MIPI CSI-2 Virtual Channel Aggregation

Reference Design

FPGA-RD-02051-1.2

September 2019

Disclaimers

Lattice makes no warranty, representation, or guarantee regarding the accuracy of information contained in this document or the suitability of its products for any particular purpose. All information herein is provided AS IS and with all faults, and all risk associated with such information is entirely with Buyer. Buyer shall not rely on any data and performance specifications or parameters provided herein. Products sold by Lattice have been subject to limited testing and it is the Buyer's responsibility to independently determine the suitability of any products and to test and verify the same. No Lattice products should be used in conjunction with mission- or safety-critical or any other application in which the failure of Lattice's product could create a situation where personal injury, death, severe property or environmental damage may occur. The information provided in this document is proprietary to Lattice Semiconductor, and Lattice reserves the right to make any changes to the information in this document or to any products at any time without notice.

Contents

Figures

Tables

Acronyms in This Document

A list of acronyms used in this document.

Acronym	Definition
AP	Application Processor
CSI-2	Camera Serial Interface 2
DDR	Double Data Rate
ECC	Error Correction Code
HS	High Speed
ID	Identification Data
LP	Low Power
MIPI	Mobile Industry Processor Interface
PLL	Phase Locked Loop
GPLL	General Purpose PLL
RX	Receiver
TDM	Time Domain Multiplexing
TX	Transmitter
VC	Virtual Channel

Supported Device and IP

This reference design supports the following devices with IP versions.

Device Family	Part Number	Compatible IP
CrossLink	LIF-MD6000 LIA-MD6000	D-PHY Receiver IP version 1.2 and 1.3 D-PHY Transmitter IP version 1.1 and 1.2
CrossLinkPlus	LIF-MDF6000	D-PHY Receiver IP version 1.3 D-PHY Transmitter IP version 1.2

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

1. Introduction

The majority of image sensors and application processors (AP) in the consumer market use the Mobile Industry Processor Interface (MIPI®) Camera Serial Interface 2 (CSI-2) as a video signal interface. In some cases, the AP has to take multiple image data for various applications without increasing the physical interface signals.

The Lattice Semiconductor MIPI CSI-2 Virtual Channel Aggregation reference design for CrossLink™ devices offers up to five-channel aggregation. A different virtual channel identification (ID) is assigned to each receiver (RX) channel. CrossLink has two MIPI hard macro IPs, which can be used as MIPI TX or RX module (D-PHY Hard IP). The RX module can also be realized by a soft macro utilizing general DDR modules (D-PHY Soft IP).

1.1. Features List

- Two to five independent RX channels can be aggregated.
- You can assign a unique virtual channel ID (0 to 15) to each RX channel.
- Each RX channel can have one, two, or four lanes as long as the total number of RX clock and data lanes by RX D-PHY Soft IP does not exceed 15.
- Number of TX lanes can be one, two, or four.
- Maximum TX bandwidth is 1.5 Gbps per lane.
- Non-continuous clock mode on RX channels is possible as long as the continuous clock can be obtained internally or fed directly from the pin. Each RX clock can be independent and does not have to come from the same clock source.
- Reference clock may be required in case a proper clock is not derived from RX side to drive TX D-PHY PLL.

1.2. Block Diagram

Figure 1.1 shows the block level diagram of the MIPI CSI-2 Virtual Channel Aggregation reference design with five RX channels. It is not recommended to use Hard D-PHY for RX channel due to IP issues until the new version (1.4) of RX D-PHY IP is released.

Since TX D-PHY PLL has an input clock frequency requirement of between 24 MHz and 30 MHz (or a multiple of between 24 and 30 MHz), another on-chip GPLL may have to be used to create an appropriate clock.

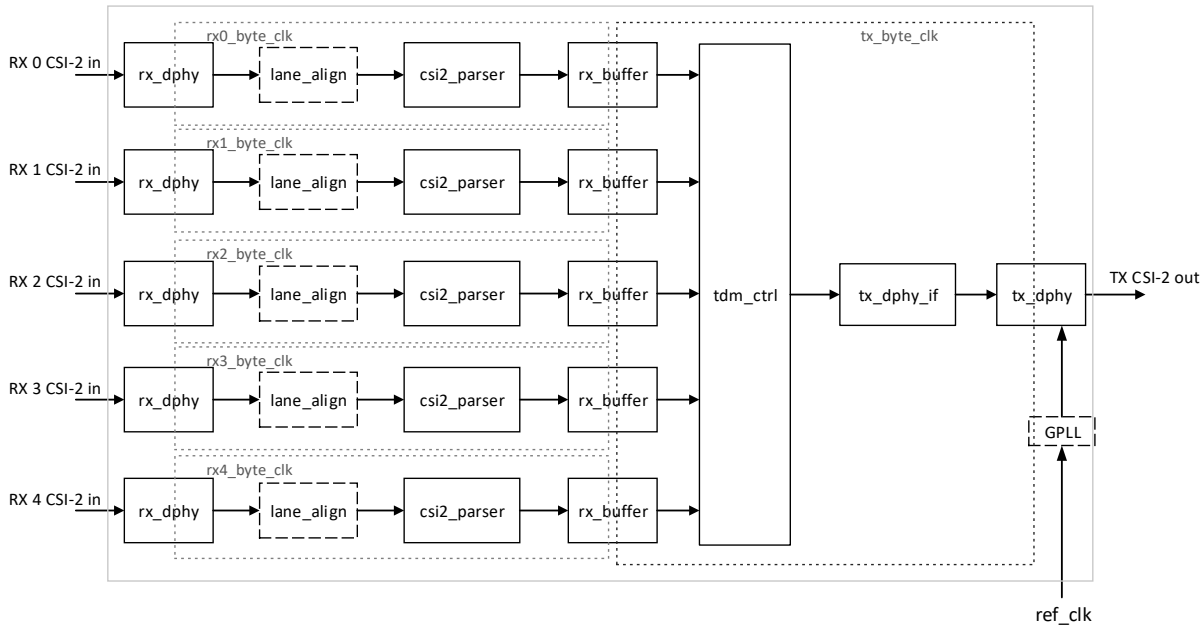

Figure 1.1. CSI-2 Virtual Channel Aggregation Block Diagram

2. Parameters and Port List

There are two directive files for this reference design:

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

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

2.1. Synthesis Directives

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

Table 2.1. Synthesis Directives

Category	Directive	Remarks
External reference clock	EXT_REF_CLK	Enable this when the reference clock is fed from a pin.
RX channel count	NUM_RX_CH_2	Number of RX channels aggregated. Only one of these four directives must be defined.
	NUM_RX_CH_3	
	NUM_RX_CH_4	
	NUM_RX_CH_5	
RX channel lane count	NUM_RX0_LANE_1	Number of lanes in RX channel 0. Only one of these three directives must be defined.
	NUM_RX0_LANE_2	
	NUM_RX0_LANE_4	
	NUM_RX1_LANE_1	Number of lanes in RX channel 1. Only one of these three directives must be defined.
	NUM_RX1_LANE_2	
	NUM_RX1_LANE_4	
	NUM_RX2_LANE_1	Number of lanes in RX channel 2. Only one of these three directives must be defined. Effective when RX channel count is 3 or more.
	NUM_RX2_LANE_2	
	NUM_RX2_LANE_4	
	NUM_RX3_LANE_1	Number of lanes in RX channel 3. Only one of these three directives must be defined. Effective when RX channel count is 4 or more.
	NUM_RX3_LANE_2	
	NUM_RX3_LANE_4	
	NUM_RX4_LANE_1	Number of lanes in RX channel 4. Only one of these three directives must be defined. Effective when RX channel count is 5.
	NUM_RX4_LANE_2	
	NUM_RX4_LANE_4	
RX D-PHY Clock Gear	RX0_GEAR_8	Only one of these directives must be selected. Gear 16 can be used for only 1- and 2-lane configurations.
	RX0_GEAR_16	
	RX1_GEAR_8	Only one of these directives must be selected. Gear 16 can be used for only 1- and 2-lane configurations.
	RX1_GEAR_16	
	RX2_GEAR_8	Only one of these directives must be selected. Gear 16 can be used for only 1- and 2-lane configurations.
	RX2_GEAR_16	
	RX3_GEAR_8	Only one of these directives must be selected. Gear 16 can be used for only 1- and 2-lane configurations.
	RX3_GEAR_16	
	RX4_GEAR_8	Only one of these directives must be selected. Gear 16 can be used for only 1- and 2-lane configurations.
	RX4_GEAR_16	
RX Hard D-PHY channel	RX0_DPHY_HARD	Specify RX channel that uses Hard D-PHY. Only one of these five directives must be defined. If none of these is defined, all RX channels use Soft D-PHY (up to four channels). Do not define any of these until the new version (1.4) of RX D-PHY IP is released.
	RX1_DPHY_HARD	
	RX2_DPHY_HARD	
	RX3_DPHY_HARD	
	RX4_DPHY_HARD	

Category	Directive	Remarks
RX D-PHY Clock Mode[1]	RX0_CLK_MODE_HS_ONLY	RX D-PHY Clock mode on channel 0. Only one of these two directives must be defined.
	RX0_CLK_MODE_HS_LP	
	RX1_CLK_MODE_HS_ONLY	RX D-PHY Clock mode on channel 1. Only one of these two directives must be defined.
	RX1_CLK_MODE_HS_LP	
	RX2_CLK_MODE_HS_ONLY	RX D-PHY Clock mode on channel 2. Only one of these two directives must be defined.
	RX2_CLK_MODE_HS_LP	
	RX3_CLK_MODE_HS_ONLY	RX D-PHY Clock mode on channel 3. Only one of these two directives must be defined.
	RX3_CLK_MODE_HS_LP	
	RX4_CLK_MODE_HS_ONLY	RX D-PHY Clock mode on channel 4. Only one of these two directives must be defined.
	RX4_CLK_MODE_HS_LP	
Hard D-PHY word alignment	WORD_ALIGN	Enable word aligner in RX Hard D-PHY IP. Ignored when Hard D-PHY is not used.
RX lane alignment[2]	RX0_LANE_ALIGN	Enable lane aligner when lane count is 2 or 4.
	RX1_LANE_ALIGN	Enable lane aligner when lane count is 2 or 4.
	RX2_LANE_ALIGN	Enable lane aligner when lane count is 2 or 4.
	RX3_LANE_ALIGN	Enable lane aligner when lane count is 2 or 4.
	RX4_LANE_ALIGN	Enable lane aligner when lane count is 2 or 4.
Frame Start Detection	TX_WAIT_LESS_15MS	Always enable this directive
VC ID pass through[3]	RX0_VC_PASS_THROUGH	Enable when VC ID is passed through without change.
	RX1_VC_PASS_THROUGH	Enable when VC ID is passed through without change.
	RX2_VC_PASS_THROUGH	Enable when VC ID is passed through without change.
	RX3_VC_PASS_THROUGH	Enable when VC ID is passed through without change.
	RX4_VC_PASS_THROUGH	Enable when VC ID is passed through without change.
Multiple VC ID	RX0_MULTI_VC	Enable when multiple VC ID exist on this channel.
	RX1_MULTI_VC	Enable when multiple VC ID exist on this channel.
	RX2_MULTI_VC	Enable when multiple VC ID exist on this channel.
	RX3_MULTI_VC	Enable when multiple VC ID exist on this channel.
	RX4_MULTI_VC	Enable when multiple VC ID exist on this channel.
RX channel VC value[4]	RX0_NEW_VC {value}	Channel 0 Virtual Channel ID; 0 – 15
	RX1_NEW_VC {value}	Channel 1 Virtual Channel ID; 0 – 15
	RX2_NEW_VC {value}	Channel 2 Virtual Channel ID; 0 – 15
	RX3_NEW_VC {value}	Channel 3 Virtual Channel ID; 0 – 15
	RX4_NEW_VC {value}	Channel 4 Virtual Channel ID; 0 – 15
RX Channel Frame Counter[5]	RX0_FRAME_COUNT	Enable when Data Field of Frame Start/End Short Packet is replaced with the Frame Counter value.
	RX1_FRAME_COUNT	Enable when Data Field of Frame Start/End Short Packet is replaced with the Frame Counter value.
	RX2_FRAME_COUNT	Enable when Data Field of Frame Start/End Short Packet is replaced with the Frame Counter value.
	RX3_FRAME_COUNT	Enable when Data Field of Frame Start/End Short Packet is replaced with the Frame Counter value.
	RX4_FRAME_COUNT	Enable when Data Field of Frame Start/End Short Packet is replaced with the Frame Counter value.
Maximum value of the frame counter	RX0_FRAME_COUNT_MAX {value}	Frame counter value goes back to 1 after it reaches this value. Effective when RX0_FRAME_COUNT is defined. Must be 2 – 65535.

Category	Directive	Remarks
	RX1_FRAME_COUNT_MAX {value}	Frame counter value goes back to 1 after it reaches this value. Effective when RX1_FRAME_COUNT is defined. Must be 2 – 65535.
	RX2_FRAME_COUNT_MAX {value}	Frame counter value goes back to 1 after it reaches this value. Effective when RX2_FRAME_COUNT is defined. Must be 2 – 65535.
	RX3_FRAME_COUNT_MAX {value}	Frame counter value goes back to 1 after it reaches this value. Effective when RX3_FRAME_COUNT is defined. Must be 2 – 65535.
	RX4_FRAME_COUNT_MAX {value}	Frame counter value goes back to 1 after it reaches this value. Effective when RX4_FRAME_COUNT is defined. Must be 2 – 65535.
RX Buffer Depth[6]	RX0_BUFFER_DEPTH_*	RX channel 0 FIFO Depth. * must be 512, 1024, or 2048.
	RX1_BUFFER_DEPTH_*	RX channel 1 FIFO Depth. * must be 512, 1024, or 2048.
	RX2_BUFFER_DEPTH_*	RX channel 2 FIFO Depth. * must be 512, 1024, or 2048.
	RX3_BUFFER_DEPTH_*	RX channel 3 FIFO Depth. * must be 512, 1024, or 2048.
	RX4_BUFFER_DEPTH_*	RX channel 4 FIFO Depth. * must be 512, 1024, or 2048.
TX channel lane count	NUM_TX_LANE_1	Number of lanes in TX channel. Only one of these three directives must be defined.
	NUM_TX_LANE_2	
	NUM_TX_LANE_4	
TX D-PHY Clock Gear	TX_GEAR_8	TX D-PHY Clock Gear. Only one of these two directives must be defined.
	TX_GEAR_16	
TX D-PHY Clock Mode	TX_CLK_MODE_HS_ONLY	TX D-PHY Clock mode. Only one of these two directives must be defined.
	TX_CLK_MODE_HS_LP	

Notes:

1. HS_LP mode means *non-continuous clock mode* and HS_ONLY means *continuous clock mode*. HS_LP mode works only if RX byte clock for corresponding RX channel can be generated internally or directly fed from I/O pin.
2. This enables instantiating a lane aligner that is different from the one in RX D-PHY IP. You must disable the lane aligner in RX D-PHY IP when it is created by Clarity in Diamond.
3. RX*_VC_PASS_THROUGH must be defined when RX*_MULTI_VC is defined. You cannot assign the same VC value on different RX channels when this is defined.
4. Incoming VC values on RX CSI-2 data are overwritten by these VC values when RX*_VC_PASS_THROUGH is not defined. Values 4 and above are only supported by CSI-2 version 2.0 and above. If the opponent device supports only CSI-2 version 1.1, VC values of 4-15 should not be used.
5. When this is defined, the Data Field is replaced with the 16-bit counter value which begins with 1 after power on/reset and goes back to 1 after it reaches RX*_FRAME_COUNT_MAX. The Data Field is passed through when this is not defined.
6. This value affects necessary EBR used in the device. Number of necessary EBR per RX channel is
   (BUFFER_DEPTH/512) × 4 (in case of NUM_TX_LANE_4 and TX_GEAR_16), or
   (BUFFER_DEPTH/512) × 2 (others).
   Total number of EBR must not exceed 20.

2.2. Simulation Directives

Table 2.2 shows the simulation directives for this reference design.

Table 2.2. Simulation Directives

Category	Directive	Remarks
Simulation	SIM	Select behavioral models for simulation.
Reference clock period	REF_CLK_PERIOD {value}	Reference clock period in ps
RX D-PHY clock period	RX0_DPHY_CLK_PERIOD {value}	RX DPHY clock period on Channel 0 in ps
	RX1_DPHY_CLK_PERIOD {value}	RX DPHY clock period on Channel 1 in ps
	RX2_DPHY_CLK_PERIOD {value}	RX DPHY clock period on Channel 2 in ps
	RX3_DPHY_CLK_PERIOD {value}	RX DPHY clock period on Channel 3 in ps
	RX4_DPHY_CLK_PERIOD {value}	RX DPHY clock period on Channel 4 in ps
RX Byte clock frequency	RX0_FREQ_TGT {value}	RX byte clock frequency on Channel 0 in MHz
	RX1_FREQ_TGT {value}	RX byte clock frequency on Channel 1 in MHz
	RX2_FREQ_TGT {value}	RX byte clock frequency on Channel 2 in MHz
	RX3_FREQ_TGT {value}	RX byte clock frequency on Channel 3 in MHz
	RX4_FREQ_TGT {value}	RX byte clock frequency on Channel 4 in MHz
TX byte clock frequency	TX_FREQ_TGT {value}	TX byte clock frequency in MHz
TX D-PHY clock period	TX_DPHY_CLK_PERIOD {value}	TX DPHY clock period in ps
Frame Start Detection	TX_WAIT_LESS_15MS	Always enable this directive.
VC value on RX channel	VC_CH0 {value}	VC value on incoming RX Channel 0; 0 – 3
	VC_CH1 {value}	VC value on incoming RX Channel 1; 0 – 3
	VC_CH2 {value}	VC value on incoming RX Channel 2; 0 – 3
	VC_CH3 {value}	VC value on incoming RX Channel 3; 0 – 3
	VC_CH4 {value}	VC value on incoming RX Channel 4; 0 – 3
Frame Number in Frame Start/End Short Packets	CH0_FNUM_EMBED	Frame number is embedded in Data Field of Frame Start/End Short packet. The value “0” is filled when undefined.
	CH1_FNUM_EMBED	Frame number is embedded in Data Field of Frame Start/End Short packet. The value “0” is filled when undefined.
	CH2_FNUM_EMBED	Frame number is embedded in Data Field of Frame Start/End Short packet. The value “0” is filled when undefined.
	CH3_FNUM_EMBED	Frame number is embedded in Data Field of Frame Start/End Short packet. The value “0” is filled when undefined.
	CH4_FNUM_EMBED	Frame number is embedded in Data Field of Frame Start/End Short packet. The value “0” is filled when undefined.
Maximum value of Frame Number	CH0_FNUM_MAX {value}	The maximum value of the frame number; 2 - 65535
	CH1_FNUM_MAX {value}	The maximum value of the frame number; 2 – 65535
	CH2_FNUM_MAX {value}	The maximum value of the frame number; 2 – 65535
	CH3_FNUM_MAX {value}	The maximum value of the frame number; 2 – 65535
	CH4_FNUM_MAX {value}	The maximum value of the frame number; 2 - 65535
Initial delay on RX channel	CH0_DELAY {value}	Initial delay to activate RX Channel 0 in ps
	CH1_DELAY {value}	Initial delay to activate RX Channel 1 in ps
	CH2_DELAY {value}	Initial delay to activate RX Channel 2 in ps
	CH3_DELAY {value}	Initial delay to activate RX Channel 3 in ps
	CH4_DELAY {value}	Initial delay to activate RX Channel 4 in ps
Gap (LP) time between active lines on RX Channel	CH0_DPHY_LPS_GAP {value}	Gap time on RX Channel 0 in ps
	CH1_DPHY_LPS_GAP {value}	Gap time on RX Channel 1 in ps
	CH2_DPHY_LPS_GAP {value}	Gap time on RX Channel 2 in ps
	CH3_DPHY_LPS_GAP {value}	Gap time on RX Channel 3 in ps
	CH4_DPHY_LPS_GAP {value}	Gap time on RX Channel 4 in ps

Category	Directive	Remarks
Gap (LP) time between Frame End and Frame Start on RX Channel	CH0_DPHY_FRAME_GAP {value}	Gap time on RX Channel 0 in ps
	CH1_DPHY_FRAME_GAP {value}	Gap time on RX Channel 1 in ps
	CH2_DPHY_FRAME_GAP {value}	Gap time on RX Channel 2 in ps
	CH3_DPHY_FRAME_GAP {value}	Gap time on RX Channel 3 in ps
	CH4_DPHY_FRAME_GAP {value}	Gap time on RX Channel 4 in ps
Video data configuration on RX Channel 0	CH0_NUM_FRAMES {value}	Number of frames to feed
	CH0_NUM_LINES {value}	Number of active lines per frame
	CH0_NUM_PIXELS {value}	Number of pixels per line
	CH0_DT_*	Data Type; * must be RAW8, RAW10, RAW12, RGB888, YUV420_10, YUV420_8, YUV422_10, or YUV422_8.
Video data configuration on RX Channel 1	CH1_NUM_PIXELS {value}	Number of pixels per line
	CH1_DT_*	Data Type; * must be RAW8, RAW10, RAW12, RGB888, YUV420_10, YUV420_8, YUV422_10, or YUV422_8.
	CH1_DT_*	Data Type; * must be RAW8, RAW10, RAW12, RGB888, YUV420_10, YUV420_8, YUV422_10, or YUV422_8.
	CH1_DT_*	Data Type; * must be RAW8, RAW10, RAW12, RGB888, YUV420_10, YUV420_8, YUV422_10, or YUV422_8.
Video data configuration on RX Channel 2	CH2_NUM_FRAMES {value}	Number of frames to feed
	CH2_NUM_LINES {value}	Number of active lines per frame
	CH2_NUM_PIXELS {value}	Number of pixels per line
	CH2_DT_*	Data Type; * must be RAW8, RAW10, RAW12, RGB888, YUV420_10, YUV420_8, YUV422_10, or YUV422_8.
Video data configuration on RX Channel 3	CH3_NUM_FRAMES {value}	Number of frames to feed
	CH3_NUM_LINES {value}	Number of active lines per frame
	CH3_NUM_PIXELS {value}	Number of pixels per line
	CH3_DT_*	Data Type; * must be RAW8, RAW10, RAW12, RGB888, YUV420_10, YUV420_8, YUV422_10, or YUV422_8.
Video data configuration on RX Channel 4	CH4_NUM_FRAMES {value}	Number of frames to feed
	CH4_NUM_LINES {value}	Number of active lines per frame
	CH4_NUM_PIXELS {value}	Number of pixels per line
	CH4_DT_*	Data Type; * must be RAW8, RAW10, RAW12, RGB888, YUV420_10, YUV420_8, YUV422_10, or YUV422_8.
Internal signal monitoring	MISC_ON	Enables internal signal monitored by the testbench. Always enable this directive.

2.3. Top-Level I/O

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

Table 2.3. CSI-2 VC Aggregation Top Level I/O

Port Name	Direction	Description
Clocks and Resets		
ref_clk_i (optional)	I	Input reference clock. Used to feed a clock to TX D-PHY PLL directly or indirectly. This port is declared only when EXT_REF_CLK is defined in synthesis_directives.v.
reset_n_i	I	Asynchronous active low system reset
CSI-2 RX Interface		
rx0_clk_p_i	I	Positive differential RX Ch0 D-PHY input clock
rx0_clk_n_i	I	Negative differential RX Ch0 D-PHY input clock
rx0_d0_p_i	I	Positive differential RX Ch0 D-PHY input data 0
rx0_d0_n_i	I	Negative differential RX Ch0 D-PHY input data 0
rx0_d1_p_i	I	Positive differential RX Ch0 D-PHY input data 1 (in case of 2-lane or 4-lane configuration)
rx0_d1_n_i	I	Negative differential RX Ch0 D-PHY input data 1 (in case of 2-lane or 4-lane configuration)
rx0_d2_p_i	I	Positive differential RX Ch0 D-PHY input data 2 (in case of 4-lane configuration)
rx0_d2_n_i	I	Negative differential RX Ch0 D-PHY input data 2 (in case of 4-lane configuration)
rx0_d3_p_i	I	Positive differential RX Ch0 D-PHY input data 3 (in case of 4-lane configuration)
rx0_d3_n_i	I	Negative differential RX Ch0 D-PHY input data 3 (in case of 4-lane configuration)
rx1_clk_p_i	I	Positive differential RX Ch1 D-PHY input clock
rx1_clk_n_i	I	Negative differential RX Ch1 D-PHY input clock
rx1_d0_p_i	I	Positive differential RX Ch1 D-PHY input data 0
rx1_d0_n_i	I	Negative differential RX Ch1 D-PHY input data 0
rx1_d1_p_i	I	Positive differential RX Ch1 D-PHY input data 1 (in case of 2-lane or 4-lane configuration)
rx1_d1_n_i	I	Negative differential RX Ch1 D-PHY input data 1 (in case of 2-lane or 4-lane configuration)
rx1_d2_p_i	I	Positive differential RX Ch1 D-PHY input data 2 (in case of 4-lane configuration)
rx1_d2_n_i	I	Negative differential RX Ch1 D-PHY input data 2 (in case of 4-lane configuration)
rx1_d3_p_i	I	Positive differential RX Ch1 D-PHY input data 3 (in case of 4-lane configuration)
rx1_d3_n_i	I	Negative differential RX Ch1 D-PHY input data 3 (in case of 4-lane configuration)
rx2_clk_p_i	I	Positive differential RX Ch2 D-PHY input clock
rx2_clk_n_i	I	Negative differential RX Ch2 D-PHY input clock
rx2_d0_p_i	I	Positive differential RX Ch2 D-PHY input data 0
rx2_d0_n_i	I	Negative differential RX Ch2 D-PHY input data 0
rx2_d1_p_i	I	Positive differential RX Ch2 D-PHY input data 1 (in case of 2-lane or 4-lane configuration)
rx2_d1_n_i	I	Negative differential RX Ch2 D-PHY input data 1 (in case of 2-lane or 4-lane configuration)
rx2_d2_p_i	I	Positive differential RX Ch2 D-PHY input data 2 (in case of 4-lane configuration)
rx2_d2_n_i	I	Negative differential RX Ch2 D-PHY input data 2 (in case of 4-lane configuration)
rx2_d3_p_i	I	Positive differential RX Ch2 D-PHY input data 3 (in case of 4-lane configuration)
rx2_d3_n_i	I	Negative differential RX Ch2 D-PHY input data 3 (in case of 4-lane configuration)
rx3_clk_p_i	I	Positive differential RX Ch3 D-PHY input clock
rx3_clk_n_i	I	Negative differential RX Ch3 D-PHY input clock
rx3_d0_p_i	I	Positive differential RX Ch3 D-PHY input data 0
rx3_d0_n_i	I	Negative differential RX Ch3 D-PHY input data 0
rx3_d1_p_i	I	Positive differential RX Ch3 D-PHY input data 1 (in case of 2-lane or 4-lane configuration)
rx3_d1_n_i	I	Negative differential RX Ch3 D-PHY input data 1 (in case of 2-lane or 4-lane configuration)
rx3_d2_p_i	I	Positive differential RX Ch3 D-PHY input data 2 (in case of 4-lane configuration)
rx3_d2_n_i	I	Negative differential RX Ch3 D-PHY input data 2 (in case of 4 lane configuration)

Port Name	Direction	Description
rx3_d3_p_i	I	Positive differential RX Ch3 D-PHY input data 3 (in case of 4 lane configuration)
rx3_d3_n_i	I	Negative differential RX Ch3 D-PHY input data 3 (in case of 4 lane configuration)
rx4_clk_p_i	I	Positive differential RX Ch4 D-PHY input clock
rx4_clk_n_i	I	Negative differential RX Ch4 D-PHY input clock
rx4_d0_p_i	I	Positive differential RX Ch4 D-PHY input data 0
rx4_d0_n_i	I	Negative differential RX Ch4 D-PHY input data 0
rx4_d1_p_i	I	Positive differential RX Ch4 D-PHY input data 1 (in case of 2-lane or 4-lane configuration)
rx4_d1_n_i	I	Negative differential RX Ch4 D-PHY input data 1 (in case of 2-lane or 4-lane configuration)
rx4_d2_p_i	I	Positive differential RX Ch4 D-PHY input data 2 (in case of 4-lane configuration)
rx4_d2_n_i	I	Negative differential RX Ch4 D-PHY input data 2 (in case of 4-lane configuration)
rx4_d3_p_i	I	Positive differential RX Ch4 D-PHY input data 3 (in case of 4-lane configuration)
rx4_d3_n_i	I	Negative differential RX Ch4 D-PHY input data 3 (in case of 4-lane configuration)
CSI-2 TX Interface		
tx_clk_p_o	O	Positive differential TX D-PHY output clock
tx_clk_n_o	O	Negative differential TX D-PHY output clock
tx_d0_p_o	O	Positive differential TX D-PHY output data 0
tx_d0_n_o	O	Negative differential TX D-PHY output data 0
tx_d1_p_o	O	Positive differential TX D-PHY output data 1 (in case of 2-lane or 4-lane configuration)
tx_d1_n_o	O	Negative differential TX D-PHY output data 1 (in case of 2-lane or 4-lane configuration)
tx_d2_p_o	O	Positive differential TX D-PHY output data 2 (in case of 4-lane configuration)
tx_d2_n_o	O	Negative differential TX D-PHY output data 2 (in case of 4-lane configuration)
tx_d3_p_o	O	Positive differential TX D-PHY output data 3 (in case of 4-lane configuration)
tx_d3_n_o	O	Negative differential TX D-PHY output data 3 (in case of 4-lane configuration)

3. Design and Module Description

The top-level design (csi2_aggregation_vc.v) consists of the following modules:

- rx_dphy
- lane_align
- csi2_parser
- rx_buffer
- tdm_ctrl
- tx_dphy_if
- tx_dphy

The top-level design has a reset synchronization logic. In addition, GPLL may be added if necessary according to RX and TX configurations.

3.1. rx_dphy

This module must be created for each RX channel according to channel conditions, such as the number of lanes, bandwidth, and others. Figure 3.1 shows an example of IP interface settings in Clarity for the CSI-2/DSI D-PHY Receiver Submodule IP. Create a separate rx_dphy IP for each RX channel even though their configurations are same. Refer to CSI-2/DSI D-PHY Receiver Submodule IP User Guide (FPGA-IPUG-02025) for details.

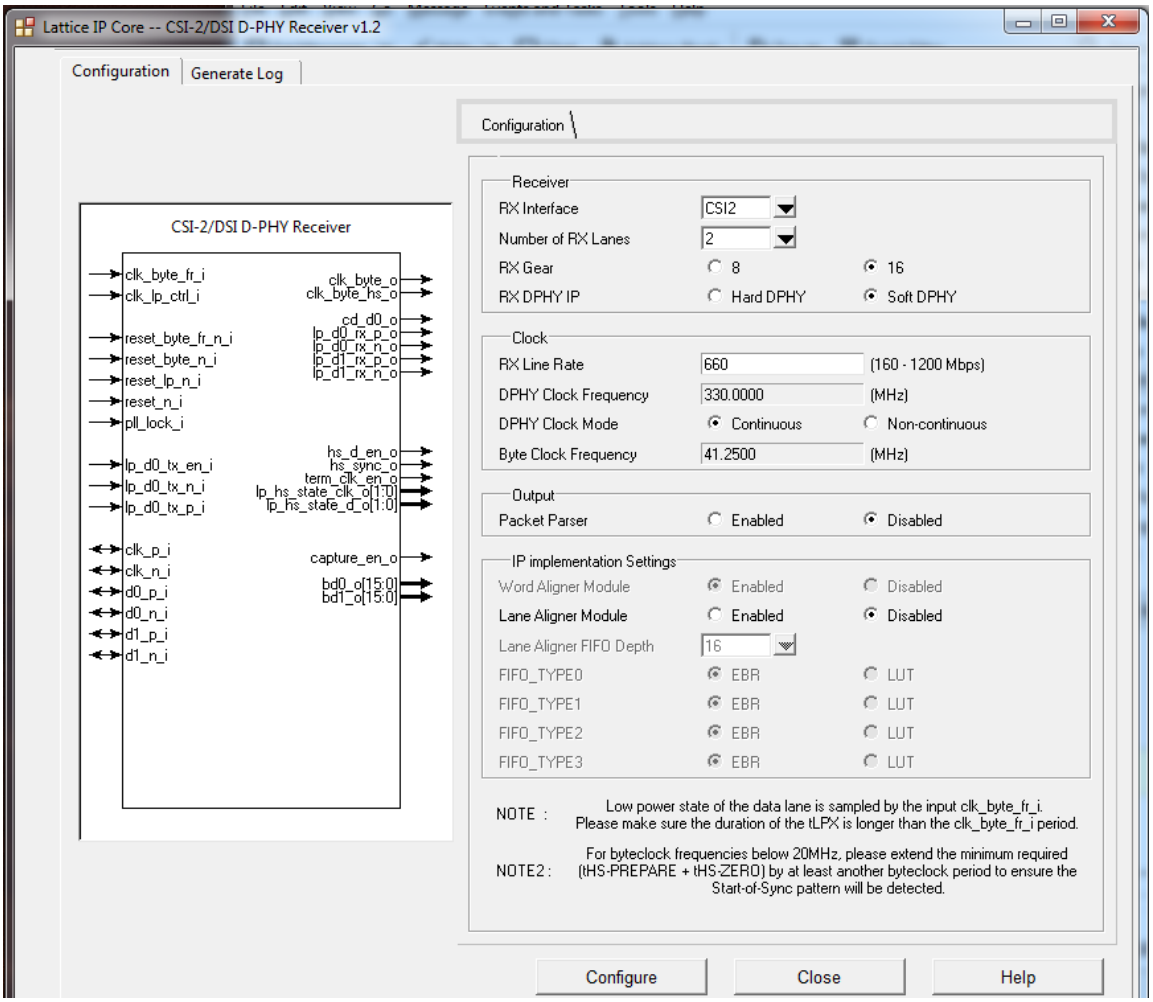

Figure 3.1. rx_dphy IP Creation in Clarity Designer

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

- RX Interface – Select CSI-2.
- Number of RX Lanes – Set according to channel configuration. The value must match NUM_RX*_LANE_* setting.
- RX Gear – Select 8 or 16; 16 is supported for 1-lane and 2-lane configurations only. 16 is recommended when the RX byte clock speed exceeds 100 MHz with Gear 8.
- RX D-PHY IP – Hard D-PHY can be selected only for one RX channel, others must be Soft D-PHY. The setting must match RX*_D-PHY_HARD setting. Use Soft D-PHY and avoid using Hard D-PHY as long as enough clock/data ports are available for Soft D-PHY.
- RX Line Rate – Set according to channel configuration. 800 or below is recommended for 4-lane configuration.
- D-PHY Clock Mode – Select Continuous or Non-continuous. Must match RX*_CLK_MODE_* setting (Continuous = HS_ONLY, Non-continuous = HS_LP).
- Packet Parser – Select Disabled.
- Word Aligner Module –Select Enabled.
- Lane Aligner Module – Select Disabled. The lane aligner is instantiated outside of this IP when RX*_LANE_ALIGN directive is defined.

This module takes serial CSI-2 data and outputs byte data after de-serialization in CSI-2 High Speed mode. It is recommended to set the design name to *rx_dphy* and module names to *rx_ch0, rx_ch1, …, rx_ch4* so that you do not need to modify the instance names of these IPs in the top-level design as well as the simulation setup file. Otherwise, you have to modify the names accordingly.

RX Gear 16 is supported for only 1-lane and 2-lane configurations.

Configuring the CSI-2/DSI D-PHY Receiver Submodule IP as Hard D-PHY may cause compilation and simulation issues. Therefore, it is not recommended unless the reference design configuration requires greater than 15 pairs of differential I/O (four RX channels with 4/4/4/4, 4/4/4/2, 4/4/4/1, or 4/4/2/2 lanes) or five RX channels.

3.2. lane_align

This module resides outside of the RX D-PHY IP and takes the byte data coming out from rx_dphy. It then checks D-PHY sync word (0xB8) to align the byte data timing among lanes when RX*LANE_ALIGN directive is defined. It is highly recommended to be enabled. A lane aligner in the RX D-PHY IP is available but should not be enabled. Use this lane aligner by defining RX*LANE_ALIGN in synthesis_directives.v.

3.3. csi2_parser

This module is instantiated for each RX channel to handle CSI-2 protocol decoding and VC overwrite. Upon receiving byte data from rx_dphy or lane_align, csi2_parser detects short and long packet headers. When RX*_VC_PASS_THROUGH is not defined, it replaces VC values with the value specified by RX*_NEW_VC and calculates ECC value based on this new VC value and other packet data. As a result, VC value and ECC are replaced with new values and sent to the next module (rx_buffer). Since VC values of 4 and above are only supported by CSI-2 version 2.x, using 0 to 3 would be safe in case of 2- to 4-channel aggregation. The downstream device must support CSI-2 version 2.x in case that VC = 4 or above is used. On the other hand, packet header data including VC values are not changed and passed through when RX*_VC_PASS_THROUGH is defined.

Figure 3.2 shows short packet capture and a new VC assignment when RX*VC_PASS_THROUGH is not defined. The original VC=0 is replaced with VC=1 along with new ECC value calculated applying the new VC. 16-bit Data Field ({bd2_i, bd1_i}) is replaced with the frame counter value if RX*FRAME_COUNT is defined. If not, the Data Field is passed through.

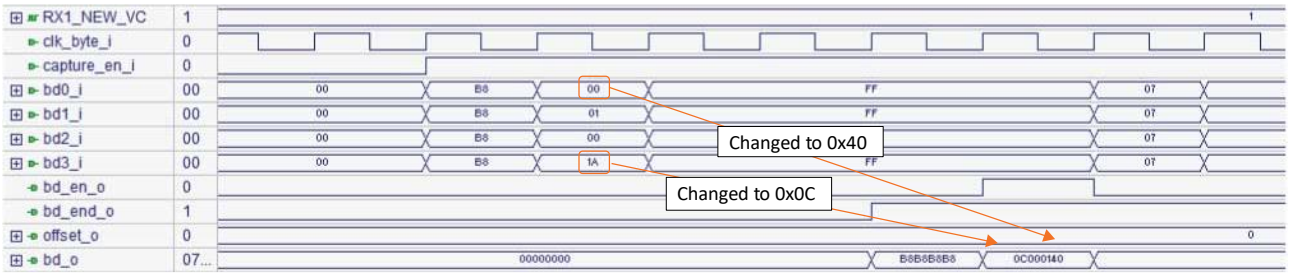

Figure 3.2. Short Packet Detection and VC Replacement

RX*_MULTI_VC must be defined when the RX channel in question carries multiple VC IDs (already VC aggregated). In that case, VC value replacement does not make sense and RX*_VC_PASS_THROUGH must be defined. RX*_MULTI_VC is useful when over 5:1 aggregation is necessary. It is possible to do 16:1 aggregation using 5 CrossLink devices with 4 CrossLink devices in the first stage, all taking 4 RX channels and 5th CrossLink device takes the output of 4 CrossLink to do another 4:1 aggregation as the 2nd stage. In this case RX*VC_MULTI should be defined in the 2nd stage CrossLink since each first stage TX output contains 4 channel data aggregated. This can be done with four CrossLink devices (three 5:1 aggregation (using Hard D-PHY IP on one RX channel) in the 1st stage and one 4:1 aggregation in the 2nd stage) after D-PHY Receiver IP 1.4 is available.

This module also detects the end of current High Speed transmission by detecting the trailer byte and asserts the end flag sent to rx_buffer along with the offset info. Offset is a 3-bit data that indicates the number of remaining bytes in the final data transmission. This is modulo of 8 considering 64-bit transmission in case of four lanes with Gear 16 on TX side. Figure 3.3 and Figure 3.4 show the beginning and end of long packet capture and transfer.

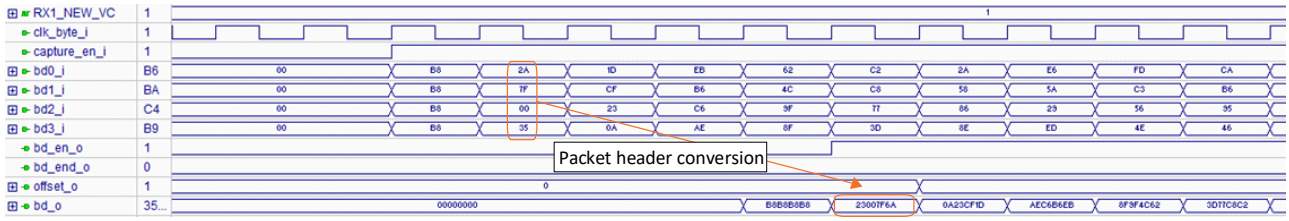

Figure 3.3. Long Packet Detection and VC Replacement

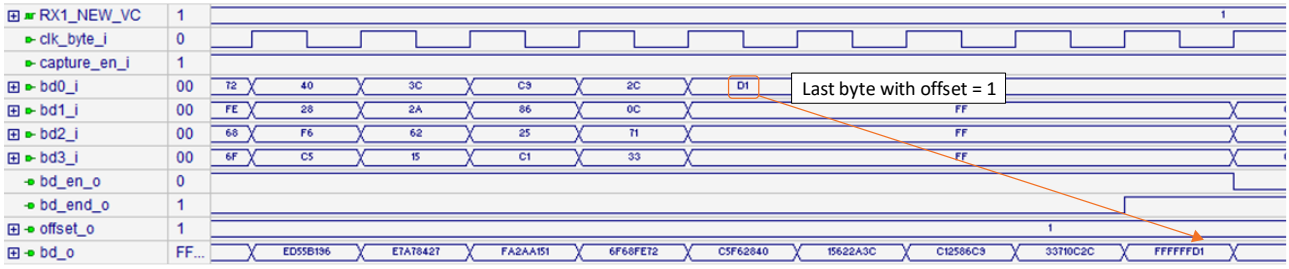

Figure 3.4. End of Long Packet with Trailer Bytes

3.4. rx_buffer

This module is instantiated for each RX channel to store HS transmission data to be read out by tdm_ctrl. Default buffer depth is 512, which means the buffer can store 512 × 64 bytes (NUM_TX_LANE_4 and TX_GEAR_16) or 512 × 32 bytes (others). 512 × 64 bytes FIFO requires four EBRs, therefore all 20 EBRs are required for 5:1 aggregation with 4-lane TX and TX Gear 16 configuration. Buffer depth of 1024 (or deeper) is possible as long as the total number of EBR does not exceed 20 (no other modules require EBR). Data size of one HS transmission must not exceed FIFO size (for example, 1920 pixels of RAW10 makes 1920 × 5/4 = 2400 bytes, which is less than 1024 × 32, but more than 512 × 32). In case the data transfer to tdm_ctrl of the RX channel in question is in a cue and needs to wait for the completion of other channel data transmission, the FIFO has to begin storing the next line data. Therefore, it is safer to make the FIFO

depth cover close to two HS transmission data. The first FIFO data is read out by this module itself with ready flag assertion. The FIFO has extra data width to contain data end flag and offset data. This FIFO is a boundary between RX byte clock (write side) and TX byte clock (read side) domain.

Figure 3.5 shows an example of short packet write transaction. An extra data write happens after the completion of every HS data transmission. This dummy write is necessary to ease the FIFO read timing to quit HS data read when it gets the end data flag. tdm_ctrl can stop fifo_re_i assertion in the next cycles when it sees bd_end_o =1. This module automatically reads the first data of each HS transmission. Along with ready flag (bd_rdy_o) assertion, it notifies tdm_ctrl that HS data is ready to be acquired from this channel. In this example, the offset is 4 since TX Gear is 16, which means 8 bytes are passed to tdm_ctrl in a single read cycle.

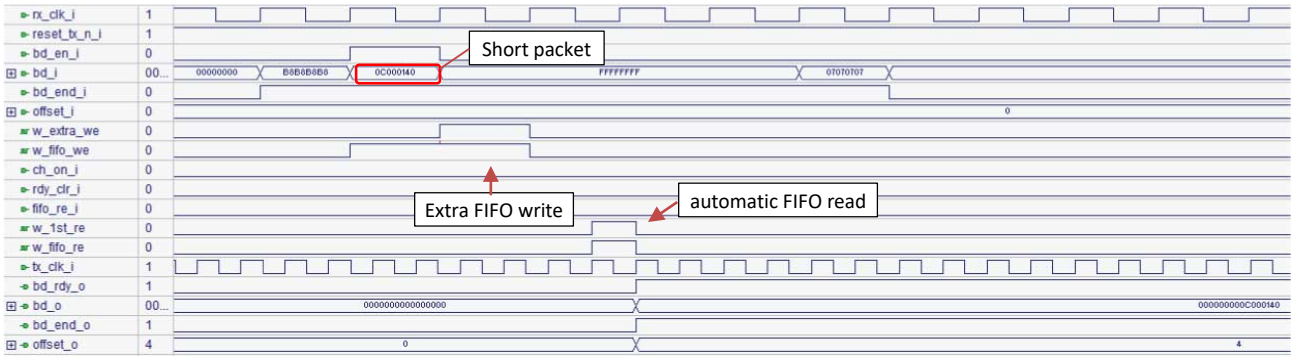

Figure 3.5. Short Packet Write

Figure 3.6 and Figure 3.7 show the beginning and end of long packet write transaction. In case of TX Gear 16, write happens every two cycles to form 64-bit byte data. Upon the detection of bd_end_i, an extra write enable is generated to write dummy data followed by the automatic first data read with bd_rdy_o assertion.

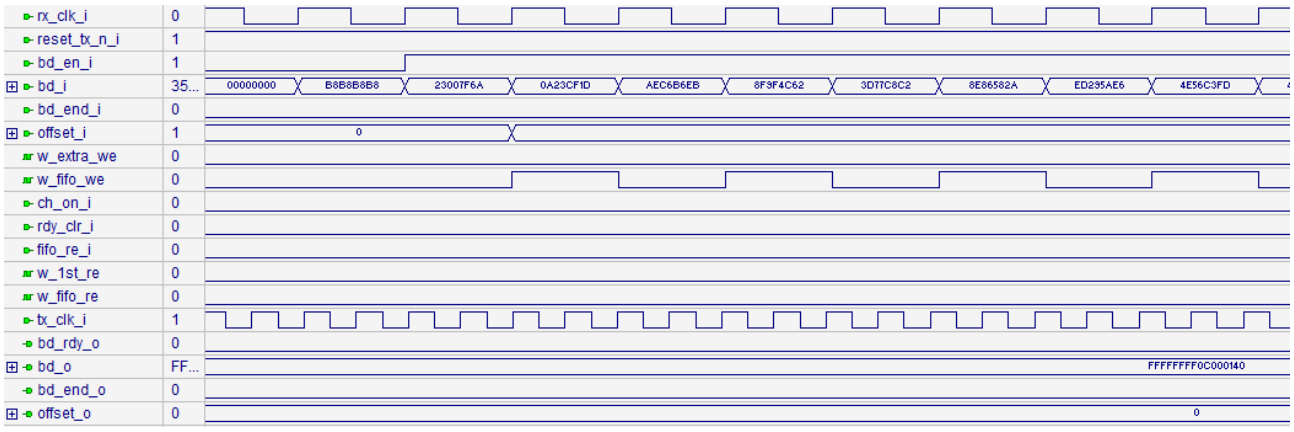

Figure 3.6. Beginning of Long Packet Write

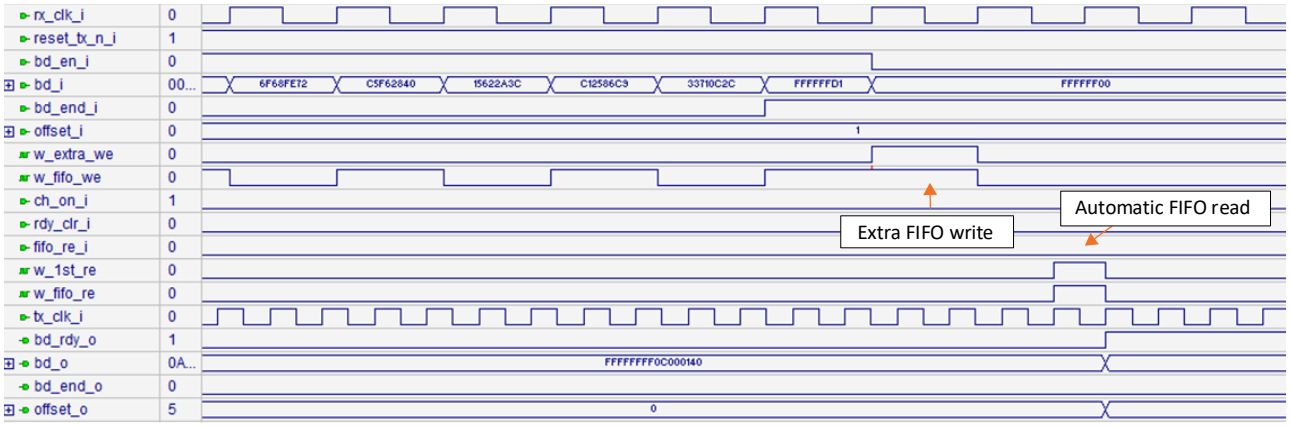

Figure 3.7. End of Long Packet Write

Figure 3.8 shows short packet read transaction. rdy_clr_i comes from tdm_ctrl to clear bd_rdy_o. The valid data (0x0c000140) is already on bd_o when fifo_re_i is asserted due to the automatic first data read. Therefore, this fifo_re_i reads an extra dummy data and discarded by tdm_ctrl. bd_end_o is used to terminate fifo_re_i by tdm_ctrl.

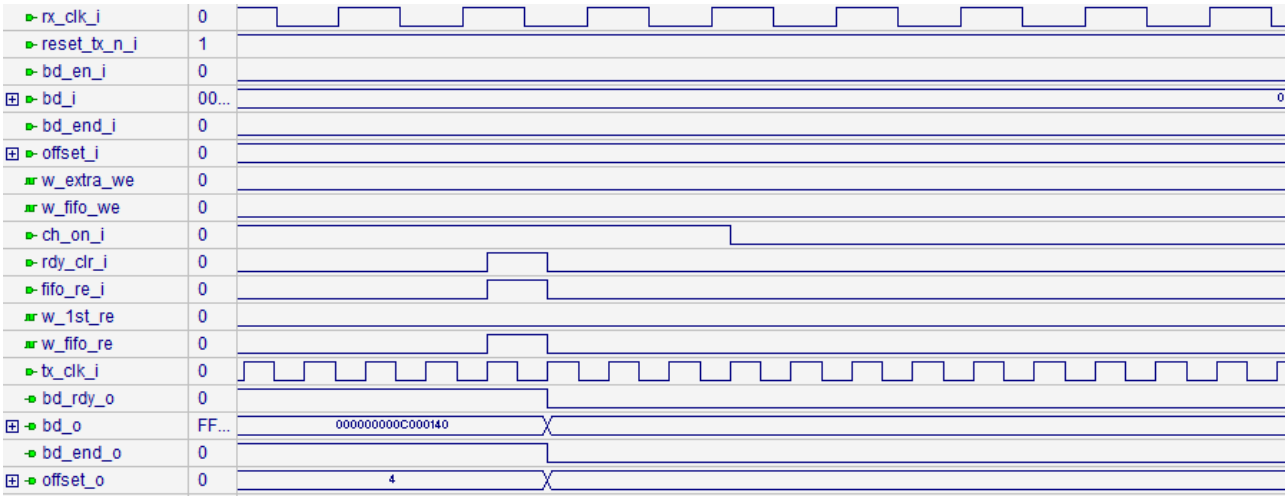

Figure 3.8. Short Packet Read

Figure 3.9 shows end of long packet read. Upon the detection of bd_end_o, fifo_re_i is de-asserted. In this example, the last data is “0xD133710C2C” since offset_o is 5.

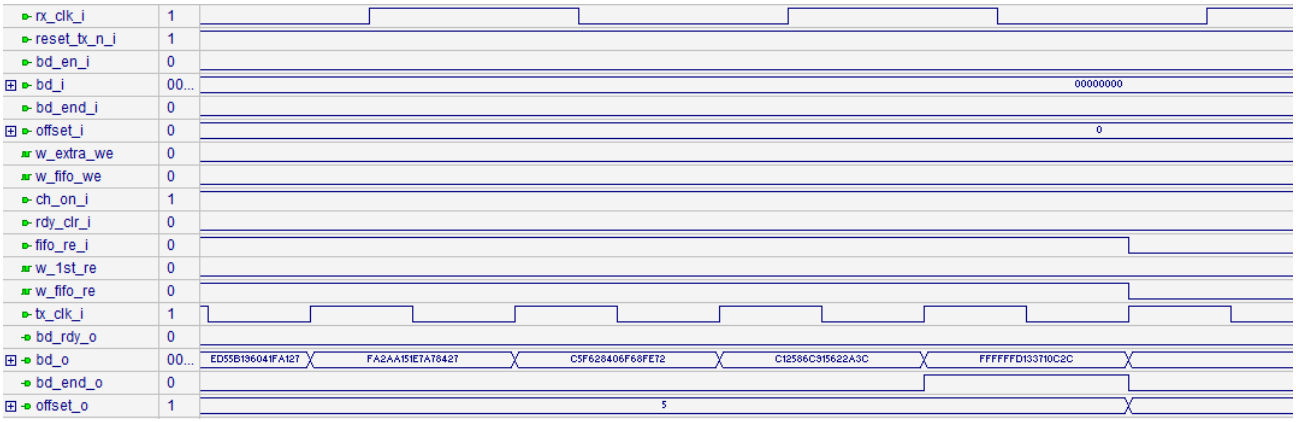

Figure 3.9. End of Long Packet Read

3.5. tdm_ctrl

This module monitors the ready flags of the RX channels and takes the HS transmission data of the selected channel. TX D-PHY data lane goes into LP mode after every HS data transfer of a single RX channel and come back to HS mode when the next HS transmission begins with the new RX channel data. The channel selection is based on a round robin fashion when multiple RX channel data are available signified by the assertion of the ready flags. Incoming data width is either 32 or 64, but the outgoing data width is adjusted to TX Lane count * TX_GEAR, which means 8, 16, 32, or 64. In case of 8 or 16, FIFO read enable is asserted once every 4 or 2 TX byte clock cycles. When the data end flag is detected, the trailer bytes are appended. Beginning of trailer byte within 16/32/64 bytes is judged by offset data obtained from rx_buffer when the end flag is detected.

Figure 3.10 shows an example of 4-channel aggregation global sequence of tdm_ctrl. tdm_ctrl monitors all ready signals (rx0_rdy_i, … rx3_rdy_i) and decides which RX channel data is aggregated for the next transmission. In this example, channel 0 is selected since ready flag is asserted only on channel 0. tdm_bgn_i is a trigger signal to begin reading FIFO data from rx_buffer. rx0_rdy_clr_o is asserted to clear rx0_rdy_i and rx0_fifo_re_o is asserted to begin reading FIFO data from rx0_buffer. Read data are sent to the next module (tx_dphy_if) for transmission. Other channel data are also transferred in the same manner.

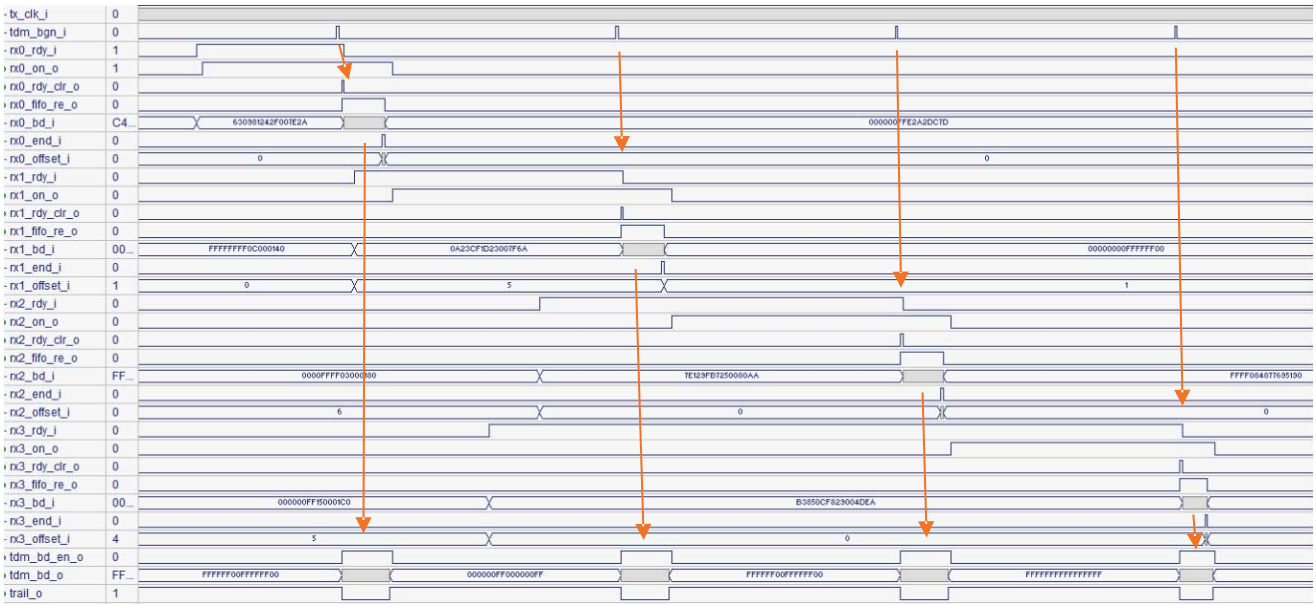

Figure 3.10. Global Sequence of tdm_ctrl

Trailer byte appending is another important feature of tdm_ctrl in addition to TDM of RX channel data. Figure 3.11 shows an example. The last valid data is 0x0C07D0 (rx3_offset_i = 3 when rx3_end_i = 1). Trailer bytes against these are 0xFFFF00. MSByte of the previous data is 0x59 and trailer byte against this is 0xFF. Therefore, 0xFFFFFF00FF is appended as trailer bytes, which appears as upper 5 bytes with the last valid data (0x0C07D0) followed by two sets of 0xFFFFFF00 along with trail_o assertion.

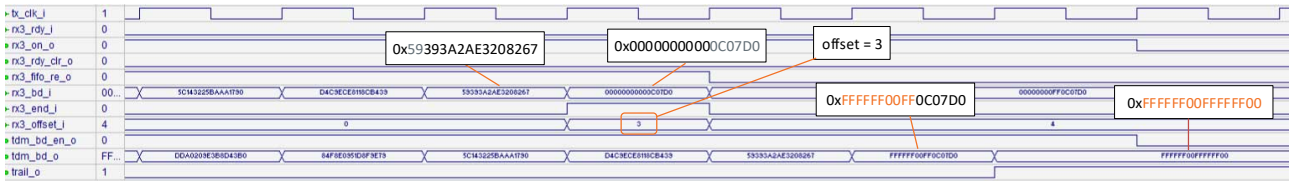

Figure 3.11. Trailer Byte Appending

3.6. tx_dphy_if

This module reallocates byte data coming from tdm_ctrl to 64-bit data path and sends to tx_dphy according to TX lane and TX Gear value. In addition, it monitors ready flag from rx_buffer modules as well as ready flag from tx_dphy to control tdm_bgn_o that triggers TDM operation in tdm_ctrl. Figure 3.12 shows global operation of tx_dphy_if including tx_dphy signals. tx_dphy_if asserts clk_hs_en_o and d_hs_en_o to make both clock and data lane from LP mode to HS mode. After that d_hs_rdy_i is asserted by tx_dphy. Confirming at least one of ready flags from rx_buffer is asserted, tdm_bgn_o is asserted to let tdm_ctrl begins a new HS data transmission.

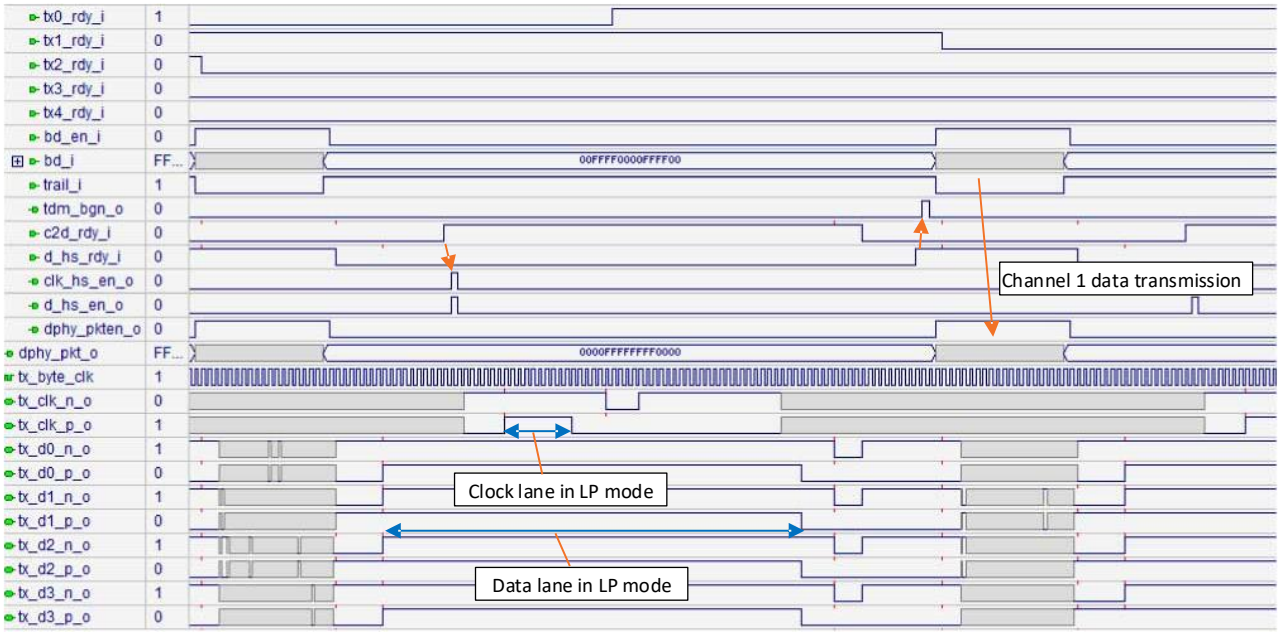

Figure 3.12. Global Operation of tx_dphy_if

3.7. tx_dphy

You must create this module according to channel conditions, such as number of lanes, bandwidth, and others. Figure 3.13 shows an example IP interface setting in Clarity Designer for the CSI-2/DSI D-PHY Transmitter Submodule IP. Refer to CSI-2/DSI D-PHY Transmitter Submodule IP User Guide (FPGA-IPUG-02024) for details.

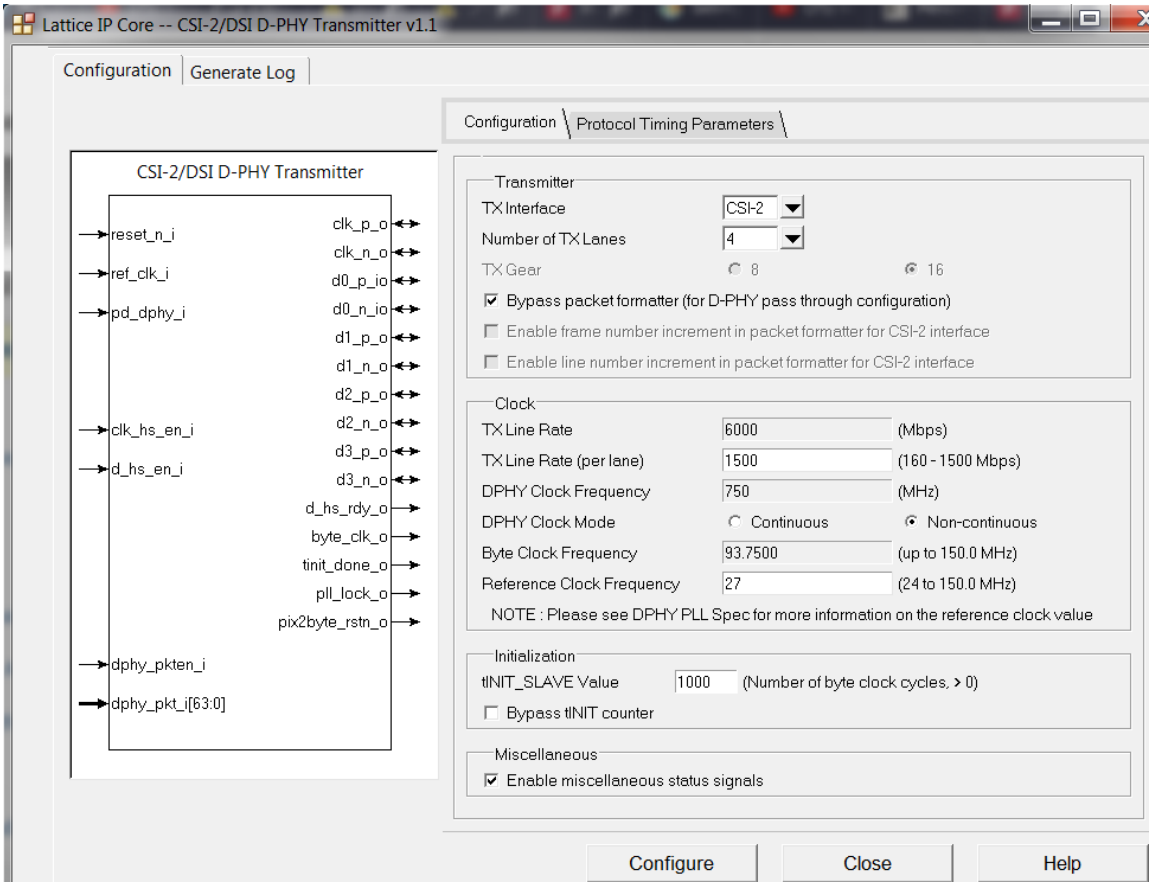

Figure 3.13. tx_dphy IP Creation in Clarity Designer

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

- TX Interface – Select CSI-2.
- Number of TX Lanes – Set according to channel configuration. Must match NUM_TX_LANE_* setting.
- TX Gear – Automatically set according to TX Line Rate.
- Bypass packet formatter – Select checkbox to enable.
- TX Line Rate – Set according to channel configuration. Total bandwidth must be fast enough to aggregate all RX channel data with extra overhead for the transitions between HS and LP modes.
- D-PHY Clock Mode – Set according to channel configuration. Select Continuous when the TX bandwidth margin is small. Must match TX*_CLK_MODE_* setting (Continuous = HS_ONLY, Non-continuous = HS_LP).
- Reference Clock Frequency – Set the appropriate value which can be obtained from ref_clk_i pin, one of continuous rx*_byte_clk, or on-chip GPLL. This clock frequency must be in one of the following ranges:
  - 24-30 MHz
  - 48-60 MHz
  - 72-90 MHz
  - 96-150 MHz
- tINIT_SLAVE Value – 1000 (default) is recommended.
- Bypass tINIT counter – Disabled (unchecked) is recommended.

- Enable miscellaneous status signals – must be set to enabled (checked).
- Protocol Timing Parameters tab – Default values are recommended.

This module takes the byte data and outputs CSI-2 data after serialization in CSI-2 High Speed mode. It is recommended to set the design name to *tx_dphy* and module name to *tx_ch* so that you do not need to modify the instance name of this IP in the top level design as well as simulation setup file. Otherwise, you need to modify the names accordingly.

TX Gear 8 is automatically selected by Clarity Designer when the lane bandwidth is less than 900 Mbps, which means TX byte clock could be ~112 MHz. If this causes timing violations in Static Timing Analysis (STA), TX Gear should be changed to 16. Since the current version of TX DPHY IP creation does not allow manual setting of TX Gear, please consult with Lattice.

TX Line Rate setting is one of the key factors in this RD. Calculating the required TX lane bandwidth is derived from the following equation:

$$TX\_lane\_bandwidth = \frac{1}{m}\sum_{k=0}^{n-1} number\_of\_lanes\_in\_RX[k] \ * lane\_bandwidth\_in\_RX[k]$$

where m is number of TX lanes and n is number of RX channels.

Example : 5 to 1 aggregation with the following configuration

RX0 : 4 lanes, 500 Mbps / lane,

RX1 : 4 lanes, 400 Mbps / lane,

RX2 : 4 lanes, 400 Mbps / lane,

RX3 : 2 lanes, 300 Mbps / lane,

RX4 : 1 lane, 200 Mbps / lane,

TX : 4 lanes

Then TX lane bandwidth = (4 x 500 + 4 x 400 + 4 x 400 + 2 x 300 + 1 x 200) / 4 = 1.5 Gbps.

The above configuration does not work if RX channels do not have enough length of blanking (LP) periods. The TX bandwidth is fast enough to cover all of RX channel HS data transmissions, but transition time going back and forth between HS and LP mode cannot be proportionally shrunk by increasing TX bandwidth. The guideline of minimum blanking period per active line is (500 ns x number of RX channels) if the above equation is used to get TX lane bandwidth, assuming continuous clock (HS_ONLY mode) is set in TX D-PHY. In case of non-continuous clock (HS_LP mode), the required blanking period is doubled. TX lane bandwidth has to be raised from the above value in case that RX channel cannot have enough blanking periods. Increasing RX FIFO size could be an alternative option in some cases, but most likely this is not as effective.

You should monitor FIFO overflow flags of all rx_buffer modules (rx*_fifo_full) instantiated and increase the TX bandwidth if any of rx_buffer FIFO overflow happens.

You also should be aware of the relationship between the reference clock and DPHY clock. DPHY clock is generated by the internal PLL of TX D-PHY IP. Following is the equation to generate DPHY clock:

$$TX\_Line\_Rate\_per\_lane = \frac{1}{NI} * ref\_clk\_frequency * \frac{M}{NO}$$

where NI = 1, 2, 3, 4, or 5; M = 16, 17, …, or 255; NO = 1, 2, 4, or 8. The following restrictions also exist:

$$24MHz \leq \frac{1}{NI} * ref\_clk\_frequency \ \leq 30\ MHz,$$

$$640MHz \leq \frac{1}{NI} * ref\_clk\_frequency * M \leq 1500MHz$$

You must set the appropriate TX Line Rate (per lane) which can be obtained by the above equations applying the given reference clock frequency.

3.8. Clock Distribution

In case of non-continuous clock (HS_LP) mode in RX, each RX channel requires two clock trees (non-continuous RX byte clock and continuous RX byte clock) in CrossLink device. Since CrossLink has only eight primary clock trees, it is not possible to use HS_LP mode in all RX channels in case of 4:1 or 5:1 aggregation when all RX channels come from different clock sources. Even in case of 2:1 or 3:1 aggregation, it may be difficult to generate continuous RX byte clocks for all RX channels if these channels are from different clock sources unless the continuous byte clocks for all RX channels are fed directly from the I/O pins. Therefore, you have to make sure that the continuous clock can be obtained to set the non-continuous clock mode in RX D-PHY on RX channels. The following are possible candidates of the continuous clock:

- PLL outputs driven by the external reference clock or a continuous byte clock from other channels
- Continuous byte clock from other channels (either directly or after divider)
- Continuous clock fed from I/O pin (either directly or after divider)

The sample design (vsi2_aggregation_vc.v) assumes that RX channel #2 is in HS-LP mode and other RX channels are in HS_ONLY mode. In this example, the continuous byte clock for RX channel #2 and TX byte clock are generated by the on-chip GPLL taking the external clock as a reference clock. The code snippets are shown below. rx*clk_lp_ctrl (clock signal for LP and HS mode control module for clock lane) could be different from rx*_clk_byte_fr (continuous byte clock for RX channel *), but recommended to be the same to save the primary clock tree resources. *int_gpll* is the name used in this top level design. This name has to be changed if the different name is used.

```
assign rx2_clk_byte_fr = pll_clk_op;      // continuous clock as fast as rx2_clk_byte_hs
assign rx2_clk_lp_ctrl = pll_clk_op;      // > 20 MHz
…

assign tx_refclk = pll_clk_os;     // use this if GPLL is necessary
…

int_gpll pll_inst (
  .CLKI (ref_clk_i),
  .CLKOP (pll_clk_op),
  .CLKOS (pll_clk_os),
  .LOCK  (pll_lock)
);
```

On TX side, using continuous or non-continuous clock mode does not affect the number of necessary clock trees (always uses one clock tree). To feed a clock to TX D-PHY IP, the external clock is necessary if continuous RX byte clocks are not appropriate to generate the desired clock for TX D-PHY. The clock to TX D-PHY must be continuous and within one of the following ranges:

- 24-30 MHz
- 48-60 MHz
- 72-90 MHz
- 96-150 MHz

4. Design and File Modifications by User

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

4.1. Top Level RTL

The current top-level file (csi2_aggreagation_vc.v) takes the secondary GPLL clock to feed a clock to TX D-PHY when EXT_REF_CLK is defined in synthesis_directives.v and takes the byte clock of RX channel 0 when EXT_REF_CLK is not defined. This part must be modified if necessary according to the clocking scheme. The code snippet is shown below.

rx*_clk_byte_fr, rx*_clk_lp_ctrl, and tx_refclk assignments also have to be modified if necessary as shown in the Clock Distribution section.

```
////////////////////////////////////////////////////////////////////////////////
///// Reference Clock generation to TX D-PHY                                /////
///// Customer has to modify here according to the available clock for TX-DPHY /////
///// tx_refclk must be in the range of 24-30, 48-60, 72-90, or 96-150 MHz.    /////
///// Crosslink GPLL must be used to avoid frequency holes of Mixel PLL if     /////
///// ref_clk_i nor rx*_clk_byte_fr is in these ranges.                      //////
////////////////////////////////////////////////////////////////////////////////

`ifdef EXT_REF_CLK
//     assign tx_refclk = ref_clk_i;
//     assign tx_refclk = pll_clk_op;     // use this if GPLL is necessary
       assign tx_refclk = pll_clk_os;     // use this if GPLL is necessary
`else
       assign tx_refclk = rx0_clk_byte_fr; // could be from other channels
//     assign tx_refclk = pll_clk_op; // use this if GPLL is necessary
`endif
///// below is just a reference only ////////////////////////////////////////////
/*

//// CrossLink GPLL
int_gpll pll_inst (
  .CLKI (ref_clk_i),
  .CLKOS (pll_clk_os),
  .CLKOP (pll_clk_op),

  .LOCK  (pll_lock)
);

int_gpll pll_inst (
  .CLKI (rx0_clk_byte_fr),
  .CLKOP (pll_clk_op),
  .LOCK  (pll_lock)
);
*/
////////////////////////////////////////////////////////////////////////////////
```

In addition, instance names of RX/TX D-PHY (rx_ch0, rx_ch1, …, tx_ch) have to be modified if you created these IP with different names.

4.2. TX D-PHY IP

After creating TX D-PHY IP in Clarity Designer with non-continuous clock mode (TX_CLK_MODE_HS_LP), you must add an output signal *c2d_rdy_o* since this signal exists internally, but not brought out to the port.

In tx_ch.v (assuming you created TX D-PHY IP with the instance name *tx_ch*), add those two lines:

output c2d_rdy_o, // additional output, around line 77, after *output d_hs_rdy_o,*

.c2d_rdy_o (c2d_rdy_o), // added to a submodule, around line 113, after *.d_hs_rdy_o (d_hs_rdy_o),*

In tx_ch_dphy_tx.v, add one line and modify one line:

output c2d_rdy_o, // additional output, around line 60, *after output d_hs_rdy_o,*

.c2d_ready_o (c2d_rdy_o), // fill in the port name *c2d_rdy_o*, around line 206 (currently empty)

In tx_ch_tx_global_operation.v, modify one line:

.c2d_ready_o (c2d_ready_o), // fill in the port name *c2d_ready_o*, around line 67 (currently empty)

The above change is required only for non-continuous mode (TX_CLK_MODE_HS_LP). No change is required for continuous clock mode (TX_CLK_MODE_HS_ONLY).

When this IP is created, Clarity Designer automatically sets TX Gear value to 8 or 16 according to the lane bandwidth. The bandwidth threshold is 900 Mbps to switch 8 and 16, which means the highest TX byte clock is 112.5 MHz. In some cases, timing violation may occur and Gear 16 may be necessary when the lane bandwidth is between 800 and 900 Mbps. Please consult with Lattice if it is necessary to manually modify the TX Gear setting. Another possible workaround is to change the TX clock mode to HS_ONLY if the current configuration is HS_LP. This reduces the overhead of TX data transmission and may enable to use the lower lane bandwidth.

4.3. RX D-PHY IP

RX D-PHY IP version 1.2 has issues related to Hard D-PHY. It is therefore not recommended to use Hard D-PHY in RX channel. On the other hand, Hard D-PHY is necessary in following configurations due to the limitations of numbers of available MIPI I/O ports for Soft D-PHY:

- 4 to1 aggregation with 4/4/4/4, 4/4/4/2, 4/4/4/1, or 4/4/2/2 RX lane configurations
- All 5 to 1 aggregation configurations

In case RX Hard D-PHY is required, it is recommended to set word aligner option enabled. Please consult with Lattice if one of the above configurations is necessary.

5. Design Simulation

The script file (csi2_aggregation_vc_fsim.do) and testbench files are provided to run the functional simulation by Active HDL. If you follow the naming recommendations regarding design name and instance name when RX and TX D-PHY IPs are created by Clarity Designer, the following are the only changes required in the script file:

- Diamond installation directory path
- User project directory
- Comment out if GPLL is not in use
- Comment out or remove lines for unused RX channel(s)
- Number of TX lanes

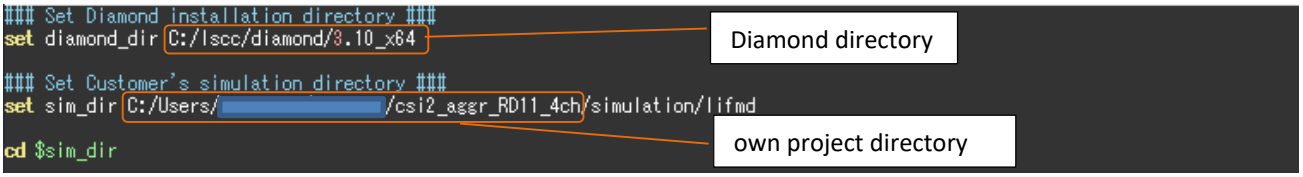

Figure 5.1. Script Modification #1

Figure 5.2. Script Modification #2

You need to modify simulation_directives.v according to your configuration (refer to Simulation Directives for details). By executing the script in Active HDL, compilation and simulation are executed automatically. The testbench takes all data comparison between the expected data and output data from the RD, including VC and ECC data modifications. It shows following statements while running and doing data comparison:

KERNEL: [93725383860][DPHY_CHK] payload data = 0b f6 6c bd --- Data matches ch0 : 0b f6 6c bd

KERNEL: [93726027380][DPHY_CHK] payload data = ea c6 c4 f1 --- Data matches ch0 : ea c6 c4 f1

KERNEL: [93726670660][DPHY_CHK] payload data = 7d 35 29 3e --- Data matches ch0 : 7d 35 29 3e

KERNEL: [93727314180][DPHY_CHK] payload data = 83 01 35 71 --- Data matches ch0 : 83 01 35 71

When the simulation is finished, the following statements are shown:

KERNEL: CH #0 (VC = 0) : 140 / 140 HS transmission completed successfully

KERNEL: CH #1 (VC = 1) : 81 / 81 HS transmission completed successfully

KERNEL: CH #2 (VC = 2) : 84 / 84 HS transmission completed successfully

KERNEL: CH #3 (VC = 3) : 115 / 115 HS transmission completed successfully

KERNEL:

KERNEL: ### Simulation Completed

One HS transmission is either Frame Start/End short packet or long packet of one active video line. The result would be fine if the numerator is equal to denominator in the statements.

You should set small values in CH*_NUM_LINES and CH*_NUM_PIXELS directives in simulation_directives.v file, especially in the first simulation trial to minimize simulation time. On the other hand, it makes sense to set the actual value to CH*_NUM_PIXELS directives and CH*_DPHY_LPS_GAP directives when TX bandwidth cannot have extra margin against total RX bandwidth. By setting realistic values, FIFO overflow could be detected when the margin is not large enough.

Figure 5.3 shows an example of 4-channel aggregation. tx*_bd_rdy = 1 indicates the waiting period from the completion of one HS transmission data write (to rx_buffer) to the start of FIFO read by tdm_ctrl. This waiting period gets longer when the channel in question is in a cue to be read by tdm_ctrl.

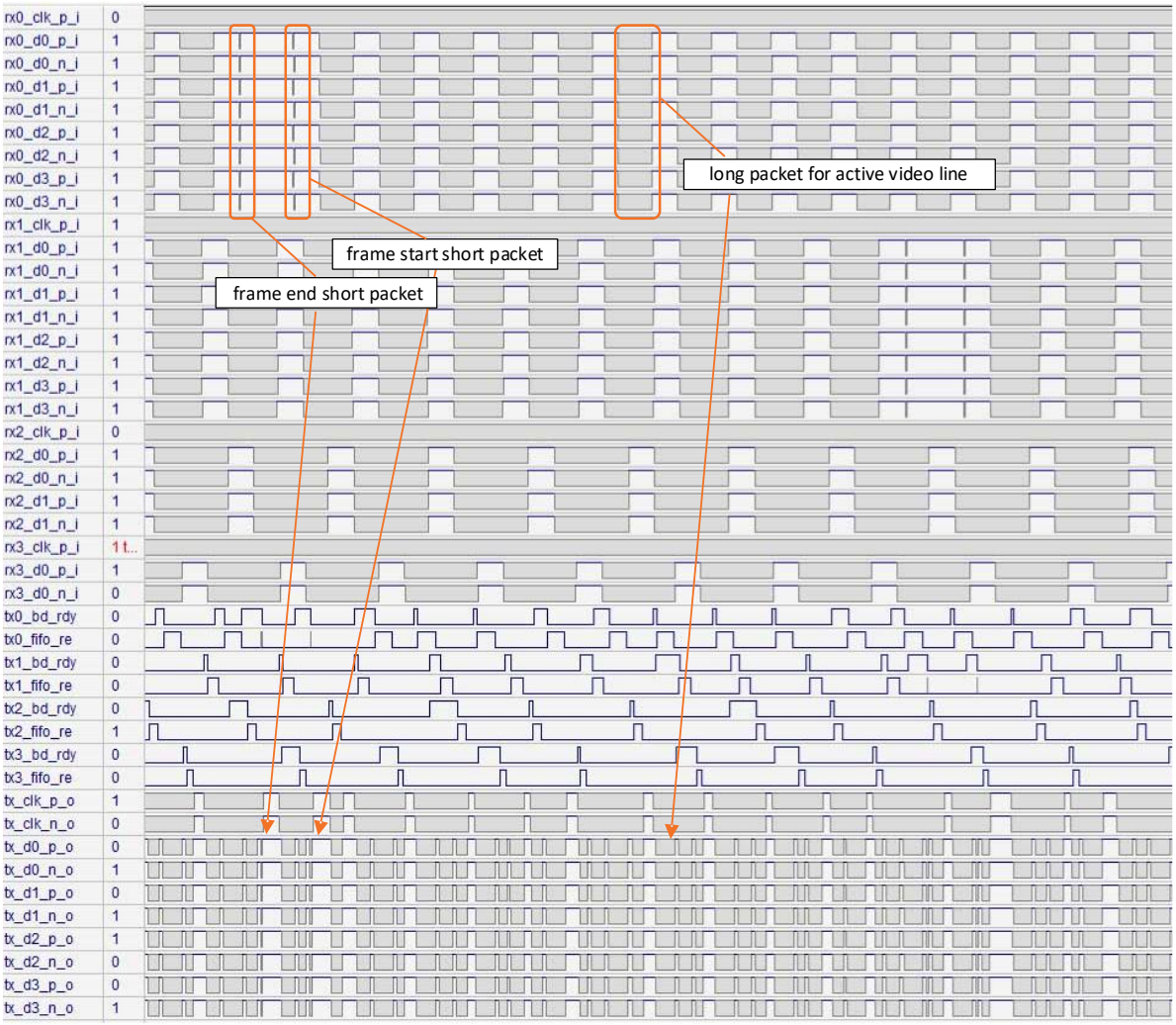

Figure 5.3. Functional Simulation Example

Figure 5.4 shows an example of FIFO overflow in rx_buffer. tx0_bd_rdy = 1 period get long and that leads to FIFO overflow on channel 0. Simulation automatically stops when FIFO overflow happens.

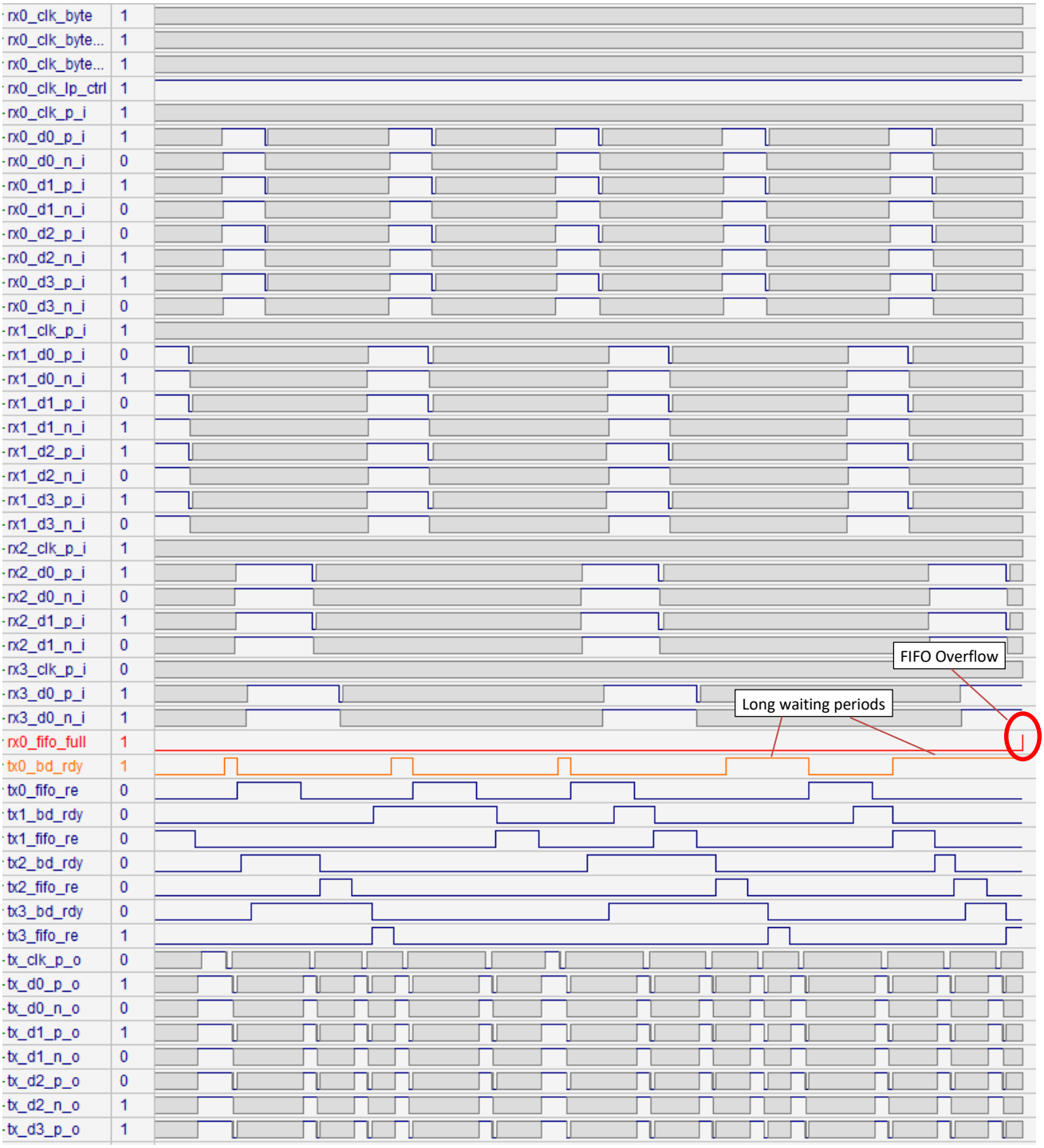

Figure 5.4. FIFO Overflow

6. Known Limitations

Following are the limitations of this reference design:

- All incoming MIPI data lanes must go into LP mode during horizontal blanking periods. Using null and/or blanking long packet to fill the blanking period is not allowed.
- RX Gear 16 with 4-lane configuration is not supported.
- RX channel must use HS_ONLY mode (continuous clock mode) when the corresponding continuous clock is not obtained.
- TX Gear cannot be set manually (refer to the TX D-PHY IP section).
- Using Hard D-PHY on RX is not recommended (refer to the RX D-PHY IP section).
- Total number of RX clock and data lanes by RX D-PHY Soft IP cannot exceed 15.

7. Design Package and Project Setup

MIPI CSI-2 Virtual Channel Aggregation Reference Design for CrossLink is available on www.latticesemi.com. Figure 7.1 shows the directory structure. The design is targeted for LIF_MD6000-6KMG80I. synthesis_directives.v and simulation_directives.v are set to configure four RX channels as an example shown below:

- RX CH #0 : 2 lanes with Soft D-PHY with Gear 16 in continuous clock mode
- RX CH #1 : 4 lanes with Soft D-PHY with Gear 8 in continuous clock mode
- RX CH #2 : 1 lanes with Soft D-PHY with Gear 16 in non-continuous clock mode
- RX CH #3 : 2 lanes with Soft D-PHY with Gear 8 in continuous clock mode
- TX : 4 lanes with Gear 8

You can modify the directives for own configuration.

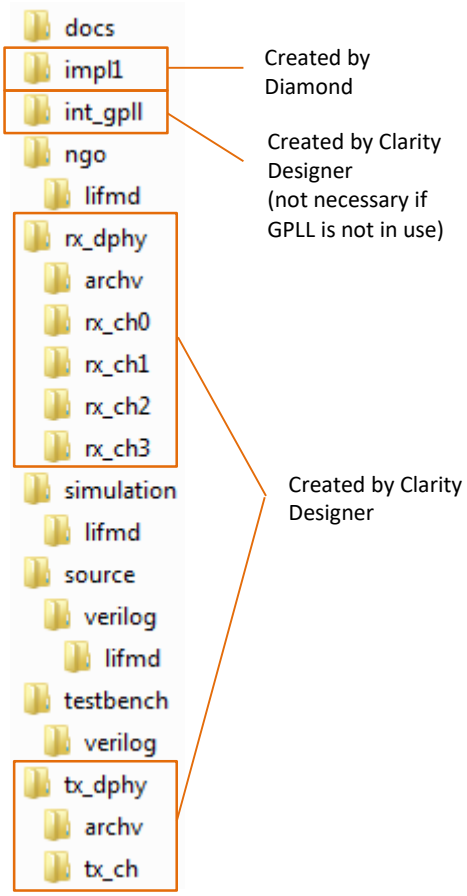

Figure 7.1. Directory Structure

Folders *rx_dphy* and *tx_dphy* are expected to be added under your project directory when Clarity Designer creates RX and TX D-PHY IPs. The *ngo\lifmd* folder contains mapped netlists of *csi2_parser*, *rx_buffer*, and *tdm_ctrl* with all possible configurations designated by suffixes. Following shows explanation of the suffixes:

- _VC*# : * VC mode; P (pass through), R (replace), # : multiple VC ID; M (multiple VC IDs), S (single VC ID)
- _FC* : * Frame Counter mode; ON (replace), PT (pass through)
- _C* : Number of RX channels; 2 – 5
- _RL*G** : * Number of RX lanes; 1, 2, or 4, ** RX Gear; 8 or 16
- _TL*G**B*** : * Number of TX lanes; 1, 2, or 4, ** TX Gear; 8 or 16, *** RX Buffer Depth; 512, 1024, or 2048

The blackbox files (*_bb.v) are also contained in the *ngo\lifmd* folder. You can include corresponding _bb.v files as design files in Diamond according to own configuration. Alternatively, it is possible to include all _bb.v files as long as *csi2_aggregation_vc* is specified as the top-level design. Figure 7.2 shows design files used in the Diamond project.

Clarity Designer creates three .sbx files. In this example, all 144 _bb.v files are included in the project. By specifying csi2_aggregation_vc as a top-level design, all unnecessary files are ignored. GPLL design file must be added if necessary.

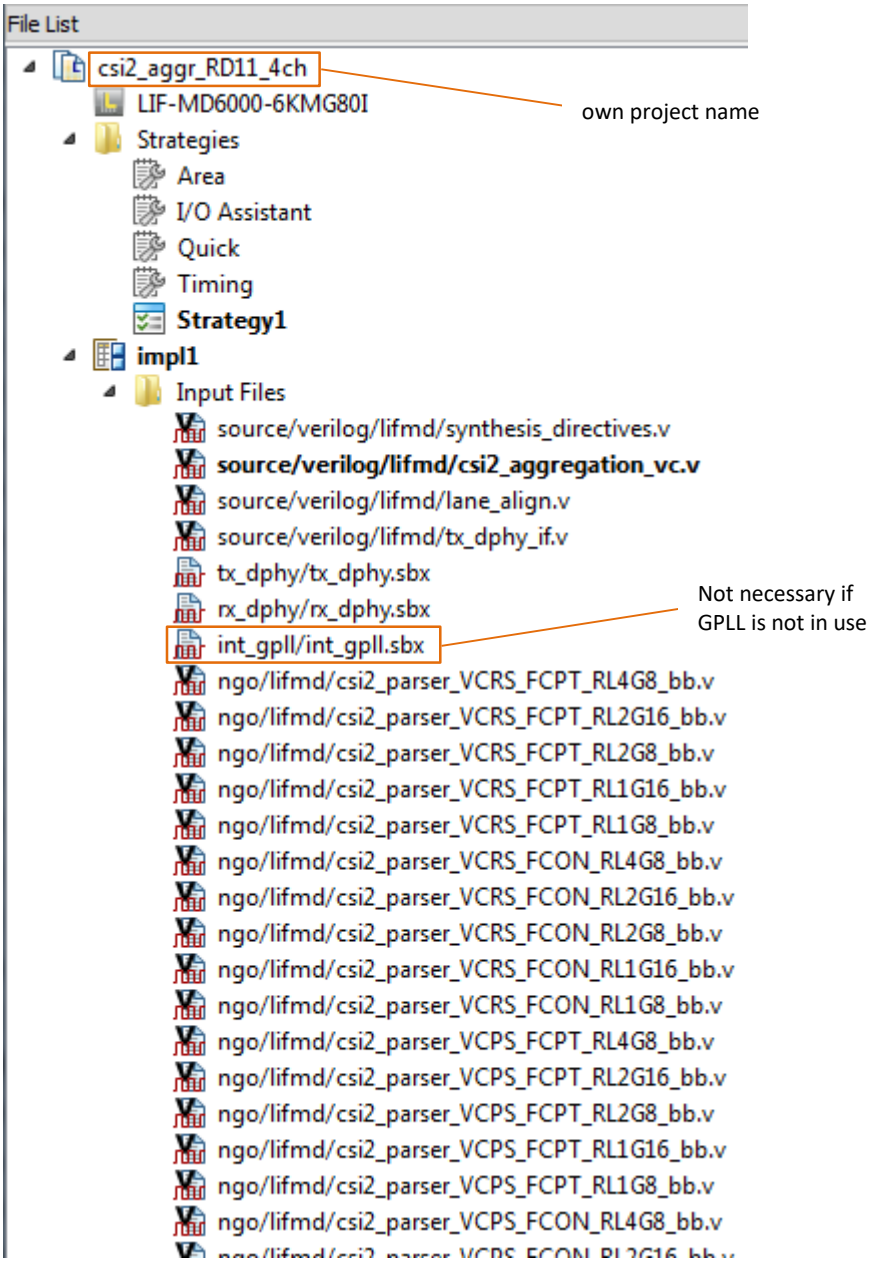

Figure 7.2. Project Files

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

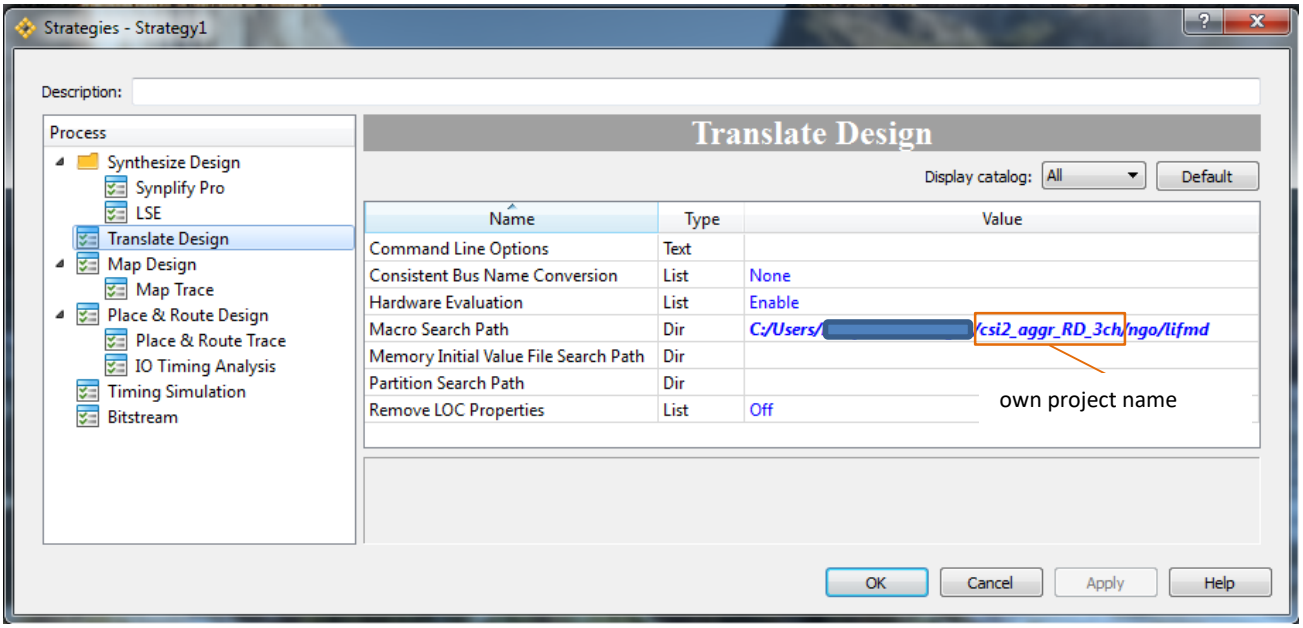

Figure 7.3. Path Setting for .ngo Files

8. Resource Utilization

Resource utilization depends on the configurations. Table 8.1 shows resource utilization examples under certain configurations. Actual usage may vary.

Table 8.1. Resource Utilization Examples

Configuration	LUT %	FF %	EBR	I/O
4 RX channels with 4/4/2/1 lanes, TX Gear 16	78	48	16	32
3 RX channels with 4/4/4 lanes, TX Gear 16	74	47	12	32
5 RX channels with 4/4/2/1 lanes, TX Gear 8	70	45	8	22

References

- MIPI® Alliance Specification for D-PHY Version 1.1
- MIPI® Alliance Specification for Camera Serial Interface 2 (CSI-2) Version 1.1
- MIPI® Alliance Specification for Camera Serial Interface 2 (CSI-2) Version 2.0
- CSI-2/DSI D-PHY Receiver Submodule IP User Guide (FPGA-IPUG-02025)
- CSI-2/DSI D-PHY Transmitter Submodule IP User Guide (FPGA-IPUG-02024)

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

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

Technical Support Assistance

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

Revision History

Revision 1.2, September 2019

Section	Change Summary
All	Added Disclaimers section.
Supported Device and IP	Newly added to support CrossLinkPlus.

Revision 1.1, March 2019

Section	Change Summary
Features List	Updated item regarding support for non-continuous clock mode.
Synthesis Directives	Updated section to reflect the following changes: • RX Gear 16 is supported in 1-lane and 2-lane configurations. • HS_LP mode is allowed on RX channels. • VC related directives are added to support pass through mode and multiple VC mode. • Frame counter related directives are added to support Data Field replacement by FPGA's own frame counter.
Simulation Directives	Frame Number related directives are added to support enabling/disabling frame number in Data Filed of Frame Start/End short packet.
rx_dphy	Updated description related to HS_LP and Gear 16 support.
csi2_parser	Added description related to VC pass through mode and multiple VC mode.
Clock Distribution	Added description related to HS_LP mode.
Top Level RTL	Updated description due to the configuration change of the sample design.
Design Simulation	Updated Figure 5.2 to reflect configuration change.
Known Limitations	Updated description due to HS_LP mode and RX Gear 16 support.
Design Package and Project Setup	• Updated description due to the configuration change of the sample design. • Added description related to VC pass through mode and multiple VC mode. • Added description related to Frame Counter mode.

Revision 1.0, March 2019

Section	Change Summary
All	Initial release.

www.latticesemi.com