CSI-2/DSI D-PHY Receiver Submodule IP

User Guide

FPGA-IPUG-02025 Version 1.0

July 2017

Contents

Figures

Tables

1. Introduction

The Lattice Semiconductor CSI-2/DSI D-PHY Receiver Submodule IP converts DSI or CSI-2 data to either 64-bit or 32-bit data for Lattice Semiconductor CrossLink™ devices. This is useful for wearable, tablet, human-machine interfacing, medical equipment and many other applications.

Mobile Industry Processor Interface (MIPI®) D-PHY has become the industry's primary high-speed PHY solution for camera and display interconnection in mobile devices. It is typically used in conjunction with MIPI Camera Serial Interface-2 (CSI-2) and MIPI Display Serial Interface (DSI) protocol Specifications. It meets requirements of low-power, low-noise generation, and high-noise immunity that mobile phone designs demand.

The CSI-2/DSI D-PHY Receiver Submodule IP is intended for use in applications that require a D-PHY receiver in the FPGA logic.

This user guide is for CSI-2/DSI D-PHY Receiver Submodule IP design version 1.0.

1.1. Quick Facts

Table 1.1. provides quick facts about the CSI-2/DSI D-PHY Receiver Submodule IP for CrossLink device.

Table 1.1. CSI-2/DSI D-PHY Receiver Submodule IP Quick Facts

		MIPI D-PHY Receiver Submodule IP Configuration	
		4-Lane, Gear 16, Soft logic implementation with packet parser and lane aligner	**4-Lane, Gear 8, Soft logic implementation with packet parser and lane aligner**
Core Requirements	FPGA Families Supported	CrossLink	
Resource Utilization	Targeted Device	LIF-MD6000-6MG81I	
	Data Path Width	16 bits per lane, 64 bits total for 4 lanes	8 bits per lane, 32 bits total for 4 lanes
	LUTs	2375	1318
	sysMEM™ EBRs	4	4
	Registers	1069	742
	Programmable IOs	10	10
	Hard D-PHY	0	0
Design Tool Support	Lattice Implementation	Lattice Diamond® 3.9	
	Synthesis	Lattice Synthesis Engine	
		Synopsys® Synplify Pro® L-2016.03L	
	Simulation	Aldec® Active-HDL™ 10.3 Lattice Edition	

1.2. Features

The key features of the MIPI D-PHY Interface for CSI-2/DSI Submodule IP include:

- Compliant with MIPI D-PHY v1.1, MIPI DSI v1.1 and MIPI CSI-2 v1.1 Specifications
- Selection between Hard Rx D-PHY or Soft Rx D-PHY implementation
- Supports MIPI D-PHY interfacing from 160 Mb/s up to 1.5 Gb/s (Hard D-PHY)
- Supports 1, 2, 3 or 4 data lanes and one clock lane
- Supports continuous and non-continuous MIPI D-PHY clock
- Supports all MIPI DSI Video Mode of operation
    - Non-Burst Mode with Sync Pulses
    - Non-Burst Mode with Sync Events
    - Burst Mode
- Optional packet parsing or parallel data translation only
- Supports all MIPI DSI compatible video formats
- Supports all MIPI CSI-2 compatible video formats

1.3. Conventions

1.3.1. Nomenclature

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

1.3.2. 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.

1.3.3. Signal Names

Signal names that end with:

- *"_n"* are active low
- *"_i"* are input signals

  Some signals are declared as bidirectional (IO) but are only used as input. Hence, "_i" identifier is used.
- *"_o"* are output signals

  Some signals are declared as bidirectional (IO) but are only used as output. Hence, "_o" identifier is used.
- *"_io"* are bidirectional signals

2. Functional Description

The MIPI D-PHY Interface for CSI-2/DSI Submodule IP can be configured to include a packet parser, which decodes the contents of the incoming CSI-2 or DSI packets, or to simply output the D-PHY packets without decoding its contents. Table 2.1 describes the functionality of the IP block that includes the packet parser.

2.1. I/O Port Definitions

Table 2.1. MIPI D-PHY Receiver Submodule IP Pin Function Description

Port Name	Direction	Function Description
Clock and Reset		
reset_n_i	I	Asynchronous system reset (active low)
clk_byte_fr_i	I	Continuously running byte clock. This should be div8 (in gear16) or div4 (in gear8) of the input D-PHY clock. This also clocks the logic that detects the Rx D-PHY data lane transitions (lp_hs_ctrl_d0-3 modules). This is also used by the word_align, lane_align and capture_control modules. Payload output is also in this clock domain.
reset_byte_fr_n_i	I	Reset the nets in the clk_byte_fr clock domain. The signal driving this port must already be synchronized to the clk_byte_fr.
clk_lp_ctrl_i	I	Clocks the logic that detects the Rx D-PHY clock lane LP<->HS transitions. The frequency of this clock should be at least twice the low-power clock frequency of the Rx D-PHY clock, i.e., clock period of clk_lp_ctrl_i should at most be half of T_{LPX} to properly sample the LP to HS clock lane transitions. No need to drive this clock if the Rx clock mode is HS_ONLY.
reset_lp_n_i	I	Reset the nets in the clk_lp_hs_ctrl clock domain. The signal driving this port must already be synchronized to the clk_lp_hs_ctrl.
clk_byte_o	O	Generated byte clock from the D-PHY module based on the input D-PHY clock lane, used to latch the internal parallel byte data from dphy_rx_wrap. This is div4 or div8 of the D-PHY clock lane frequency. This is only active when the data lanes are in high-speed mode.
reset_byte_n_i	I	Reset the logic clocked by the clk_byte_o.
clk_byte_hs_o	O	Generated byte clock from the D-PHY module based on the input D-PHY clock lane, active only when the clock lanes are in high-speed mode. This clock is the same as the clk_byte_o when the submodule is in HS_ONLY mode and D-PHY implementation is Soft DPHY. We recommend to connect this to the clk_byte_fr_i during HS_ONLY mode.
MIPI Interface		
clk_p_i, clk_n_i	IO	MIPI D-PHY clock lane
d0_p_io, d0_n_io	IO	MIPI D-PHY data lane 0. Available only for MIPI DSI configuration.
d0_p_i, d0_n_i	IO	MIPI D-PHY data lane 0. Available only for MIPI CSI-2 configuration.
d1_p_i, d1_n_i	IO	MIPI D-PHY data lane 1. Available only for configurations with two or more data lanes.
d2_p_i, d2_n_i	IO	MIPI D-PHY data lane 2. Available only for configurations with four data lanes
d3_p_i, d3_n_i	IO	MIPI D-PHY data lane 3. Available only for configurations with four data lanes.
Fabric Interface (for low-power communications)		
lp_d0_rx_p_o, lp_d0_rx_n_o	O	Low-power values of data lane 0 true and differential lines, respectively.
lp_d1_rx_p_o, lp_d1_rx_n_o	O	Low-power values of data lane 1 true and differential lines, respectively.
lp_d2_rx_p_o, lp_d2_rx_n_o	O	Low-power values of data lane 2 true and differential lines, respectively.
lp_d3_rx_p_o, lp_d3_rx_n_o	O	Low-power values of data lane 3 true and differential lines, respectively.
lp_d0_tx_p_i, lp_d0_tx_n_i	I	Input port to drive the low-power values of data lane 0 when low-power data transmission is enabled (for reverse direction communications).
lp_d0_tx_en_i	I	When driven high, this puts the data lane 0 in transmit mode for LPDT reverse direction communications. This must be driven low during forward direction.

Table 2.1. MIPI D-PHY Receiver Submodule IP Pin Function Description *(Continued)*

Port Name	Direction	Function Description
Fabric Interface (without packet parser)		
bd0_o	O	Parallel data from lane 0. This is 8-bit wide if the design is configured as gear8, or 16-bit wide if gearing is 16.
bd1_o	O	Parallel data from lane 1. This is 8-bit wide if the design is configured as gear8, or 16-bit wide if gearing is 16.
bd2_o	O	Parallel data from lane 2. This is 8-bit wide if the design is configured as gear8, or 16-bit wide if gearing is 16.
bd3_o	O	Parallel data from lane 3. This is 8-bit wide if the design is configured as gear8, or 16-bit wide if gearing is 16.
capture_en_o	O	This functions as a valid signal for bd0_o to bd3_o. This is enabled from the start of synchronization pattern 'B8 up to the last data captured before detecting LP-11 state (of any lane, if Soft D-PHY; of data lane0 for Hard D-PHY). This is the same as hs_sync_o.
Fabric Interface (with packet parser)*		
sp_en_o	O	This pulse asserts when a short packet is detected.
lp_en_o	O	This pulse asserts when long packet is detected.
lp_av_en_o	O	This pulse asserts when a valid video data type is detected. The valid data type is defined by the input signal ref_dt_i.
vc_o	O	This is the 2-bit virtual channel ID of the received packet.
wc_o	O	This is the 16-bit Word Count field. This denotes the number of bytes in the payload of a long packet. In a short packet, this contains a 2-byte data.
dt_o	O	This is the 6-bit CSI-2 or DSI data type field.
ecc_o	O	This is the received 8-bit error correction code (ECC). The submodule does not detect or correct any ECC errors.
payload_o	O	This is the payload of a long packet arranged the way it was received in the data lanes. The width of this bus is the number of gear bits multiplied by the number of lanes (RX_GEAR*NUMBER_OF_LANE).
payload_en_o	O	This signifies the arrival of valid payload data.
ref_dt_i	I	Input reference data type. The long packet video valid signal indicator lp_av_en_o asserts if the incoming packet data type is equal to this input.
Miscellaneous Signals (status and debug)		
pll_lock_i	I	PLL lock indicator, if a PLL is used to generate a free-running byte clock. Set this to 1 if a PLL is not used.
bd_o	O	Valid only for Rx gear 8. This is the DPHY byte data.
term_clk_en_o	O	Active-high enable signal for the line termination of the D-PHY clock lane. This is asserted on detection of transition from LP-11 to LP-01 of the clock lane, and deasserts upon detection of LP-11 after a high-speed mode.
hs_d_en_o	O	Active-high high-speed mode enable for data lane d0. For Hard D-PHY IP, this signal is also used for HS mode enable for the other data lanes.
cd_d0_o	O	Contention detection indicator
hs_sync_o	O	This indicates the successful detection of the synchronization code 'B8 in the data lanes. This signal asserts from the start of synchronization pattern 'B8 up to the last data captured before detecting LP-11 state (of any lane, if Soft D-PHY; of data lane0 for Hard D-PHY).
lp_hs_state_clk_o	O	2-bit state encoding of the D-PHY clock controller
lp_hs_state_d_o	O	2-bit state encoding of the D-PHY data lane 0 controller

There is another set of fabric interface signals when the packet parser is enabled and the payload_o data width is greater than 32 bits. This set is valid when there are two packets received within one byte clock cycle. This is discussed in more detail in subsequent sections.

2.2. Submodule without Packet Parser

This submodule can be instantiated without the packet parser. In this configuration, the output is the received bytes from the D-PHY starting from the reception of the Start of Transmit (SoT) code in all the active data lanes until the detection of the LP-11 signifying the end of high-speed transmission. This includes the trail bits and any glitches that may be seen on the D-PHY bus. It is up to the interfacing logic to obtain and decode the valid data packets from the trail. This configuration is useful for bridging D-PHY packets without going to the protocol level.

In Figure 2.1, the data reflected to the output side is highlighted in yellow.

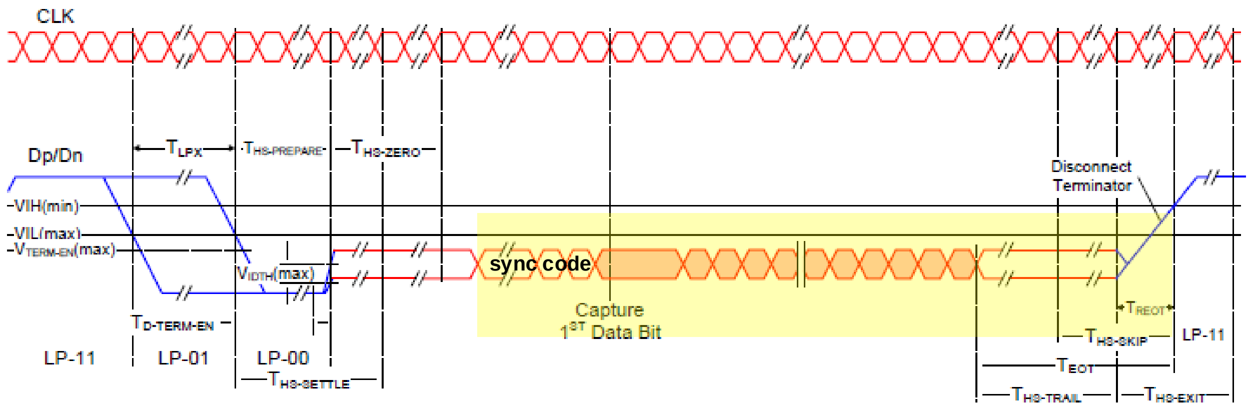
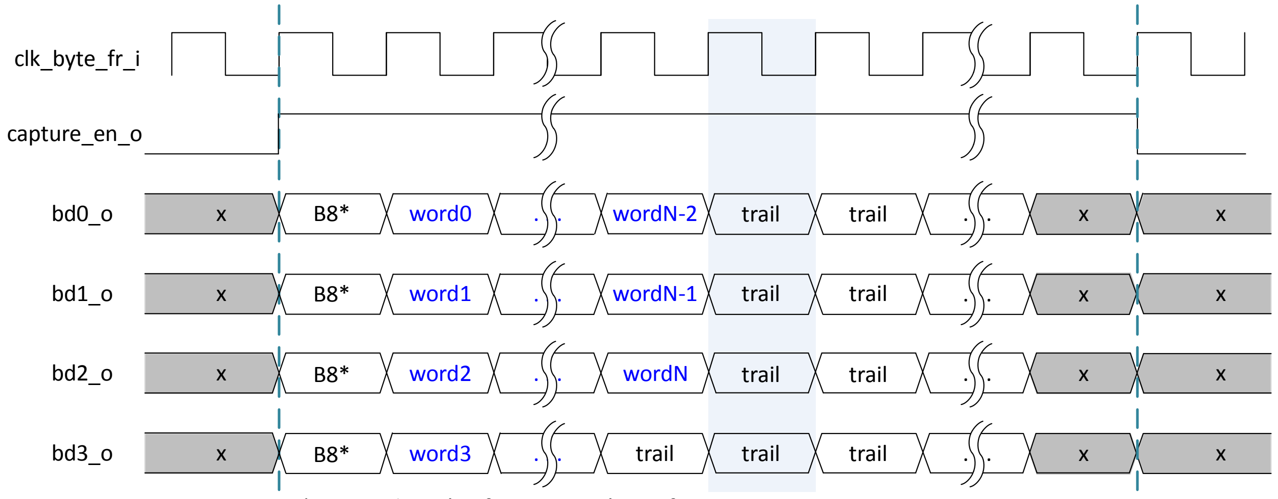


Figure 2.1. MIPI D-PHY Rx IP captured Data

Figure 2.2 shows the fabric interface timing diagram.



* sync code – 8'B8 for gear8; 16'B800 for gear16

Figure 2.2. Fabric Interface Timing Diagram of the MIPI Rx D-PHY IP without Packet Parser

There must be at least one clock cycle where all the data lanes are sending trail data (shown in blue in Figure 2.2). HS-skip logic is also not provided in this submodule. We assume that the parts of data that the interfacing logic decodes are valid by parsing the data type and wordcount fields.

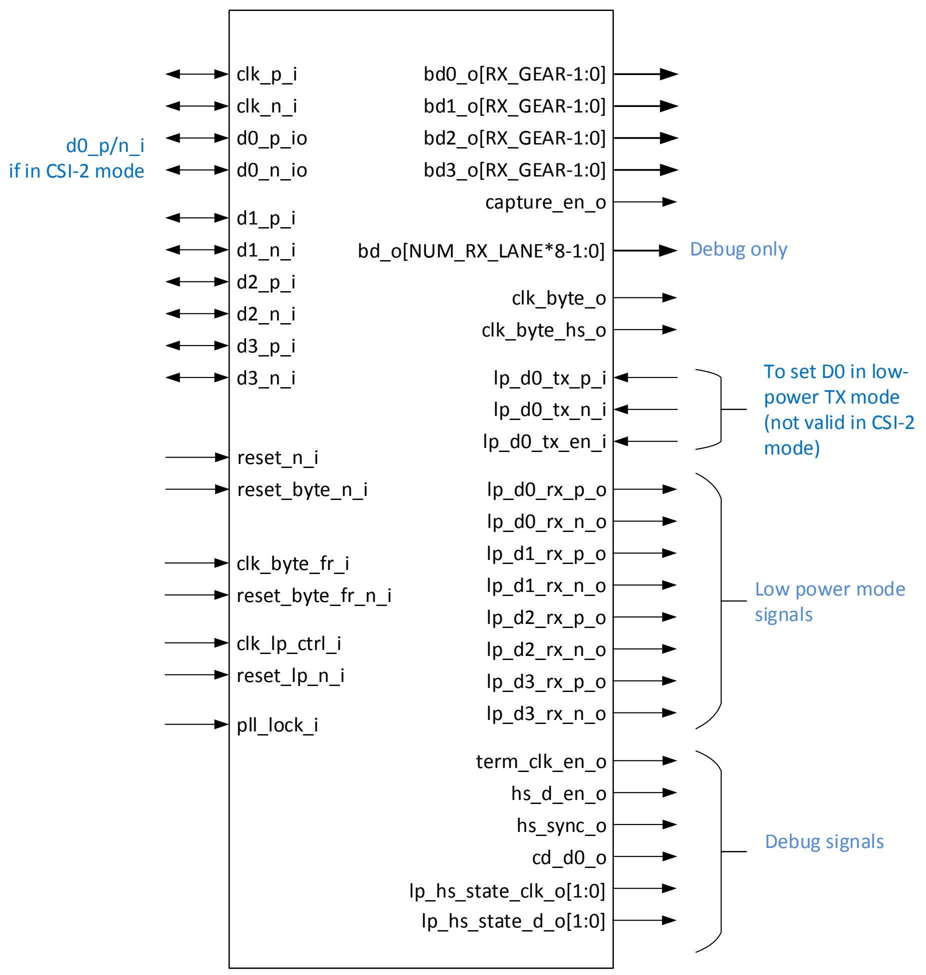


Figure 2.3. Block Diagram of MIPI Rx D-PHY IP without Packet Parser

2.3. Submodule with Packet Parser

This submodule can be instantiated with a DSI or a CSI-2 packet parser. The parser checks the incoming data for a valid data type and corresponding packet fields.

Figure 2.4 shows the timing diagram of the submodule interface with the packet parser enabled.

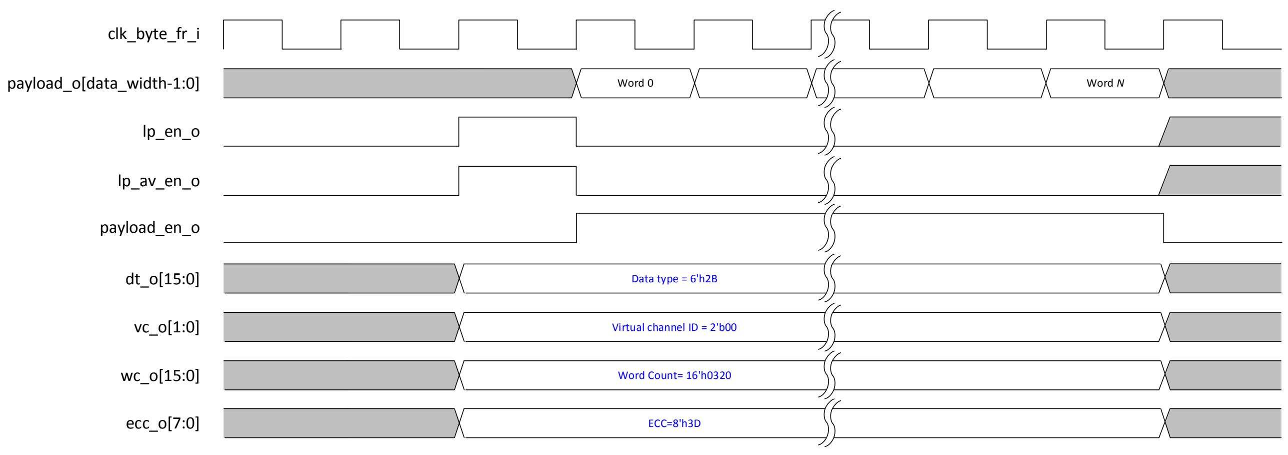


Figure 2.4. Rx D-PHY with Packet Parser Output Timing Diagram

The lp_en_o or sp_en_*o* signal will assert when a valid data type is received. These signals also indicate the valid data type, virtual channel ID, wordcount and ECC fields.

The lp_av_en_o only asserts with the lp_en_o, if the long packet received is the same as the input reference data type ref_dt_i. This is to differentiate active video packets from other long packets, such as null or blanking. Consequently, this signal does not assert on any video data type other than the defined ref_dt_i value.

The payload_en_o signal indicates that the data in the payload_o bus contain the valid payload bytes. The width of the payload, data_width, is the number of gear bits multiplied by the number of data lanes. Upper data bytes for the last payload data must be ignored if the wordcount is not a multiple of data_width/8. The interfacing module should be responsible for extracting the correct payload bytes based on the valid output wordcount *wc_o*.

Figure 2.5 shows the block diagram of the submodule with the packet parser included.

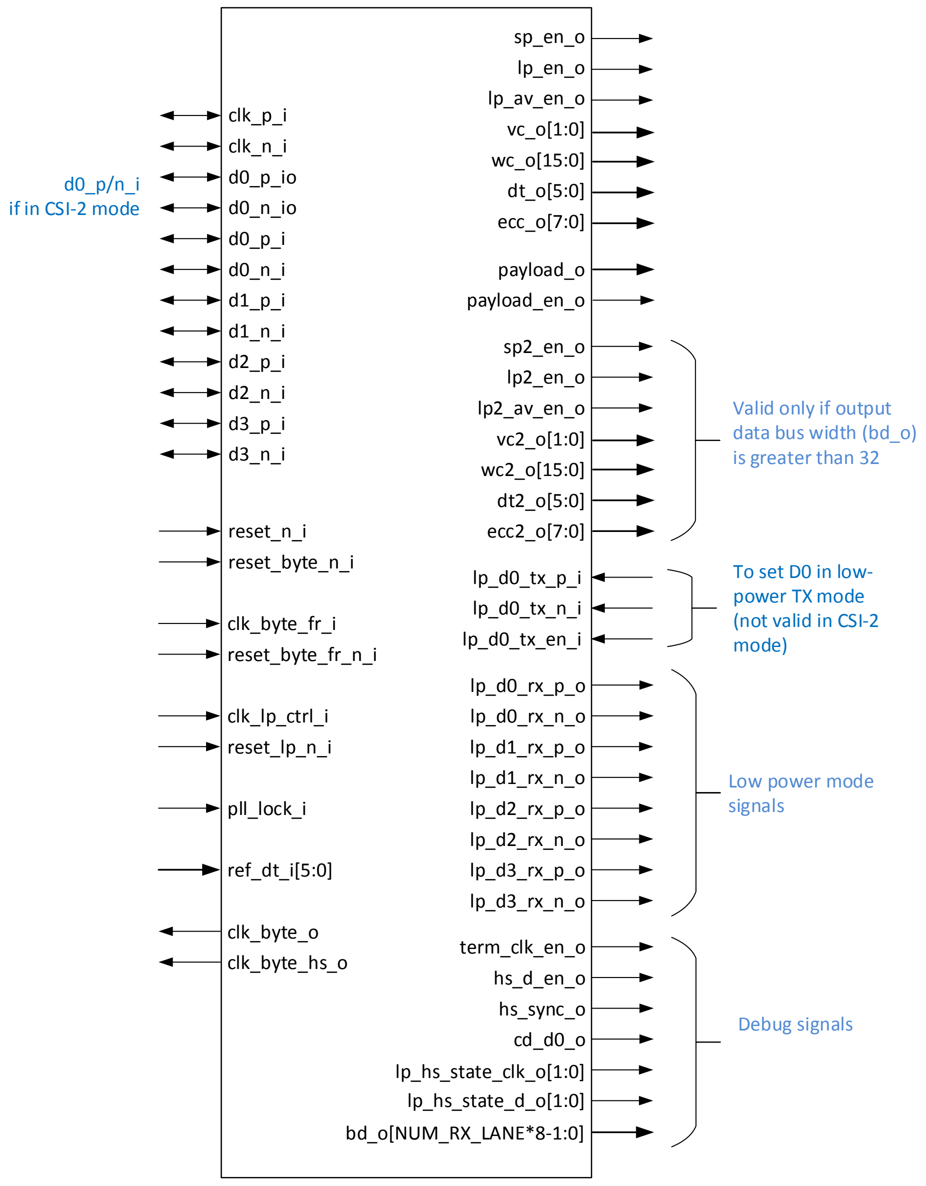


Figure 2.5. Block Diagram of MIPI Rx D-PHY IP with Packet Parser

As shown in the block diagram, there is a second set of decoded packet fields. This set is valid when the input data bus going to the packet parser is greater than 32. In this configuration, 2 packet headers may simultaneously be decoded within the same byte clock cycle. This scenario is illustrated in Figure 2.6.

The interval between the assertion of the sp_en_o or the lp_en_o and the reception of the valid input data are not fixed; this depends on the gearing and number of lanes.

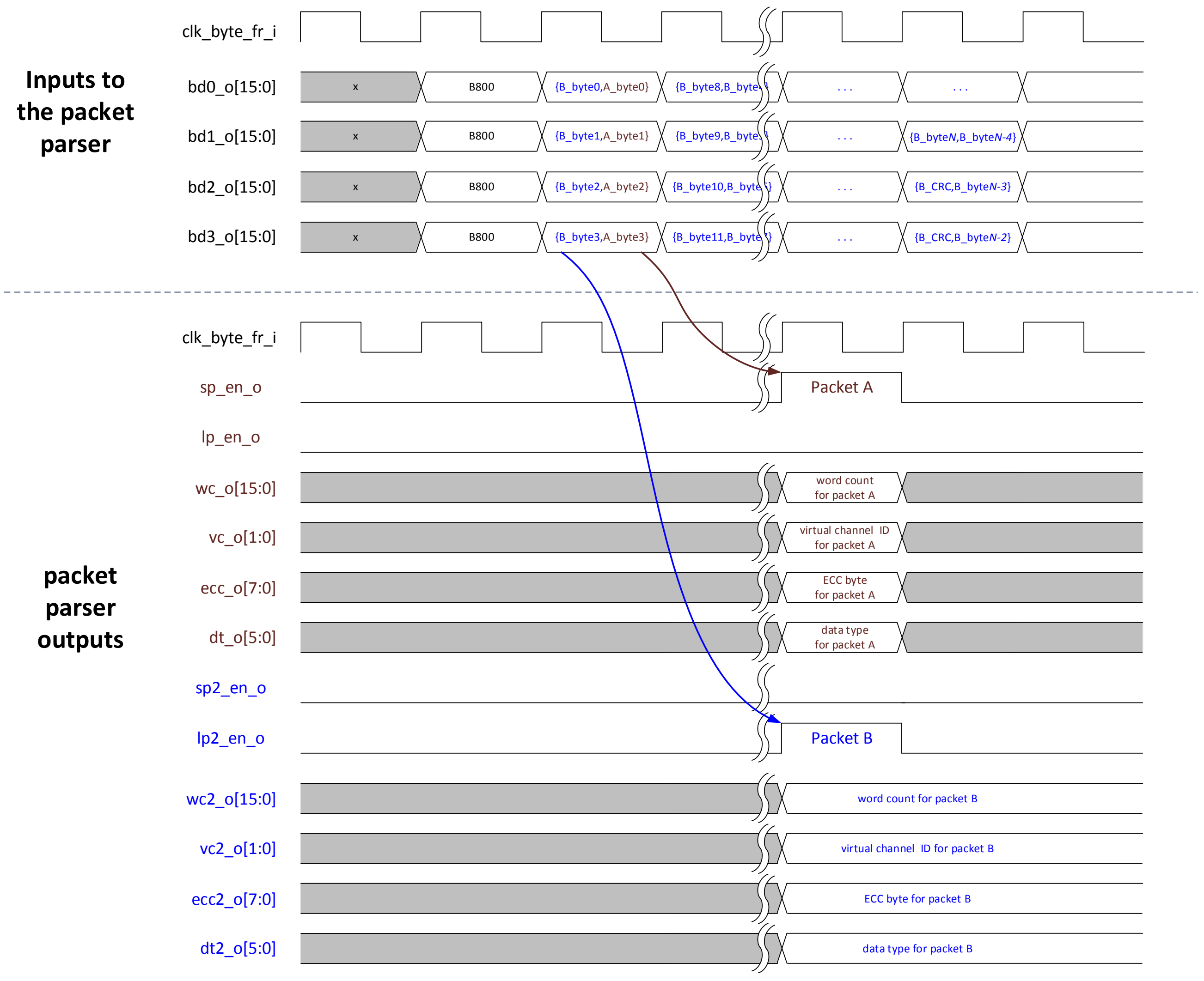


Figure 2.6. Output Timing Diagram with Valid Second Set of Packet Information

2.4. Submodule Description

Figure 2.7 shows the three major blocks of the Rx D-PHY submodule.

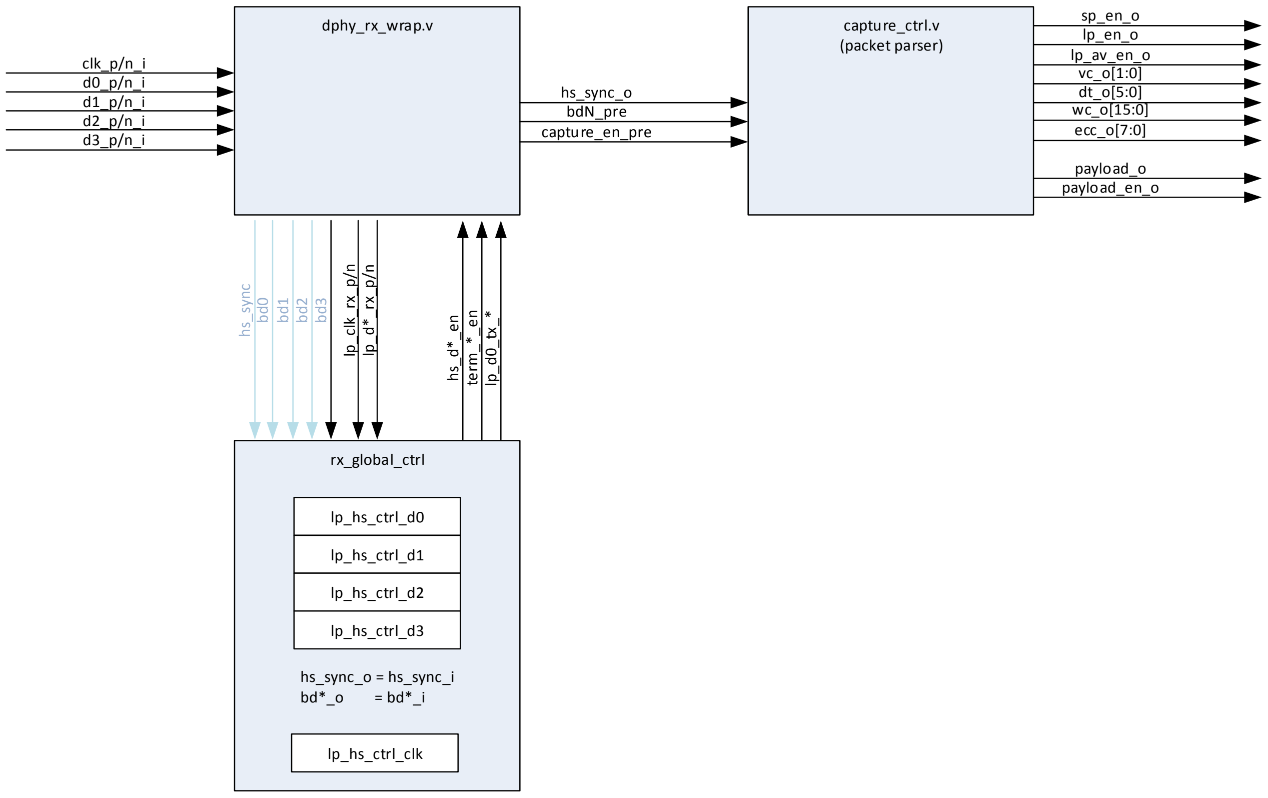


Figure 2.7 Rx D-PHY Submodule Components

2.4.1. D-PHY Common Interface Wrapper

This block instantiates and configures either the hard or the soft D-PHY IP to receive MIPI D-PHY high-speed data from all enabled data lanes. Figure 2.8 shows the differences between the soft and hard IP configurations. The interface to the external logic remains the same regardless of the implementation.

The Start-of-Transmit pattern (SoTp) is included in the output data bytes from the Soft IP D-PHY implementation. To mimic this behavior, the dphy_rx_wrap module appends the SoTp in the Hard IP implementation.

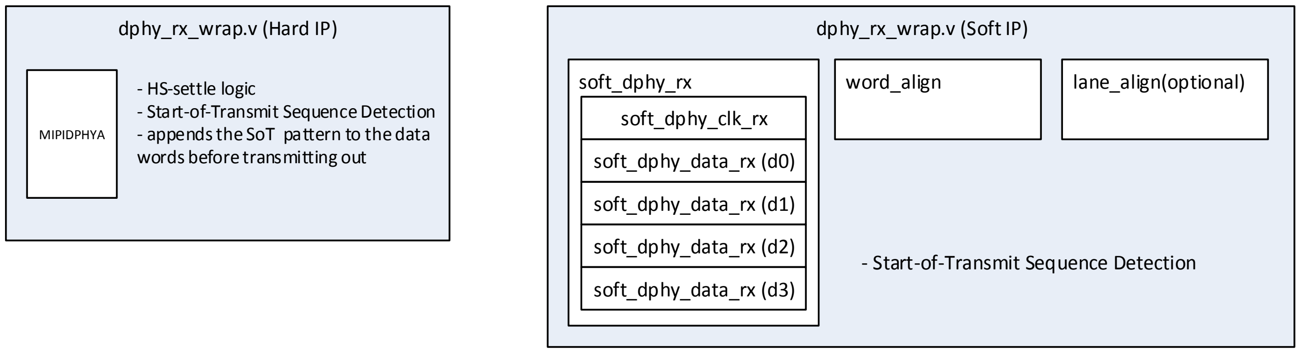


Figure 2.8 D-PHY Rx Implementation

Byte data are transferred from the stoppable byte clock domain to the continuous byte clock domain using multicycle registers. Data enable signal from this block becomes active when SoT sync pattern is successfully detected by hard D-PHY IP from all enabled data lanes, and becomes inactive when MIPI D-PHY data lanes go to Stop state (LP11).

A word aligner logic detects the SoT pattern from each lane and ensures the parallel data are word (byte) aligned. The design assumes that input data lanes are driven at the same time, and skew between data lanes are less than 1 UI. However, an optional lane aligner module may be instantiated to further ensure that the lanes are aligned with each other.

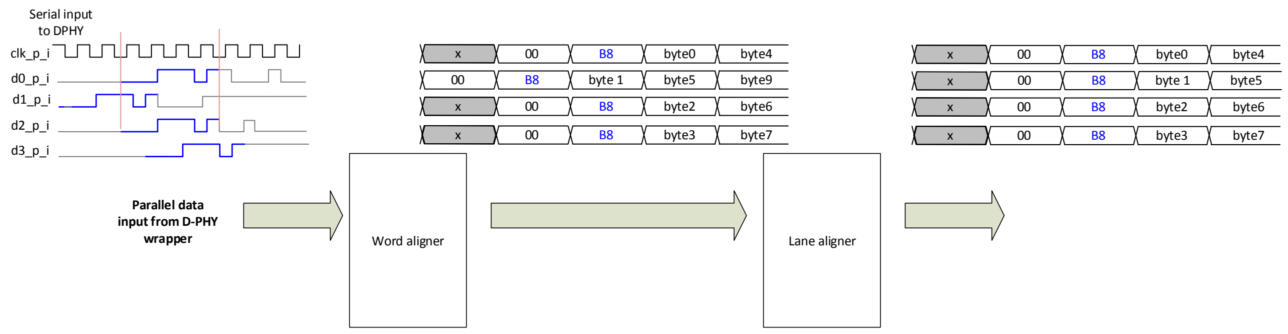


Figure 2.9 Aligner Modules

2.4.2. Rx Global Operations Controller

This block controls the high-speed termination enable of MIPI D-PHY clock and data lanes. When MIPI D-PHY clock is continuous, the HS termination enable of clock lane is tied to VCC. When MIPI D-PHY clock is non-continuous, the HS termination enable of clock lane becomes active right after proper LP to HS transition is observed. A reference clock input is required for this function. Figure 2.10 shows the required LP to HS transition on clock lane as per MIPI D-PHY Specification version 1.1.

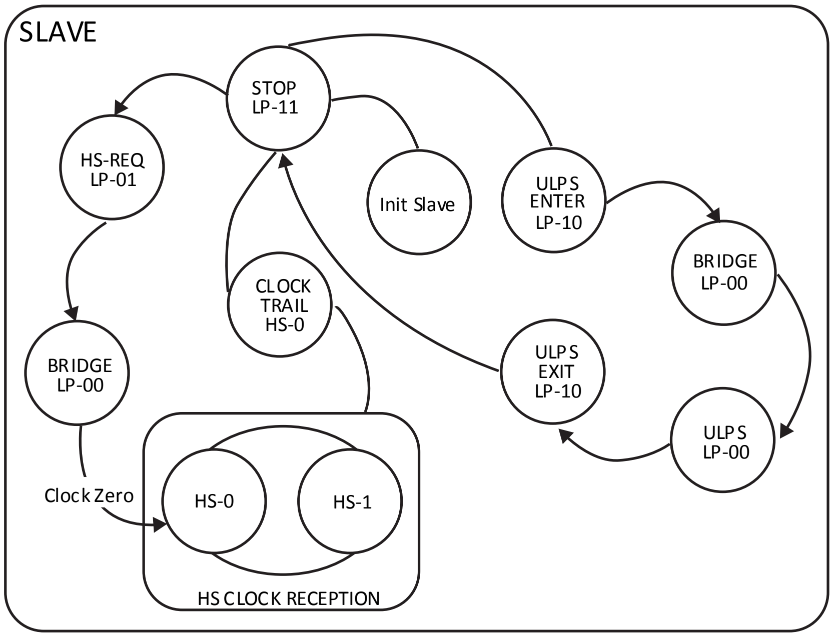


Figure 2.10. MIPI D-PHY Clock Lane Module State Diagram

Similarly, HS termination enable of data lanes becomes high after proper LP to HS transition is detected on data lane 0. A free-running byte clock is used for this function. Figure 2.11 shows the required LP to HS transition on data lanes as per MIPI D-PHY Specification version 1.1.

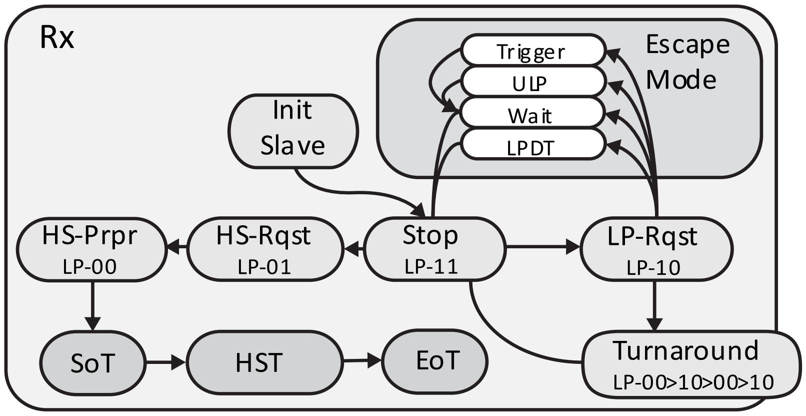


Figure 2.11. MIPI D-PHY Data Lane Module State Diagram

2.4.3. Capture Controller (Packet Parser)

This block parses the data bytes from D-PHY Common Interface Wrapper, and detects short and long packets defined by MIPI DSI or MIPI CSI-2. This block extracts video data and other control parameters from the packets.

There are no signals from the external logic to this block to control the flow of output data. The interfacing logic must provide ample buffering to ensure the continuous flow of data from this submodule is transferred correctly.

2.5. Reset and Clocking

Asynchronous active low reset input (reset_n_i) is used as a system reset. To reset logic in continuous byte clock domain, system reset passes through synchronization registers to create asynchronous reset assertion and synchronous reset deassertion. The system reset input must be asserted for at least three times the byte clock cycle.

When MIPI D-PHY clock is continuous, it is expected to be in high-speed mode at power on of the device. The HS termination enable of clock lane is tied to VCC. Continuous byte clock is generated by the D-PHY IP. We suggest to use this output byte clock as the input free-running clock clk_byte_fr_i.

When MIPI D-PHY clock is non-continuous, an external clock input is required for detecting the LP to HS transition of clock (clk_lp_ctrl_i). This could be the same input free-running byte clock clk_byte_fr_i, as long as the clk_lp_ctrl_i clock period is half the T_{LPX} (transmitted length of any low-power state period) .

Table 2.2 lists the frequency calculations. DCK refers to MIPI D-PHY clock frequency.

Table 2.2. Clock Frequency Calculations

Clock	Formula
D-PHY clock	DCK
Rx line rate	DCK * 2
Byte clocks (clk_byte_o and clk_byte_fr)	DCK / (Rx gear / 2)
LP Control (clk_lp_ctrl_i)	2*(Rx D-PHY low-power mode frequency)

2.5.1. Clock Domains

Figure 2.12 shows the three clock domains within the D-PHY Rx Submodule.

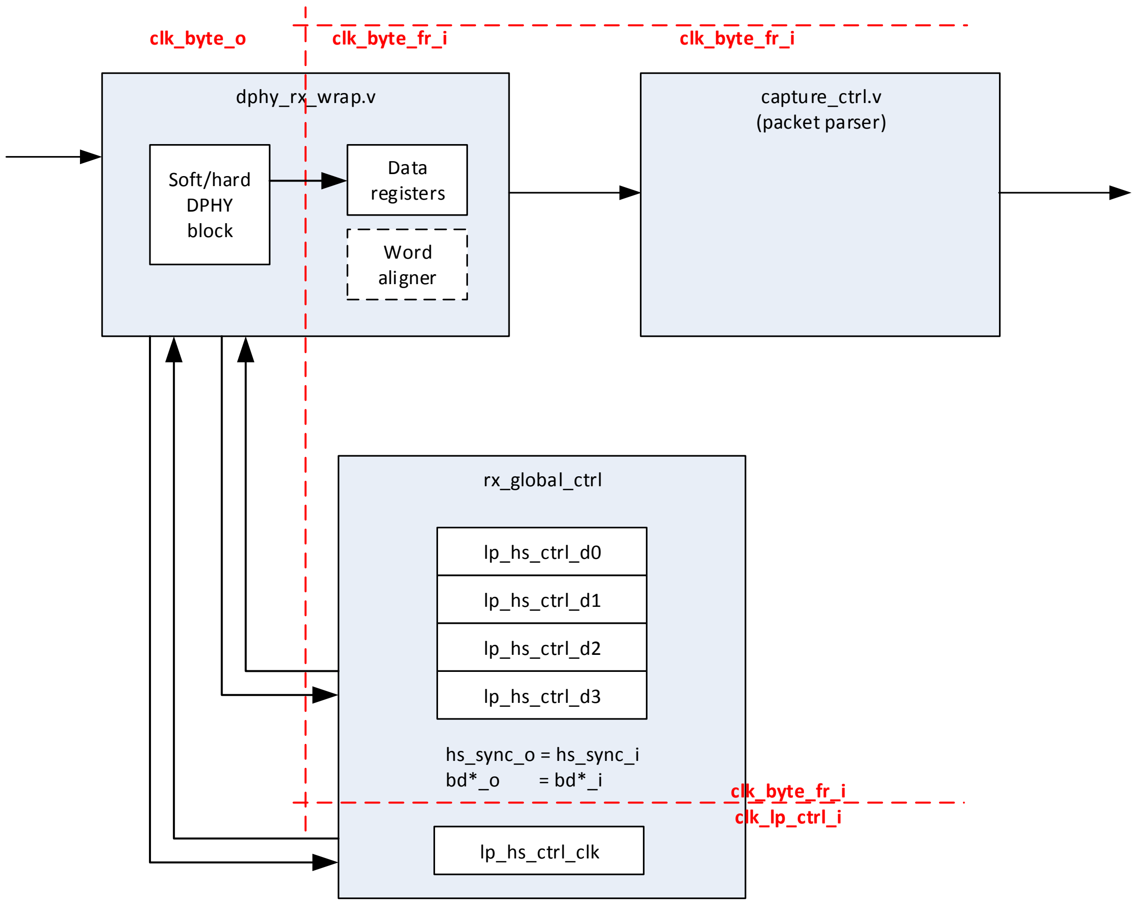


Figure 2.12. Clock Domain Diagram

The clk_lp_ctrl_i clock must be continuous and be twice the LP mode clock frequency to ensure correct D-PHY clock lane state detection.

The output signal clk_byte_o is non-continuous and is only active when the Rx D-PHY data lanes are in high-speed mode. The 4-pipeline multicycle registers are used to synchronize the parallel data from this clock to the free-running clk_byte_fr domain. There is no flow control in this data path, therefore, the clk_byte_o must be the exact same frequency of the clk_byte_fr.

3. Compiler Directives and Parameter Settings

This section lists the compiler directives and parameters used to configure the D-PHY Receiver Submodule.

3.1. RTL Compiler Directives

Table 3.1. MIPI D-PHY Rx Submodule IP RTL Compiler Directives

Compiler Directive	Value	Description	Operation
DPHY_RX	MIXEL or LATTICE	Implementation type. Use MIXEL if using Hard IP, LATTICE if using Soft IP logic implementation	`define DPHY_RX_<val>
PARSER	ON or OFF	Includes the packet parser in the design	`define PARSER_<val>
RX_CLK_MODE	HS_LP or HS_ONLY	Clock mode	`define RX_CLK_MODE_<val>
NUM_RX_LANE	1, 2, 3 or 4	Number of D-PHY data lanes	`define NUM_RX_LANE _<val>
RX_GEAR	8 or 16	Number of bits per lane in one byte clock cycle	`define RX_GEAR_<val>
RX_TYPE	DSI or CSI2	Protocol type	`define RX_TYPE_<val>
LANE_ALIGN	—	Includes the lane aligner module in the design. This is only valid for Rx Soft IP implementation	`define LANE_ALIGN
BYTECLK_MHZ	20 - 150	Integer byte clock value (whole number, rounded down)	`define BYTECLK_MHZ=<val>
LANE_FIFO _DEPTH	8, 16 or 32	Lane aligner FIFO depth	`define LANE_FIFO_DEPTH=<val>
FIFO_TYPE0 FIFO_TYPE1 FIFO_TYPE2 FIFO_TYPE3	LUT	Uses LUTs to implement the lane aligner FIFO. The FIFO defaults to EBR if it is not defined. This reduces the total EBR usage at the expense of higher LUT count.	`define FIFO_TYPE0_LUT `define FIFO_TYPE1_LUT `define FIFO_TYPE2_LUT `define FIFO_TYPE3_LUT

3.2. RTL Global Parameters

Most of the compiler directives control the value of a design parameter. These parameters are listed in Table 3.2.

Table 3.2. MIPI D-PHY Receiver Submodule IP RTL Parameter Settings

Compiler Directive	Value	Description	Default values
RX_TYPE	"DSI" or "CSI2"	Protocol type	parameter RX_TYPE = "DSI"
PARSER	"ON" or "OFF"	Includes the packet parser in the design	parameter PARSER="ON"
RX_CLK_MODE	"HS_LP" or "HS_ONLY"	Clock mode	parameter RX_CLK_MODE="HS_LP"
NUM_RX_LANE	1, 2, 3 or 4	Number of D-PHY data lanes	parameter NUM_RX_LANE=4
RX_GEAR	8 or 16	Number of bits per lane in one byte clock cycle	parameter RX_GEAR=16
DPHY_RX_IP	"MIXEL" or "LATTICE"	Implementation type. Use MIXEL if using Hard IP, LATTICE if using Soft IP logic implementation	parameter DPHY_RX_IP="MIXEL"
LANE_ALIGN	"ON" or "OFF"	Enables or disables the lane aligner module when DPHY_RX_IP is "LATTICE"	parameter LANE_ALIGN ="OFF"

Table 3.2. MIPI D-PHY Receiver Submodule IP RTL Parameter Settings *(Continued)*

Compiler Directive	Value	Description	Default values
BYTECLK_MHZ	20 - 150	Target byte clock frequency	parameter BYTECLK_MHZ=111
FIFO_DEPTH	8, 16 or 32	Lane aligner FIFO depth	parameter FIFO_DEPTH=16
FIFO_TYPE0 FIFO_TYPE1 FIFO_TYPE2 FIFO_TYPE3	"EBR" or "LUT"	Lane aligner FIFO implementation per lane	parameter FIFO_TYPE0="EBR"

3.3. GUI Parameters

Table 3.3 lists the GUI parameters used to generate MIPI D-PHY Receiver Submodule IP.

Table 3.3. MIPI D-PHY Receiver Submodule IP GUI Parameter Settings

Parameter	Attribute	Options	Description
Rx Interface	User-Input	DSI or CSI-2	Receive interface
Packet Parser	User-Input	ON or OFF	Includes or excludes the packet parser in the design.
Number of Rx lanes	User-Input	1, 2, 3 or 4	Number of MIPI D-PHY data lanes. 3-lane configuration is only valid when packet parser is disabled.
Rx gearing	User-Input	8, 16	Rx gear is automatically computed based on selected configuration and D-PHY data rate.
Rx D-PHY IP Implementation	User-Input	Hard D-PHY or Soft D-PHY	MIPI D-PHY Implementation. Soft D-PHY uses programmable MIPI I/Os while Hard D-PHY implementation uses hardened logic blocks.
Rx Line Rate	User-Input	160 Mb/s - 1500 Mb/s	Bit rate per Rx data lane in high-speed mode. This is twice the D-PHY clock lane frequency. Data rate is only up to 1200 Mb/s for Soft D-PHY implementation.
D-PHY Clock Mode	User-Input	Continuous or Non-continuous	Rx D-PHY is in Continuous clock mode when the D-PHY clock lanes are in high-speed mode. The termination is always enabled in this mode. The clock mode is non-continuous if the clock lane goes to low-power mode in between high-speed data transmission.
Byte Clock Frequency	Read-only	—	Byte clock frequency
Reference Clock Frequency	Read-only	—	Reference clock frequency
Lane Aligner module	User-Input	Enabled or Disabled	Enables or disables the lane aligner module when implementing the D-PHY as Hard IP
Lane Aligner FIFO Depth	User-Input	8, 16 or 32	Sets the lane aligner FIFO depth when the lane aligner is enabled. The width of the FIFO per lane is the same as the gearing.
FIFO_TYPE0/1/2/3	User-Input	EBR or LUT	Lane aligner FIFO implementation per lane. This has no change in functionality. Modify the implementation type depending on the resource requirement of the overall design.

4. IP Generation and Evaluation

This section provides information on how to generate Lattice CSI-2/DSI D-PHY Receiver Submodule IP using the Diamond Clarity Designer, and how to run simulation, synthesis and hardware evaluation.

4.1. Licensing the IP

The CSI-2/DSI D-PHY Receiver Submodule IP license is available free of charge, but an IP-specific license is required to enable full, unrestricted use of the CSI-2/DSI D-PHY Receiver Submodule IP in a complete, top-level design.

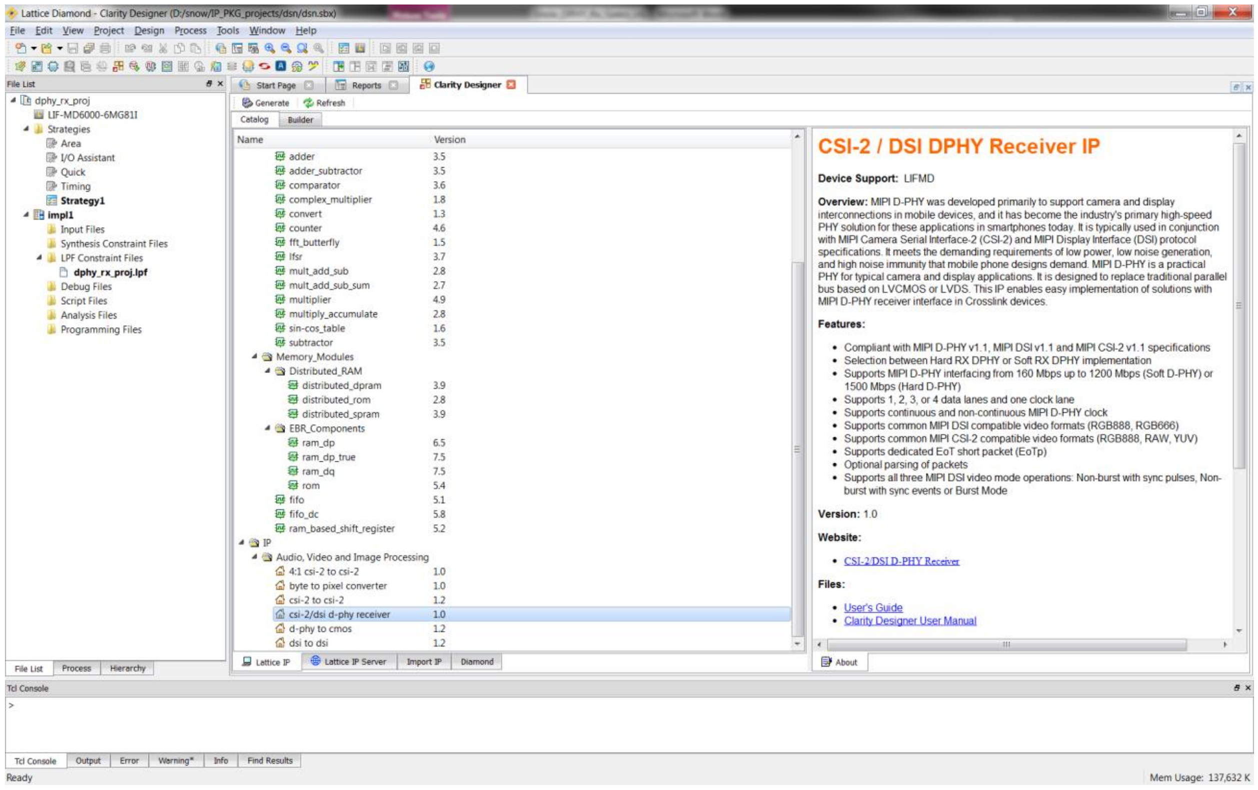
Please request your free license by sending an email to lic_admn@latticesemi.com attaching your existing Lattice Diamond license or providing your MacID along with the IP details.

You may download or generate the CSI-2/DSI D-PHY Receiver Submodule IP and fully evaluate it through functional simulation and implementation (synthesis, map, place and route) without the IP license. The CSI-2/DSI D-PHY Receiver Submodule IP also supports Lattice's IP hardware evaluation capability, see the Hardware Evaluation section on page 26 for further details.

However, IP license is required to enable timing simulation, to open the design in Diamond EPIC tool, or to generate bitstreams that do not include the hardware evaluation timeout limitation.

4.2. Getting Started

The CSI-2/DSI D-PHY Receiver Submodule IP is available for download from the Lattice IP Server using the Clarity Designer tool. The IP files are automatically installed using ispUPDATE technology in any customer-specified directory. After the IP has been installed, the IP is available in the Clarity Design GUI as shown in Figure 4.1.



dphy_rx_proj
- LIF-MD6000-6MG81I
- Strategies
  - Area
  - I/O Assistant
  - Quick
  - Timing
  - **Strategy1**
- **impl1**
  - Input Files
  - Synthesis Constraint Files
  - LPF Constraint Files
    - **dphy_rx_proj.lpf**
  - Debug Files
  - Script Files
  - Analysis Files
  - Programming Files

Name	Version
adder	3.5
adder_subtractor	3.5
comparator	3.6
complex_multiplier	1.8
convert	1.3
counter	4.6
fft_butterfly	1.5
lfsr	3.7
mult_add_sub	2.8
mult_add_sub_sum	2.7
multiplier	4.9
multiply_accumulate	2.8
sin-cos_table	1.6
subtractor	3.5
Memory_Modules	
Distributed_RAM	
distributed_dpram	3.9
distributed_rom	2.8
distributed_spram	3.9
EBR_Components	
ram_dp	6.5
ram_dp_true	7.5
ram_dq	7.5
rom	5.4
fifo	5.1
fifo_dc	5.8
ram_based_shift_register	5.2
IP	
Audio, Video and Image Processing	
4:1 csi-2 to csi-2	1.0
byte to pixel converter	1.0
csi-2 to csi-2	1.2
csi-2/dsi d-phy receiver	1.0
d-phy to cmos	1.2
dsi to dsi	1.2

CSI-2 / DSI DPHY Receiver IP

Device Support: LIFMD

Overview: 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 today. It is typically used in conjunction with MIPI Camera Serial Interface-2 (CSI-2) and MIPI Display 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. This IP enables easy implementation of solutions with MIPI D-PHY receiver interface in Crosslink devices.

Features:

- Compliant with MIPI D-PHY v1.1, MIPI DSI v1.1 and MIPI CSI-2 v1.1 specifications
- Selection between Hard RX DPHY or Soft RX DPHY implementation
- Supports MIPI D-PHY interfacing from 160 Mbps up to 1200 Mbps (Soft D-PHY) or 1500 Mbps (Hard D-PHY)
- Supports 1, 2, 3, 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)
- Supports common MIPI CSI-2 compatible video formats (RGB888, RAW, YUV)
- Supports dedicated EoT short packet (EoTp)
- Optional parsing of packets
- Supports all three MIPI DSI video mode operations: Non-burst with sync pulses, Non-burst with sync events or Burst Mode

Version: 1.0

Website:

- CSI-2/DSI D-PHY Receiver

Files:

- User's Guide
- Clarity Designer User Manual

Figure 4.1. Clarity Designer Window

4.3. Generating IP in Clarity Designer

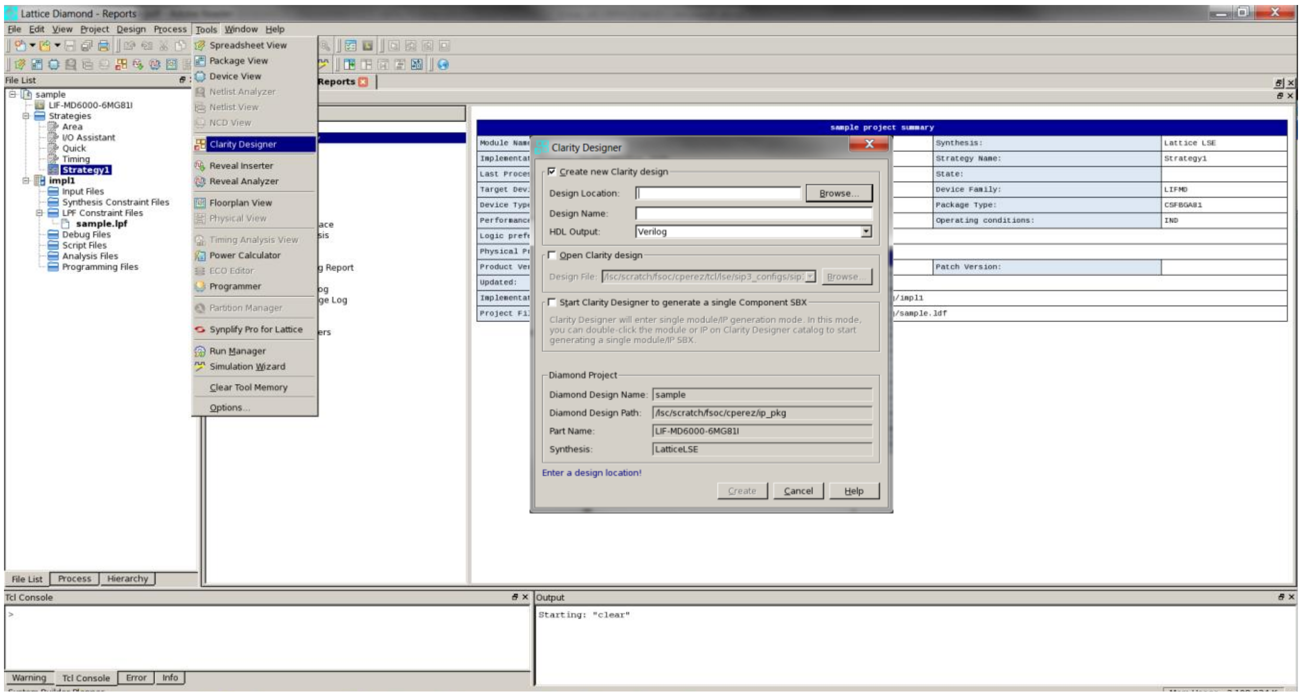
The Clarity Designer tool is used to customize modules and IPs and place them into the device's architecture. The following describes the procedure for generating CSI-2/DSI D-PHY Receiver Submodule IP in Clarity Designer.

Clarity Designer is started from the Diamond design environment.

To start Clarity Designer:

1. Create a new Diamond project for CrossLink family devices.
2. From the Diamond main window, choose **Tools** > **Clarity Designer**, or click in Diamond toolbox. The Clarity Designer project dialog box is displayed.
3. Select and fill out the following items as shown in Figure 4.2:
   - **Create new Clarity design** - Click this to create a new Clarity Design project directory in which the CSI-2/DSI D-PHY Receiver Submodule IP will be generated.
   - **Design Location** - Clarity Design project directory path.
   - **Design Name** - Clarity Design project name.
   - **HDL Output** - Hardware Description Language Output Format (Verilog).

   The Clarity Designer project dialog box also allows you to open an existing Clarity Designer project by selecting the following:
   - **Open Clarity design** - Open an existing Clarity Design project.
   - **Design File** - Name of existing Clarity Design project file with .sbx extension.
4. Click the **Create** button. A new Clarity Designer project is created.



Clarity Designer

[✓] Create new Clarity design
Design Location:
Design Name:
HDL Output: Verilog

[ ] Open Clarity design
Design File: /lsc/scratch/fsoc/cperez/tcl/lse/sip3_configs/sip

[ ] Start Clarity Designer to generate a single Component SBX
Clarity Designer will enter single module/IP generation mode. In this mode, you can double-click the module or IP on Clarity Designer catalog to start generating a single module/IP SBX.

Diamond Project
Diamond Design Name: sample
Diamond Design Path: /lsc/scratch/fsoc/cperez/ip_pkg
Part Name: LIF-MD6000-6MG81I
Synthesis: LatticeLSE

Enter a design location!

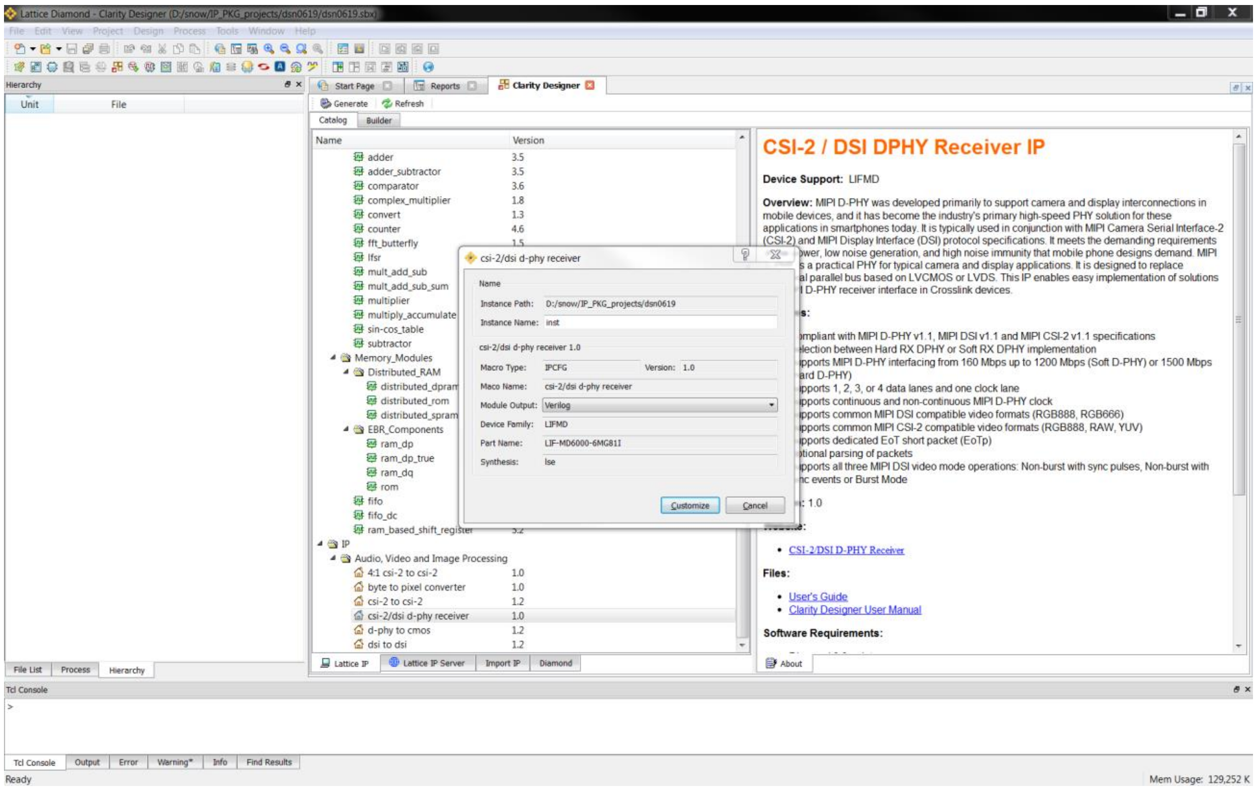
Synthesis:	Lattice LSE		
Strategy Name:	Strategy1		
State:			
Device Family:	LIFMD		
Package Type:	CSFBGA81		
Operating conditions:	IND		
Patch Version:			

```
Starting: "clear"
```

Figure 4.2. Starting Clarity Designer from Diamond Design Environment

To configure the CSI-2/DSI D-PHY Receiver Submodule IP in Clarity Designer:

1. Double-click **csi-2/dsi d-phy receiver** in the IP list of the Catalog view. The **csi-2/dsi d-phy receiver** dialog box is displayed as shown in Figure 4.3.

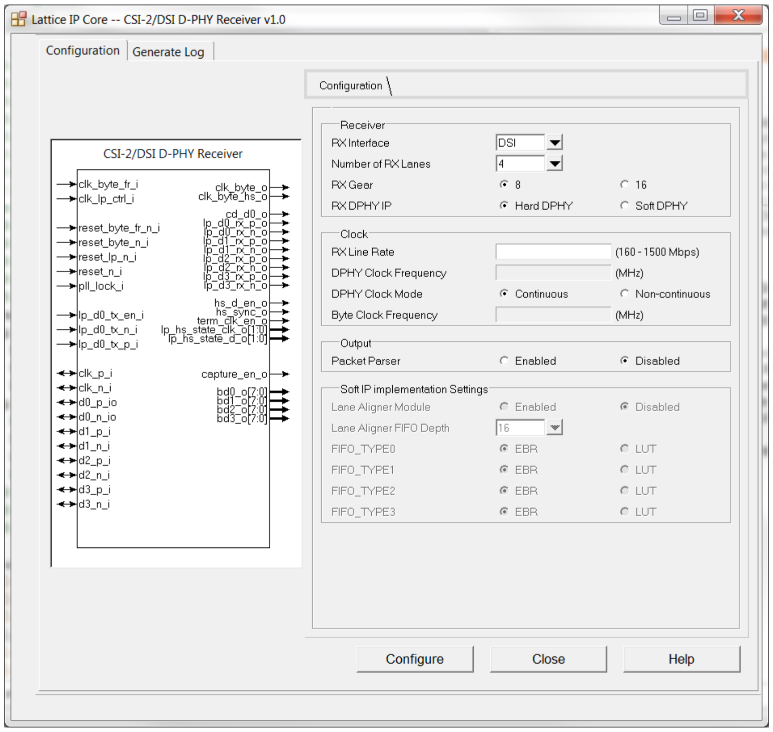

csi-2/dsi d-phy receiver

Name
Instance Path: D:/snow/IP_PKG_projects/dsn0619
Instance Name: inst

csi-2/dsi d-phy receiver 1.0
Macro Type: IPCFG Version: 1.0
Maco Name: csi-2/dsi d-phy receiver
Module Output: Verilog
Device Family: LIFMD
Part Name: LIF-MD6000-6MG81I
Synthesis: lse

Figure 4.3. Configuring CSI-2/DSI D-PHY Receiver Submodule IP in Clarity Designer

2. Enter the Instance Name.
3. Click the **Customize** button. An IP configuration interface is displayed as shown in Figure 4.4. From this dialog box, you can select the IP parameter options specific to your application.


Lattice IP Core -- CSI-2/DSI D-PHY Receiver v1.0

Receiver
RX Interface: DSI
Number of RX Lanes: 4
RX Gear: [✓ 8] 16
RX DPHY IP: [✓ Hard DPHY] Soft DPHY

Clock
RX Line Rate: (160 - 1500 Mbps)
DPHY Clock Frequency: (MHz)
DPHY Clock Mode: [✓ Continuous] Non-continuous
Byte Clock Frequency: (MHz)

Output
Packet Parser: Enabled [✓ Disabled]

Soft IP implementation Settings
Lane Aligner Module: Enabled [✓ Disabled]
Lane Aligner FIFO Depth: 16
FIFO_TYPE0: [✓ EBR] LUT
FIFO_TYPE1: [✓ EBR] LUT
FIFO_TYPE2: [✓ EBR] LUT
FIFO_TYPE3: [✓ EBR] LUT

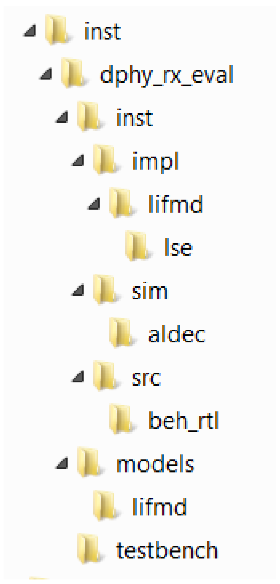
Figure 4.4. Configuration Tab in IP GUI

4. Select the required parameters, and click the **Configure** button.
5. Click **Close**.
6. Click Generate in the toolbox. Clarity Designer generates all the IPs and modules, and creates a top module to wrap them.

For detailed instructions on how to use the Clarity Designer, please refer to the Lattice Diamond software user guide.

4.4. Generated IP Directory Structure and Files

Figure 4.5 shows the directory structure of the generated IP files.



```
inst
  dphy_rx_eval
    inst
      impl
        lifmd
          lse
      sim
        aldec
      src
        beh_rtl
    models
      lifmd
    testbench
```

Figure 4.5. CSI-2/DSI D-PHY Receiver Submodule IP Directory Structure

The design flow for the IP created with Clarity Designer uses post-synthesized modules (NGO) of the IP core modules for synthesis and uses protected models for simulation. The post-synthesized modules are customized when you configure the IP and are created automatically when the IP is generated. The protected models are common to all configurations.

Table 4.1 provides a list of key files and directories created by Clarity Designer with details on how they are used.

Table 4.1. Files Generated by Clarity Designer

File	Description
<instance_name>.v	Verilog top-level module of CSI-2/DSI D-PHY Receiver Submodule IP used for both synthesis and simulation.
<instance_name>_*.v	Verilog submodules for simulation. Files that do not have equivalent black box modules are also used for synthesis.
<instance_name>_*_beh.v	Protected Verilog models for simulation.
<instance_name>_*_bb.v	Verilog black box modules for synthesis.
<instance_name>_*.ngo	GUI configured and synthesized modules for synthesis.
<instance_name>_params.v	Verilog parameters file which contains required compiler directives to successfully configure IP during synthesis and simulation.
<instance_name>.lpc	Lattice Parameters Configuration file. This file records all the IP configuration options set through Clarity Designer. It is used by the IP generation script to generate configuration-specific IP. It is also used to reload parameter settings in the IP GUI in Clarity Designer when it is being reconfigured.
<instance_name>_inst.v/vhd	Template for instantiating the generated soft IP top-level in another user-created top module.

All IP files are generated inside `\<project_dir>` directory (test folder in Figure 4.5). The `\<project_dir>` is `<design_location>\<design_name>\<instance_name>`, see the Generating IP in Clarity Designer section on page 19. A separate `\<project_dir>` is created each time CSI-2/DSI D-PHY Receiver Submodule IP is created with a different IP instance name.

The `\dphy_rx_eval` and subdirectories provide files supporting push-button IP evaluation through functional simulations, design implementation (synthesis, map) and hardware `evaluation`. Inside the `\dphy_rx_eval` is the `\<instance_name>` folder (inst folder in Figure 4.5) which contains protected behavioral files in `\<instance_name>\src\beh_rtl` and a pre-built Diamond project in `\<instance_name>\impl\lifmd\<synthesis_tool>`. The `<instance_name>` is the IP instance name specified by user in Clarity Designer. The simulation part of user evaluation provides testbench and test cases supporting RTL simulation for Active-HDL simulator under \testbench folder. Separate directories located at `\<project_dir>\dphy_rx_eval\<instance_name>\sim\aldec` are provided and contain specific pre-built simulation script files. See the Running Functional Simulation section below for details.

4.5. Running Functional Simulation

The generated IP package includes the behavioral models (`<instance_name>_*_beh.v`) provided in `\<project_dir>\dphy_rx_eval\<instance_name>\src\beh_rtl` for functional simulation. The testbench files are provided in `\<project_dir>\dphy_rx_eval\testbench`.

To run the evaluation simulation on Active-HDL (Windows only) follow these steps:

1. Create new project using Lattice Diamond for Windows.
2. Open **Active-HDL Lattice Edition** GUI tool.
4. To customize the testbench parameters, edit the file **tb_setup_params.v** inside the *<project_dir>\<instance_name>\dphy_rx_eval\testbench* folder. See Table 4.2 for the list of valid testbench compiler directives.
5. Click **Tools**, then click **Execute macro**.
6. Select the ***<instance_name>*_run.do** file inside the *<project_dir>\<instance_name>\dphy_rx_eval\<instance_name>\sim\aldec* folder.
7. Wait for the simulation to finish.
8. Input and output log files are saved in the **sim** directory.

Testbench parameters and directives can be modified by setting the define in the vlog command in the *.do file. Table 4.2 lists the testbench directives common for DSI and CSI-2 Rx type.

Table 4.2. Testbench Directives Common for DSI and CSI-2

Directive	Description
NUM_FRAMES	Set the number of video frames
NUM_LINES	Set the number of lines per frame
VIRTUAL_CHANNEL	Set the virtual channel number
DPHY_DEBUG_ON	Enable or disable debug messages 0 – Debug messages disabled 1 – Debug messages enabled
DPHY_CLK	Set the D-PHY clock period (in ps)
REF_CLK	Override the default clk_byte_fr_i clock period for HS_LP clock mode.
FRAME_LPM_DELAY	Set the low-power mode delay between frames (in ps).
INIT_DELAY	Set the delay before data is transmitted to the design.

The testbench has default settings for D-PHY timing parameters. Refer to Table 14 of MIPI D-PHY Specification version 1.1 for information regarding D-PHY timing requirements. To modify the D-PHY timing parameters, set the following testbench directives:

Table 4.3. Testbench Directives for D-PHY Timing Parameters

Directive	Description
DPHY_LPX	Set T-LPX (in ps)
DPHY_CLK_PREPARE	Set T-CLK-PREPARE (in ps)
DPHY_CLK_ZERO	Set T-CLK-ZERO (in ps)
DPHY_CLK_PRE	Set T-CLK-PRE (in ps)
DPHY_CLK_POST	Set T-CLK-POST (in ps)
DPHY_CLK_TRAIL	Set T-CLK-TRAIL (in ps)
DPHY_HS_PREPARE	Set T-HS-PREPARE (in ps)
DPHY_HS_ZERO	Set T-HS-ZERO (in ps)
DPHY_HS_TRAIL	Set T-HS-TRAIL (in ps)
DPHY_INIT	Set the T-INIT timing (in ps)

Table 4.4 is a list of testbench directives for DSI Rx type.

Table 4.4. Testbench Directives for DSI Rx Type

File	Description
LP_BLANKING	Drive low-power blanking. If this is not defined, the testbench drives HS data as blanking
NON_BURST_SYNC_PULSE	DSI video modes
NON_BURST_SYNC_EVENTS	
BURST_MODE	
RGB888	DSI data types
RGB666	
RGB666_LOOSE	
DSI_VACT_PAYLOAD	Number of bytes of active pixels per line
DSI_HSA_PAYLOAD	Number of bytes of Horizontal Sync Active Payload (used for Non-burst sync pulse)
DSI_BLLP_PAYLOAD	Number of bytes of BLLP Payload (used for HS data blanking)
DSI_HBP_PAYLOAD	Number of bytes of Horizontal Back Porch Payload (used for HS data blanking, and in LP blanking for Non-burst sync pulse mode)
DSI_HFP_PAYLOAD	Number of bytes of Horizontal Front Porch Payload (used for HS data blanking, and in LP blanking for Non-burst sync pulse mode)
DSI_VSA_LINES	Number of Vertical Sync Active Lines
DSI_VBP_LINES	Number of Vertical Back Porch Lines
DSI_VFP_LINES	Number of Vertical Front Porch Lines
DSI_EOTP_ENABLE	Enable/disable transmission of EoTP packet 0 – EoTP packet is disabled 1 – EoTP packet is enabled
DSI_LPS_BLLP_DURATION	Set the duration (in ps) for BLLP low-power state (used for LP blanking)
DSI_LPS_HBP_DURATION	Set the duration (in ps) for Horizontal Back Porch low-power state (used for LP blanking in Non-burst sync events and Burst mode)
DSI_LPS_HFP_DURATION	Set the duration (in ps) for Horizontal Front Porch low-power state (used for LP blanking in Non-burst sync events and Burst mode)

Table 4.5 is a list of testbench directives for CSI-2 Rx type.

Table 4.5. Testbench Directives for CSI-2 Rx Type

File	Description
CSI2_LPS_GAP	Set low-power state delay between HS transactions (in ps)
CSI2_NUM_PIXELS	Set the number of pixels per line
CSI2_LS_LE_EN	Enable/disable D-PHY model transmission of line start and line end packets 0 – No Line start and Line end packets 1 – Line start and Line end packets enable
RGB888 RAW8 RAW10 RAW12 YUV420_10 YUV420_10_CSPS YUV420_8 YUV420_8_CSPS LEGACY_YUV420_8 YUV422_10 YUV422_8	CSI-2 data types

4.6. Simulation Strategies

This section describes the simulation environment which demonstrates basic MIPI D-PHY Rx functionality. Figure 4.6 shows the block diagram of simulation environment.

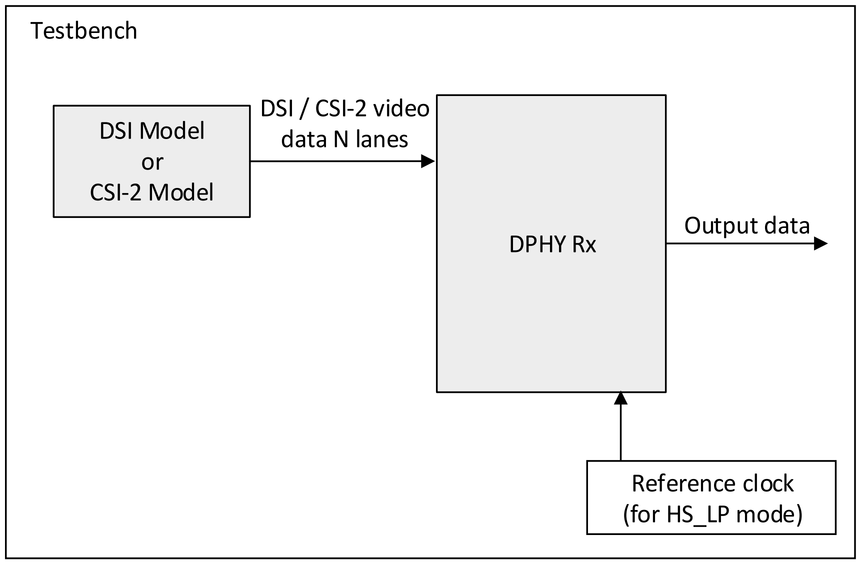

Testbench
DSI Model or CSI-2 Model
DSI / CSI-2 video data N lanes
DPHY Rx
Output data
Reference clock (for HS_LP mode)

Figure 4.6. Simulation Environment Block Diagram

4.7. Simulation Environment

The simulation environment is made up of the DSI model instance if Rx type is DSI, or made up of CSI-2 model if Rx type is CSI-2. The instantiated model is connected to the D-PHY Rx IP core instance in the testbench. The DSI model or CSI-2 model is configured based from the D-PHY Rx IP core configurations and testbench configurations. Please refer to testbench `tb_setup_params.v` file for details on how to set testbench parameters. The DSI or CSI-2 model transmits the video data to the CSI-2/DSI D-PHY Receiver Submodule IP core after reset. The testbench also transmits reference clock to the CSI-2/DSI D-PHY Receiver Submodule IP core if clock mode is non-continuous (HS_LP). If clock mode is continuous (HS_ONLY), the clk_byte_hs_o is connected to clk_byte_fr_i.

The video data transmitted by the DSI model can viewed in the waveform, see Figure 4.7.

- tb.dsi_ch0.data0 – refers to the data bytes transmitted in D-PHY data lane 0
- tb.dsi_ch0.data1 – refers to the data bytes transmitted in D-PHY data lane 1
- tb.dsi_ch0.data2 – refers to the data bytes transmitted in D-PHY data lane 2
- tb.dsi_ch0.data3 – refers to the data bytes transmitted in D-PHY data lane 3

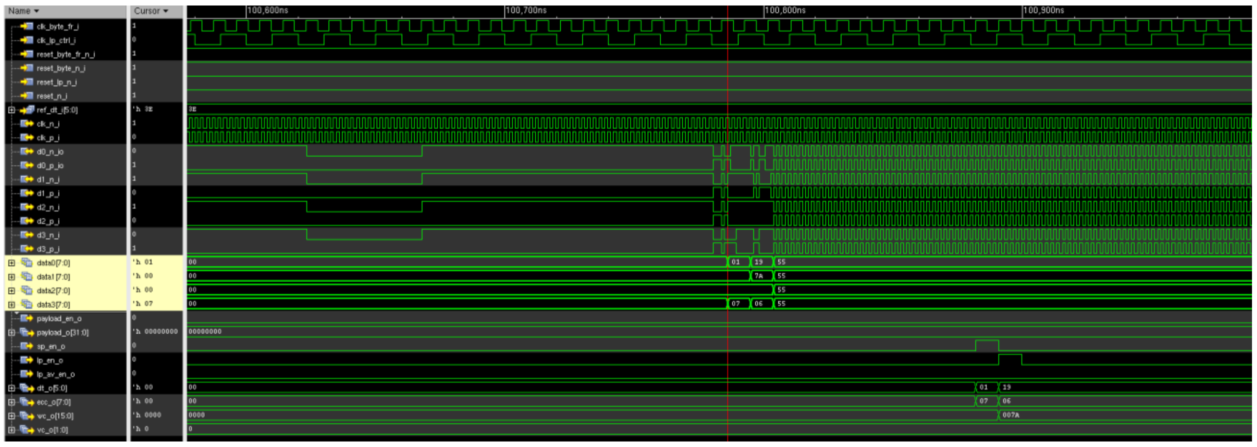


Figure 4.7. DSI Model Video Data

The video data transmitted by the CSI-2 model can viewed in the waveform, see Figure 4.8.

- tb.csi2_ch0.data0 – refers to the data bytes transmitted in D-PHY data lane 0
- tb.csi2_ch0.data1 – refers to the data bytes transmitted in D-PHY data lane 1
- tb.csi2_ch0.data2 – refers to the data bytes transmitted in D-PHY data lane 2
- tb.csi2_ch0.data3 – refers to the data bytes transmitted in D-PHY data lane 3

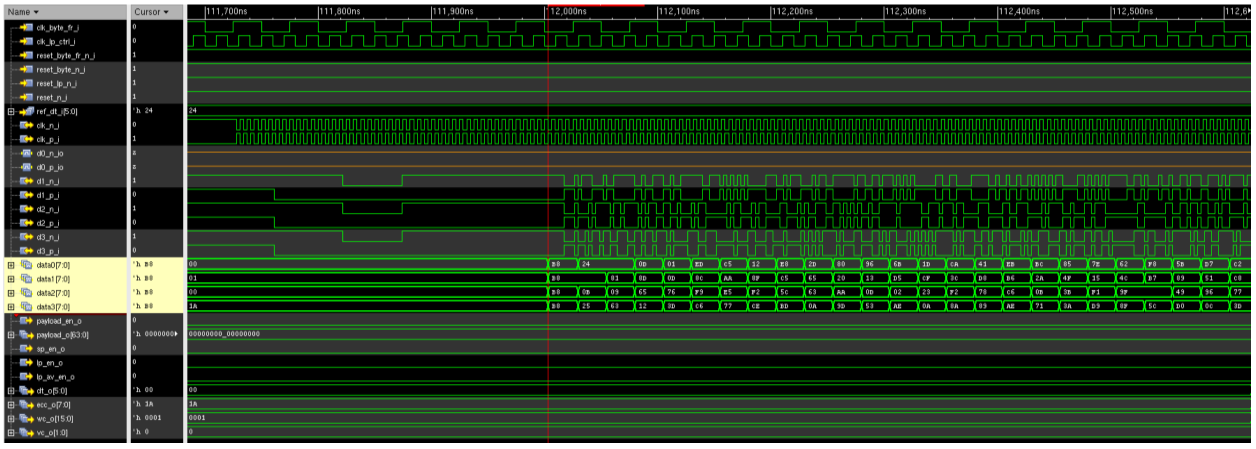


Figure 4.8. CSI-2 Model Video Data

4.8. Instantiating the IP

The core modules of CSI-2/DSI D-PHY Receiver Submodule IP are synthesized and provided in NGO format with black box Verilog source files for synthesis. A Verilog source file named `<instance_name>_dphy_rx.v` instantiates the black box of core modules. The top-level file `<instance_name>.v` instantiates `<instance_name>_dphy_rx`.

The IP instances do not need to be instantiated one by one manually. The top-level file and the other Verilog source files are provided in `\<project_dir>`. These files are refreshed each time the IP is regenerated.

A Verilog instance template `<instance_name>_inst.v` or VHDL instance template `<instance_name>_inst.vhd` is also provided as a guide if the design is to be included in another top level module.

4.9. Synthesizing and Implementing the IP

In Clarity Designer, the Clarity Designer project file (.sbx) is added to Lattice Diamond as a source file after IP is generated. All required Verilog source files for implementation are invoked automatically. The IP can be directly synthesized, mapped and placed/routed in the Diamond design environment after the IP is generated. Note that default Diamond strategy (.sty) and default Diamond preference file (.lpf) are used. When using the .sbx approach, import the recommended strategy and preferences from `\<project_dir>\dphy_rx_eval\<instance_name>\impl\lifmd\lse` or `\<project_dir>\dphy_rx_eval\<instance_name>\impl\lifmd\synplify` directories and set them as active strategy and active preference file.

Push-button implementation of this IP with either Lattice Synthesis Engine (LSE) or Synopsys Synplify Pro RTL synthesis is supported via the pre-built Diamond project file `<instance_name>_top.ldf` located in `\<project_dir>\dphy_rx_eval\<instance_name>\impl\lifmd\lse` or `\<project_dir>\dphy_rx_eval\<instance_name>\impl\lifmd\synplify` directories.

To use the pre-built Diamond project file:

1. Choose **File > Open > Project**.
2. In the **Open Project** dialog box browse to `\<project_dir>\dphy_rx_eval\<instance_name>\impl\lifmd\<synthesis_tool>`
3. Select and open `<instance_name>_top.ldf`. At this point, all of the files needed to support top-level synthesis and implementation are imported to the project.
4. Select the **Process** tab in the left-hand GUI window.
5. Implement the complete design via the standard Diamond GUI flow.

4.10. Hardware Evaluation

The CSI-2/DSI D-PHY Receiver Submodule IP supports Lattice's IP hardware evaluation capability, so you can create versions of the IP that operate in hardware for a limited period of time without requiring the request of an IP license. It may also be used to evaluate the IP in hardware in user-defined designs.

4.10.1. Enabling Hardware Evaluation in Diamond

Choose **Project** > **Active Strategy** > **Translate Design Settings**. The hardware evaluation capability may be enabled or disabled in the **Strategy** dialog box. It is enabled by default.

4.11. Updating/Regenerating the IP

The Clarity Designer interface allows you to update the local IPs from the Lattice IP server. The updated IP can be used to regenerate the IP instance in the design. To change the parameters of the IP used in the design, the IP must also be regenerated.

4.11.1. Regenerating an IP in Clarity Designer

To regenerate IP in Clarity Designer:

1. In the **Builder** tab, right-click the IP instance to be regenerated and select **Config** from the menu as shown in Figure 4.9.

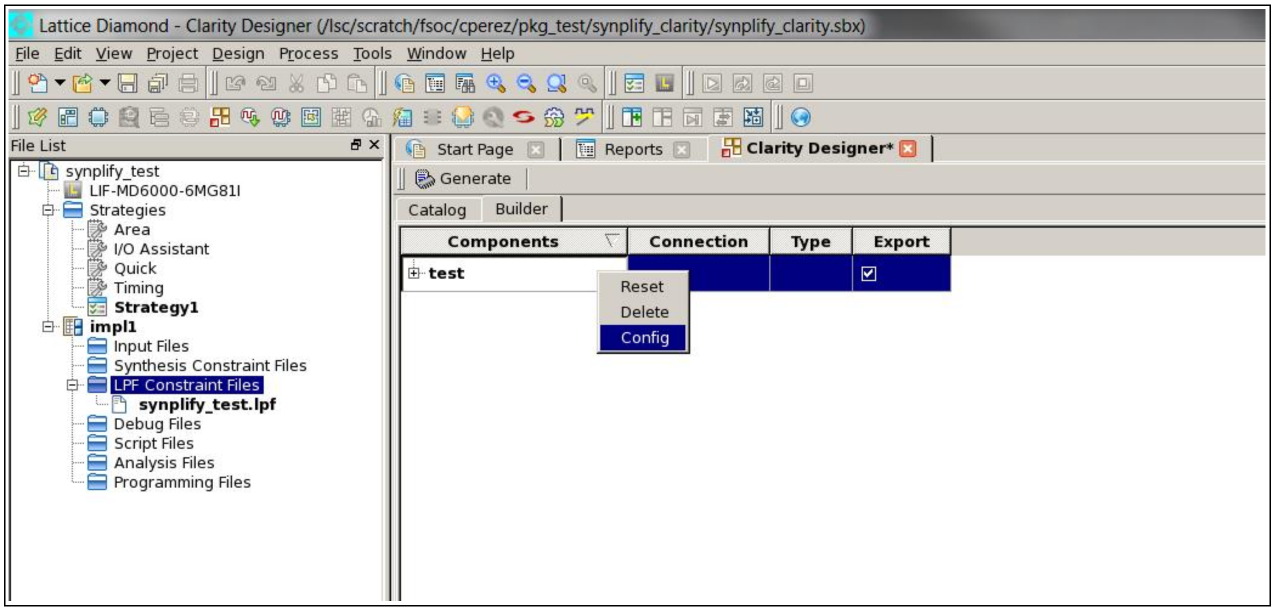


Figure 4.9. Regenerating IP in Clarity Designer

2. The IP Configuration GUI is displayed. Change the parameters as required and click the **Configure** button.
3. Click Generate in the toolbox. Clarity Designer regenerates all the IP instances which are reconfigured.

References

For more information about CrossLink devices, refer to FPGA-DS-02007, CrossLink Family Data Sheet.

For further information on interface standards refer to:

- MIPI Alliance Specification for D-PHY, version 1.1, November 7, 2011, www.mipi.org
- MIPI Alliance Specification for DSI, version 1.1, November 22, 2011,www.mipi.org
- MIPI Alliance Specification for Camera Serial Interface 2 (CSI-2) version 1.1, July 18, 2012, www.mipi.org

Software documentation:

- Clarity Designer User Guide
- Lattice Diamond User Guide

Technical Support Assistance

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

Appendix A. Resource Utilization

Table A.1 lists resource utilization for Lattice CrossLink FPGAs using the CSI-2/DSI D-PHY Receiver Submodule IP.

The values shown below are based on map reports. The performance and utilization data target an LIF-MD6000-6MG81I device with –6 speed grade using Lattice Diamond 3.9 and Lattice Synthesis Engine. Performance may vary when using a different software version or targeting a different device density or speed grade within the CrossLink family.

The Target f_{MAX} is the target byte clock frequency of each configuration. Actual f_{MAX} may vary depending on the complete top level design.

Table A.1. Resource Utilization

IP User-Configurable Parameters	Slices	LUTs	Registers	sysMEM EBRs	Programmable IOs	Target f_{MAX} (MHz)
Gear16, Hard D-PHY, Packet parser enabled	942	1264	787	0	0	93.75
Gear8, Hard D-PHY, Packet parser enabled	454	580	488	0	0	150.0
Gear16, Soft D-PHY, Packet parser enabled, Lane aligner enabled, EBR FIFO depth 16	1710	2375	1069	4	10	75.0
Gear8, Soft D-PHY, Packet parser enabled, Lane aligner enabled, EBR FIFO depth 16	974	1318	742	4	10	150.0
Gear16, Soft D-PHY, Packet parser enabled, Lane aligner disabled	1605	2197	1003	0	10	75.0
Gear8, Soft D-PHY, Packet parser enabled, Lane aligner disabled	874	1144	681	0	10	150
Gear16, Soft D-PHY, Packet parser disabled, Lane aligner disabled	919	1156	561	0	10	75.0
Gear8, Soft D-PHY, Packet parser disabled, Lane aligner disabled	553	688	369	0	10	150

Above configurations use 4-lanes, DSI mode, continuous D-PHY clock.

Appendix B. What is Not Supported

The MIPI D-PHY Rx submodule IP does not support the following features:

- PHY Protocol Interface (PPI)
- Low-level protocol error detection (SoT Error, SoT Sync Error, and so on)
- ECC check and error detection/correction of packet header in a short and a long packet
- Checksum calculation and error detection in long packet
- Command mode operation in MIPI DSI
- DCS parsing in MIPI DSI
- CCI communication in MIPI CSI-2
- Optional line synchronization packets in MIPI CSI-2

The CSI-2/DSI D-PHY Receiver Submodule IP has the following design limitations:

- Maximum value of word count in a long packet is 16'hFFF5.
- To ensure proper detection of low-power state transitions, the minimum duration of the MIPI D-PHY low-power states (T_{LPX}) of the data lanes should be at least two times the byte clock period. Similarly, the minimum duration of the low-power states of the clock lane should be twice the clk_lp_ctrl_i clock period.
- Maximum fabric speed is 150 MHz. Configurations that result in byte clock frequencies greater than 150 MHz cause timing violations.
- Ports for Low-Power Data Transmissions (LPDT) communications are provided. However, these are not fully tested in simulations.
- To ensure that high-speed data are registered correctly, the design requires at least 6 byte clock cycles of LP-11 and LP-01 between high-speed data transactions (that is $T_{HS\text{-}EXIT} + T_{LPX} \geq 6$ byte clock period).
- For non-continuous Rx clock mode, $T_{CLK\text{-}POST}$ should be at least 6 byte clock cycles to ensure the data is transferred from the stoppable byte clock domain to the free-running byte clock domain.
- DSI Null Packets are ignored by the packet parser. The signals lp_av_en_o and lp_en_o will not assert in the event that a null packet is received.
- The header for Blanking Packets are parsed and lp_en_o will assert in the event that a blanking packet is received. However, lp_av_en_o will remain deasserted. Payload data will not be buffered and will not be transmitted by the packet parser.
- The design supports all DSI and CSI-2 compatible formats. However, only common video DSI and CSI-2 formats were used to simulate the design:
    - DSI – RGB888, RGB666
    - CSI-2 – RGB888, RAW, YUV

  See Table 4.4 and Table 4.5 for the list of data types used in simulation.

Revision History

Date	Version	Change Summary
July 2017	1.0	Initial release