4:1 MIPI CSI-2 Camera Interface Bridge Soft IP

User Guide

FPGA-IPUG-02020-1.1

April 2019

Contents

Figures

Tables

1. Introduction

The majority of mobile image sensors and application processors in the consumer market use Mobile Industry Processor Interface (MIPI®) Camera Serial Interface 2 (CSI-2). Mobile influenced products often use multiple image sensors for depth perception as well as object and gesture detection. In some cases, mobile Application Processors (AP) may not have enough interfaces to support the number of image sensor inputs required for a particular application. In other cases, the processing latency between image sensors and imaging data may be too large.

The Lattice Semiconductor Quad MIPI CSI-2 to Single MIPI CSI-2 Bridge IP for the Lattice Semiconductor CrossLink™ devices allows you to mux between four image sensors and then merge the two chosen video streams to a single MIPI CSI-2 video image stream output. This is useful for augmented and virtual reality, drone, interactive gaming, and 360° camera applications.

This user guide is for 4:1 MIPI CSI-2 Camera Interface Bridge Soft IP design version 1.x.

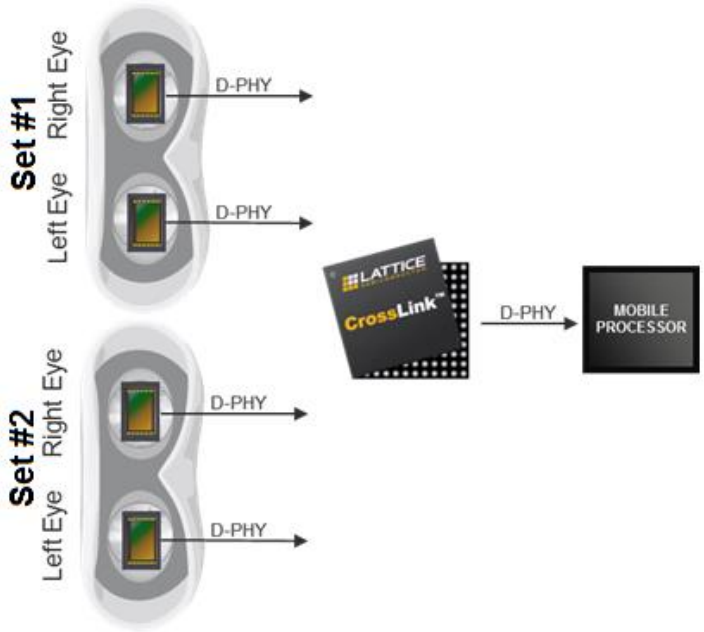

Figure 1.1. Quad Camera Aggregation System Diagram for Lattice CrossLink

1.1. Quick Facts

Table 1.1. provides quick facts about the Lattice 4:1 MIPI CSI-2 Bridge IP for CrossLink devices.

Table 1.1. Lattice 4:1 MIPI CSI-2 Bridge IP Quick Facts

		4:1 MIPI CSI-2 IP Configuration
		Left-Right Merge
		2-Lane HSLP Configuration
Core Requirements	FPGA Families Supported	CrossLink
Resource Utilization	Target Device	LIF-MD6000-6MG81I
	LUTs	3922
	sysMEM™ EBRs	8
	Registers	2313
Design Tool Support	Lattice Implementation	Lattice Diamond® 3.8 or later
	Synthesis	Lattice Synthesis Engine (LSE) Synopsys® Synplify Pro® L-2016.03L or later
	Simulation	Aldec® Active HDL™ 10.3 Lattice Edition

1.2. Features

The key features of the Quad MIPI CSI-2 to Single MIPI CSI-2 Bridge IP are:

- Supports MIPI D-PHY Specification version 1.1 and MIPI CSI-2 Specification version 1.1
- Output data rate up to 1.5 Gb/s/lane
- Configurable to 1 or 2 data lanes for each image sensor
- Supports all MIPI CSI-2 compatible data types
- Provides input pin to mux which two out of four camera inputs get merged
- Left-Right (LR) Merge – merges video data packets from two channels chosen to form a single packet for each pixel line. Data type of both inputs must be the same

1.3. Conventions

1.3.1. Nomenclature

The nomenclature used in this document is based on the Verilog language. 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.

Single-bit data stream from each MIPI D-PHY Rx data lane is deserialized into 8-bit parallel data where bit 0 is the first received bit.

LR Merge method requires the data type from both channels to be the same.

1.3.3. Signal Names

Signal names that end with:

- *_i* are input pins.
- *_o* are output pins.
- *_io* are bi-directional pins.
- *_n_i* are active low.

2. Functional Description

MIPI CSI-2 is an interface between a peripheral device (camera image sensor) and a host processor (baseband, application engine). The Lattice Quad MIPI CSI-2 to Single CSI-2 Interface Bridge IP combines data streams from two image sensors to be sent out to an Application Processor. Figure 2.1 shows the top level block diagram of 4:1 MIPI CSI-2 Bridge IP.

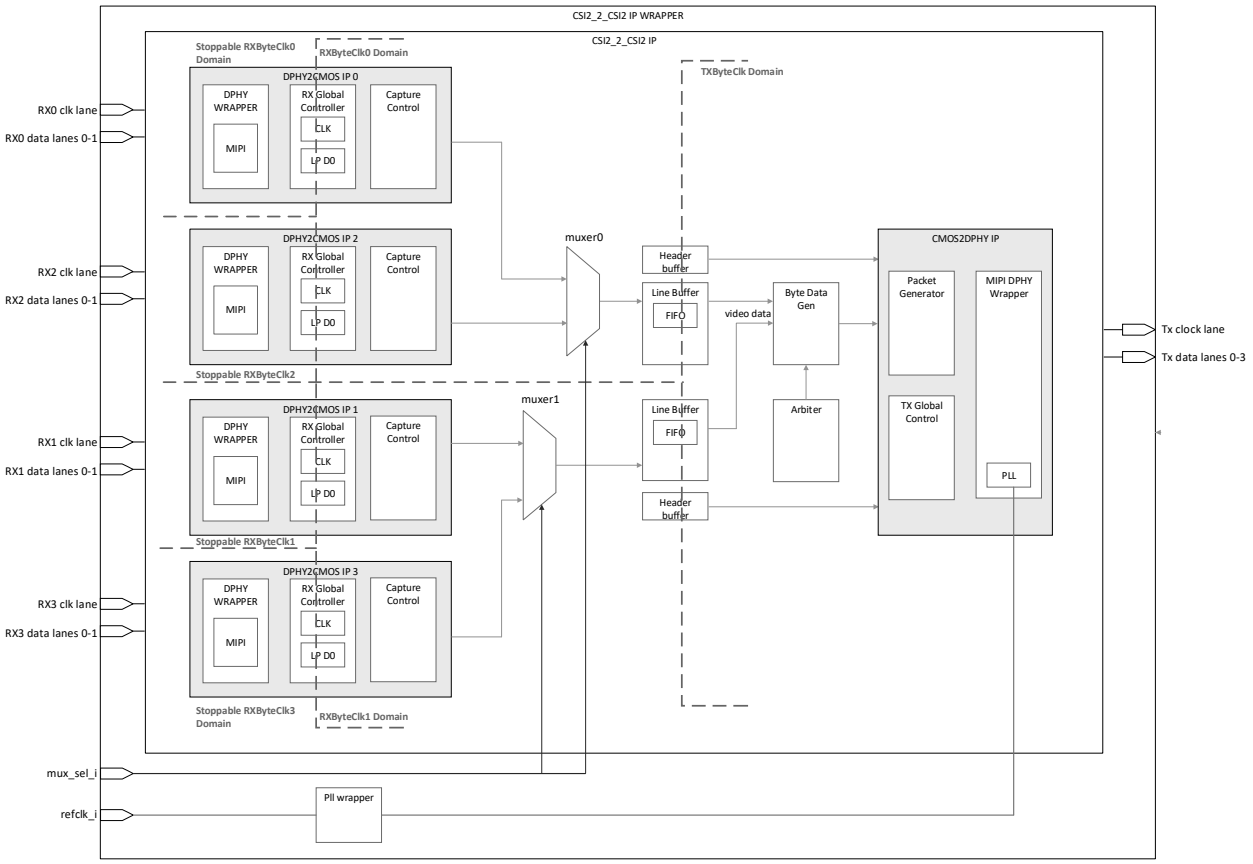

Figure 2.1. 4:1 MIPI CSI-2 Bridge IP Block Diagram

Table 2.1. 4:1 MIPI CSI-2 Top Pin Function Description

Signal	Direction	Description
Clocks and Resets		
ref_clk_i	I	Input reference clock (must be the same as the byte clock frequency). This is only needed for Non-continuous Rx Clock Mode
reset_n_i	I	Asynchronous active low system reset
Mux Control		
mux_sel_i	I	Chooses which image sensor inputs to merge. 0 – Sensors 0 and 1 1 – Sensors 2 and 3
CSI-2 Rx Interface		
clk_ch0_p_i	I	Positive differential Rx Ch0 D-PHY input clock
clk_ch0_n_i	I	Negative differential Rx Ch0 D-PHY input clock
clk_ch1_p_i	I	Positive differential Rx Ch1 D-PHY input clock
clk_ch1_n_i	I	Negative differential Rx Ch1 D-PHY input clock
clk_ch2_p_i	I	Positive differential Rx Ch0 D-PHY input clock

Signal	Direction	Description
clk_ch2_n_i	I	Negative differential Rx Ch0 D-PHY input clock
clk_ch3_p_i	I	Positive differential Rx Ch1 D-PHY input clock
clk_ch3_n_i	I	Negative differential Rx Ch1 D-PHY input clock
d0_ch0_p_i	I/O	Positive differential Rx Ch0 D-PHY input data 0
d0_ch0_n_i	I/O	Negative differential Rx Ch0 D-PHY input data 0
d1_ch0_p_i	I/O	Positive differential Rx Ch0 D-PHY input data 1
d1_ch0_n_i	I/O	Negative differential Rx Ch0 D-PHY input data 1
d0_ch1_p_i	I/O	Positive differential Rx Ch1 D-PHY input data 0
d0_ch1_n_i	I/O	Negative differential Rx Ch1 D-PHY input data 0
d1_ch1_p_i	I/O	Positive differential Rx Ch1 D-PHY input data 1
d1_ch1_n_i	I/O	Negative differential Rx Ch1 D-PHY input data 1
d0_ch2_p_i	I/O	Positive differential Rx Ch2 D-PHY input data 0
d0_ch2_n_i	I/O	Negative differential Rx Ch2 D-PHY input data 0
d1_ch2_p_i	I/O	Positive differential Rx Ch2 D-PHY input data 1
d1_ch2_n_i	I/O	Negative differential Rx Ch2 D-PHY input data 1
d0_ch3_p_i	I/O	Positive differential Rx Ch3 D-PHY input data 0
d0_ch3_n_i	I/O	Negative differential Rx Ch3 D-PHY input data 0
d1_ch3_p_i	I/O	Positive differential Rx Ch3 D-PHY input data 1
d1_ch3_n_i	I/O	Negative differential Rx Ch3 D-PHY input data 1
CSI-2 Tx Interface		
clk_p_o	O	Positive differential Tx D-PHY output clock
clk_n_o	O	Negative differential Tx D-PHY output clock
d0_n_o	O	Positive differential Tx D-PHY output data 0
d0_p_o	O	Negative differential Tx D-PHY output data 0
d1_n_o	O	Positive differential Tx D-PHY output data 1
d1_p_o	O	Negative differential Tx D-PHY output data 1
Miscellaneous Debug Ports		
mux0_sp_en_o	O	Short packet enable from Rx channel 0 or 2
mux0_lp_en_o	O	Long packet enable from Rx channel 0 or 2
mux0_hs_sync_o	O	Signal indicator that the sync pattern has been detected by the dphy2cmos0/2
mux1_sp_en_o	O	Short packet enable from Rx channel 1 or 3
mux1_lp_en_o	O	Long packet enable from Rx channel 1 or 3
mux1_hs_sync_o	O	Signal indicator that the sync pattern has been detected by the dphy2cmos1/3
tx_pll_lock_o	O	Tx PLL lock indicator
tinit_done_o	O	Indicates that slave initialization period is over
tx_byte_data_vld	O	Data valid signal indicator from byte_data_gen going to Tx channel cmos2dphy

2.1. Design and Module Description

The top level design *(csi2_to_csi2_ip_wrapper.v)* consists of the following modules:

- Reset synchronizer to synchronize reset deassertion with the Rx byte clock
- Instance of the *csi2_to_csi2_ip* module

Within the *csi2_to_csi2_ip.v* block are instances of:

- Double flip-flop synchronizer.v to synchronize reset deassertion with the Tx byte clock. It also waits for the Tx tinit_done signal assertion before deasserting the output reset signal.
- dphy_2_cmos_ip.v
- line_buffer.v
- tx_byte_data_gen.v
- arbiter_fsm.v
- hdr_buffer.v
- cmos_2_dphy.v

2.1.1. dphy_2_cmos_ip

There are four instances of this module, one for each Rx channel.

This module includes the programmable I/O D-PHY wrapper (*dphy_rx_wrap.v*). It converts the incoming serial data from D-PHY data lanes to 8-bit (8:1 gearing) or 16-bit (16:1 gearing) words per lane.

This module also instantiates the Rx global controller (*rx_global_ctrl.v*) which contains FSMs that detect the state transitions of the clock and data lanes. This requires a free running clock during initialization. This is clocked by the refclk_i in non-continuous Rx clock mode, or by the Rx D-PHY generated free running byteclk_fr in continuous Rx clock mode. As such, it is recommended that the minimum duration of MIPI D-PHY low-power states (T_{LPX}) should be at least two times the byte clock period.

The capture control module (*capture_ctrl.v*) is also instantiated. It checks for long and short packet types and strips off the trail and checksum.

In the continuous Rx D-PHY clock mode configuration, the free-running byte clock of Rx channels 2 and 3 comes from the byte clocks of Rx channels 0 and 1, respectively.

2.1.2. muxer

This module multiplexes between the two pairs of Rx channels.

2.1.3. line_buffer

This module stores the payload of a long packet. It contains a dual port FIFO large enough to store one video line. The depth should be configured to be deep enough to hold one complete video line. The width is 1 byte per lane (32 bits for 4-lane, 16 bits for 2-lane, and 8 bits for single lane configuration).

2.1.4. tx_byte_data_gen

This module merges the data from the two Rx line buffers in LR Merge configuration.

2.1.5. arbiter_fsm

In VC interleave, this module arbitrates packets coming from the paired Rx channels for transmission. In LR Merge, it checks the state of the cmos2dphy module before initiating reading from the buffers.

2.1.6. hdr_buffer

This module receives and stores packet header fields from the dphy_2_cmos_ip module. This assures that the packet sequence from each channel is maintained.

2.1.7. cmos_2_dphy

This module encapsulates the merged video data and control signals into packets (*pkt_header_csi2_2bg.v*). It creates the packet header, ECC for the footer and checksum bits (CRC16).

Within this module is the Tx Global Operations module (*tx_global_operation.v*) that controls the transitions of the Tx D-PHY clock and data lanes. The Tx Global Operations controls the HS request path and timing using parameters. Currently, LP-request, escape mode and turnaround path are not supported. This block follows the requirements from D-PHY Specification version 1.2 section 6 – operating modes for control and high-speed data transmission.

This module also contains the hard MIPI D-PHY Tx wrapper (*dci_wrapper.v*) that serializes the packet data. The Tx PLL is included with the hard D-PHY. In continuous Rx D-PHY clock mode, the reference clock of this PLL comes from the Rx Channel 0 byte clock.

2.1.8. pll_wrapper

This module is instantiated only for certain clock rates. This is used to generate a valid reference clock to the Tx PLL, in place of the free-running Rx byte clock. In non-continuous Rx clock mode, its own reference clock comes from the input refclk pin. In continuous Rx clock mode, the reference clock comes from Rx channel 0.

3. Parameter Settings

Table 3.1 lists the user parameters available for the Lattice 4:1 MIPI CSI-2 Bridge IP.

Table 3.1. 4:1 CSI-2 to CSI-2 IP Parameter Settings

Parameter	Attribute	Options	Description
Number of Rx Lanes	User-Input	1, 2	Selects the number of Rx D-PHY data lanes.
Rx Gear	Read-Only	Gear8	Input serial bits are converted into 8-bit data bus.
Rx D-PHY Clock Mode	User-Input	Continuous or Non-continuous	In continuous mode, the input D-PHY clock lanes are always in high-speed mode. The DSI to DSI bridge utilizes this clock to generate the Rx byte clock for internal logic. The Rx byte clock is also used as the reference clock of the Tx PLL. In non-continuous mode, the input D-PHY clock lanes go to low-power states. As such, an external clock, refclk_i, is needed. The refclk_i is used as the reference clock of the Tx PLL. The generated byte clock is also used to clock the logic of the Rx channels.
Lane Aligner FIFO Type	User-Input	EBR or LUT	If the lane aligner is enabled, the FIFOs of the lane aligner modules are implemented as EBR by default. This can be changed to LUT to reallocate EBR resources.
Enable Lane Aligner	User-Input	ON or OFF	The word aligner modules are sufficient to handle data lane skews within 1 byte clock period. Option to use lane aligner is available in case the input to the Rx D-PHY have more than 1 byte clock period of skew.
Line Buffer Depth	User-Input	256 , 512, 1024 or 2048	The width of each line buffers is the number of bits transferred within one Rx byte clock (8 bits*lane_width). Each line buffer must be deep enough to store one complete pixel line. This is set to 2048 by default, but may be adjusted to a lower value to free up EBR resources.
Number of Tx Lanes	Read-Only	1, 2	The number of Tx data lanes is always the same as the number of Rx data lanes.
Tx Gear	User-Input	Gear8 or Gear16	The Tx channel (cmos2dphy module) translates the input data to 8-bit or 16-bit data bus respectively. Selecting Tx gear 16 results in lower Tx byte clock, thus saving some power, while Tx gear 8 results in smaller logic resources. Tx line rates greater than 1200 Mb/s automatically sets the Tx gear to 16.
Tx D-PHY Clock Mode	User-Input	Continuous or Non-continuous	In continuous mode, the output D-PHY clock lane is always in high-speed mode. In non-continuous mode, the output D-PHY clock lane switches to low-power states.
Rx Line Rate	Read Only	80-750	Rx line rate in Mb/s. This is always half of the Tx line rate.
Tx Line Rate	User-Input	160-1200 (Tx gear16) 320–1500 (Tx gear8)	Target Tx line rate in Mb/s.
tINIT_SLAVE Value	User-Input	1–32767	This parameter can be configured to ensure that the initialization period of the D-PHY slave is met. The D-PHY specification places a minimum period of 100 µs, but this parameter may be increased depending on the receiver requirement. During this period, all incoming data is ignored by the bridge.
Merging Options	User-Input	Left-Right Merge	The bridge merges payloads of long packets from both left and right channels to form a single packet. Short packets such as frame start and frame end only come from Rx channel 0. Data type from both channels should be the same.

Parameter	Attribute	Options	Description
t_HS-PREPARE	User-Input	1–99	This sets the $T_{HS-PREPARE}$ in terms of number of byte clock cycles. It is the time the transmitter drives the Data Lane LP-00 Line state immediately before the HS-0 Line state starting the HS transmission
t_HS-ZERO	User-Input	1–99	This sets the $T_{HS-ZERO}$ in terms of number of byte clock cycles. It is the time the transmitter drives the HS-0 state prior to transmitting the Sync sequence.
t_CLK-PRE	User-Input	1–99	This sets the $T_{CLK-PRE}$ in terms of number of byte clock cycles. This is the time that the transmitter drives the HS clock prior to any associated Data Lane beginning the transition from LP to HS mode.
t_CLK-POST	User-Input	1–99	This sets the $T_{CLK-POST}$ in terms of number of byte clock cycles. This is the time that the transmitter 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 $T_{HS-TRAIL}$ to the beginning of $T_{CLK-TRAIL}$.
Enable Miscellaneous Status Signals	User-Input	Enable Disable	If this parameter is enabled, internal signals are ported out for debug purposes. The number of signals that can be ported out is limited by the number of available PIO pins.

4. IP Generation and Evaluation

This section provides information on how to generate the Lattice 4:1 MIPI CSI-2 Bridge 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 4:1 MIPI CSI-2 Bridge IP in a complete, top level design. The 4:1 MIPI CSI-2 Bridge IP is available free of charge.

Request your license by going to the link http://www.latticesemi.com/en/Support/Licensing and request the free Lattice Diamond license. In this form, select the desired CrossLink IP for your design.

You may download or generate the 4:1 MIPI CSI-2 Bridge IP and fully evaluate the IP through functional simulation and implementation (synthesis, map, place and route) without the IP license. The 4:1 MIPI CSI-2 Bridge 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 for further details.

HOWEVER, THE 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 Lattice 4:1 MIPI CSI-2 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 user interface as shown in Figure 4.1.

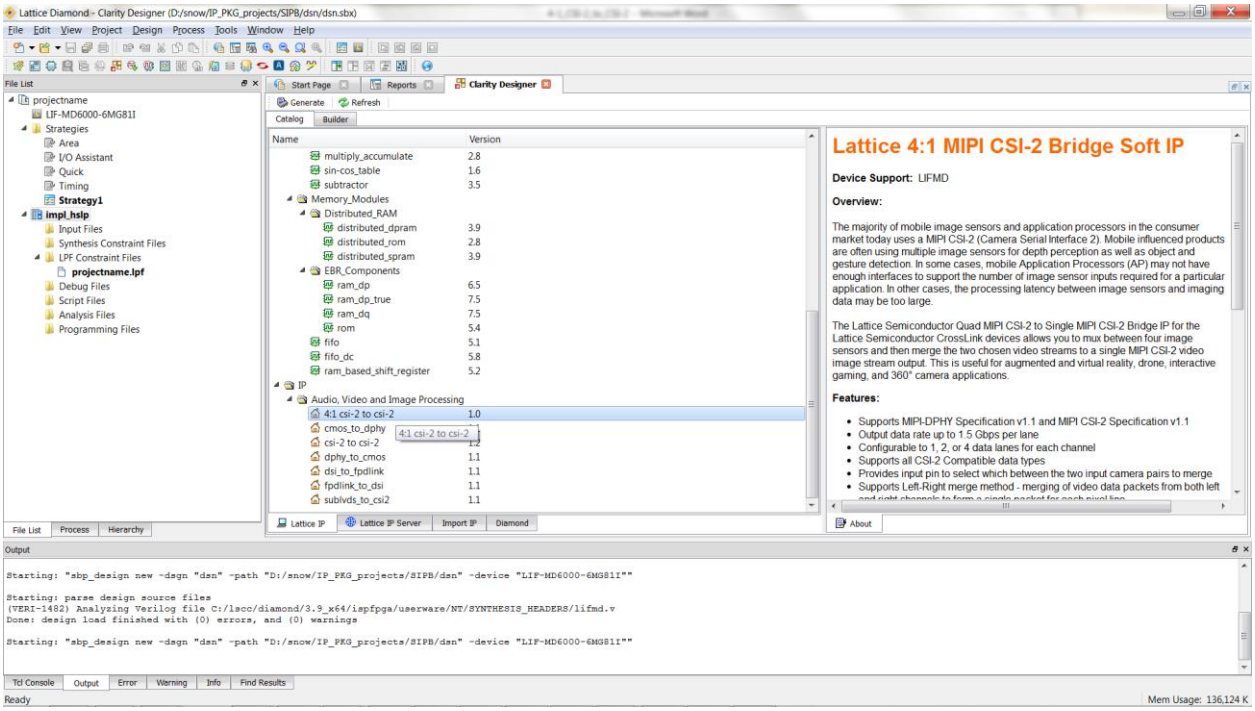

Figure 4.1. Clarity Designer Window

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.

The following describes the procedure for generating 4:1 MIPI CSI-2 Bridge IP in Clarity Designer.

Clarity Designer is started from the Diamond design environment.

To start Clarity Designer:

1. Create a new empty Diamond project for LIFMD 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** – Choose to create a new Clarity Design project directory in which the CSI-2 to CSI-2 will be generated.
   - **Design Location** – Clarity Design project directory path.
   - **Design Name** – Clarity Design project name.
   - **HDL Output** – Hardware Description Language Output Format. Select **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.

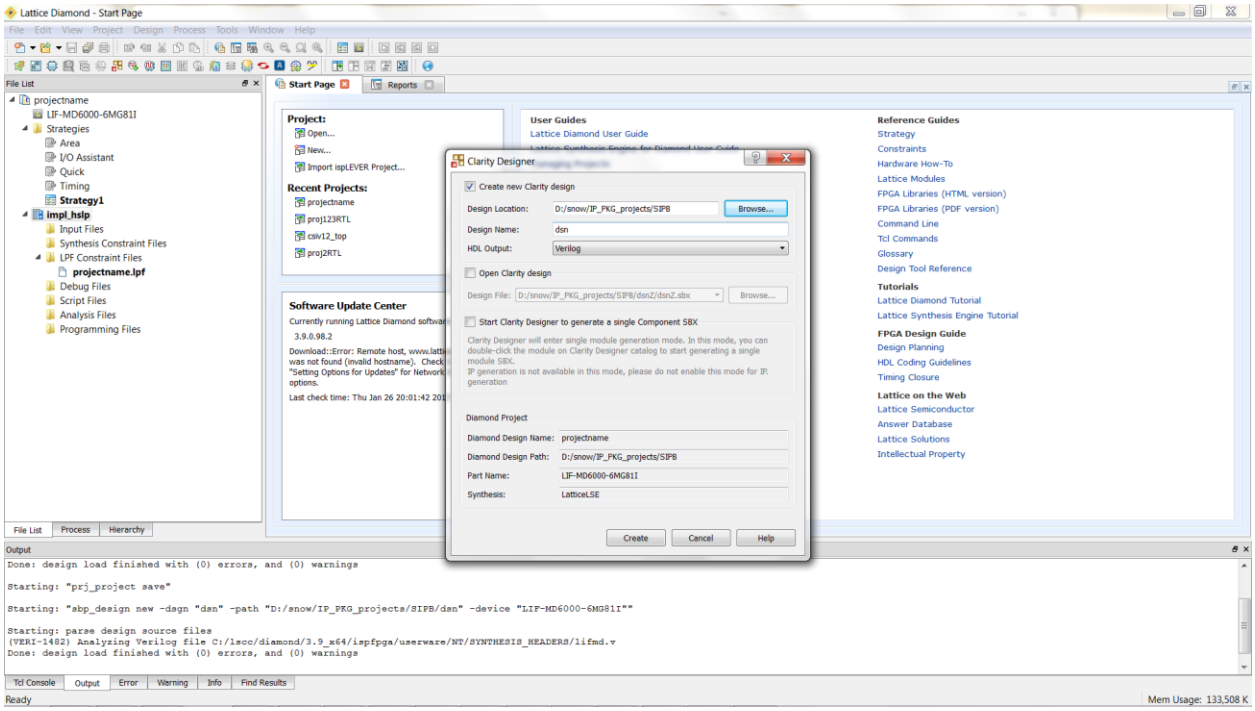

Figure 4.2. Starting Clarity Designer from Diamond Design Environment

To configure the CSI-2 to CSI-2 IP in Clarity Designer:

1. Double-click **CSI-2 to CSI-2** in the IP list of the Catalog view. The CSI2_to_CSI2 dialog box is displayed as shown in Figure 4.3.

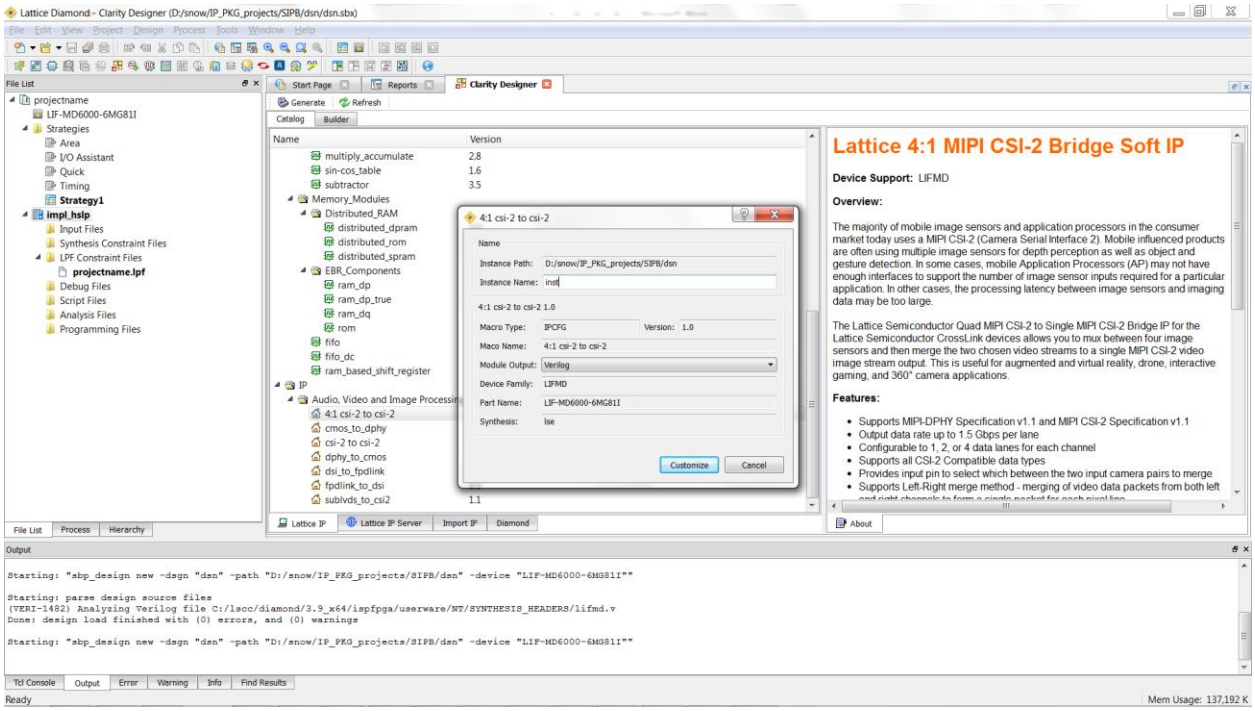

Figure 4.3. Configuring CSI-2 to CSI-2 Bridge IP in Clarity Designer

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

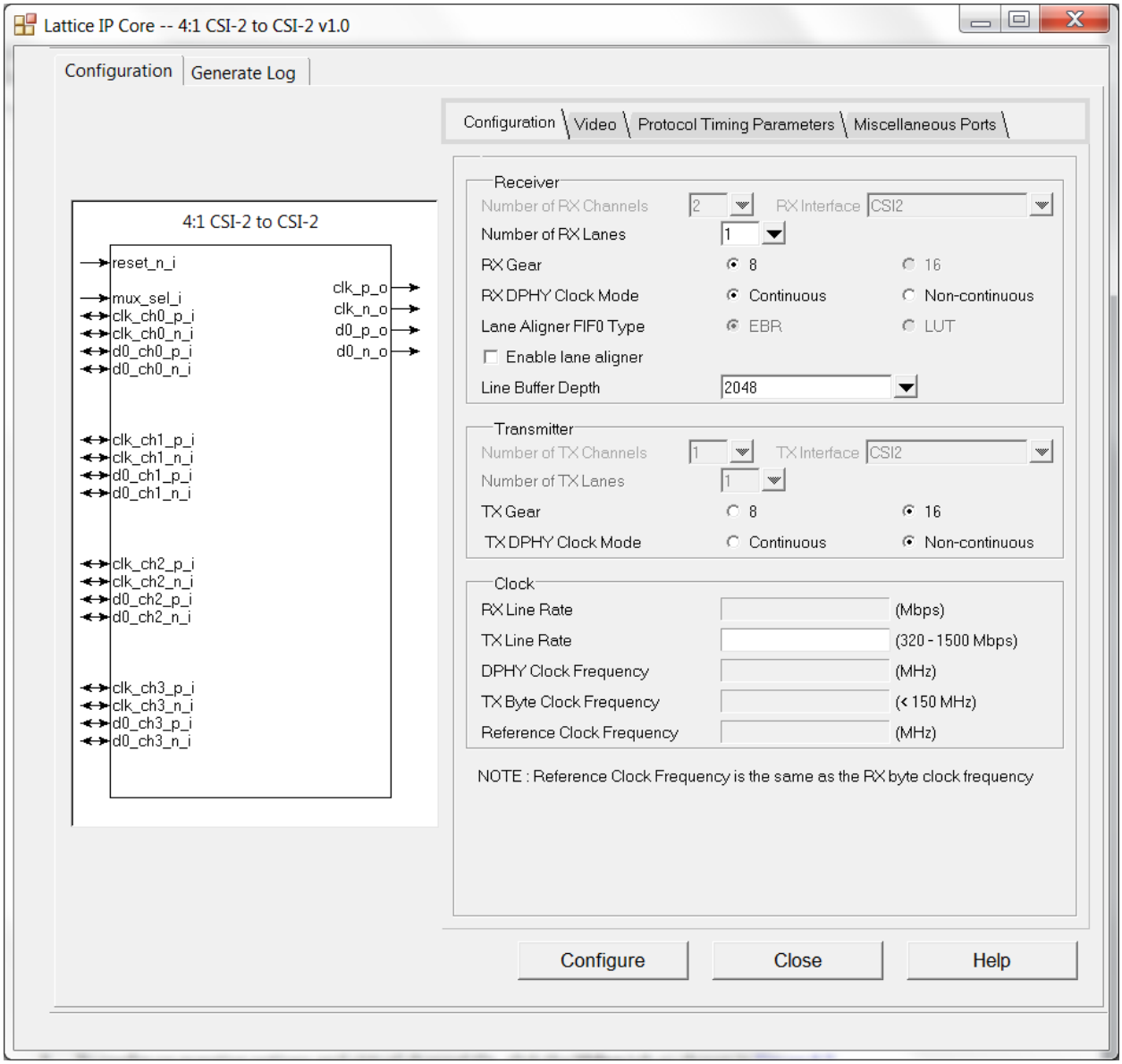

Figure 4.4. Configuration Tab in IP User Interface

5. To configure merging options and virtual channel IDs, click the **Video** tab as shown in Figure 4.5.

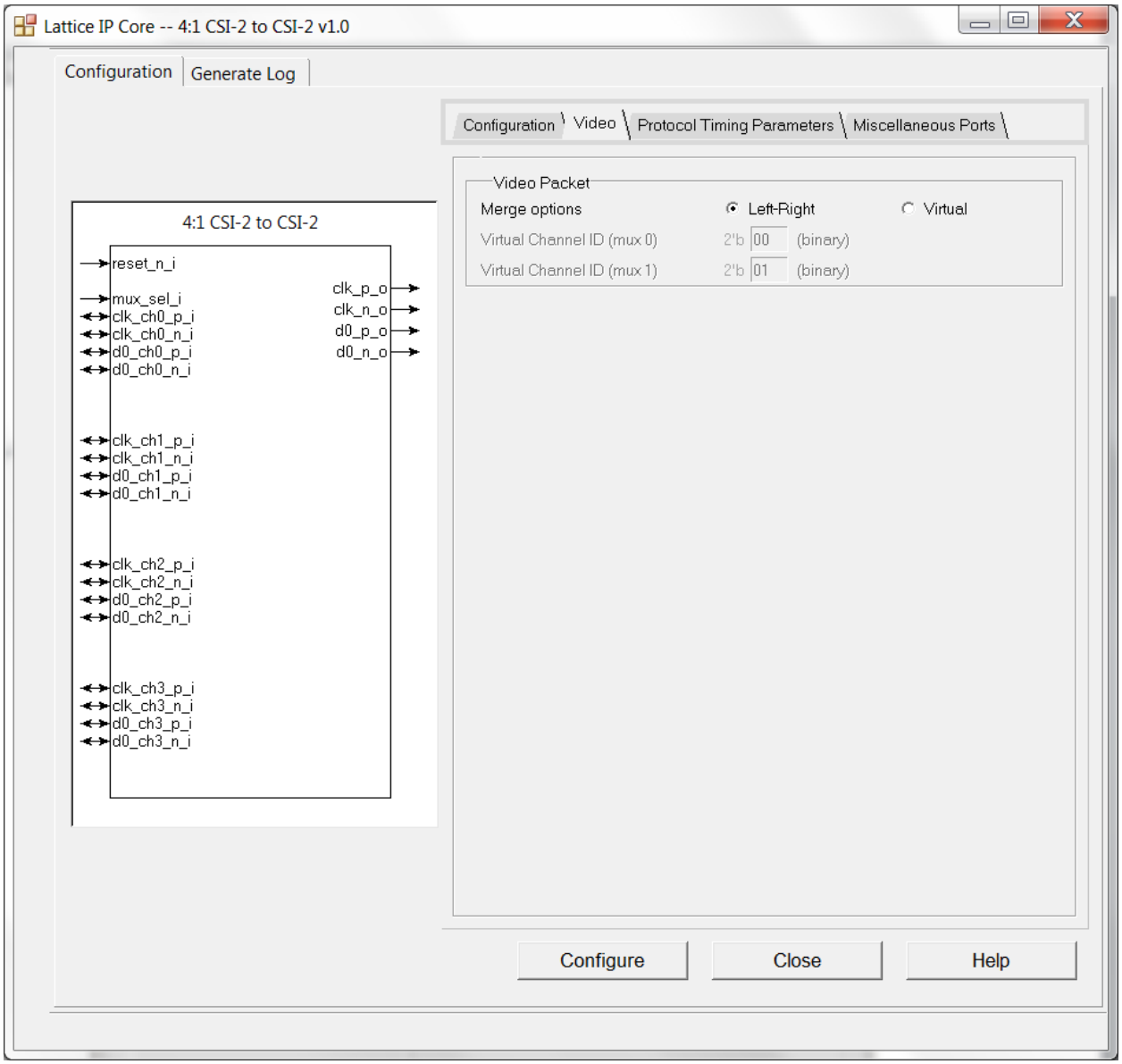

Figure 4.5. Video Tab in IP User Interface

6. To modify D-PHY timing parameters, click the **Protocol Timing Parameters** tab as shown in Figure 4.6.

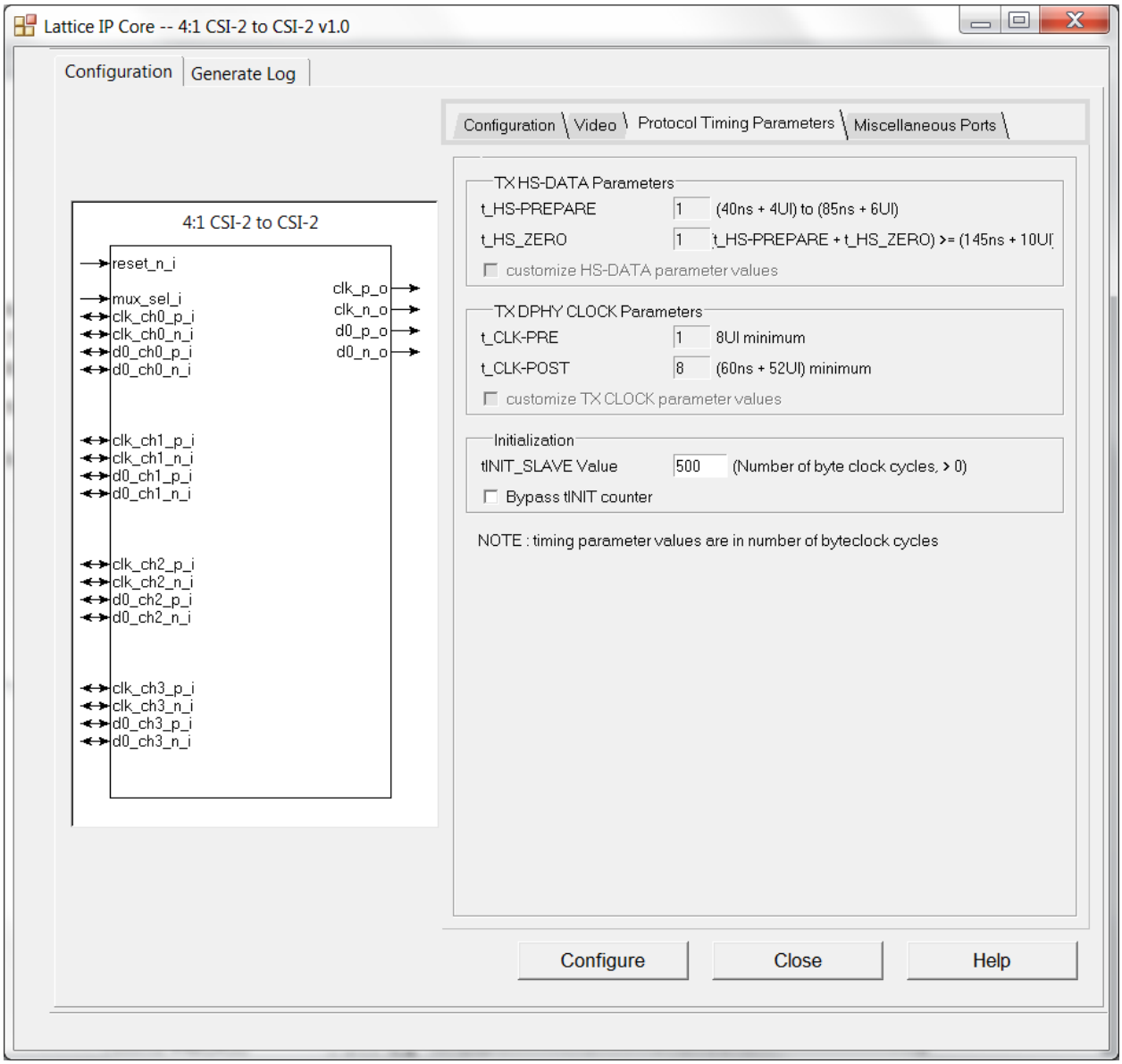

Figure 4.6. Protocol Timing Parameters Tab in IP User Interface

7. To add debug pins to monitor internal signals, click the **Miscellaneous Ports** tab as shown in Figure 4.7. The number of available pins may vary depending on the configuration and device package.

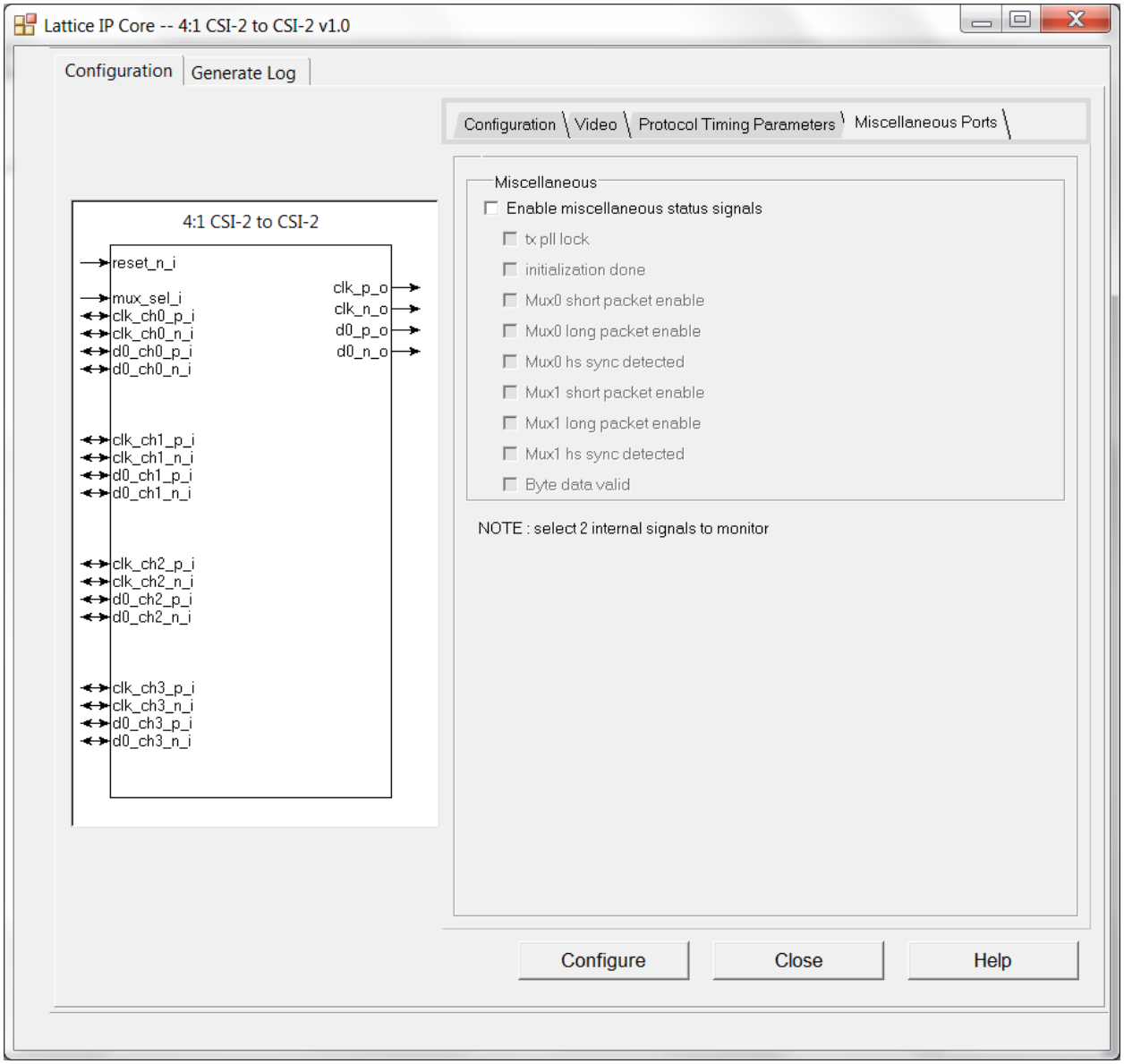

Figure 4.7. Miscellaneous Ports Tab in IP User Interface

8. Select the required parameters, and click the **Configure** button.
9. Click **Close**.
10. 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.

4.4. Generated IP Directory Structure and Files

The IP and supporting files generated in Clarity Designer and IP Express have similar folder architecture and files. The directory structure of the generated files is shown in Figure 4.8.

Figure 4.8. CSI-2 to CSI-2 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. Besides the post-synthesized module (NGO) and protected simulation model, all other files are 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 MIPI D-PHY to CMOS 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>_inst.v/vhd	Template for Instantiating the generated soft IP top-level in another user-created top module.
<instance_name>_*_beh.v	Protected Verilog models for simulation.
<instance_name>_*_bb.v	Verilog black box modules for synthesis.
<instance_name>_*.ngo	User interface 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 user interface in Clarity Designer when it is being reconfigured.

Aside from the files listed in the tables, most of the files required to evaluate the 4:1 MIPI CSI-2 Bridge IP are available under the directory *\csi2_2_csi2_qm_eval*. This includes the simulation model, testbench and simulation script files for running the simulation in Active HDL.

The *\<instance_name>* folder 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 *\csi2_2_csi2_qm_eval* and subdirectories provide files supporting 4:1 MIPI CSI-2 Bridge IP evaluation that includes files/folders with content that is constant for all configurations of the 4:1 MIPI CSI-2 Bridge IP. The *\csi2_2_csi2_qm_eval* directory is created by Clarity Designer the first time the IP is generated when multiple CSI-2 to CSI-2 IPs are generated in the same root directory and updated each time the IP is regenerated.

The simulation part of the user evaluation provides testbench and test cases supporting RTL simulation for Active-HDL simulators under *<project root>\testbench*.

Separate directories located at *\<project_dir>\csi2_2_csi2_qm_eval\<instance_name>\sim\Aldec\rtl* are provided and contain pre-built simulation script files. See Running Functional Simulation section below for details.

4.5. Running Functional Simulation

The functional simulation includes a configuration-specific behavioral model of the 4:1 MIPI CSI-2 Bridge, which is instantiated in an FPGA top level along with some other logic (such as PLLs and registers with Read/Write Interface). This FPGA top is instantiated in an evaluation testbench that provides appropriate stimulus for the 4:1 MIPI CSI-2 Bridge. The testbench files are provided in *<project_dir>\csi2_to_csi2_qm_eval\testbench*.

The generated IP package includes the configuration-specific behavior model (*<instance_name>_beh.v*, provided in *<project_dir>\csi2_to_csi2_eval\<instance_name>\src\beh_rtl\<family>*) for functional simulation. Models for simulation are provided in the corresponding \models folder if required.

To run the simulation in Active-HDL (Windows only):

1. Modify the ***.do** file located in **\<project_dir>\<IPinstance_name>\<instance_name>_eval\<IPinstance_name>\sim\aldec**.
   a. Specify the working directory (sim_working_folder). For example, set sim_working_folder **C:/my_design**.
   b. Specify the workspace name that will be created in the working directory. For example, set workspace_name **design_space**.
   c. Specify the design name. For example, set design_name **DesignA**.
   d. Specify the design path where the IP Core generated using Clarity Designer is located. For example, set design_path **C:/my_designs/DesignA**.
   e. Specify the design instance name (same as the instance name specified in Clarity Designer). For example, set design_inst DesignA_inst.
   f. Specify the Lattice Diamond Primitive path (diamond_dir) to where it is installed. For example, set diamond_dir **C:/lscc/diamond/3.9_x64**.
   g. Update the testbench parameters to customize data size, clock, and/or other settings. See Table 4.2 for the list of valid testbench compiler directives.
2. From the **Tools** menu, select **Active-HDL**.
3. In Active-HDL window, under **Tools** tab, select **Execute macro**.
4. Select the ***.do** file.
5. Click **OK**.
6. 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+NUM_FRAMES=60 \
+define+NUM_LINES=1080 \
....
```

You can override the default timing parameters using the below information.

Table 4.2. Testbench Directives

Directive	Description
NUM_PIXELS	Number of pixels per line.
NUM_LINES	Number of lines per frame.
NUM_FRAMES	Number of frames to be transmitted.
DPHY_LPX	Timing parameters (in ps) are set to minimum based on MIPI Specifications.
DPHY_CLK_PREPARE	
DPHY_CLK_ZERO	
DPHY_CLK_TRAIL	
DPHY_CLK_PRE	
DPHY_CLK_POST	
DPHY_HS_PREPARE	
DPHY_HS_ZERO	
DPHY_HS_TRAIL	
DPHY_INIT_DRIVE_DELAY	Delay from reset deassertion or tinit_done assertion before ch0 starts driving data. If tinit_done signal does not exist, testbench skips waiting for tinit_done signal. The tester can adjust this parameter to meet the initialization before driving data.
CH0_INIT_DELAY CH1_INIT _DELAY CH2_INIT _DELAY CH3_INIT _DELAY	Number of byte clocks before the specified channel starts transmitting data. Default is 0.
DT_RAW8	Used to set the data transaction to RAW8.
DT_RAW10	Used to set the data transaction to RAW10. This is the default value if data type is not specified.
DT_RAW12	Used to set the data transaction to RAW12.
DT_RGB888	Used to set the data transaction to RGB888.
DT_YUV420_10	Used to set the data transaction to YUV420_10.
DT_YUV420_8	Used to set the data transaction to YUV420_8.
DT_YUV422_10	Used to set the data transaction to YUV422_10.
DT_YUV422_8	Used to set the data transaction to YUV422_8.
PIX_CLK	Clock period in ps; used as reference clock.
DPHY_CLK_PERIOD	Used to set the D-PHY clock frequency of the design.
DPHY_LPS_GAP	Used to inject LPS delay (in ps) in between HS transactions 0-3.
DPHY_FRAME_GAP	Used to inject LPS delay (in ps) in between frames.
VC_CH0 VC_CH1 VC_CH2 VC_CH3	Used to set virtual channel of D-PHY CSI-2 models 0-3.
LS_LE_EN	Used to enable D-PHY model transmission of LS/LE short packets.

4.6. Simulation Strategies

This section describes the simulation environment which demonstrates basic 4:1 MIPI CSI-2 Bridge IP functionality.

Figure 4.9 shows the block diagram of simulation environment.

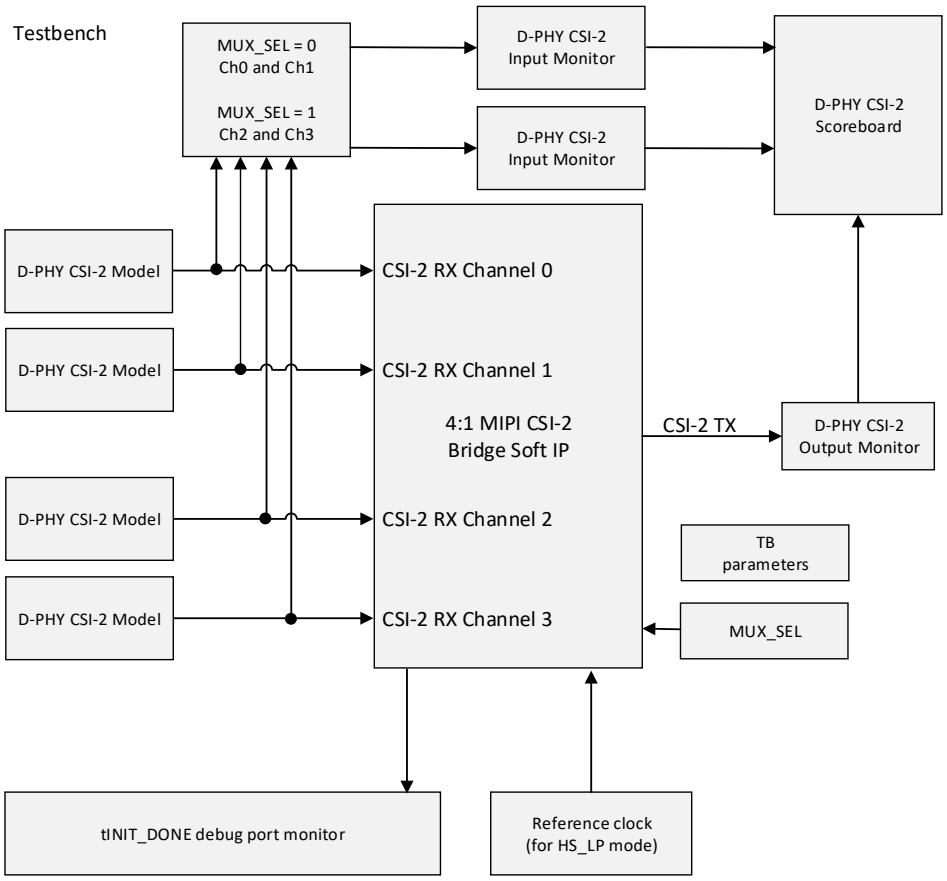

Figure 4.9. Simulation Environment Block Diagram

4.7. Simulation Environment

The simulation environment is made up of the D-PHY CSI-2 model instances connected to the input of the IP instance in the testbench. The D-PHY CSI-2 models are configured based on the IP core configurations and testbench configurations. You can also adjust testbench parameters to modify parameters such as D-PHY timing, data type to be transmitted, enabling of LS/LE transmission, and so on.

The testbench transmits reference clock to the CSI2_2_CSI2 IP core if clock mode is non-continuous (HS_LP). The MUX_SEL selects which D-PHY channels will transmit data to the output. Depending on the selector, the monitors sample data from Ch0 and Ch1 if MUX_SEL=0, otherwise the monitors sample data from Ch2 and Ch3. The monitors then pass the information to the scoreboard for checking. The scoreboard only performs data comparison, there are no protocol/timing checks included. The testbench waits for tinit_done or pll_lock (if miscellaneous signals are available) or waits for some time before transmitting data to the IP core.

Figure 4.10 shows a sample simulation where tINIT done port is included.

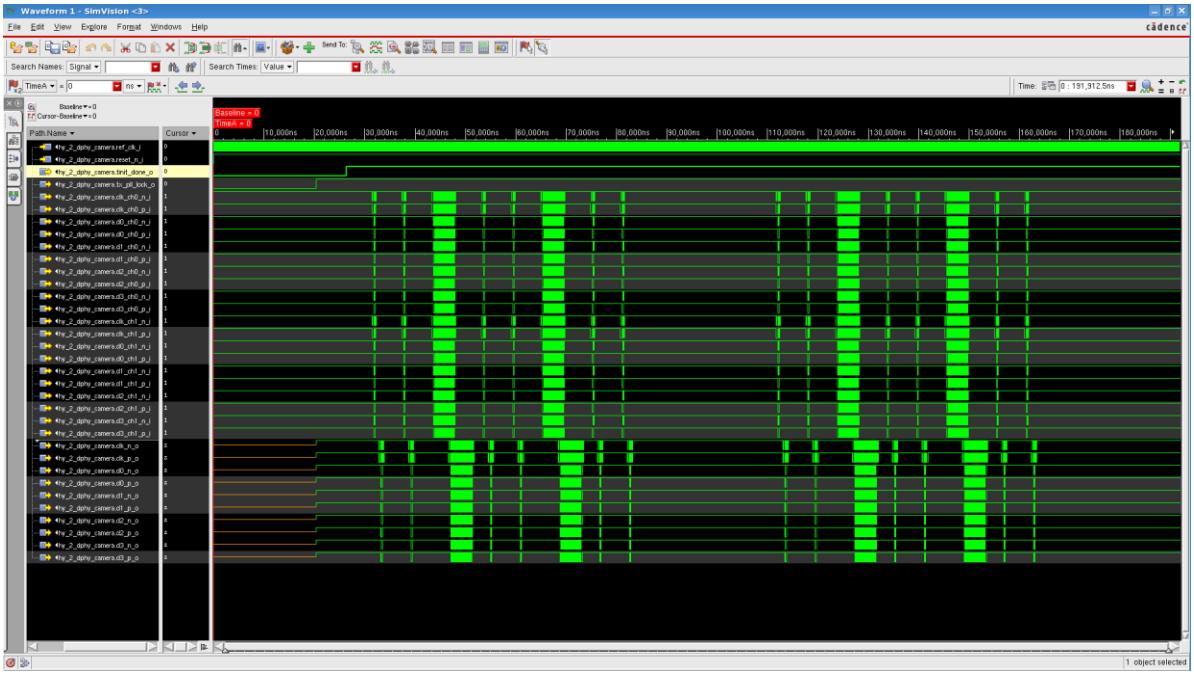

Figure 4.10. tinit_done Assertion Before Start of Transaction

The data transmitted by the D-PHY CSI-2 model can be viewed in the waveform (see Figure 4.11):

- tb.dphy_ch0.data0 – Refers to the data bytes transmitted in D-PHY data lane 0
- tb.dphy_ch0.data1 – Refers to the data bytes transmitted in D-PHY data lane 1
- tb.dphy_ch0.data2 – Refers to the data bytes transmitted in D-PHY data lane 2
- tb.dphy_ch0.data3 – Refers to the data bytes transmitted in D-PHY data lane 3

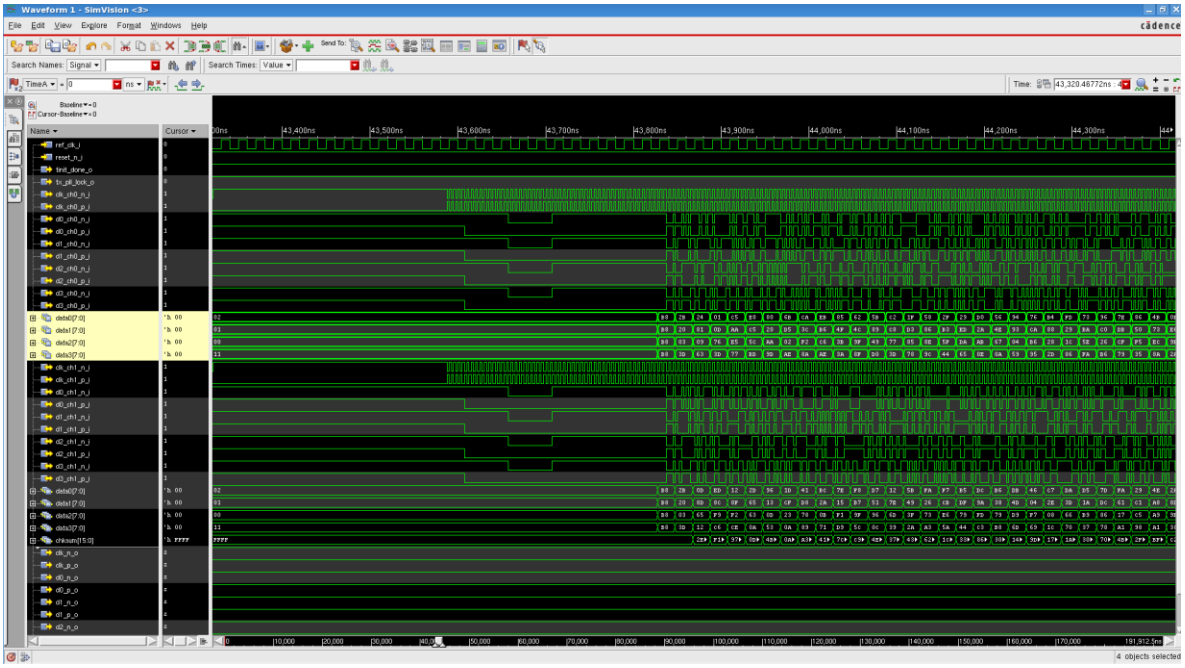

Figure 4.11. D-PHY CSI-2 Model Data

4.8. Instantiating the IP

The core modules of the 4:1 MIPI CSI-2 to CSI-2 Interface Bridge IP are synthesized and provided in NGO format with black box Verilog source files for synthesis. A Verilog source file named *<instance_name>_csi2_2_csi2_ip.v* instantiates the black box of core modules. The top-level file *<instance_name>.v* instantiates *<instance_name>_csi2_2_csi2_ip.v.*

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

For example, if the Clarity Designer project file is *cdprj.sbx*, the automatically generated wrapper file is *cdprj.v* in which all generated IPs are instantiated. You do not need to instantiate the IP instances one by one manually. The *cdprj.v* is refreshed each time the IPs in the design are 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 a Top-Level Design

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 the default Diamond strategy (.sty) and default Diamond preference file (.lpf) are used. When using .sbx approach, import the recommended strategy and preferences from *\<project_dir>\csi2_2_csi2_qm_eval\<instance_name>\impl\lifmd\[lse | 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 IPs are generated.

To use the pre-built Diamond project files:

1. Choose **File > Open > Project**.
2. Browse to **\<project_dir>\csi2_to_csi2_qm_eval\<instance_name>\impl\lifmd\[lse|synplify]** in the **Open Project** dialog box.
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 user interface window.
5. Implement the complete design via the standard Diamond user interface flow.

4.10. Hardware Evaluation

The Lattice 4:1 MIPI CSI-2 Bridge Soft IP supports Lattice's IP hardware evaluation capability, which makes it possible to create versions of the IP that operate in hardware for a limited period of time (approximately four hours) 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 user 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** or **Planner** tab, right-click the IP instance to be regenerated and select **Config** in the menu as shown in Figure 4.12.
2. The IP Configuration user interface is displayed. Change the parameters as required and click the **Configure** button.
3. Update the pin connection in **Builder** tab for configuration changes.
4. Click 

in the toolbox. Clarity Designer regenerates all the instances which are reconfigured.

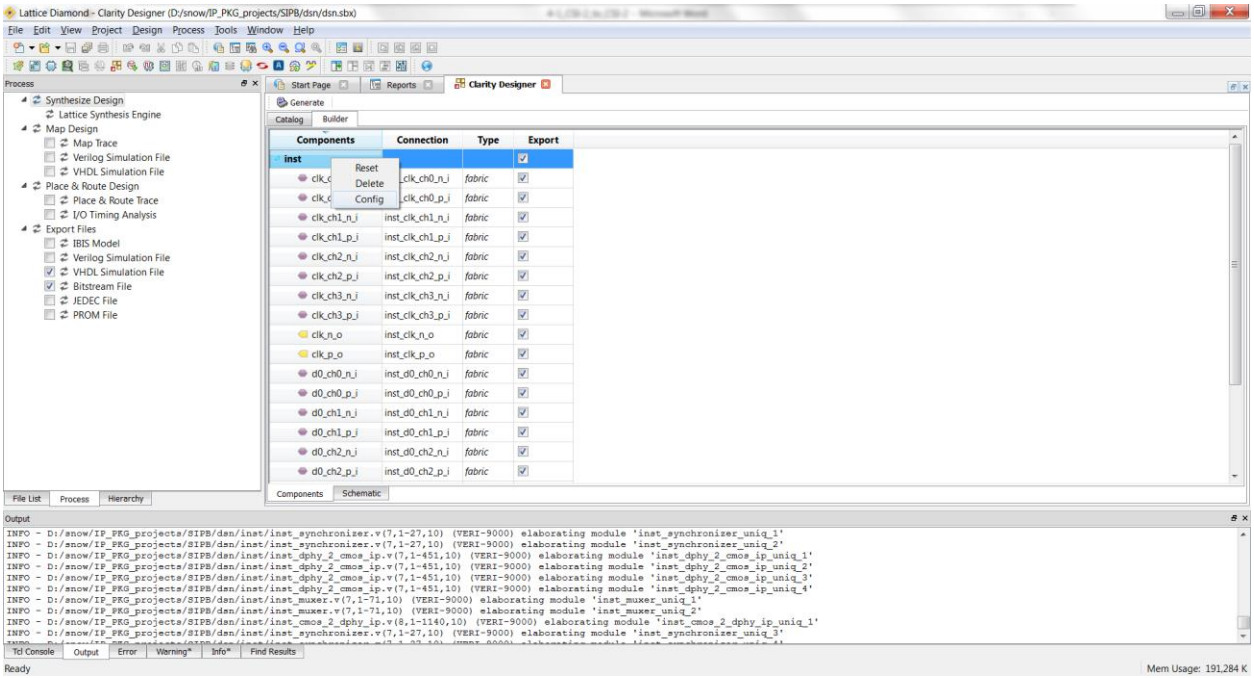

Figure 4.12. Regenerating IP in Clarity Designer

References

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

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 Camera Serial Interface 2 (CSI-2) version 1.1, July 18, 2012, www.mipi.org

Technical Support Assistance

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

Appendix A. Resource Utilization

This appendix provides resource utilization information for Lattice FPGAs using the 4:1 MIPI CSI-2 Bridge IP.

LIF-MD6000 Utilization

Table A.1 lists the resource utilization for Lattice CrossLink FPGAs using the 4:1 MIPI CSI-2 Bridge IP with default values.

The performance and utilization data target an LIF-MD6000-6MG81I device using Lattice Diamond 3.9 and LSE software. Performance may vary when using different software version or targeting a different device density or speed grade within the CrossLink family. The values of f_{MAX} shown are based on byte clock. The Target f_{MAX} column shows the target byte clock frequency for each configuration.

Table A.1. Resource Utilization

IP User-Configurable Parameters	Slices	LUTs	Registers	sysMEM EBRs	Actual f_{MAX} (MHz)	Target f_{MAX} (MHz)
Tx gear 16, 1-lane	1896	2624	1677	4	120.89	93.75
Tx gear 16, 2-lane	2465	3922	2313	8	100.573	93.75

Appendix B. What is Not Supported

- The 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 error detection and correction.
  - Checksum error detection
  - Camera Control Interface (CCI) communication
  - Long packets with zero payload, including Null, are not supported.
- All long packet payloads are merged, including payloads from Null and blanking packets.
- The word count of the resulting transmit packet must be in multiples of the number of lanes multiplied by the number of bytes transmitted per byte clock. See Table B.1 for the supported word counts for each configuration.

Table B.1. Supported Tx Word Count

Tx Gear	Number of Lanes	Output Word Count Multiple
16	4	8
16	2	4
16	1	2
8	4	4
8	2	2
8	1	1

Revision History

Revision 1.1, IP Version 1.0, April 2019

Section	Change Summary
Introduction	Specified that this user guide can be used for IP design versions 1.x.
IP Generation and Evaluation	In Licensing the IP, modified the instructions for requesting free license.
Appendix B. What is Not Supported	Revised grouping of items.
Revision History	Updated revision history table to new template.
All	Minor adjustments in style and formatting.

Revision 1.0, IP Version 1.0, February 2017

Section	Change Summary
All	Initial release.

www.latticesemi.com