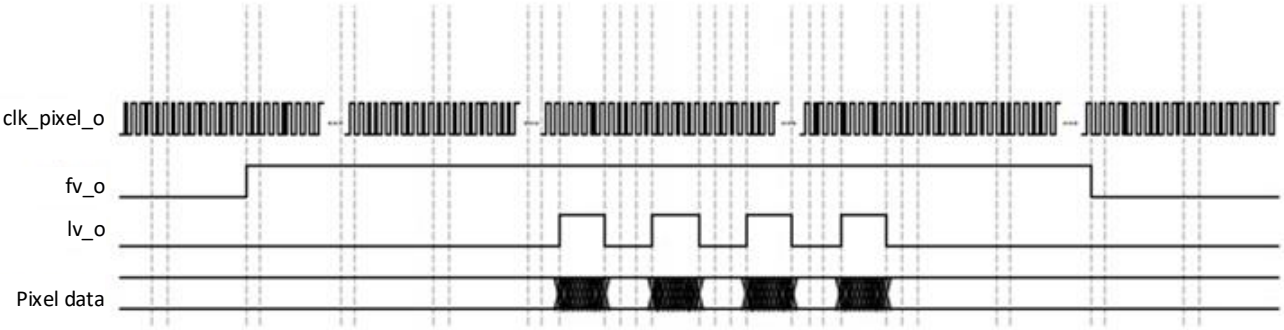

Figure 1.5. Parallel Transmit Interface Timing Diagram (CSI-2)

1.4. Conventions

1.4.1. Nomenclature

The nomenclature used in this document is based on Verilog HDL. This includes radix indications and logical operators.

1.4.2. Data Ordering and Data Types

The highest bit within a data bus is the most significant bit.
1-bit data stream from each MIPI D-PHY data lane is deserialized into 8-bit parallel data where bit 0 is the first received bit.

Table 1.1 lists the pixel data order coming from the core module.

Table 1.1. Pixel Data Order

Data Type	Format
RGB	{Red[MSB:0], Green[MSB:0], Blue[MSB:0]}
RAW	RAW[MSB:0]
YUV	YUV[MSB:0]

1.4.3. Signal Names

Signal names that end with:

- *_n* is active low
- *_i* are input signals
- Some signals are declared as bidirectional (I/O) but are only used as input. Hence, *_i* identifier is used.
- *_o* are output signals
- Some signals are declared as bidirectional (I/O) but are only used as output. Hence, *_o* identifier is used.
- *_io* are bidirectional signals

2. Parameters and Port List

There are two directive files for this reference design:

- synthesis_directives.v – used for design compilation by Lattice Diamond software 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, Byte to Pixel, GPLL and OSC IP settings created by Lattice Diamond.

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
RX Interface	RX_TYPE_DSI	D-PHY Receive Interface. Only one of these directives must be defined. Must match with RX D-PHY IP GUI options.
	RX_TYPE_CSI2	
Number of RX Lanes	NUM_RX_LANE_1	Number of lanes in RX interface. Only one of these three directives must be defined. Must match with RX D-PHY and Byte To Pixel IP GUI options.
	NUM_RX_LANE_2	
	NUM_RX_LANE_4	
RX D-PHY Clock Gear	RX_GEAR_8	Only one of these directives must be selected. Must match with RX D-PHY and Byte To Pixel IP GUI options.
	RX_GEAR_16	
Sync Signal Polarity	SYNC_POLARITY_POS	Sync signal (VSYNC, HSYNC) polarity. Only one of these two directives must be defined. Applicable only to DSI. Must match with Byte to Pixel IP GUI options.
	SYNC_POLARITY_NEG	
RX D-PHY Clock Mode[1]	RX_CLK_MODE_HS_ONLY	RX D-PHY Clock mode. Only one of these two directives must be defined. Must match with RX D-PHY IP GUI options.
	RX_CLK_MODE_HS_LP	
RX D-PHY type	RX_DPHY_HARD	Specify Hard D-PHY implementation. If this define is not used, Soft D-PHY implementation is used. Must match with RX D-PHY IP GUI options.
RX Data Type	RX_RGB888	Data type on the D-PHY RX interface. Only one of these directives must be defined. Must match with Byte to Pixel IP GUI options. Supported MIPI DSI data types: RGB888, RGB666, RGB666_LOOSE Supported MIPI CSI-2 data types: RGB888, RAW8, RAW10, RAW12, YUV_420_8, YUV_420_8_CSPS, LEGACY_YUV_420_8, YUV_420_10, YUV_420_10_CSPS, YUV_422_8, YUV_422_10
	RX_RGB666	
	RX_RGB666_LOOSE	
	RX_RAW8	
	RX_RAW10	
	RX_RAW12	
	YUV_420_8	
	YUV_420_8_CSPS	
	LEGACY_YUV_420_8	
	YUV_420_10	
	YUV_420_10_CSPS	
	YUV_422_8	
	YUV_422_10	
Number of Pixels Per Clock	RX_PEL_PER_CLK {value}	Number of pixels per clock at output where x = 1, 2, or 4. Only one of these values must be defined. Must match with Byte to Pixel IP GUI options.
I^2C Target Address MSB	I2C_TARGET_ADR_MSB {value}	5-bits MSB for the target address. Must match with I2C IP GUI options.
Software Reset Register	SW_RST_N {value}	Default value of the active low software reset register of I^2C Target Module.

Note: HS_LP mode means *non-continuous clock mode* and HS_ONLY means *continuous clock mode*. HS_LP mode works only if RX byte clock for corresponding RX channel can be generated internally or directly fed from I/O pin.

2.2. Simulation Directives

Table 2.2 shows the simulation directives for this reference design.

Table 2.2. Simulation Directives

Category	Directive	Remarks
Video data configuration on RX Interface	NUM_FRAMES {value}	Number of video frames fed by the testbench
	NUM_LINES {value}	Number of active lines per frame
	NUM_PIXELS {value}	Number of active video pixels per line
Blanking Mode	LP_BLANKING	Enable the Low Power Blanking during Vertical Blanking Period. If not defined, then High-Speed Blanking is used by default.[1]
Sync Type of Video Data	NON_BURST_SYNC_EVENTS	Enable Non-Burst Sync Event mode. If not defined, then Non-Burst Sync Pulse mode is used by default. Must be matched with Byte To Pixel IP GUI option.*
RX D-PHY clock period	DPHY_CLK {value}	RX D-PHY clock period in ps. Must be matched with RX D-PHY IP GUI option.
Reference clock period	REF_CLK {value}	Reference clock period in ps. Must be matched with PLL IP GUI option.
Vertical Sync Width	VSYNC_WIDTH {value}	Vertical SYNC width. This value must match “number of HSYNC pulses inside VSYNC active region” setting of Byte to Pixel IP.*
Horizontal Sync Width	HSYNC_WIDTH {value}	Horizontal SYNC width. This value must match the *number of pix clock cycles HSYNC remains active* setting of the Byte to Pixel IP.*
Internal signal monitoring	DPHY_DEBUG_ON	Enables D-PHY internal signal monitoring by the testbench.

Note:

1. Applicable to DSI Mode only.

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 configurations. All necessary I/O ports are automatically declared by compiler directives.

Table 2.3. MIPI to Parallel Top Level I/O

Port Name	Direction	Description
Clocks and Resets		
ref_clk_i (optional)	I	External input reference clock. Used as PLL reference clock as well as I^2C internal reference clock.
reset_n_i	I	Asynchronous active low system reset
I^2C Bus Interface		
scl_io	I/O	I^2C clock signal.
sda_io	I/O	I^2C data signal.
MIPI D-PHY RX Interface		
clk_p_io	I/O	Positive differential RX D-PHY input clock
clk_n_io	I/O	Negative differential RX D-PHY input clock
d_p_io[BUS_WIDTH – 1:0][1]	I/O	Positive differential RX D-PHY input data
d_n_io[BUS_WIDTH – 1:0][1]	I/O	Negative differential RX D-PHY input data
Parallel Interface		
clk_pixel_o	O	Pixel clock generated from internal PLL
vsync_o	O	Vertical Sync Indicator (active high/low). Goes high/low when VSYNC start short packet is received. Goes low/high when VSYNC end short packet is received. Available only for MIPI DSI mode[2]

Port Name	Direction	Description
hsync_o	O	Horizontal Sync Indicator (active high/low). Goes high when either VSYNC/HSYNC start or VSYNC end short packet is received. Goes low when HSYNC end short packet is received. Available only for MIPI DSI mode[3]
de_o	O	Data Enable Indicator (active high). Goes high at the start of valid pixel data and goes low at the end of valid pixel data. Available only for MIPI DSI mode.
fv_o	O	Frame Valid Indicator (active high). Goes high when frame start short packet is received and goes low when frame end short packet is received. Available only for MIPI CSI-2 mode.
lv_o	O	Line Valid Indicator (active high). Goes high at the start of valid pixel data and goes low at the end of valid pixel data. Available only for MIPI CSI-2 mode.
pd_o[PD_BUS_WIDTH*RX_PEL_PER_CLK - 1:0][4,5]	O	Pixel data output. Data width depends on selected data type and Pixel per clock configurations.
p_odd_o[1:0]	O	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. 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

Notes:

1. BUS_WIDTH – Number of D-PHY Lanes 1, 2, or 4 (available on the user interface).
2. In case of Non-Burst Sync Events, VSYNC end short packet is not received, and in that case vsync_o goes high/low as specified in VSYNC_WIDTH in simulation_directives.v
3. In case of Non-Burst Sync Events, HSYNC end short packet is not received, and in that case hsync_o goes high/low as specified in HSYNC_WIDTH in simulation_directives.v
4. PD_BUS_WIDTH - It can be 8, 10, 12, 18 or 24 as per the selected data type. Refer to FPGA-IPUG-02027 Byte-to-Pixel Converter IP user guide for more information.
5. RX_PEL_PER_CLK – Number of Pixels per Clock as defined in the synthesis_directives.v.

3. Design and Module Description

The top-level design (mipi2parallel_top.v) consists of the following modules:

- mipi2parallel
- int_pll
- i2c_target

In addition, more PLLs may be added if necessary, depending on RX D-PHY configurations.

3.1. mipi2parallel

This module act as a wrapper for the 2 main modules in this design:

- rx_dphy
- b2p

This module also provides reset synchronization for the various clock domains as required by the IPs.

3.1.1. rx_dphy

This module must be created for the RX interface according to the required configuration, such as the number of lanes, bandwidth, and others. Figure 3.1 show an example of IP interface settings in the Clarity Designer of the Lattice Diamond for the CSI-2/DSI D-PHY Receiver Submodule IP. Refer to CSI2/DSI D-PHY Receiver IP Core User Guide (FPGA-IPUG-02025) for details.

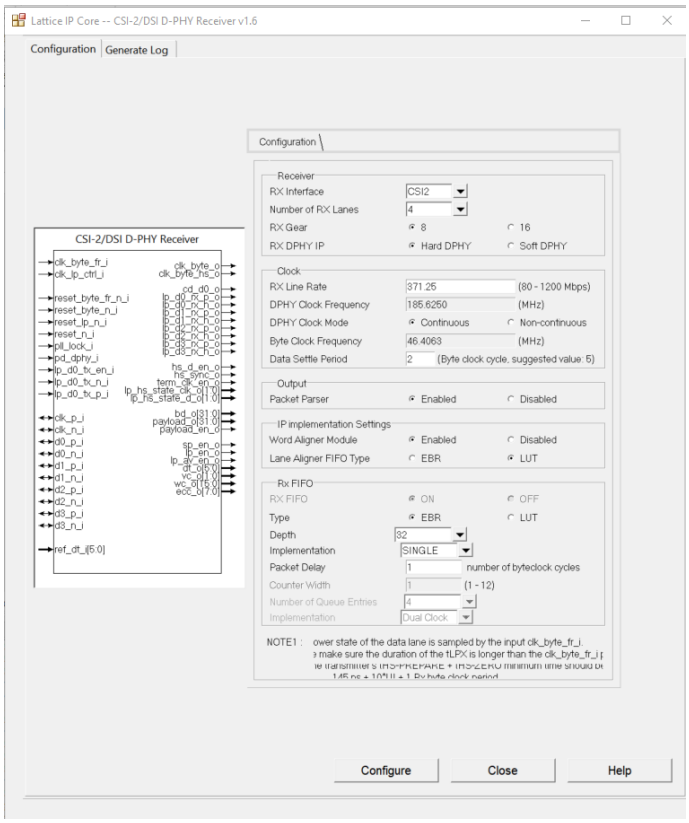

Figure 3.1. rx_dphy IP Creation in Clarity Designer

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

- RX Interface – Select CSI-2 or DSI. Must match with synthesis directive.
- D-PHY RX IP – Select Hard D-PHY or Soft D-PHY. The setting must match RX_DPHY_HARD directive.
- Number of RX Lanes – Set according to RX interface 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 150 MHz or timing error occurs with Gear 8.
- RX Line Rate – Set according to the RX interface configuration. 1200 is the maximum for Soft D-PHY and 1500 is the maximum for Hard D-PHY configuration.
- DPHY Clock Mode – Select Continuous or Non-continuous. Must match RX_CLK_MODE_* setting (Continuous = HS_ONLY, Non-continuous = HS_LP).
- Data Settle Period – Corresponds to $T_{HS-SETTLE}$ D-PHY parameters in byte clock domain. Refer to the Excel file doc/mipi2parallel_lifmd.xlsx for the guidance.
- Packet Parser – Enabled.
- Word Aligner Module – Enabled.
- Enable Miscellaneous Status Signals – Select disabled (unchecked).
- Enable Lane Aligner Module – Select checkbox to enable (checked) for Soft D-PHY with 2-lane and 4-lane configurations.
- Lane Aligner FIFO type – Select LUT.
- RX_FIFO Enable – Select disabled (unchecked) for continuous clock mode with Soft D-PHY, select checkbox to enable (checked) for all other cases.
- Implementation – Select LUT for continuous clock mode with Hard D-PHY, refer to for continuous clock mode with Hard D-PHY, refer to RX FIFO section for non-continuous clock mode.
- Depth – Select 16 for continuous clock mode with Hard D-PHY, refer to RX FIFO section for non-continuous clock mode.
- Type – Select SINGLE for continuous clock mode with Hard D-PHY, refer to RX FIFO section for non-continuous clock mode.
- Packet Delay – Set 1 for continuous clock mode with Hard D-PHY, refer to RX FIFO section for non-continuous clock mode.
- Clock Mode – Select DC (dual clock).

This module takes serial CSI-2/DSI data and outputs byte data after de-serialization in MIPI High Speed mode. The .sbx file included in the project (ipk/ipk.sbx) can be used to reconfigure the IP as per the user configuration requirements. If you are creating this IP from scratch, it is recommended to set the design name to *rx_dphy* so that you do not need to modify the instance names of these IPs in mipi2parallel.v as well as the simulation setup file. Otherwise, you need to modify the names accordingly.

In case of four lanes, RX Gear = 16 in DSI mode, there is a possibility to encounter STA (Static Timing Analysis) failure in Lattice Diamond, especially in higher bandwidth. Most likely this affects in case that there exist two short and/or long packet headers within consecutive eight bytes in the MIPI DSI stream. In this case, DSI stream must be changed to increase the interval between two packet headers.

3.1.1.1. RX FIFO

RX FIFO is useful especially in non-continuous clock mode and the continuous byte clock cannot have the exact same frequency as the non-continuous byte clock used in D-PHY RX IP. It resides after the word aligner in case of Hard D-PHY RX IP and resides before the word aligner in case of Soft D-PHY RX IP.

Hard D-PHY in Continuous Clock Mode

In this case, the minimum configuration of RX FIFO is recommended as mentioned above (LUT based, Depth = 16, type = SINGLE, Packet Delay = 1, Clock Mode = DC).

Soft D-PHY in Continuous Clock Mode

In this case, RX FIFO is not necessary and RX_FIFO Enable should be disabled (unchecked).

Non-Continuous Clock Mode

In this case, RX FIFO configuration depends on the relationship between the non-continuous byte clock in D-PHY RX IP and the continuous byte clock, which is most likely generated by GPLL. The non-continuous byte clock is used to write the data to RX FIFO and the continuous byte clock is used to read the data from RX FIFO.

- Continuous Byte Clock = Non-Continuous Byte Clock

  In this case, the minimum configuration of RX FIFO is recommended (LUT based, Depth = 16, type = SINGLE, Packet Delay = 1, Clock Mode = DC). This is the recommended settings for this design.

- Continuous Byte Clock < Non-Continuous Byte Clock

  In this case type = SINGLE and Packet Delay = 1 is recommended and others depend on the frequency ratio between these two clocks. When the clock speed difference gets larger, the required depth of RX FIFO gets larger. First, it is important to know the horizontal blanking period of the incoming RX channel. For example, in case that one-line active video period is 40 us and the horizontal blanking is 4 µs, then we have 10 % of extra time to process the active data. This means the continuous byte clock can be as slow as ~-10% comparing to the non-continuous byte clock to avoid RX FIFO overflow.

- Continuous Byte Clock > Non-Continuous Byte Clock

  There are two options in this case:

  - Use type = SINGLE with Large Packet Delay

    Set the Depth large enough to contain the necessary data to avoid RX FIFO underflow after FIFO read begins after the time specified by Packet Delay. In general, Packet Delay must be set close the depth of the RX FIFO. This configuration can be used when we have enough time interval between the last active line and the frame end short packet so that the frame end short packet is not written to RX FIFO while it still contains the last active line of video data.

  - Use type = QUEUE with Number of Queue Entries = 2

    This is useful when the time interval between the last active line and frame end short packet is short or unknown. Depth must be set large enough to contain one active line data (plus some more for short packet data). This mode is also useful when line start and the line end short packets exist in the incoming RX stream. In that case, Number of Queue Entries = 4 and extra depth is required (one line plus two short packet data). FIFO read begins after each HS data transaction is completed. EBR must be used. Counter Width is determined by the amount of the one-line video data plus extra overheads by preceding HS zero data and trail byte in the end of HS transmission.

  - Frequency relationship is unknown

    When the continuous byte clock is within the certain range against the non-continuous byte clock (for example, two clocks come from different clock sources which have ppm tolerance), we have no idea which clock is faster. The simplest way is to use type = SINGLE with setting Packet Delay to the midpoint of FIFO depth when the tolerance is in ppm level. QUEUE can also be used as described above.

In case you do not have detailed information regarding RX data (whether containing lane start/end short packet, interval of the horizontal blanking period against active line period), the safest way is to set the continuous byte clock faster than the non-continuous byte clock and use type = QUEUE with Number of entries = 4 even though it might require more EBR resources comparing to type = SINGLE.

3.2. b2p

This module must be created for the RX interface according to the required configuration, such as data type, the number of lanes, RX Gear, and others. Figure 3.2 shows an example of IP interface settings in Clarity Designer software for the byte2pixel IP. Refer to Byte-to-Pixel Converter Converter IP User Guide (FPGA-IPUG-02027) for details.

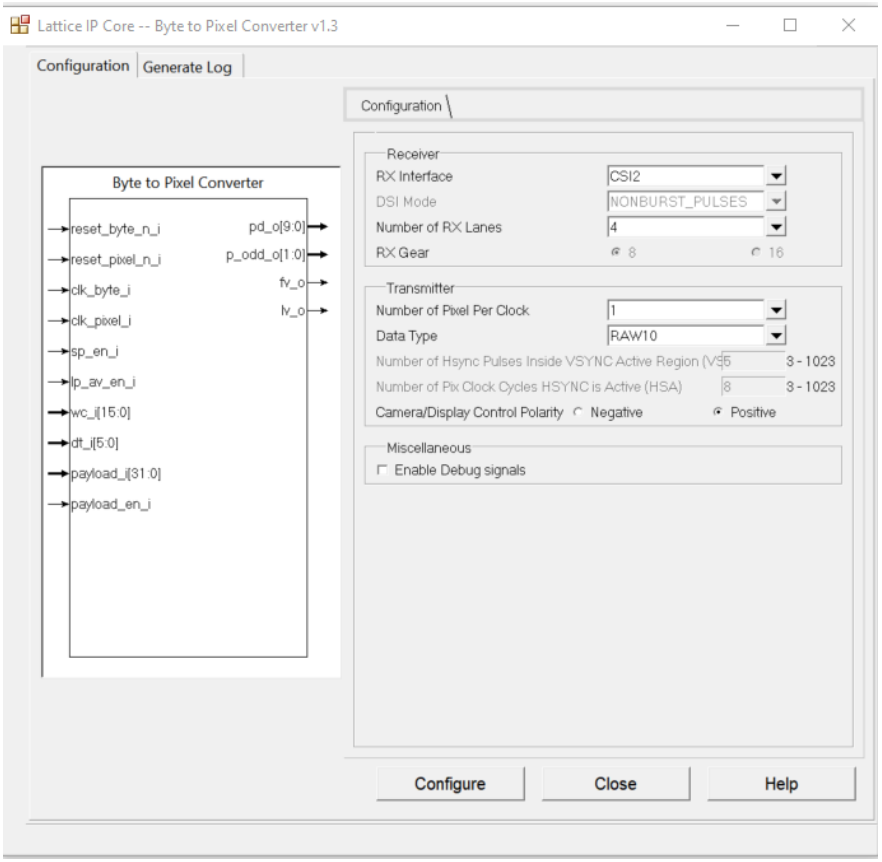

Figure 3.2. b2p IP Creation in Clarity Designer

The following shows 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.
- DSI Mode – Select Non-Burst Pulses (applicable to DSI Interface only)
- Number of RX Lanes – Select 1, 2 or 4. Set the same value as RX D-PHY IP.
- RX Gear – Select 8 or 16. Set the same value as RX D-PHY IP.
- Number of Pixel Per Clock – Select 1, 2 or 4.
- Data Type – Must be matched to the Video Data Type in synthesis directive.
- Camera/Display Control Polarity – Select Positive.
- Enable Debug Ports – Select disabled (unchecked).

The Byte-to-Pixel Converter IP converts the D-PHY CSI-2/DSI standard based byte data stream to standard pixel data format. The .sbx file included in the project (ipk/ipk.sbx) can be used to reconfigure the IP as per the user configuration requirements. If you are creating this IP from scratch, it is recommended to set the design name to b2p so that you do not need to modify the instance name of this IP in the mipi2parallel.v design as well as in the simulation setup file. Otherwise, you need to modify the names accordingly.

3.3. int_pll

The frequency relationship between byte clock and pixel clock is as follows:

*Pixel Clock = (byte clock * number of RX lanes * 8) / data width*

In most cases, you need to create PLL module to generate clocks required by the mipi2parallel module. The reference clock for the PLL in this example is routed from the 27MHz external oscillator clock. CLKOP is used to drive the clk_lp_ctrl_i clock pin, CLKOS is used to generate the free running byte clock clk_byte_fr_i, and CLKOS2 is used as the pixel clock clk_pixel_i, as well as the output pixel clock clk_pixel_o.

Note that for the continuous clock mode, clk_byte_fr_i pin is driven directly from the clk_byte_hs_o pin of the rx_dphy module. clk_lp_ctrl_i clock is not required to be driven in the continuous clock mode too. Thus, in the case of non-continuous clock mode, CLKOP and CLKOS pins are not being used to drive any blocks in this example design.

As a guide, user can refer to the Excel file doc/mipi2parallel_lifmd.xlsx from the design package file to assist with the calculation required for the clock's frequencies.

Figure 3.3 shows an example of IP interface settings in Clarity Designer for the PLL IP. Refer to CrossLink sysCLOCK PLL/DLL Design and Usage Guide (FPGA-TN-02015) for details.

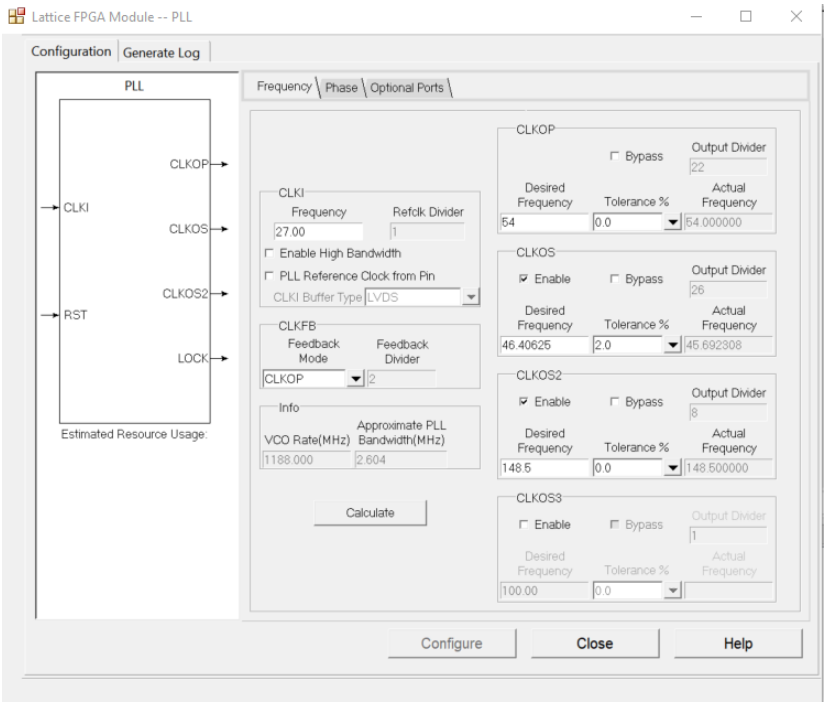

Figure 3.3. int_pll IP Creation in Clarity Designer

The .sbx file included in the project (ipk/int_pll.sbx) can be used to reconfigure the IP as per the user configuration requirements. If you are creating this IP from scratch, it is recommended to set the design name to int_*pll* so that you do not need to modify the instance names of these IPs in mipi2parallel_top.v file. Otherwise, you need to modify the name accordingly.

The following code snippet shows the condition in which clocks are generated from int_pll in case.

```
int_pll int_pll_inst (
    .CLKI    (ref_clk_i),       // 27 MHz Ref clock
    .RST     (~reset_n_i),
    .CLKOP   (clk_lp_ctrl_w),   // 54 MHz for non-continuous clock mode
    .CLKOS   (clk_byte_fr_w),   // Only used in non-continuous clock mode
    .CLKOS2  (clk_pixel_w),     // Pixel clock
    .LOCK    (pll_lock_w)
);
```

More than one PLL can be used if the required clock frequencies for the rx_clk_byte_fr_pll and clk_pixel cannot be generated using a single PLL.

3.4. i2c_target

This module is instantiated to allow users to control or overwrite any parameters or configurations dynamically via I^2C bus. This module act as a wrapper for the I2C Hard IP (i2c_s) and map the addresses to each of the configurations.

Register map that being used in this design is showed in Table 3.1. In this example design, only bit 0 of the sub-address 0 is being used, which is act as SW Reset for the mipi2parallel module. To modify or add the register map, user would need to modify the i2c_target_inst instance declaration in the mipi2parallel_top.v file.

Table 3.1 I^2C Target Register Map

Sub address	Name	Bits	Description
0	SW Reset	[0]	Software reset register. When this is active (1'b1), mipi2parallel module is in reset condition. Active low. Default value is defined by the SW_RST_N directive.
0	—	[1:7]	Reserved
1 – 15	User Defined	[7:0]	User can customize this address for their specific applications

3.4.1. i2c_s

The user can use the .sbx file (ipk/ipk.sbx) included in the sample project and re-configure. In case that user make this IP from scratch, it is recommended to set the design name to i2c_s so that user do not need to modify the instance name of this IP in the top-level design as well as simulation setup file. Otherwise, the user needs to modify the names accordingly. There exist two I2C Hard IP modules in CrossLink and I2C0 is used in this IP as shown in Figure 3.4. The user must change the setting if the other IP (I2C1) is used. Also, the System Bus clock frequency must be changed according to the clock fed to this module. In this design, the System Bus clock is driven by the external ref_clk_i pin.

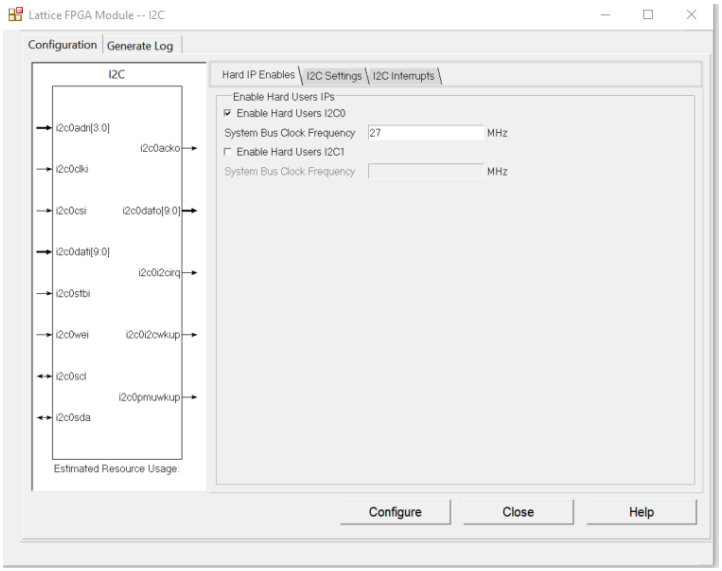

Figure 3.4. i2c_s IP Creation in Clarity Designer (1/3)

Figure 3.5 shows the basic settings of I2C IP. The user 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 the case of I2C0, LSB 2 bits are fixed to 2'b10.

Figure 3.6 shows the interrupt settings. At least Tx/Rx Ready must be enabled (checked).

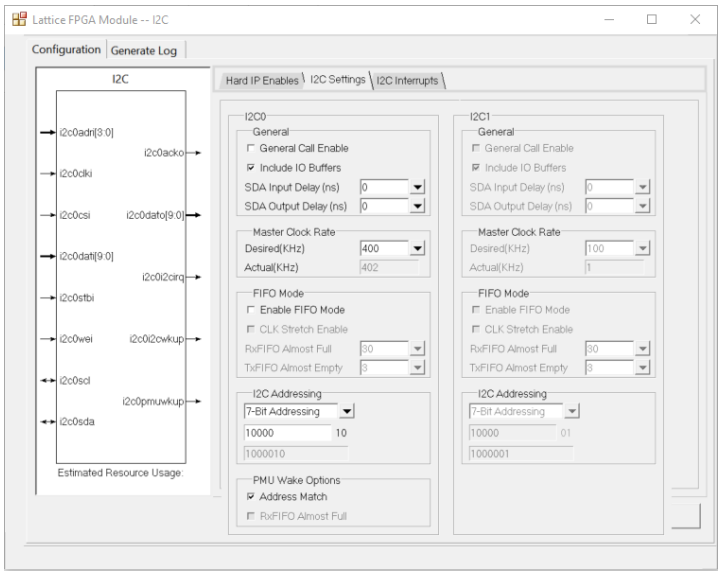

Figure 3.5. i2c_s IP Creation in Clarity Designer (2/3)

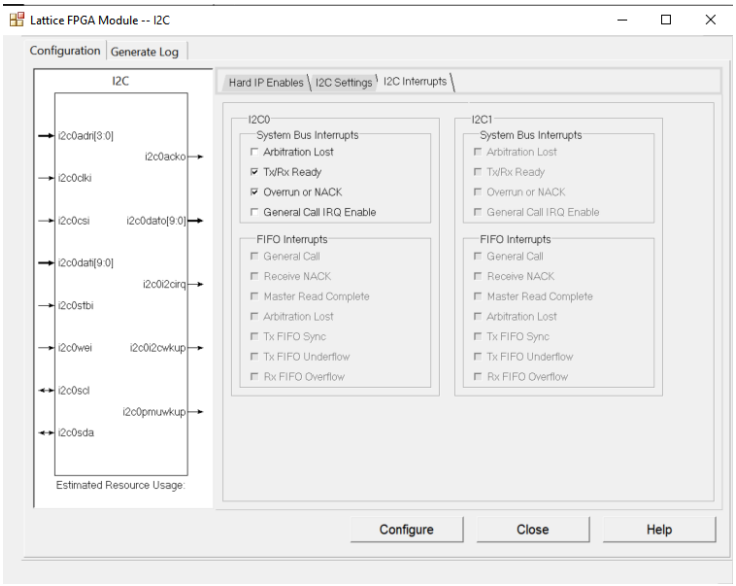

Figure 3.6. i2c_s IP Creation in Clarity Designer (3/3)

This module is equipped with parameter registers of 4-bit address area of I2C sub address shown in Table 3.1.

4. Design and File Modifications

This RD is based on version 1.6 of the RX D-PHY IP and version 1.3 of the Byte2Pixel IP. Some modifications are required depending on user configuration in addition to the two directive files (synthesis_directives.v and simulation_directives.v).

4.1. Top Level RTL

The PLL setting in the current top-level file (mipi2parallel_top.v) is based on the configuration selected for this reference design. According to the configuration you use, you need to modify the PLL instantiation as described in int_pll section. In addition, an instance names of RX D-PHY (rx_dphy), Byte to Pixel (b2p), I^2C target (i2c_s) and internal PLL (int_pll) need to be modified if you create this IP(s) with different name(s).

To assist on modifying the design, user can use the Excel file (doc/mipi2parallel_lifmd.xlsx) to determine the recommended settings. User is required to key-in or select the values based on the design in the highlighted cells as shown in Figure 4.1.

	A	B	C	D	E	F	G
1	**Items**	**Values**	**Warnings**		**Legend:**		
2	**RX Line Rate (Mbps)**	371.25					Users to Key in
3	**RX Number of Lanes**	4					Calculated values
4	**RX Gear**	8					
5	**Data Type**	RAW10					
6	**Byte to Pixel pixel per clock output**	1					
7	**RX D-PHY Data Settle Period**	2					
8	**Byte To Pixel Output Width**	10					
9	**Recommended clk_byte_fr frequency (MHz)**	46.40625					
10	**Required clk_pixel frequency (MHz)**	148.5					
11							

Figure 4.1. Excel calculator file.

5. Design Simulation

The script file (simulation/simulation.spf) and testbench files are provided to run the functional simulation by Modelsim. If you follow the naming recommendations regarding design name and instance name when the RX D-PHY, Byte2Pixel, Internal I^2C, Internal PLL IPs are created by Clarity Designer, user can execute the .spf file as below:

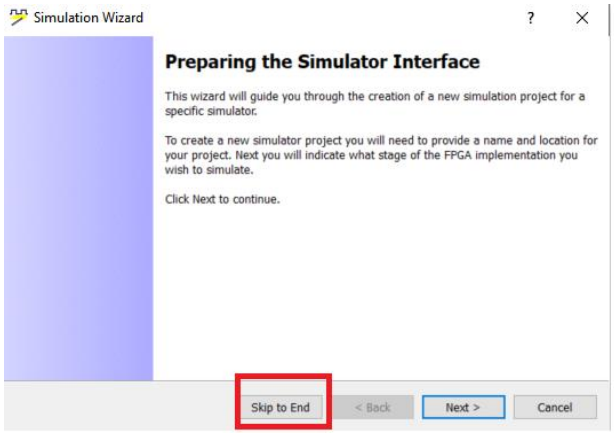

Figure 5.1. Launching simulation 1/2

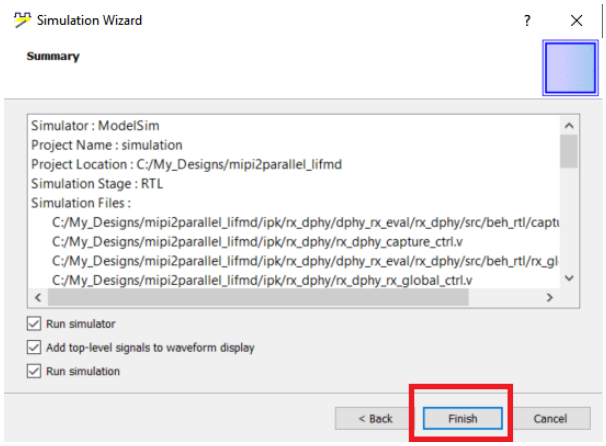

Figure 5.2. Launching simulation 2/2

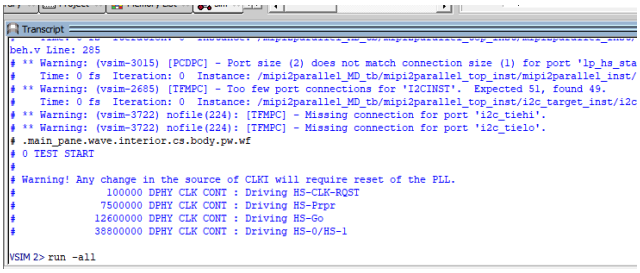

Figure 5.3. Running the simulation by typing "run -all" in the transcript window

You need to modify *simulation_directives.v* according to your configuration (refer to Simulation Directives for details). By executing the script in Modelsim, compilation and simulation are executed automatically. The testbench takes all data comparison between the expected data and output data from the RD.

```
#           15099120800 Waiting for PLL lock...
#
#           15099120800 PLL lock DONE
#
#           15109120800 Releasing SW_RST thru I2C
#
#           23619120800 MIPI D-PHY Clock begins...
#
#           23629120800 Activating DSI/CSI-2 model
#
#           23629120800 DPHY CSI-2 model activated
#
#           31629120800 DPHY CSI-2 FS begins...
…
…
##### LV de-assertion #####
#           41416001600 DPHY CLK : Driving HS-CLK-RQST
#           41423401600 DPHY CLK : Driving HS-Prpr
#           41428501600 DPHY CLK : Driving HS-Go
#           41454701600 DPHY CLK : Driving HS-0/HS-1
#           41456856000 DPHY DATA : Driving HS-RQST
#           41464256000 DPHY DATA : Driving HS-Prpr
#           41469356000 DPHY CLK : Driving HS-Go
#           41481894900 DPHY CLK : Driving SYNC Data
#           41484183800 DPHY Driving Data header
#           41484183800 DPHY Driving Data header for data = 2b e8 03 1e
##### LV assertion #####
#           42024938200 DPHY Driving CRC[15:8] = 5f; CRC[7:0] = 4b
#           42024938200 DPHY 0 Lane 2 : Driving trail bytes
#           42024938200 DPHY 0 Lane 3 : Driving trail bytes
#           42027092600 DPHY 0 Lane 0 : Driving trail bytes
#           42027092600 DPHY 0 Lane 1 : Driving trail bytes
#           42037766700 DPHY 0 Lane 0 : Driving stop
#           42037766700 DPHY 0 Lane 1 : Driving stop
#           42037766700 DPHY 0 Lane 2 : Driving stop
#           42037766700 DPHY 0 Lane 3 : Driving stop
#           42058196800 DPHY CLK : Driving CLK-Trail
#           42064196800 DPHY CLK : Driving CLK-Stop
##### LV de-assertion #####
#           42564296800 DPHY CLK : Driving HS-CLK-RQST
#           42571696800 DPHY CLK : Driving HS-Prpr
#           42576796800 DPHY CLK : Driving HS-Go
#           42602996800 DPHY CLK : Driving HS-0/HS-1
#           42605151200 DPHY DATA : Driving HS-RQST
#           42612551200 DPHY DATA : Driving HS-Prpr
#           42617651200 DPHY CLK : Driving HS-Go
#           42630190100 DPHY CLK : Driving SYNC Data
#           42632479000 DPHY Driving Data header
#           42632479000 DPHY Driving Data header for data = 2b e8 03 1e
##### LV assertion #####
#           43173233400 DPHY Driving CRC[15:8] = f2; CRC[7:0] = 36
#           43173233400 DPHY 0 Lane 2 : Driving trail bytes
#           43173233400 DPHY 0 Lane 3 : Driving trail bytes
#           43175387800 DPHY 0 Lane 0 : Driving trail bytes
#           43175387800 DPHY 0 Lane 1 : Driving trail bytes
#           43186061900 DPHY 0 Lane 0 : Driving stop
#           43186061900 DPHY 0 Lane 1 : Driving stop
#           43186061900 DPHY 0 Lane 2 : Driving stop
#           43186061900 DPHY 0 Lane 3 : Driving stop
#           43206492000 DPHY CLK : Driving CLK-Trail
#           43212492000 DPHY CLK : Driving CLK-Stop
##### LV de-assertion #####
#           43712592000 DPHY CLK : Driving HS-CLK-RQST
#           43719992000 DPHY CLK : Driving HS-Prpr
#           43725092000 DPHY CLK : Driving HS-Go
#           43751292000 DPHY CLK : Driving HS-0/HS-1
#           43753446400 DPHY DATA : Driving HS-RQST
#           43760846400 DPHY DATA : Driving HS-Prpr
```

Figure 5.4 shows the simulation waveform of the full view of 10 lines and 1 frame for the CSI-2 interface.

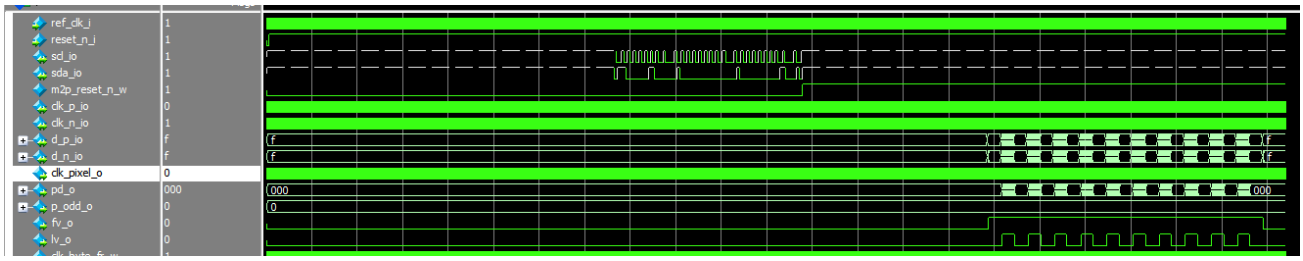

Figure 5.4. Simulation Waveform

The simulation waveform can be accessed by opening the *vsim.wlf* file in the Modelsim from the simulation directory. More signals of a module can be added to the waveform as required.

6. Known Limitations

The following are the limitations of this reference design:

- Only the following data types are supported for MIPI DSI interface: RGB888, RGB666, RGB666_LOOSE
- Only the following data types are supported for MIPI CSI-2 interface: RGB888, RAW8, RAW10, RAW12, YUV_420_8, YUV_420_8_CSPS, LEGACY_YUV_420_8, YUV_420_10, YUV_420_10_CSPS, YUV_422_8, YUV_422_10

7. Design Package and Project Setup

The MIPI to Parallel with CrossLink Reference Design is available on www.latticesemi.com. Figure 7.1 shows the directory structure. The design is targeted for LIF-MD6000-6MG81I. *synthesis_directives.v* and *simulation_directives.v* are set to configure the design with following configuration:

- RX – 4-lanes, Gear 8 with HARD D-PHY in continuous clock mode
- TX – CSI-2, RAW10 parallel data with 1 pixels/clock

You can modify the directives for your own configuration.

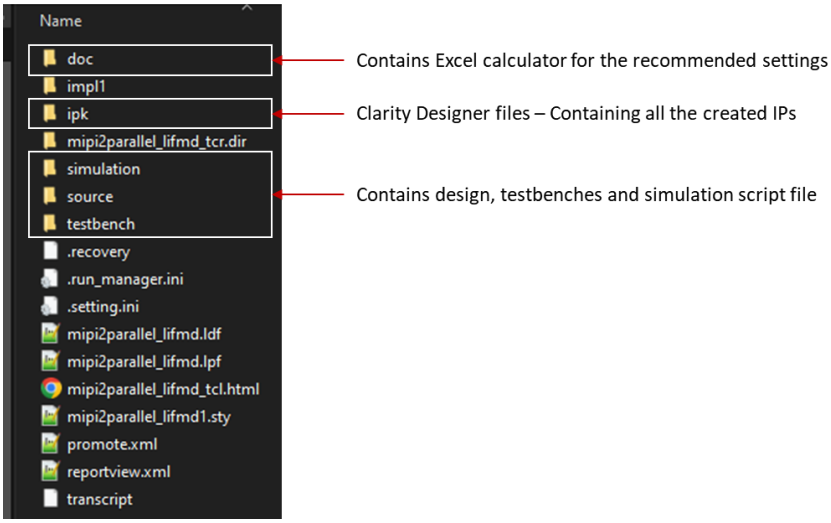

Figure 7.1. Directory Structure

Figure 7.2 shows the design files used in the Lattice Diamond project. All of the IPs (PLL, I^2C, RX DPHY, B2P) are lumped into a single .sbx file. By specifying mipi2parallel_top as a top-level design, all unnecessary files are ignored. Constraint file (mipi2parallel_lifmd.lpf) is also included in the project for reference. You can modify it according to your own configuration.

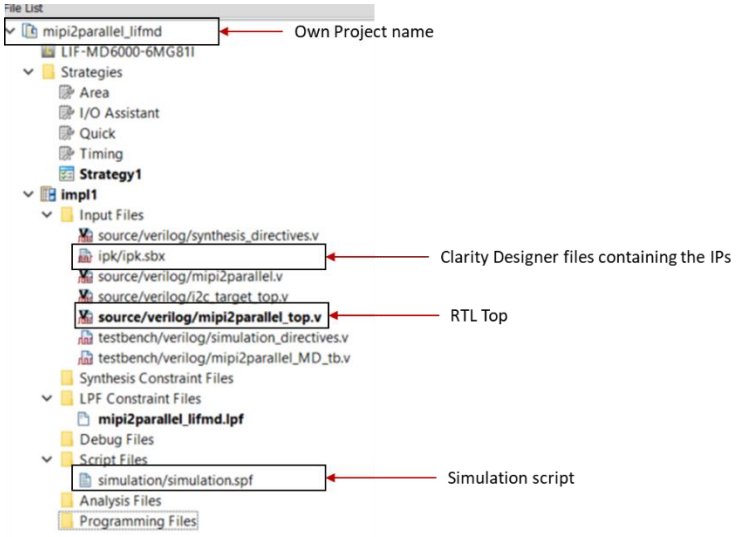

Figure 7.2. Project Files

8. Resource Utilization

Resource utilization depends on the configuration used. Table 8.1 shows the resource utilization examples under certain configurations targeting LIF-MD-6000. This is just a reference and actual usage varies, especially when EBR usage depends on the horizontal resolution of each RX channels and clock frequency relationship between non-continuous byte clock and continuous byte clock when non-continuous clock mode is used.

Table 8.1. Resource Utilization Examples

Configuration	LUT	FF	EBR	I/O
4-lane, Gear 8, Hard D-PHY, CSI-2, RAW10, 1 Pixels/clock	1610	969	3	19
1-lane, Gear 16, Hard D-PHY, DSI, RGB888, 1 Pixels/clock	912	658	2	34
2-lane, Gear 8, Soft D-PHY, DSI, RGB666, 1 Pixels/clock	1106	667	1	34
1-lane, Gear 8, Soft D-PHY, CSI-2, RAW8, 1 Pixels/clock	658	447	1	21

References

For more information refer to:

- CrossLink web page
- CSI2/DSI D-PHY Receiver IP Core User Guide (FPGA-IPUG-02025)
- Byte-to-Pixel Converter Converter IP User Guide (FPGA-IPUG-02027)
- CrossLink sysCLOCK PLL/DLL Design and Usage Guide (FPGA-TN-02015)
- MIPI Alliance Specification for D-PHY Version 1.1
- MIPI Alliance Specification for Display Serial Interface 2 (DSI) Version 1.1
- MIPI Alliance Specification for Camera Serial Interface 2 (CSI-2) Version 1.1
- Lattice Diamond FPGA design software
- Lattice Diamond 3.13 User Guide
- Lattice Insights for Lattice Semiconductor training courses and learning plans

Technical Support Assistance

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

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

Revision History

Revision 1.0, January 2024

Section	Change Summary
All	Initial release.

www.latticesemi.com