



OpenLDI/FPD-LINK/LVDS to MIPI DSI Display Interface Bridge Soft IP

User Guide

FPGA-IPUG-02005 Version 1.3

January 2017

Contents

1. Introduction	4
1.1. Quick Facts	4
1.2. Features.....	5
1.3. Conventions.....	5
1.3.1. Nomenclature.....	5
1.3.2. Data Ordering and Data Types	5
1.3.3. Signal Names	5
2. Functional Descriptions	6
2.1. Design and Module Description	9
2.2. Reset and Initialization	9
2.3. LVDS Wrapper	9
2.4. Byte Pad	9
2.5. cmos_2_dphy_ip	10
2.5.1. Pixel2Byte	10
2.5.2. Packet Formatter	10
2.5.3. Tx Global Operations Controller	10
2.5.4. D-PHY Common Interface Wrapper	11
2.5.5. dcs_rom/dcs_rom_hs.....	11
2.5.6. dcs_hs.....	11
2.5.7. dcs_control.....	11
3. Parameter Settings	12
4. IP Generation and Evaluation	14
4.1. Licensing the IP.....	14
4.2. Getting Started	14
4.3. Generating IP in Clarity Designer	15
4.4. Generated IP Directory Structure and Files.....	18
4.5. Running Functional Simulation	20
4.6. Simulation Strategies	21
4.7. Simulation Environment.....	21
4.8. Instantiating the IP	22
4.9. Synthesizing and Implementing the IP	22
4.10. Hardware Evaluation.....	23
4.10.1. Enabling Hardware Evaluation in Diamond	23
4.11. Updating/Regenerating the IP.....	23
4.11.1. Regenerating an IP in Clarity Designer	23
References	24
Technical Support Assistance	24
Appendix A. Resource Utilization	25
Appendix B. Initializing the DCS ROM	26
Low-Power Mode.....	26
High-Speed Mode	26
Appendix C. What is Not Supported	29
Revision History	30

Figures

Figure 1.1. OpenLDI/FPD-LINK/LVDS to MIPI DSI System Diagram	4
Figure 2.1. FPD-LINK Input Bus Waveform	6
Figure 2.2. Top Level Block Diagram	7
Figure 2.3. High-Speed Data Transmission Timing Diagram	10
Figure 4.1. Clarity Designer Window	14
Figure 4.2. Starting Clarity Designer from Diamond Design Environment	15
Figure 4.3. Configuring OpenLDI/FPD-LINK/LVDS to MIPI DSI Display Interface Bridge Soft IP in Clarity Designer	16
Figure 4.4. Configuration Tab in IP GUI	17
Figure 4.5. Video Tab in IP GUI	17
Figure 4.6. Protocol Timing Parameter Tab in IP GUI	18
Figure 4.7. OpenLDI/FPD-LINK/LVDS to MIPI DSI Display Interface Bridge Soft IP Directory Structure	18
Figure 4.8. Simulation Environment Block Diagram	21
Figure 4.9. Two Rx Channels Configuration with MISC_ON Disabled	21
Figure 4.10. One Rx Channel Configuration with MISCON Enabled	22
Figure 4.11. IP Regeneration in Clarity Designer	23
Figure B.1. Sample DCS ROM for DCS Low-Power Mode	26
Figure B.2. Sample DCS ROM for x4 Gear 8 DCS High-Speed Mode	27
Figure B.3. Data Ordering for a Gear 16x4 Configuration	27
Figure B.4. Sample DCS ROM for x4 Gear 16 DCS High-Speed Mode	28
Figure B.5. Directory Containing the Sample DCS ROM Initialization Files	28

Tables

Table 1.1. OpenLDI/FPD-LINK/LVDS to MIPI DSI Display Interface Bridge Soft IP Quick Facts	4
Table 2.1. OpenLDI/FPD-LINK/LVDS to MIPI DSI Display Interface Bridge IP Pin Function Description	7
Table 3.1. OpenLDI/FPD-LINK/LVDS to MIPI DSI Display Interface Bridge Soft IP Parameter Settings	12
Table 4.1. Files Generated in Clarity Designer	19
Table 4.2. Testbench Compiler Directives	20
Table A.1. Resource Utilization ¹	25

1. Introduction

The Mobile Industry Processor Interface (MIPI) provides specifications for standardization in consumer mobile devices. MIPI Display Serial Interface (DSI) and MIPI D-PHY specifications were developed to create a standardized interface for all displays used in the mobile industry. OpenLDI is an LVDS based interface commonly used internally for tablet, laptop and desktop display panels. As the industry evolves, it is common to have interface mismatches between legacy devices and displays.

The Lattice Semiconductor OpenLDI/FPD-LINK/LVDS to MIPI[®] DSI Display Interface Bridge Soft IP for Lattice Semiconductor CrossLink[™] translates video streams from a processor with an OpenLDI/FDP-Link/LVDS connection interface to a display with MIPI DSI interface.

This user guide is for OpenLDI/FPD-LINK/LVDS to MIPI DSI Display Interface Bridge Soft IP design version 1.2.



Figure 1.1. OpenLDI/FPD-LINK/LVDS to MIPI DSI System Diagram

1.1. Quick Facts

Table 1.1 provides quick facts about the OpenLDI/FPD-LINK/LVDS to MIPI DSI Display Interface Bridge Soft IP for CrossLink device.

Table 1.1. OpenLDI/FPD-LINK/LVDS to MIPI DSI Display Interface Bridge Soft IP Quick Facts

		OpenLDI/FPD-LINK/LVDS to MIPI DSI Display Interface Bridge Soft IP Configuration		
		1x7 Rx, 1x8 Tx, RGB888 configuration	2x7 Rx, 1x8 Tx, RGB888 configuration	2x7 Rx, 2x8 Tx, RGB888 configuration
IP Requirements	FPGA Families Supported	Crosslink LIF-MD6000-6MG81I		
	Targeted Device			
Resource Utilization	Data Path Width	28-bit input, serial output	56-bit input (28-bit per Rx channel), serial output	56-bit input (28-bit per Rx channel), serial output
	LUTs	1654	1802	2959
	sysMEM [™] EBRs	4	6	8
	Registers	1058	1170	1987
	HW MIPI Block	1	1	2
	Programmable I/O	5	9	9
Design Tool Support	Lattice Implementation	Lattice Diamond [®] 3.8		
	Synthesis	Lattice Synthesis Engine		
	Simulation	Synplify Pro [®] L-2016.03L		
		Aldec [®] Active HDL [™] 10.3 Lattice Edition		

1.2. Features

The key features of the OpenLDI/FPD-LINK/LVDS to MIPI DSI Display Interface Bridge IP are:

- Compliant to MIPI DSI v1.1 and MIPI D-PHY v1.1 specifications
- Supports MIPI DSI interfacing up to 12 Gb/s
- 4-8 MIPI D-PHY data lanes and 3-8 LVDS data lanes
- MIPI DCS (Display Command Set) controller to program the display for DSI Interface in either HS or HSLP mode
- Supports Non-Burst Mode with Sync Events data transaction
- Supports RGB888 and RGB666 video formats
- Supports Receiving in Dual Channel Flat Panel Display Link Protocol (7:1 LVDS)

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 most significant bit within the pixel data is the highest index. D-PHY outputs the least significant bit first.

1.3.3. Signal Names

Signal names that end with:

- “_n” are active low
- “_i” are input signals
- “_o” are output signals
- “_io” are bidirectional signals

2. Functional Descriptions

The OpenLDI/FPD-LINK/LVDS to MIPI DSI Display Interface Bridge converts a standard OpenLDI serial video interface into DSI byte packets. The input interface for the design consists of a data bus, vertical and horizontal sync flags, a data enable and a clock in OpenLDI (LVDS7:1) interface format.

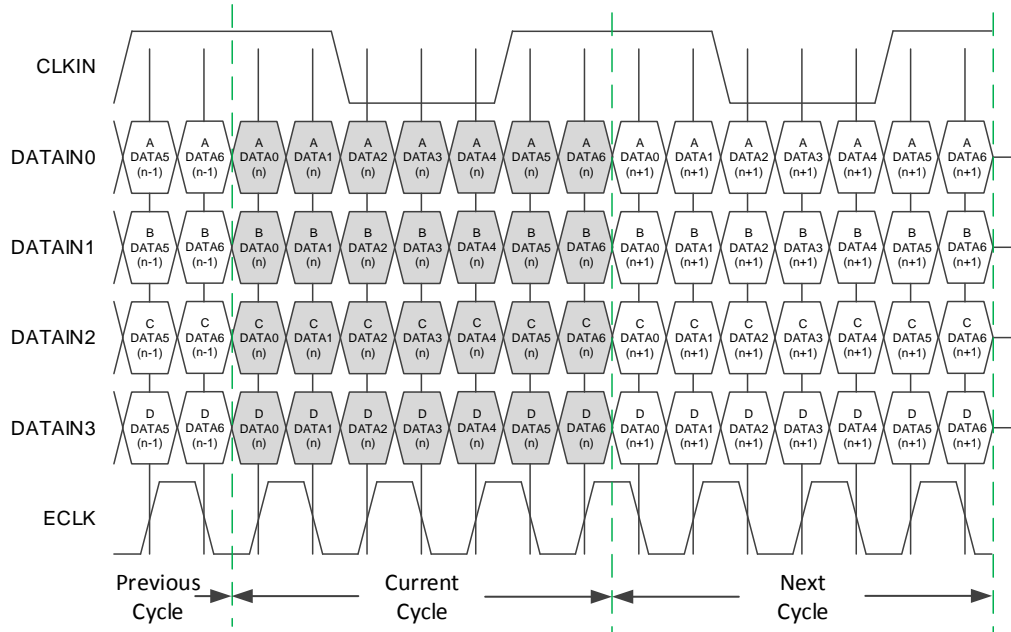


Figure 2.1. FPD-LINK Input Bus Waveform

Basic timing of LVDS7:1 interface is shown in Figure 2.1. There is a 2-bit offset between the rising edge of LVDS clock and the word boundary. Each word is 7-bit long. DATAIN0, DATAIN1, DATAIN2, and DATAIN3 are the data lanes. Clock is the LVDS clock lane. For every 7-bit data packet, LSB is the first input serial data to the receiver.

A processor sends video packet data to the FPGA chip via OpenLDI/FPD-LINK (LVDS7:1) interface. One channel of LVDS7:1 receiver has a maximum of 5 lanes. Each channel consists of 1 LVDS clock pair and 4 LVDS data pairs (RGB888) or 3 LVDS data pairs (RGB666). Maximum of two LVDS7:1 channels can be used. When dual channel is supported, additional data lanes are activated. Clock is running at 1/7th of the data rate as this is the standard for the LVDS7:1 interface. Unbalanced mode for the LVDS operating system is used as this is commonly used. A maximum of 1.2 Gb/s data rate per lane is supported for LVDS.

From the processor, data and clock are transmitted serially to the LVDS7:1 receiver. Crosslink does data processing such as converting the received LVDS pixel data to MIPI D-PHY serial data. Consequently, MIPI D-PHY also has a maximum of 5 lanes per channel. It consists of 1 clock lane and 4 data lanes. Similar to the LVDS side, the IP supports options for use of one or two MIPI D-PHY for output depending on bandwidth needs. Maximum MIPI D-PHY data rate per lane is 1.5 Gb/s but this is limited by the 1.2 Gb/s LVDS line rate. Only RGB888 and RGB666 data types are supported as MIPI D-PHY Tx will be used to interface with display using DSI protocol. Serialized MIPI D-PHY data are transmitted to the connected display via Display Serial Interface protocol.

Data Lane 0 of each MIPI D-PHY can be used to configure the display with DCS (Display Command Set) commands when DCS is in HSLP mode. When DCS is in HS mode, commands are distributed in all the data lanes depending on the number of lanes selected with data lane 0 having the 1st byte command. A module (`dcs_rom.v` or `dcs_rom_hs.v`) to assist in placing the DCS commands on the LP data lane is provided. If user wants to change the DCS commands to be compatible with the display to be used, DCS command must be provided to include the proper display commands.

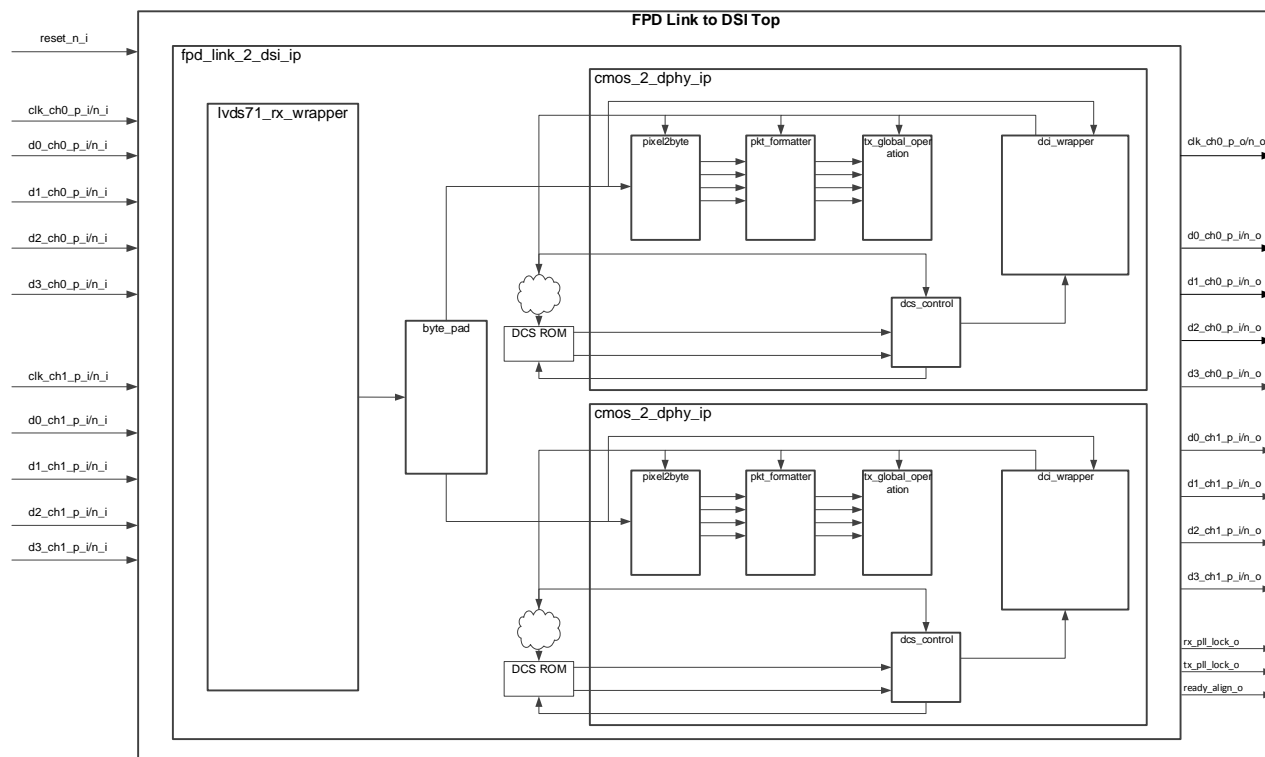


Figure 2.2. Top Level Block Diagram

Table 2.1. OpenLDI/FPD-LINK/LVDS to MIPI DSI Display Interface Bridge IP Pin Function Description

Pin Name	Direction	Function Description
Reset		
reset_n_i	I	Asynchronous active low system reset; 0 – System on reset
OpenLDI/FPD-LINK Interface		
clk_ch0_p_i	I	Positive LVDS Input clock to LVDS7:1 Rx Wrapper channel 0. For dual LVDS channel configuration, only channel 0 clock is used as reference clock for the system
clk_ch0_n_i	I	Negative LVDS Input clock to LVDS7:1 Rx Wrapper channel 0, complement of clk_ch0_p_i. For dual LVDS channel configuration, only channel 0 clock is used as reference clock for the system
clk_ch1_p_i	I	Positive LVDS Input clock to LVDS7:1 Rx Wrapper channel 1. For dual LVDS channel configuration, only channel 0 clock is used as reference clock for the system, channel 1 clock is unused
clk_ch1_n_i	I	Negative LVDS Input clock to LVDS7:1 Rx Wrapper channel 1, complement of clk_ch1_p_i. For dual LVDS channel configuration, only channel 0 clock is used as reference clock for the system, channel 1 clock is unused
d0_ch0_p_i	I	Positive LVDS Input data lane 0 to LVDS7:1 Rx Wrapper channel 0
d0_ch0_n_i	I	Negative LVDS Input data lane 0 to LVDS7:1 Rx Wrapper channel 0, complement of d0_ch0_p_i
d1_ch0_p_i	I	Positive LVDS Input data lane 1 to LVDS7:1 Rx Wrapper channel 0
d1_ch0_n_i	I	Negative LVDS Input data lane 1 to LVDS7:1 Rx Wrapper channel 0, complement of d1_ch0_p_i
d2_ch0_p_i	I	Positive LVDS Input data lane 2 to LVDS7:1 Rx Wrapper channel 0
d2_ch0_n_i	I	Negative LVDS Input data lane 2 to LVDS7:1 Rx Wrapper channel 0, complement of d2_ch0_p_i
d3_ch0_p_i ¹	I	Positive LVDS Input data lane 3 to LVDS7:1 Rx Wrapper channel 0
d3_ch0_n_i ¹	I	Negative LVDS Input data lane 3 to LVDS7:1 Rx Wrapper channel 0, complement of d3_ch0_p_i

Table 2.1. OpenLDI/FPD-LINK/LVDS to MIPI DSI Display Interface Bridge IP Pin Function Description (Continued)

Pin Name	Direction	Function Description
d0_ch1_p_i ²	I	Positive LVDS Input data lane 0 to LVDS7:1 Rx Wrapper channel 1. Drive to default when not used.
d0_ch1_n_i ²	I	Negative LVDS Input data lane 0 to LVDS7:1 Rx Wrapper channel 1, complement of d0_ch1_p_i. Drive to default when not used.
d1_ch1_p_i ²	I	Positive LVDS Input data lane 1 to LVDS7:1 Rx Wrapper channel 1. Drive to default when not used.
d1_ch1_n_i ²	I	Negative LVDS Input data lane 1 to LVDS7:1 Rx Wrapper channel 1, complement of d1_ch1_p_i. Drive to default when not used.
d2_ch1_p_i ²	I	Positive LVDS Input data lane 2 to LVDS7:1 Rx Wrapper channel 1. Drive to default when not used.
d2_ch1_n_i ²	I	Negative LVDS Input data lane 2 to LVDS7:1 Rx Wrapper channel 1, complement of d2_ch1_p_i. Drive to default when not used.
d3_ch1_p_i ^{1,2}	I	Positive LVDS Input data lane 3 to LVDS7:1 Rx Wrapper channel 1. Drive to default when not used.
d3_ch1_n_i ^{1,2}	I	Negative LVDS Input data lane 3 to LVDS7:1 Rx Wrapper channel 1, complement of d3_ch1_p_i. Drive to default when not used.
DSI Interface		
d0_ch0_p_o	O	Positive D-PHY data lane 0 channel 0
d0_ch0_n_o	O	Negative D-PHY data lane 0 channel 0, complement of d0_ch0_p_o
d1_ch0_p_o	O	Positive D-PHY data lane 1 channel 0
d1_ch0_n_o	O	Negative D-PHY data lane 1 channel 0, complement of d1_ch0_p_o
d2_ch0_p_o	O	Positive D-PHY data lane 2 channel 0
d2_ch0_n_o	O	Negative D-PHY data lane 2 channel 0, complement of d2_ch0_p_o
d3_ch0_p_o	O	Positive D-PHY data lane 3 channel 0
d3_ch0_n_o	O	Negative D-PHY data lane 3 channel 0, complement of d3_ch0_p_o
clk_ch0_p_o	O	Positive D-PHY output clock channel 0
clk_ch0_n_o	O	Negative D-PHY output clock channel 0, complement of clk_ch0_p_o
d0_ch1_p_o ³	O	Positive D-PHY data lane 0 channel 1
d0_ch1_n_o ³	O	Negative D-PHY data lane 0 channel 1, complement of d0_ch1_p_o
d1_ch1_p_o ³	O	Positive D-PHY data lane 1 channel 1
d1_ch1_n_o ³	O	Negative D-PHY data lane 1 channel 1, complement of d1_ch1_p_o
d2_ch1_p_o ³	O	Positive D-PHY data lane 2 channel 1
d2_ch1_n_o ³	O	Negative D-PHY data lane 2 channel 1, complement of d2_ch1_p_o
d3_ch1_p_o ³	O	Positive D-PHY data lane 3 channel 1
d3_ch1_n_o ³	O	Negative D-PHY data lane 3 channel 1, complement of d3_ch1_p_o
clk_ch1_p_o ³	O	Positive D-PHY output clock channel 1
clk_ch1_n_o ³	O	Negative D-PHY output clock channel 1, complement of clk_ch1_p_o
Miscellaneous		
ready_align_o ⁴	O	1 – Indicates that bit and word alignment are done (via BW_ALIGN) and DCS controller is done sending DCS commands to the display 0 – Indicates that bit and word alignment are not yet done (via BW_ALIGN) and DCS controller is not yet done sending DCS commands to the display
rx_pll_lock_o ⁴	O	Active high LVDS Rx PLL lock signal
tx_pll_lock_o ⁴	O	Active high Mixel Tx PLL lock signal

Notes:

1. Used only when RGB888 data type is used, otherwise, this port is unused
2. LVDS channel 1 input ports are not available when single LVDS channel is selected during IP generation.
3. MIPI D-PHY channel 1 output ports are not available when single MIPI D-PHY channel is selected during IP generation.
4. Can be turned-on if MISC_ON is selected in the GUI upon IP generation

2.1. Design and Module Description

The top level module instantiates the main module `cmos_2_dphy_ip.v`. This module contains the D-PHY related module `lvds_rx_wrapper.v` that contains the OpenLDI/FPD-LINK related modules and DCS ROMs.

DCS ROMs contains the DCS commands for display when DSI is selected.

The `cmos_2_dphy_ip.v` module instantiates the `pixel2byte.v`, `pkt_formatter.v`, `tx_global_operation.v`, and `dci_wrapper.v`, and `dcs_control.v` modules.

2.2. Reset and Initialization

Active low reset is used in the design with synchronous release. This is the system reset input connected to LVDS7:1 Rx Wrapper and CMOS2DPHY blocks and are synchronized in each of the clock domains used in the blocks.

Follow this initialization and reset sequence:

1. Assert active low system reset for at least 500 ns.
2. After power-up, the Master PHY is required to drive a Stop State (LP-11) for a period longer than t_{INIT} ($t_{INIT} \geq 100 \mu s$ as per MIPI D-PHY Specification version 1.1). PHY can force the lane module into transmit mode and generate stop state. The Master PHY is initialized by a system reset using Protocol input signal (PPI).
3. Initialize the Slave PHY when the Master PHY drives a Stop State (LP-11). The first Stop State longer than the specified t_{INIT} is called the Initialization period.
4. Wait for Rx and Tx PLL lock (`rx_pll_lock_o`, `tx_pll_lock_o`) to be asserted.
5. Wait for `ready_align_o` to be asserted. The `ready_align_o` is used to indicate LVDS7:1 Rx Wrapper data training is done and DCS controller is done sending DCS commands to the display. Only when `ready_align_o` is asserted that valid data can be sampled and correctly transmitted by the LVDS7:1 Rx Wrapper.

Note: Steps 3 and 4 are used to make sure that system is ready to receive valid data. Exact wait time cannot be specified since design's alignment is dynamic and these flag signals are used to observe when to send valid data.

The LVDS7:1 Rx Wrapper is now ready to receive valid data.

2.3. LVDS Wrapper

The 7:1 LVDS Rx interface is a source synchronous LVDS interface. Seven serial data bits are de-serialized for each cycle of the low-speed clock. This interface has a maximum of five LVDS pairs (four data, one clock). The five pairs translate to 28 parallel data bits and one clock lane. Note that there is a 2-bit offset between the clock rising edge and the word boundary. Each word is 7 bits long.

LVDS data are fed to the I/O logic IDDR71 register. LVDS clock is also fed to the I/O logic IDDR71 register, as well as to PLL. PLL is used to generate the sampling clock (ECLK) which is 3.5 times faster than the LVDS clock. Alignment IP together with PLL is used to shift ECLK to normally 90 degrees to allow the placing of the edge of clock in the middle of the data valid window. CLKDIV is used to generate a slower clock (SCLK) which is 3.5 times slower than ECLK. SCLK is used as the output clock of IDDR71 IP.

After ECLK and SCLK are generated, LVDS7:1 soft IP does bit and word alignment. Bit alignment is placing ECLK in the middle of the data training window, and word alignment is getting the correct training sequence after bit alignment. Training pattern is 7'b1100011 and alignment is done with respect to LVDS clock. It is expected for the LVDS clock and data to be edge-aligned with each other so that when alignment is done with respect to the LVDS clock, the same goes for the LVDS data lanes. Dual x4 LVDS7:1 Rx can only be supported if the two channels are sharing the same clock.

2.4. Byte Pad

This module is used to pad 0s (multiple of bytes) to the end of active pixel data per line in cases when active pixel data is not multiple of 4 for x8 Tx gearing or 8 for x16 Tx gearing which is a requirement of CMOS2DPHY IP. Only multiples of 4 or 8 bytes depending on the Tx gearing selected can be processed. Padded 0s are considered valid data by CMOS2DPHY IP.

2.5. cmos_2_dphy_ip

2.5.1. Pixel2Byte

The pixel2byte module converts pixel data to MIPI byte data based on number of bits per pixel, the data type (DT) and number of D-PHY lanes. The input to this system is the pixel data (pixdata[L:0], vsync, hsync) and a data enable for display interfaces. The output of this module provides a byte enable indicating data payload is available and payload itself. The data type output provides the packet info indicator at the output of pixel2byte.

2.5.2. Packet Formatter

The packet formatter module wraps the packet header and packet footer modules. The packet header module generates and appends the packet header and footer to the data payload. The long_en input and the byte_data bus are used together to identify when payload is available. The header field and payload size is configurable by setting VC, WC, and DT parameters. EoTp is supported by this module. The Packet footer module computes a CRC16 checksum based on incoming data and data enable. The data bus input is maximum 24-bits as the maximum parallel input width is 24-bit RGB.

2.5.3. Tx Global Operations Controller

The Tx global operations controller controls HS request path and timing using parameters. Currently LP-request, escape mode and turnaround path are not supported. This block follows the requirements as per D-PHY Specification version 1.2 section 6 – operating modes for control and high-speed data transmission.

DSI data goes into Lower-Power mode during vertical and horizontal blanking depending on the Soft IP design.

Example: HS-0/1 -> LP11(Stop) -> LP01(LP-Rqst) -> LP00(Bridge) -> HS-0/1

This module controls the timing entering HS and coming out from HS entering to LP following the D-PHY Specification version 1.1 Table 16. The delay parameters can be adjusted inside this module by changing its local parameters.

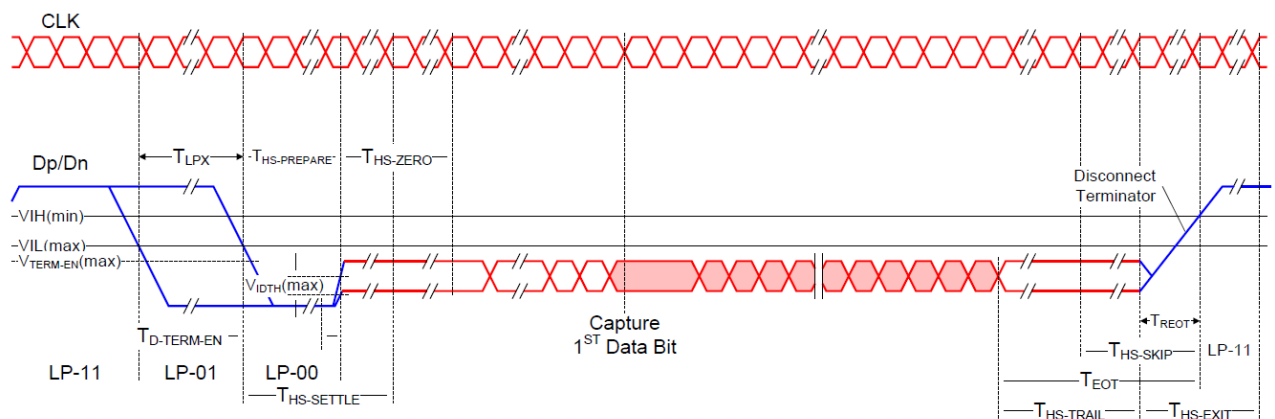


Figure 2.3. High-Speed Data Transmission Timing Diagram

Figure 2.3 shows that prior to the HS mode data transfer, all clock and data lanes are in the LP11 state (1.2 V on the P channel and 1.2 V on the N channel). The clock lane then goes to the LP01 state (0 V on the P channel and 1.2 V on the N channel) then the LP00 state (0 V on the P channel and 0 V on the N channel). After that, the clock lane goes into HS mode with SLVS200 signaling ($V_{cm}=200$ mV, $V_{diff}=\pm 100$ mV) and holds an HS0 state (differential 0 state of P channel = 100 mV and N channel = 300 mV when termination of the receiver is turned on). Then the clock starts shortly after.

When the HS clock is running, the data lanes follow a similar procedure going from LP11 to LP01, LP00, and HS0 states. Then the HS-SYNC sequence is driven on the line followed by the packet header and data payload. At the end of the transfer the data lanes first go back into LP mode by going to LP00 then LP11 states. The clock lane follows shortly after.

2.5.4. D-PHY Common Interface Wrapper

The D-PHY Common Interface (DCI) wrapper is used as the wrapper of the MIPI D-PHY IP. This is used to serialize the incoming byte data and transmits to D-PHY receiver. The DCI connects the PHY hard IP and higher protocol layers. Based on the Tx global operation state, it determines how to enable HS or LS mode for data transfer.

2.5.5. dcs_rom/dcs_rom_hs

The DCS ROM is used to store the Display Command Set for the interfaced DSI display, and to place the DCS commands on the data lane provided. See [Appendix B](#). Initializing the DCS ROM for details.

2.5.6. dcs_hs

The DCS HS is used to control the flow of DCS data when DCS is transmitted in HS mode and append proper trail bits.

2.5.7. dcs_control

The DCS LP controller is used to send MIPI Display Command Set (DCS) configuration commands through data lane 0. DCS is only used with MIPI DSI, not CSI-2. The controller provides a multiplexed path to data lane 0, where inputs to the controller are the 8-bit MIPI DCS words and the output is a one hot-encoded 2-channel signal to be transferred on the P and N channels of data lane 0. A ROM is used to hard code configuration DCS command for display at startup.

3. Parameter Settings

Table 3.1 lists the parameters used to generate the OpenLDI/FPD-LINK/LVDS to MIPI DSI Display Interface Bridge Soft IP. All parameters are either set automatically or input in the GUI during the OpenLDI/FPD-LINK/LVDS to MIPI DSI Display Interface Bridge Soft IP generation.

Table 3.1. OpenLDI/FPD-LINK/LVDS to MIPI DSI Display Interface Bridge Soft IP Parameter Settings

Parameter	Attribute	Options	Description
Number of Rx Channels	User-Input	1, 2	Number of LVDS Rx channel.
Rx Interface	Read-Only	—	Rx is set to LVDS interface.
Number of Rx Lanes	Read-Only	3, 4	Generates LVDS IOs to either three or four depending on the data type.
Rx Gear	Read-Only	7, 14	Specify the Rx gearing. Gear14 is automatically selected for Tx Line rate that is 900 Mb/s and above.
Number of Tx Channels	Read-Only	1, 2	Number of CMOS to D-PHY IP instance. When the Number of Rx Channels is set to 1, the Number of Tx Channels is automatically set to 1.
Tx Interface	Read-Only	—	Tx is set to DSI interface.
Number of Tx Lanes	Read-Only	4	Generates IOs to four HS Tx data lane. Always set to 4.
Tx Gear	Read-Only	8, 16	Specify the Tx gearing. Gear16 is automatically selected when Rx Gear 14 is selected.
Tx Clock Mode	User-Input	0, 1	Specify the Tx clock transmission mode. 1 – High Speed 0 – Low Power
Rx Total Aggregate Bandwidth	Read-Only	<Value>	Rx total line rate. Automatically computed based on target Tx line rate. This is not equal to Tx line rate since DSI control signals (hsync, vsync, de) are included in the traffic for FPDLINK interface.
Tx Total Aggregate Bandwidth	Read-Only	<Value>	Target total Tx line rate.
Tx Line Rate (per lane)	User-Input	<Value>	Target Tx line rate per lane.
D-PHY Clock Frequency	Read-Only	<Value>	D-PHY clock. Automatically computed based on target Tx line rate
Byte Clock Frequency	Read-Only	<Value>	Byte clock. Automatically computed based on target Tx line rate
Pixel Clock Frequency	Read-Only	<Value>	Pixel clock. Automatically computed based on target Tx line rate
LVDS Input Clock Frequency	Read-Only	<Value>	LVDS clock; Automatically computed based on target Tx line rate
DCS Clock Frequency	Read-Only	<Value>	DCS clock. Automatically computed based on target Tx line rate. Used when DCS is transmitted in Low-Speed mode.
DCS ROM Wait Time	User-Input	<Value>	Specify delay time for sending DCS commands. This is clocked by byte clock if DCS is in High-Speed mode and DCS clock when DCS is in Low-Speed mode. This needs to be adjusted if DCS is not meeting protocol timing requirements. Normal value is 1200 based on 111.375 MHz byte clock.
tINIT_SLAVE Value	User-Input	<Value>	Specify delay time for Tx D-PHY tINIT requirement. Must satisfy Tx D-PHY tINIT minimum requirement of 100 μ s and is clocked by byte clock.
Data Type	User-Input	RGB888, RGB666	Specify the data type depending on the Data Format selected
Virtual Channel	Read-Only	0, 1, 2, 3	Specify the Virtual Channel. Always set to 0
Word Count	User-Input	1 – 65536	Specify the total number of bytes for active pixels in a line. Word Count = (Number of active pixel per line) · (Bits per pixel) / 8 For example RGB888, 1920x1080p60 resolution Word Count = (1920) · (24) / 8 = 5760

Table 3.1. OpenLDI/FPD-LINK/LVDS to MIPI DSI Display Interface Bridge Soft IP Parameter Settings (Continued)

Parameter	Attribute	Options	Description
Number of DCS words	User-Input	<Value>	Specify the number of words for the DCS commands. For dual MIPI D-PHY configuration, when DCS for both MIPI D-PHY are enabled, it is expected that both has the same number of DCS words.
DCS Mode	User-Input	0, 1	Specify the DCS transmission mode. 1– High Speed 0 – Low Power
DCS rom initialization file	User-Input	<file>	Specify the DCS commands. See Appendix B for details.
Data HS-Prepare	User-Input	<Value>	Specify the time that the D-PHY Tx drives the Data Lane LP-00 Line state immediately before the HS-0 Line state starting the HS transmission. Timing parameter is in number of byte clock cycles. The value is automatically computed initially but can be customized.
Data HS-Zero	User-Input	<Value>	Specify the time that the D-PHY Tx drives the HS-0 state prior to transmitting the Sync sequence. Timing parameter is in number of byte clock cycles. The value is automatically computed initially but can be customized.
Clock Pre	User-Input	<Value>	Specify the time that the HS clock shall be driven by the D-PHY Tx prior to any associated Data Lane beginning the transition from LP to HS mode. Timing parameter is in number of byte clock cycles. The value is automatically computed initially but can be customized.
Clock Post	User-Input	<Value>	Specify the time that the D-PHY Tx continues to send HS clock after the last associated Data Lane has transitioned to LP Mode. Interval is defined as the period from the end of THS-TRAIL to the beginning of TCLK-TRAIL. Timing parameter is in number of byte clock cycles. The value is automatically computed initially but can be customized.

4. IP Generation and Evaluation

This section provides information on how to generate the Lattice OpenLDI/FPD-LINK/LVDS to MIPI DSI Display Interface Bridge Soft IP code using the Diamond Clarity Designer and how to run simulation, synthesis and hardware evaluation.

4.1. Licensing the IP

An IP-specific license is required to enable full, unrestricted use of the OpenLDI/FPD-LINK/LVDS to MIPI DSI Display Interface Bridge Soft IP in a complete, top level design. The OpenLDI/FPD-LINK/LVDS to MIPI DSI Display Interface Bridge Soft IP is available free of charge.

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 and generate the OpenLDI/FPD-LINK/LVDS to MIPI DSI Display Interface Bridge Soft IP and fully evaluate the IP through functional simulation and implementation (synthesis, map, place and route) without an IP license. The OpenLDI/FPD-LINK/LVDS to MIPI DSI Display Interface Bridge Soft IP also supports Lattice's IP hardware evaluation capability, which makes it possible to create versions of the IP that operate in hardware for a limited time (approximately four hours) without requiring an IP license. See the [Hardware Evaluation](#) section on page 23 for further details. However, the 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 OpenLDI/FPD-LINK/LVDS to MIPI DSI Display Interface Bridge Soft 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 Designer GUI as shown in [Figure 4.1](#).

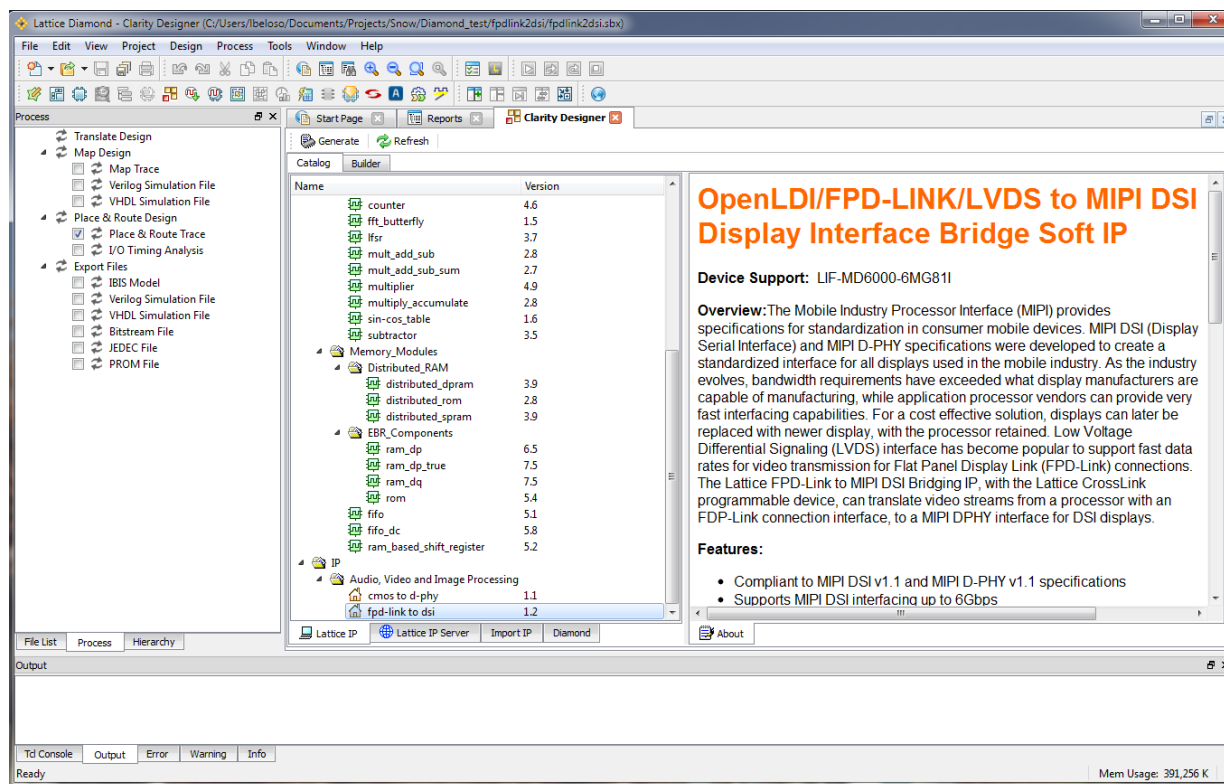


Figure 4.1. Clarity Designer Window

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. Besides configuration and generation of modules and IPs, Clarity Designer can also create a top module template in which all generated modules and IPs are instantiated.

The following describes the procedure for generating OpenLDI/FPD-LINK/LVDS to MIPI DSI Display Interface Bridge Soft IP in Clarity Designer.

Clarity Designer can be started from the Diamond design environment.

To start Clarity Designer:

1. Create a new empty 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 or fill out the following items as shown in [Figure 4.2](#).
 - **Create new Clarity design** - Choose to create a new Clarity Design project directory in which the OpenLDI/FPD-LINK/LVDS to MIPI DSI Display Interface Bridge Soft 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.

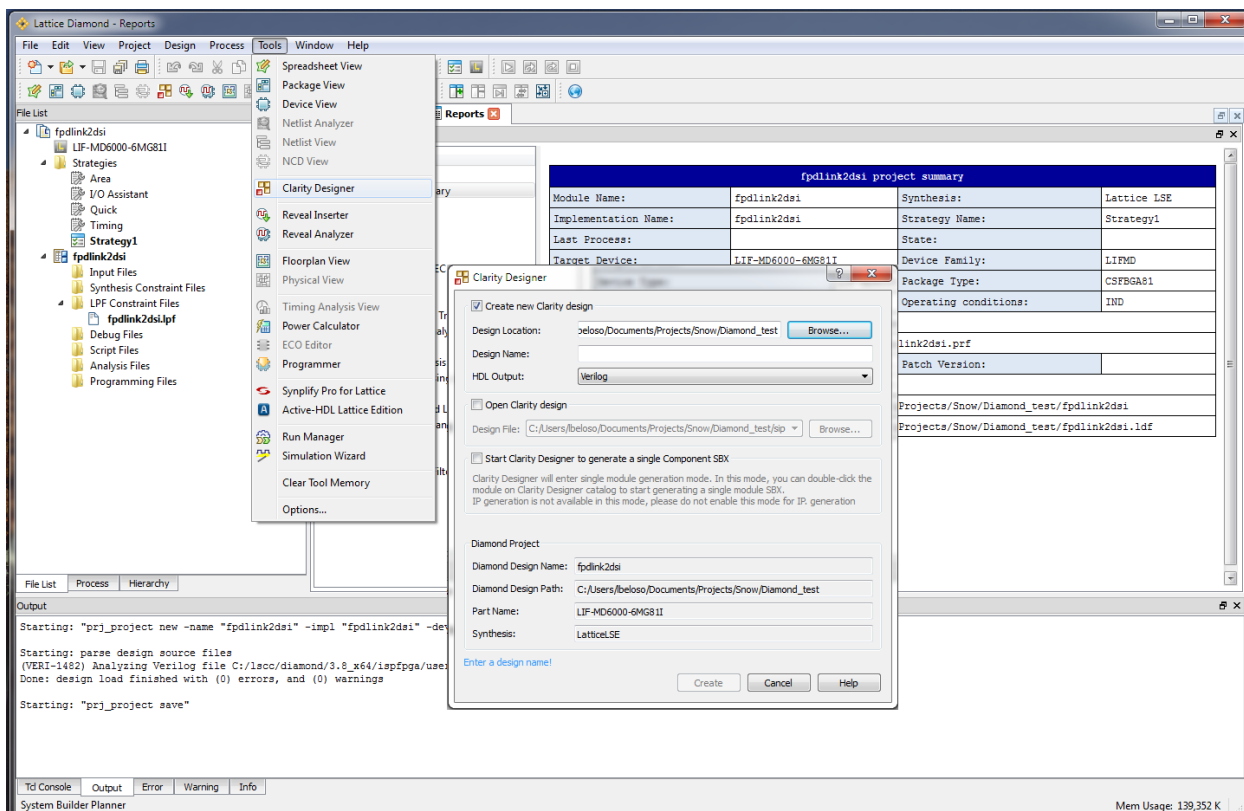


Figure 4.2. Starting Clarity Designer from Diamond Design Environment

To configure OpenLDI/FPD-LINK/LVDS to MIPI DSI Display Interface Bridge Soft IP in Clarity Designer:

1. Double-click **FPD-LINK to DSI** in the IP list of the System Catalog view. The **fpdlink_2_dsi** dialog box is displayed as shown in [Figure 4.3](#).

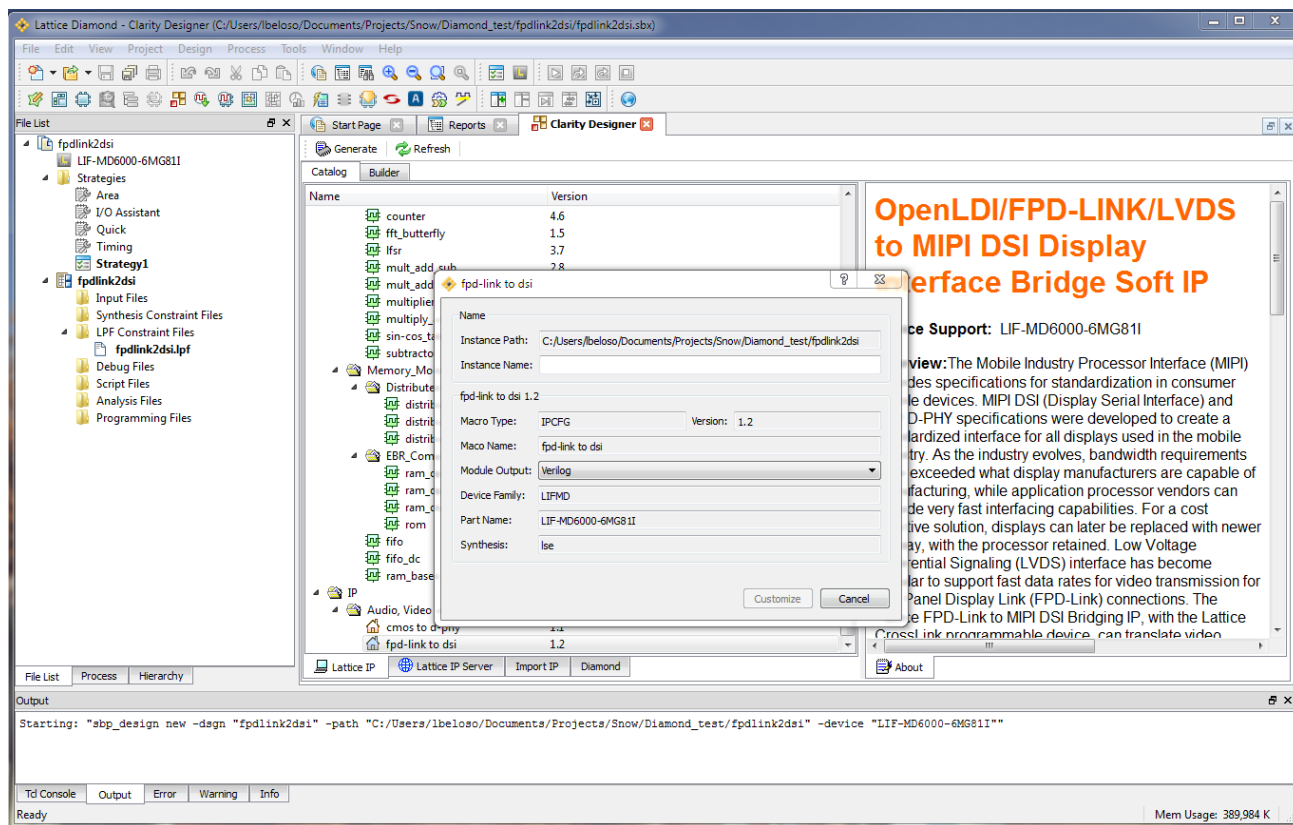


Figure 4.3. Configuring OpenLDI/FPD-LINK/LVDS to MIPI DSI Display Interface Bridge Soft IP in Clarity Designer

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 configuration specific to your application.
4. Input valid values in the required fields in the **Configuration** tab.
5. Go to **Video** tab and input valid values in the required fields. See [Figure 4.5](#).

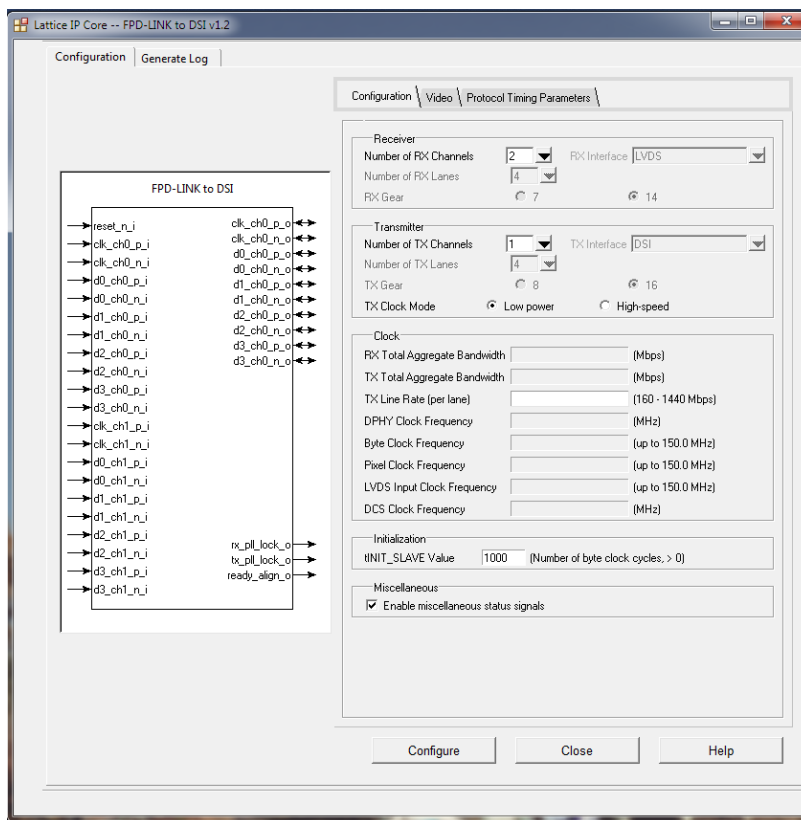


Figure 4.4. Configuration Tab in IP GUI

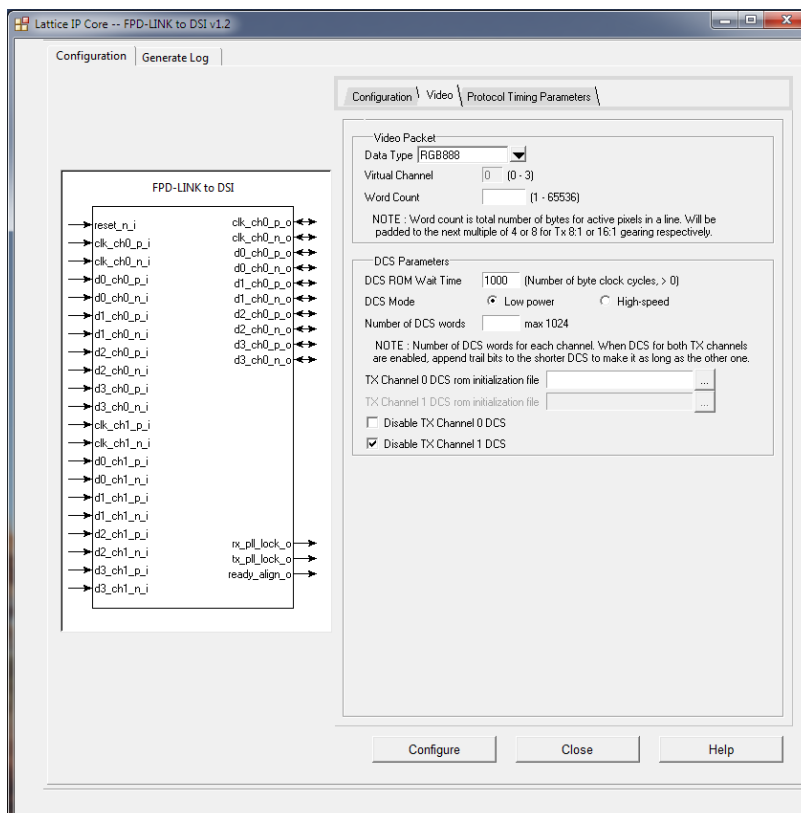



Figure 4.5. Video Tab in IP GUI

6. Go to **Protocol Timing Parameters** tab. The values shown are automatically computed. You can enter other valid values if customization is desired.
7. After selecting the required parameters, click the **Configure** button.
8. Click **Close**.
9. 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, refer to the Lattice Diamond software user guide.

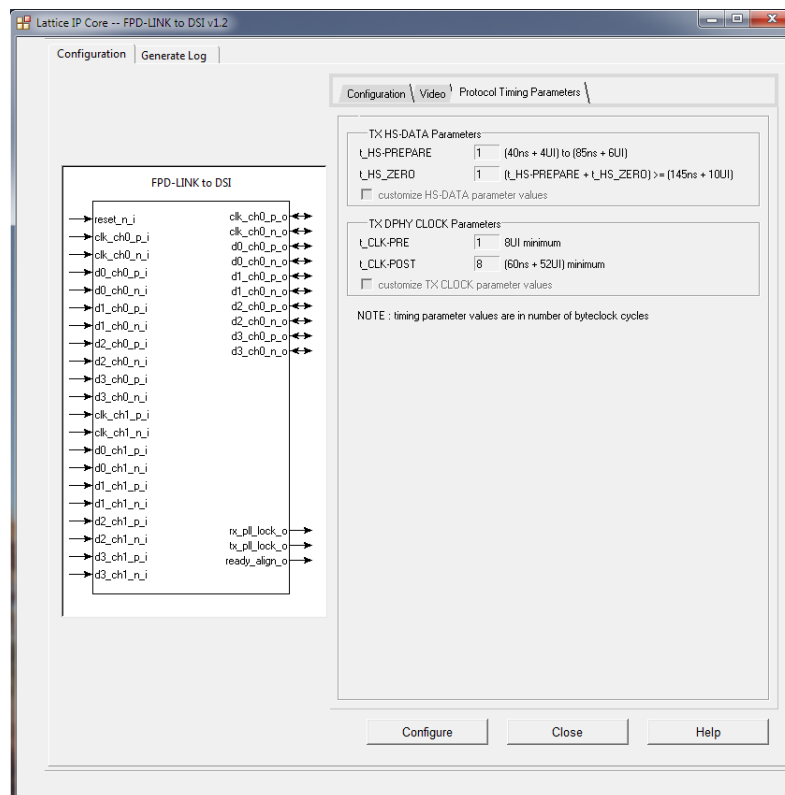


Figure 4.6. Protocol Timing Parameter Tab in IP GUI

4.4. Generated IP Directory Structure and Files

The directory structure of generated IP and supporting files is shown in Figure 4.7.

Clarity Designer IP Folder

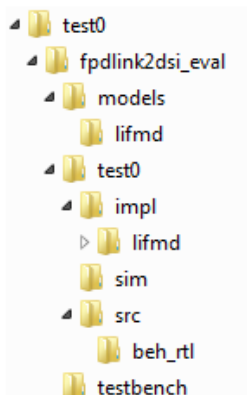


Figure 4.7. OpenLDI/FPD-LINK/LVDS to MIPI DSI Display Interface Bridge Soft IP Directory Structure

The design flow for the IP created with Clarity Designer uses a post-synthesized module (NGO) for synthesis and uses a protected model for simulation. The post-synthesized module and protected model are customized when you configure the IP and created automatically when the IP is generated.

Table 4.1 provides a list of key files and directories created by Clarity Designer and how they are used. The post-synthesized module (NGO), the protected simulation model, and all other files are also generated based on your configuration and provided as examples to use or evaluate the IP.

Table 4.1. Files Generated in Clarity Designer

File	Description
<instance_name>.v	Verilog top-level module of OpenLDI/FPD-LINK/LVDS to MIPI DSI Display Interface Bridge Soft 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 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.

Aside from the files listed in the tables, most of the files required to evaluate the OpenLDI/FPD-LINK/LVDS to MIPI DSI Display Interface Bridge Soft IP are available under the directory \<fpdlink2dsi_eval>. This includes the simulation model. Lattice Diamond project files are also included under the folder at \<fpdlink2dsi_eval>\<instance_name>\impl\lifmd\<synthesis_tool>\.

The \<instance_name> folder (test0 in Figure 4.7) contains files/folders with content specific to the <instance_name> configuration. This directory is created by Clarity Designer each time the IP is generated and regenerated with the same file name. A separate \<instance_name> directory is generated for IPs with different names, such as \<my_IP_0>, \<my_IP_1>, and others.

The folder \<instance_name>, the \fpdlink2dsi_eval and sub directories provide files supporting OpenLDI/FPD-LINK/LVDS to MIPI DSI Display Interface Bridge Soft IP evaluation that includes files/folders with content that is constant for all configurations of the OpenLDI/FPD-LINK/LVDS to MIPI DSI Display Interface Bridge Soft IP. The \fpdlink2dsi_eval directory is created by Clarity Designer the first time the IP is generated, when multiple OpenLDI/FPD-LINK/LVDS to MIPI DSI Display Interface Bridge Soft IPs are generated in the same root directory and updated each time the IP is regenerated.

You can use the prebuilt Diamond projects provided at

\<project_root>\fpdlink2dsi_eval\<instance_name>\impl\lifmd\<synthesis_tool>\ to evaluate the implementation (synthesis, map, place and route) of the IP in Lattice Diamond tool. The src directory contains the behavioral models of the black-boxed modules and the models directory provides library elements.

4.5. Running Functional Simulation

To run simulations using Active HDL follow these steps:

1. Modify the "do" file located in \<project_dir>\fpdlink2dsi_eval\<instance_name>\sim\aldec\
 - a. Specify working directory (sim_working_folder), for example
`set sim_working_folder "C:/my_design"`
 - b. Specify workspace name that will be created in working directory, for example
`set workspace_name "design_space"`
 - c. Specify design name, for example
`set design_name "DesignA"`
 - d. Specify design path where the IP Core generated using Clarity Designer is located, for example
`set design_path "C:/my_designs/DesignA"`
 - e. Specify design instance name (same as the instance name specified in Clarity Designer) , for example
`set design_inst "DesignA_inst"`
 - f. Specify Lattice Diamond Primitive path (diamond_dir) to where it is installed, for example
`set diamond_dir "C:/lscd/diamond/3.8_x64"`
2. Update testbench parameters to customize data size, clock and/or other settings. See [Table 4.2](#) for the list of valid testbench compiler directives.
3. Under the **Tools** menu, select **Active-HDL**.
4. In Active-HDL window, under the **Tools** tab, select **Execute Macro**.
5. Select the "do" file \<project_dir>\fpdlink2dsi_eval\<instance_name>\sim\aldec*.do
6. Click **OK**.
7. Wait for the simulation to finish.

[Table 4.2](#) is a list of testbench directives which can be modified by setting the define in the vlog command in the "do" file.

Example:

```
vlog \
+define+SIP5_NUM_FRAMES=2 \
+define+SIP5_TOTAL_LINES=5 \
....
```

Table 4.2. Testbench Compiler Directives

Compiler Directive	Description
SIP5_NUM_FRAMES	Used to set the number of video frames
SIP5_NUM_LINES	Used to set the number of lines per frame
HFRONT	Number of cycles before HSYNC signal asserts (Horizontal Front Blanking)
HPULSE	Number of cycles HSYNC signal asserts
HBACK	Number of cycles after HSYNC signal asserts (Horizontal Rear Blanking)
VFRONT	Number of cycles before VSYNC signal asserts (Vertical Front Blanking)
VPULSE	Number of cycles VSYNC signal asserts
VBACK	Number of cycles after VSYNC signal asserts (Vertical Rear Blanking)
SIP5_CLK	LVDS7:1 input clock period

4.6. Simulation Strategies

This section describes the simulation environment which demonstrates basic FPDLINK2DSI functionality. Figure 4.8 shows a block diagram of the simulation environment.

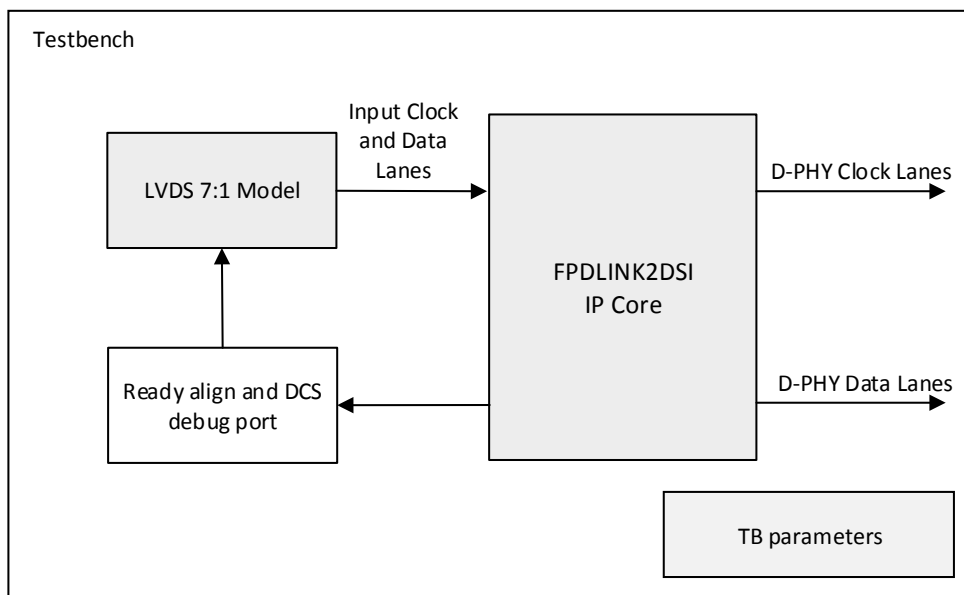


Figure 4.8. Simulation Environment Block Diagram

4.7. Simulation Environment

The simulation environment is made up of an LVDS 7:1 model instance connected to the input of FPDLINK2DSI IP core instance in the testbench. The LVDS 7:1 model is configured based on the FPDLINK2DSI IP configurations and testbench parameters. The testbench can be configured as 1 or 2 Rx channels. If miscellaneous signals such as PLL lock and ready_align debug ports are included in the IP core, the testbench monitors assertion of these signals before sending the DSI video data. See the [Running Functional Simulation](#) section on page 20 for details on how to set testbench parameters.

Figure 4.9 shows an example simulation of two Rx channels configuration with miscellaneous signals disabled.

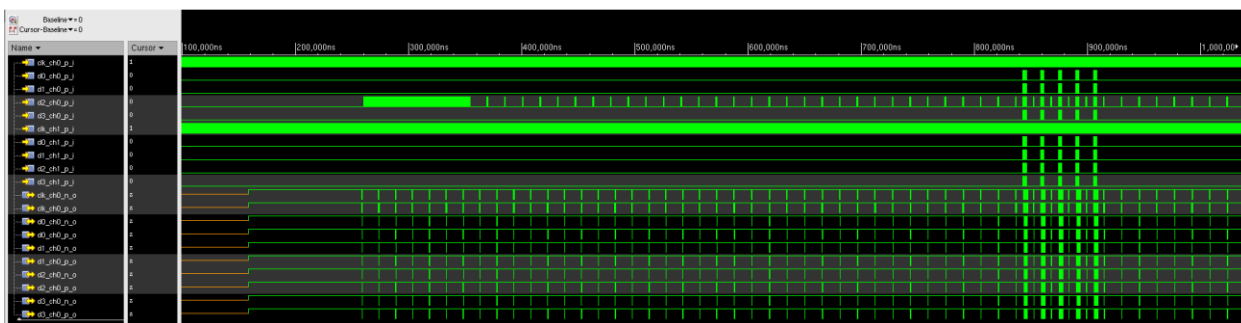


Figure 4.9. Two Rx Channels Configuration with MISC_ON Disabled

Since miscellaneous signals are disabled, user should set initial driving delay before testbench starts sending data to the IP core. In this example, ch0 and ch1 data inputs are driven as IP core supports two Rx channels.

Figure 4.10 shows an example simulation of one Rx channel configuration with miscellaneous signals enabled.

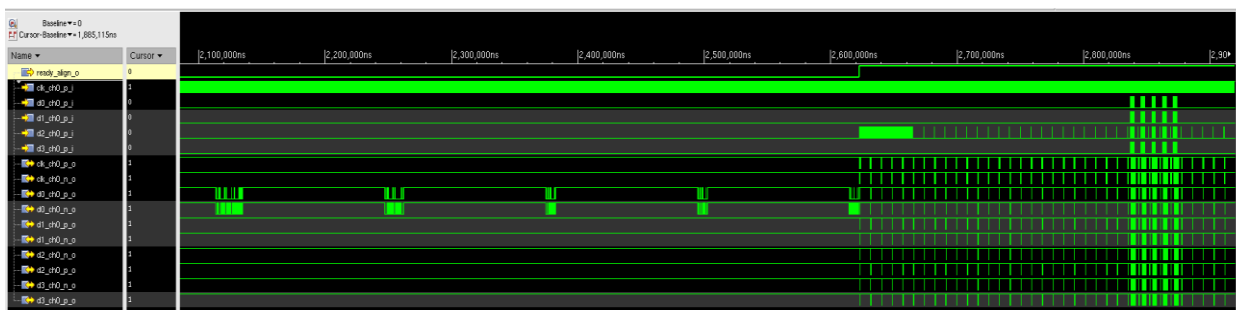


Figure 4.10. One Rx Channel Configuration with MISCON Enabled

In this example, miscellaneous signals are enabled. Testbench monitors for `ready_align_o` signal to assert indicating that DCSROM_done has completed. Then testbench drives `ch0` data inputs as the IP core supports one Rx channel.

4.8. Instantiating the IP

The core modules of the OpenLDI/FPD-LINK/LVDS to MIPI DSI Display Interface Bridge Soft IP are synthesized and provided in NGO format with black box Verilog source files for synthesis. A Verilog source file named `<instance_name>_fpd_link_2_dsi_ip.v` instantiates the black box of core modules. The top-level file `<instance_name>.v` instantiates `<instance_name>_fpd_link_2_dsi_ip.v` and DCS components.

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.

The user does not need to instantiate the IP instances 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.

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 all IPs are 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>\fpdlink2dsi_eval\<instance_name>\impl\lifmd\lse` or
`\<project_dir>\fpdlink2dsi_eval\<instance_name>\impl\lifmd\synplify` directories. All required files are invoked automatically. You can directly synthesize, map and place/par the design in the Diamond design environment after the cores are generated.

Push-button implementation of this top-level design with either Synplify or Lattice Synthesis Engine is supported via the Diamond project files `<instance_name>_top.ldf` which is located in

`\<project_dir>\fpdlink2dsi_eval\<instance_name>\impl\lifmd\<synthesis_tool>\` directory.

To use the pre-bult Diamond project files:

1. Choose **File > Open > Project**.
2. In the **Open Project** dialog box browse to
`\<project_dir>\fpdlink2dsi_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.

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, <http://mipi.org/>
- MIPI Alliance Specification for Display Serial Interface, version 1.1, November 22, 2011, <http://mipi.org/>

Technical Support Assistance

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

Appendix A. Resource Utilization

Table A.1 list resource utilization information for Lattice CrossLink FPGA using the OpenLDI/FPD-LINK/LVDS to MIPI DSI Display Interface Bridge Soft IP.

Clarity Designer is the Lattice IP configuration utility, and is included as a standard feature of the Diamond tool. For details about the usage of Clarity Designer refer to the Clarity Designer and Diamond help system.

For more information on the Diamond design tools, visit the Lattice web site at

www.latticesemi.com/Products/DesignSoftwareAndIP.

Table A.1. Resource Utilization¹

IP User-Configurable Parameters	Slices	LUTs	Registers	sysMEM EBRs	Actual f_{MAX} (MHz) ²	Target f_{MAX} (MHz) ²
1x7 Rx, 1x8 Tx, RGB888	1152	1654	1058	4	128.733	112.500
2x7 Rx, 1x8 Tx, RGB888	1223	1802	1170	6	123.870	112.500
2x14 Rx, 1x16 Tx, RGB888	1410	2135	1393	12	109.075	90.000
2x7 Rx, 2x8 Tx, RGB888	2095	2959	1987	8	116.036	112.500

- Performance and utilization data target an LIF-MD6000-6MG81I device using Lattice Diamond 3.8 and Lattice Synthesis Engine software. Performance may vary when using a different software version or targeting a different device density or speed grade within the CrossLink family. This does not show all possible configurations of the OpenLDI/FPD-LINK/LVDS to MIPI DSI Display Interface Bridge Soft IP.
- The f_{MAX} values are based on byte clock.

Appendix B. Initializing the DCS ROM

Display Command Set (DCS) initialization is used to configure the command registers of a DSI-compliant display. The bridge has an option to perform this in high-speed or in low-power mode.

In either DCS mode, the number of entries must correspond to the number of DCS Words indicated in the GUI.

There should be no empty lines within the text file. Comments within the file are not supported.

Low-Power Mode

To initialize the DCS ROM in low-power mode, the input file must contain one byte of data in each line, in hex format.

Figure B.1 shows the sample entries.

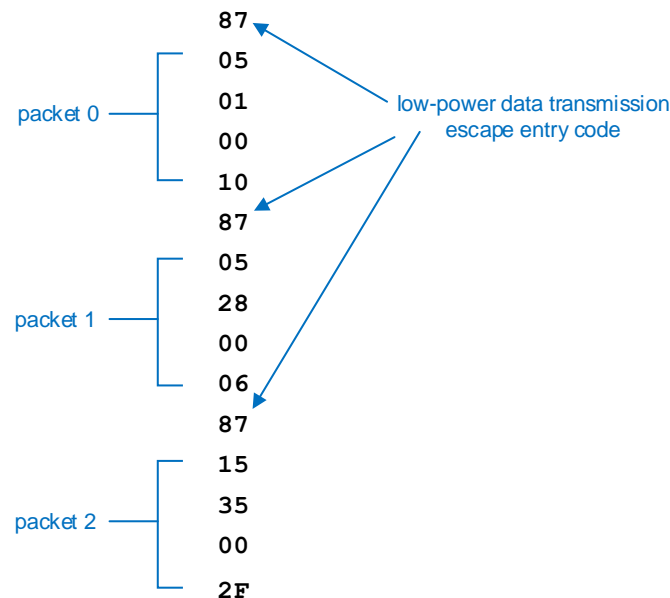


Figure B.1. Sample DCS ROM for DCS Low-Power Mode

The 8'h87 byte indicates the start of a new packet. In this example, the DCS Controller breaks down the DCS words into 3 packets. The last entry should be the last valid byte. DCS Word Count in this example is 15.

High-Speed Mode

When the DCS ROM initialization is in high-speed mode, the interval between high-speed transmissions may be set through the DCS ROM Wait Time parameter. Multiple packets may be concatenated to reduce overhead of frequent switching between low-power state and high-speed mode.

The entries within the input file should be in the following format:

```
<trail bit indicator><DCS byte lane3><DCS byte lane2><DCS byte lane1><DCS byte lane0>
```

For each high-speed transmission, each lane must start with the SoT pattern 8'hB8, and the last word should be made up of complete trail bytes with the trail indicator bit set to 1. The design checks this trail bit indicator to determine the end of the high-speed transmission.

Sample DCS for Gear 8

Figure B.2 shows the sample entries for the DCS initialization file of a 4-lane, gear 8 configuration.

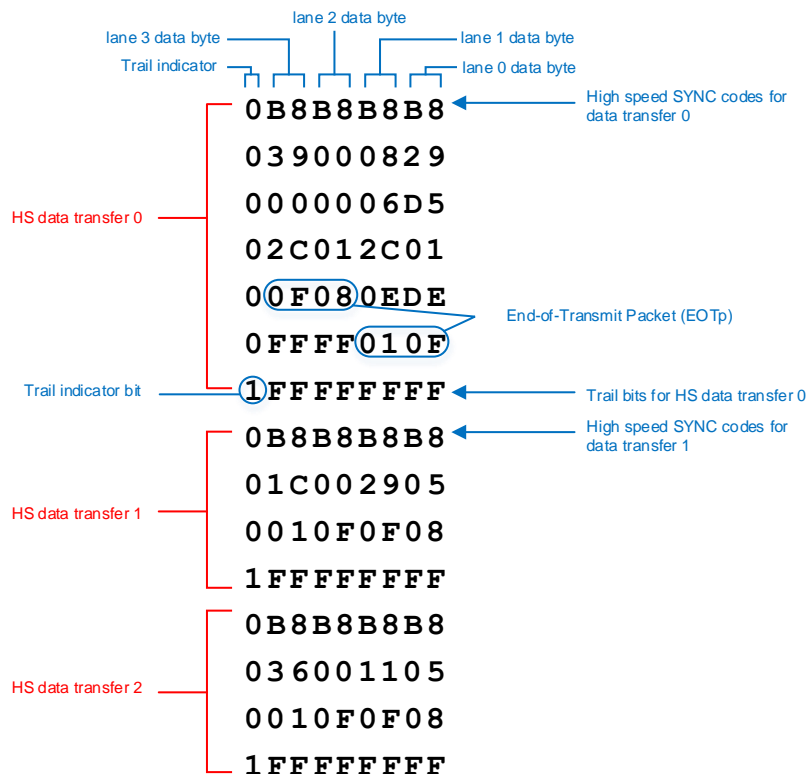


Figure B.2. Sample DCS ROM for x4 Gear 8 DCS High-Speed Mode

In this example, an End-of-Transmit packet is sent after each DCS packets. The last word after each high-speed transmission must be made up completely of trail bits. The DCS Word Count in this example is 15.

Sample DCS for Gear 16

The byte order of DCS words for gear 16 configuration should follow the order described in Figure B.3.

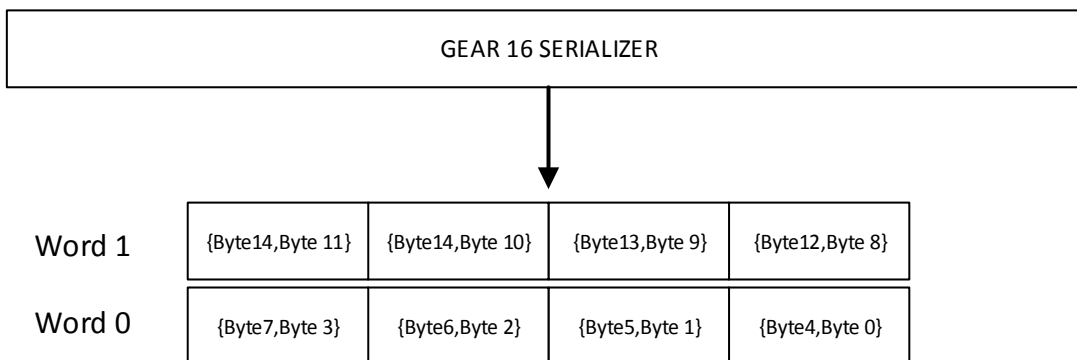


Figure B.3. Data Ordering for a Gear 16x4 Configuration

Figure B.4 shows the sample entries for 4-lane Gear 16 is configuration.

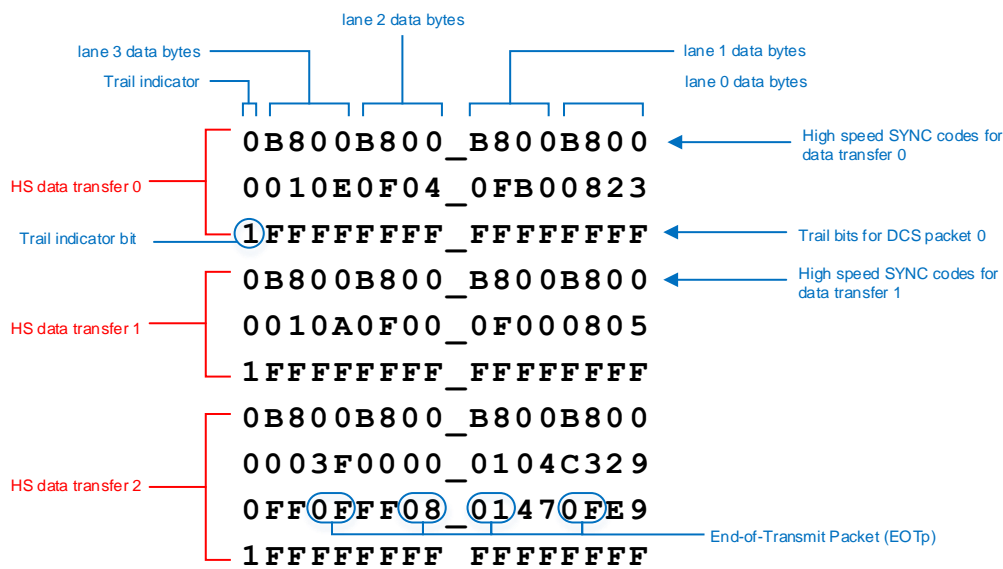


Figure B.4. Sample DCS ROM for x4 Gear 16 DCS High-Speed Mode

This example contains three DCS packets concatenated with EoTp at the end of each. The first word for each lane contains the MIPI D-PHY high-speed synchronization sequence 8'hB8 padded with zeros at the start. The zero padding is used for alignment purposes only. These may be removed, but the data bytes should be adjusted accordingly. The DCS Word Count in this example is 10.

Sample DCS ROM files are also available under `./<IPinstallation_path>/samples`.

Name	Type	Size
hsdcsrom_x4_gear16_wc109	MEM File	6 KB
hsdcsrom_x4_gear8_wc89	MEM File	5 KB
lpdcsrom_sample	Text Document	1 KB

Figure B.5. Directory Containing the Sample DCS ROM Initialization Files

Appendix C. What is Not Supported

The IP does not support configuration through registers.

The IP has the following limitation:

Pixel frequencies that do not follow the D-PHY PLL CN requirement are not supported. The D-PHY PLL which is used by the MIPI D-PHY Tx to generate the D-PHY clocks has to be programmed such that the frequency after the input divider ranges from 24 MHz to 30 MHz ($24 \leq (\text{CLKREF}/N) \leq 30$), otherwise, D-PHY PLL does not work. CLKREF corresponds to the slow clock of the system (pixel clock). Due to this requirement, there are frequency holes in the frequency range supported by the design. Refer to D-PHY PLL Specifications for more details regarding the D-PHY PLL requirement.

Revision History

Date	Version	Change Summary
January 2017	1.3	<ul style="list-style-type: none"> Added support for 2:2 LVDS to MIPI DSI Updated number of LUTs, Registers, and Programmable I/Os in Table 1.1. OpenLDI/FPD-LINK/LVDS to MIPI DSI Display Interface Bridge Soft IP Quick Facts, and added new column of 2x7 Rx, 2x8 Tx, RGB888 configuration Updated data rate to 12 Gb/s and number of MIPI D-PHY data lanes to 4-8 in Features section Updated port list in Table 2.1. OpenLDI/FPD-LINK/LVDS to MIPI DSI Display Interface Bridge IP Pin Function Description, and added Note 3 Updated Table 3.1. OpenLDI/FPD-LINK/LVDS to MIPI DSI Display Interface Bridge Soft IP Parameter Settings <ul style="list-style-type: none"> Updated description of Number of Tx Channels and Number of DCS words Added Tx Clock Mode Changed parameter name to Rx Total Aggregate Bandwidth and Tx Total Aggregate Bandwidth Updated figures in Getting Started and Generating IP in Clarity Designer sections Updated values in Table A.1. Resource Utilization, and added new row of 2x7 Rx, 2x8 Tx, RGB888
November 2016	1.2	<ul style="list-style-type: none"> Added Note 2 to Table 2.1. OpenLDI/FPD-LINK/LVDS to MIPI DSI Display Interface Bridge IP Pin Function Description Updated Word Count description in Table 3.1. OpenLDI/FPD-LINK/LVDS to MIPI DSI Display Interface Bridge Soft IP Parameter Settings Updated Licensing the IP section – Added email address lic_admn@latticesemi.com for requesting free license Added step 9 in Generating IP in Clarity Designer section
July 2016	1.1	<ul style="list-style-type: none"> Updated document number, the previous document number was IPUG124 Updated Table 1.1. OpenLDI/FPD-LINK/LVDS to MIPI DSI Display Interface Bridge Soft IP Quick Facts Added parameters Data HS-PREPARE, Data HS-Zero, Clock Pre, Clock Post to Table 3.1. OpenLDI/FPD-LINK/LVDS to MIPI DSI Display Interface Bridge Soft IP Parameter Settings Updated Generating IP in Clarity Designer section New Running Functional Simulation, Simulation Strategies, and Simulation Environment sections Updated values in Table A.1. Resource Utilization Updated Appendix B. Initializing the DCS ROM
May 2016	1.0	Initial release.



7th Floor, 111 SW 5th Avenue
Portland, OR 97204, USA
T 503.268.8000
www.latticesemi.com