JESD204B IP Core

User Guide

FPGA-IPUG-02010 Version 2.3

June 2017

Contents

1. Introduction 4
1.1. Quick Facts 4
1.2. Features 5
1.3. What is Not Supported 5
1.4. Conventions 6
1.5. Data Ordering and Data Types 6
1.6. Signal Names 6
2. Functional Description 7
2.1. Rx Core 7
2.2. Tx Core 14
3. Parameter Settings 19
3.1. Configuring JESD204B Core in IPexpress 19
3.2. Configuring JESD204B Core in Clarity Designer 21
4. IP Core Generation and Evaluation 22
4.1. Licensing the IP 22
4.2. Getting Started 22
4.3. Creating IP in Clarity Designer 23
4.4. Generating IP in IPexpress 27
4.5. Generated IP Directory Structure and Files 28
4.6. Instantiating the IP Core 29
4.7. Running Functional Simulation 30
4.7.1. Running Functional Simulation in ModelSim 30
4.7.2. Running Functional Simulation in Active HDL 30
4.8. Simulation Strategies 31
4.9. Simulation Environment 31
4.10. Synthesizing and Implementing the IP in a Top-Level Design 32
4.11. Hardware Evaluation 32
4.11.1. Enabling Hardware Evaluation in Diamond 32
4.12. Updating/Regenerating the IP 32
4.12.1. Regenerating an IP in Clarity Designer 32
5. Application Support 34
References 35
Technical Support Assistance 35
Appendix A. Resource Utilization 36
LatticeECP3-70 Utilization 36
LFE5UM-85 Utilization 36
LFE5UM5G-85 Utilization 36
Appendix B. Limitations 37
JESD204B 5G IP Core 37
Revision History 38

Figures

Figure 2.1. JESD204B Rx IP Core I/O 7
Figure 2.2. JESD204B Single Device Application 10
Figure 2.3. JESD204B IP Receiver Functions 11
Figure 2.4. JESD204B Rx 3G IP Core Block Diagram 11
Figure 2.5. JESD204B Rx 3G IP Core Block Lane Alignment Status 12
Figure 2.6. JESD204B Rx 3G IP Core User Interface 12
Figure 2.7. JESD204B Tx IP Core I/O Ports 14
Figure 2.8. JESD204B Tx 5G IP Core Block Diagram 17
Figure 3.1. Global Tab 20
Figure 3.2. Others Tab 20
Figure 3.3. PCS Tab 21
Figure 4.1. Clarity Designer Window 22
Figure 4.2. Starting Clarity Designer from Diamond Design Environment 23
Figure 4.3. Configuring JESD204B IP in Clarity Designer 24
Figure 4.4. JESD204B 5G IP Configuration Interface in Clarity Designer 25
Figure 4.5. Configuring PCS in JESD204B 5G IP Configuration Interface in Clarity Designer 25
Figure 4.6. Place and Generate JESD204B IP in Clarity Designer 26
Figure 4.7. Configuration Interface 27
Figure 4.8. JESD204B Core Directory Structure 28
Figure 4.9. Simulation Environment Block Diagram 31
Figure 4.10. Data Patterns Seen on User Interface in Simulation 31
Figure 4.11. IP Regeneration in Clarity Designer 33
Figure 5.1. HetNet and Small Cells Application 34

Tables

Table 1.1. JESD204B 3G IP Core Quick Facts 4
Table 1.2. JESD204B 5G IP Core Quick Facts 5
Table 2.1. Rx IP Core Pin Function Description 7
Table 2.2. Frame Boundary versus Octet Data at Rx User Interface for 3G IP Core (F=3) 12
Table 2.3. Frame Boundary versus Octet Data at Rx User Interface for 5G IP Core (F=3) 13
Table 2.4. Tx IP Core Pin Function Description 14
Table 2.5. Frame Boundary versus Octet Data at Rx User Interface for 3G IP Core (F=3) 17
Table 2.6. Frame Boundary versus Octet Data at Rx User Interface for 3G IP Core (F=3) 18
Table 3.1. JESD20B IP Core Parameters 19
Table 4.1. Files Generated in Clarity Designer 28
Table A.1. Resource Utilization 36
Table A.2. Resource Utilization 36
Table A.3. Resource Utilization 36
Table B.1. Core and Reference Design Minimum Device Support 37

1. Introduction

JEDEC Standard No. 204B (JESD204B) describes a serialized interface between data converters and logic devices. It contains the information necessary to allow designers to implement logic devices which can communicate with other devices (converters) that are compliant with the standard. Lattice Semiconductor's JESD204B 3G/5G IP Core offerings support both an Rx core (ADC to FPGA direction) and/or a Tx core (FPGA to DAC direction). The Rx and Tx cores can each be generated separately and with different parameters.

1.1. Quick Facts

Quick facts about the JESD204B IP Core for LatticeECP3™, ECP5™ and ECP5-5G™ devices.

Table 1.1. JESD204B 3G IP Core Quick Facts

Core Requirements	FPGA Families Supported	LatticeECP3	ECP5
Resource Utilization	Targeted Device	LFE3-70EA-6FN672C	LFE5UM-85F-8BG756C
	Max Data Rate	3 Gb/s	3 Gb/s
	Data Path Width	16 bits per lane, 32 bits total for 2 lanes	16 bits per lane, 32 bits total for 2 lanes
	LUTs	Rx:4886/Tx:651	Rx:2276/Tx:534
	sysMEM™ EBRs	Rx:2/Tx:0	Rx:2/Tx:0
	Registers	Rx:2174/Tx:266	Rx:2170/Tx:266
Design Tool Support	Lattice Implementation	Lattice Diamond® 3.2 IPexpress/Clarity Designer	Lattice Diamond® 3.2 IPexpress/Clarity Designer
	Synthesis	Synopsys® Synplify® Pro I-2013.09L	Synopsys® Synplify® Pro I-2013.09L
	Simulation	Aldec® Active-HDLTM 9.3 Lattice Edition	Aldec® Active-HDLTM 9.3 Lattice Edition
		Mentor Graphics® ModelSim® SE 6.5D	Mentor Graphics® ModelSim® SE 6.5D

Table 1.2. JESD204B 5G IP Core Quick Facts

Core Requirements	FPGA Families Supported	ECP5-5G
Resource Utilization	Targeted Device	LFE5UM5G-85F-8BG756C
	Max Data Rate	5 Gb/s
	Data Path Width	32 bits per lane, 64 bits total for 2 lanes
	LUTs	Rx:3468/Tx:1001
	sysMEM™ EBRs	Rx:0/Tx:0
	Registers	Rx:2477/Tx:621
Design Tool Support	Lattice Implementation	Lattice Diamond® 3.9 Clarity Designer
	Synthesis	Synopsys® Synplify® Pro L-2016.09L-1
	Simulation	Aldec® Active-HDLTM 10.3 Lattice Edition
		Mentor Graphics® ModelSim® SE 10.2C

1.2. Features

- Subsets of JEDEC Standard No. 204B(JESD204B.01) July 2011
- Rx core performs lane alignment based on Subclass 0 and Subclass 1
- Rx core performs frame alignment detection / monitoring and octet reconstruction
- Rx core performs user-enabled descrambling
- Rx core recovers link configuration parameters during initial lane synchronization and compares them to user selected parameters to generate a configuration mismatch error
- Tx core performs user-enabled scrambling
- Tx core generates initial lane alignment sequence
- Tx core performs alignment character generation
- Tx core sources link configuration data with user selected parameter values during initial lane synchronization sequence
- 16 bit(3G) or 32 bit(5G) fabric interface per channel for low core frequency
- One-shot frame/multi-frame boundary flags with one clock ahead of data make users easy to control the transition of the state machines for framer / de-framer

1.3. What is Not Supported

- Configuration of Rx core by link configuration parameters
- Octet to frame stream conversion in Rx core
- Frame to octet stream conversion in Tx core
- Verification of transport layer test samples within the Rx core
- Dynamic re-alignment is not supported. User can start a new CGS/ILA by driving the SYSREF high for 1 clock when alignment error is reported by the core
- Error reporting via SYNC~ interface
- Unexpected control character reporting during ILAS phase

1.4. Conventions

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

JESD204B IP Core pertains to both 3G and 5G IP packages. JESD204B 3G IP Core and JESD204B 5G IP Core are separate entities addressed in this document.

1.5. Data Ordering and Data Types

- The native JESD204B 3G IP Core operates on individual bytes only, so there is no byte ordering defined for the core. However, to minimize the FPGA fabric clock frequency, the data path within the IP cores is two bytes wide. The byte in the lower 8 bits of a 16-bit pair is the “low” byte and the byte in the upper 8 bits is the “high” byte. On the interface to user’s logic, each lane is a 16 bit interface. Within each 16-bit pair of bytes, the high byte is the first byte transmitted or received, and the low byte is the second byte transmitted or received. On the interface between the IP cores and the PCS/SERDES, the low byte is the first byte of the pair and the high byte is the second byte of the pair.
- For JESD204B 5G IP Core, in order to maintain the FPGA fabric clock frequency, the data path within the IP cores is doubled. This means that the data path will be four bytes wide when 5G data rate is selected. The same convention of data ordering within each two-byte pair is followed while between the two-byte pair, the lower two bytes are the first ones transmitted or received.
- The most significant bit within data bytes is bit 7.

1.6. Signal Names

Signal names that end with “_n” are active low.

2. Functional Description

The JESD204B IP core implements the functionality described in the JEDEC standard. The following sections describe the Rx and Tx cores.

2.1. Rx Core

Figure 2.1 shows the I/O ports on the Rx IP core.

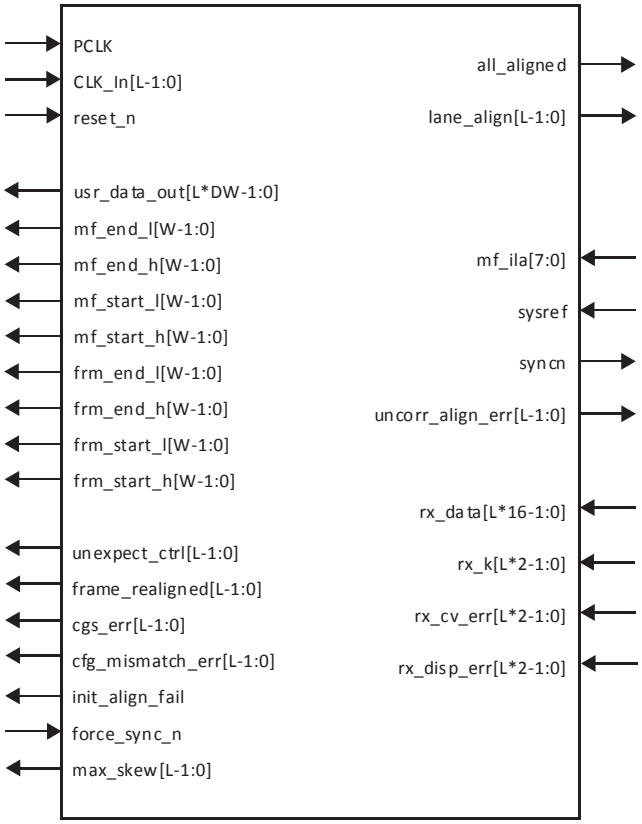

Figure 2.1. JESD204B Rx IP Core I/O

Table 2.1. Rx IP Core Pin Function Description

Pin Name	Direction	Function description
PCLK	Input	Clock for Tx core configured to run at maximum of 3 Gb/s. 1:1 clock period ratio with CLK_In. See Note.
PCLK_div2	Input	Clock for Tx core configured to run at maximum of 5 Gb/s. 1:2 clock period ratio with CLK_In. This clock is only available for 5G IP Core. See Note.
CLK_In[L-1:0]	Input	Lane clock from Rx PCS/SERDES Channel PCLK.
reset_n	Input	Active low core reset.
usr_data_out[L*DW-1]	Output	Aligned data output to user interface, DW = 16-bits per lane for 3G or 32-bits per lane for 5G.

Table 2.1. Rx IP Core Pin Function Description *(continued)*

Pin Name	Direction	Function description
mf_end_l[W-1:0]	Output	For 3G: W = 1. Multiframe boundary flag to indicate the current multiframe will end on bit[7:0] of the next aligned lane data. For 5G: W = 2. Multiframe boundary flag mf_end_l[0] to indicate the current multiframe will end on bit[7:0] or mf_end_l[1] to indicate the current multiframe will end on bit[23:16] of the next aligned lane data.
mf_end_h[W-1:0]	Output	For 3G: W = 1. Multiframe boundary flag to indicate the current multiframe will end on bit[15:8] of the next aligned lane data. For 5G: W = 2. Multiframe boundary flag mf_end_h[0] to indicate the current multiframe will end on bit[15:8] or mf_end_h[1] to indicate the current multiframe will end on bit[31:24] of the next aligned lane data.
mf_start_l[W-1:0]	Output	For 3G: W = 1. Multiframe boundary flag to indicate the current multiframe will start on bit[7:0] of the next aligned lane data. For 5G: W = 2. Multiframe boundary flag mf_start_l[0] to indicate the current multiframe will start on bit[7:0] or mf_start_l[1] to indicate the current multiframe will start on bit[23:16] of the next aligned lane data.
mf_start_h[W-1:0]	Output	For 3G: W = 1. Multiframe boundary flag to indicate the current multiframe will start on bit[15:8] of the next aligned lane data. For 5G: W = 2. Multiframe boundary flag mf_start_h[0] to indicate the current multiframe will start on bit[15:8] or mf_start_h[1] to indicate the current multiframe will start on bit[31:24] of the next aligned lane data.
frm_end_l[W-1:0]	Output	For 3G: W = 1. Frame boundary flag to indicate the current frame will end on bit[7:0] of the next aligned lane data. For 5G: W = 2. Frame boundary flag frm_end_l[0] to indicate the current frame will end on bit[7:0] or frm_end_l[1] to indicate the current frame will end on bit[23:16] of the next aligned lane data.
frm_end_h[W-1:0]	Output	For 3G: W = 1. Frame boundary flag to indicate the current frame will end on bit[15:8] of the next aligned lane data. For 5G: W = 2. Frame boundary flag frm_end_h[0] to indicate the current frame will end on bit[15:8] or frm_end_h[1] to indicate the current frame will end on bit[31:24] of the next aligned lane data.

Table 2.1. Rx IP Core Pin Function Description *(continued)*

Pin Name	Direction	Function description
frm_start_l[W-1:0]	Output	For 3G: W = 1. Frame boundary flag to indicate the current frame will start on bit[7:0] of the next aligned lane data. For 5G: W = 2. Frame boundary flag frm_start_l[0] to indicate the current frame will start on bit[7:0] or frm_start_l[1] to indicate the current frame will start on bit[23:16] of the next aligned lane data.
frm_start_h[W-1:0]	Output	For 3G: W = 1. Frame boundary flag to indicate the current frame will start on bit[15:8] of the next aligned lane data. For 5G: W = 2. Frame boundary flag frm_start_h[0] to indicate the current frame will start on bit[15:8] or frm_end_h[1] to indicate the current frame will end on bit[31:24] of the next aligned lane data.
unexpect_ctrl[L-1:0]	Output	Active high lane status signal to indicate an unexpected control character is received but not fall on the end of a frame or multiframe.
frame_realigned[L-1:0]	Output	Inherited from JESD204A Core. Not implemented in the current version of JESD204B core.
cgs_error	Output	Over 3 continuous disparity or coding violation errors detected. A new initialization sequence will be started.
cfg_mismatch_err[L-1:0]	Output	Configuration mismatch error, status signal per lane to indicate a mismatch between received parameters over link and IP configured parameters.
init_align_fail	Output	Initial lane alignment fails. all_aligned is not detected at the end of ILA.
force_sync_n	Input	Force Rx core to initiate a new synchronization sequence.
max_skew[L-1:0]	Output	Active high status signal per lane to indicate the alignment FIFO is over flow for error of some other lane.
lane_align[L-1:0]	Output	Active high status signal per lane to indicate a K28.0 is received during code group synchronization and ready to be aligned with the other lanes to the next multiframe boundary.
all_aligned	Output	Active high status signal to indicate all lanes are aligned to the multiframe boundary.
mf_ila[7:0]	Input	Sets the number of multiframes in initialization lane alignment sequence from 4-256. Currently hardcoded to 4.
sysref	Input	Generated in system level to indicate the start of a multiframe.
syncn	Output	Used in core as a request signal to perform a CGS/ILA after power on or link error (disparity/coding violation) is detected. User can drive force_sync_n from high to low to restart a new initialization sequence if necessary.
rx_data[L*16-1:0]	Input	Receive data from PCS/SERDES, 16 bits per lane.
rx_disp_err[L*2-1:0]	Input	Receive disparity error from PCS/SERDES, 2 bits per lane.

Table 2.1. Rx IP Core Pin Function Description *(continued)*

Pin Name	Direction	Function description
rx_cv_err[L*2-1:0]	Input	Receive coding violation error from PCS/SERDES, 2 bits per lane.
rx_k[L*2-1:0]	Input	Receive control character from PCS/SERDES, 2 bits per lane.

Note: Frame clock and octet clock used by framer/de-framer, if necessary, can be derived from the PCLK/PCLK_div2 as shown below:

- 3G: Frame clock = 2*PCLK/F(octets per frame), Octet clock = 2*PCLK=F*FCLK(Frame clock)
- 5G: Frame clock = PCLK/F(octets per frame), Octet clock = PCLK=F*FCLK(Frame clock)

Referring to Figure 2.2, in the Rx direction, serial I/F data is received by the logic device (FPGA) over one or more serial I/F bit-stream lanes. The number of lanes is equal to the parameter L. The Rx core finds the framing information for each lane and aligns all lanes to the next multi-frame boundary. After system power on or when link error or alignment errors are found in the Tx core, it sets the SYNC_N_RX_CORE signal active. This prompts the ADC converter to send a new initialization sequence. Once the individual lanes are aligned, the Rx core places the aligned receive data from each lane to the application logic. In the same way, the DAC converter drives SYNC_N_DAC active to request an initialization sequence to Tx core. SYSREF is used to synchronize the LMFC or multiframe boundary in both FPGA logic device and converters.

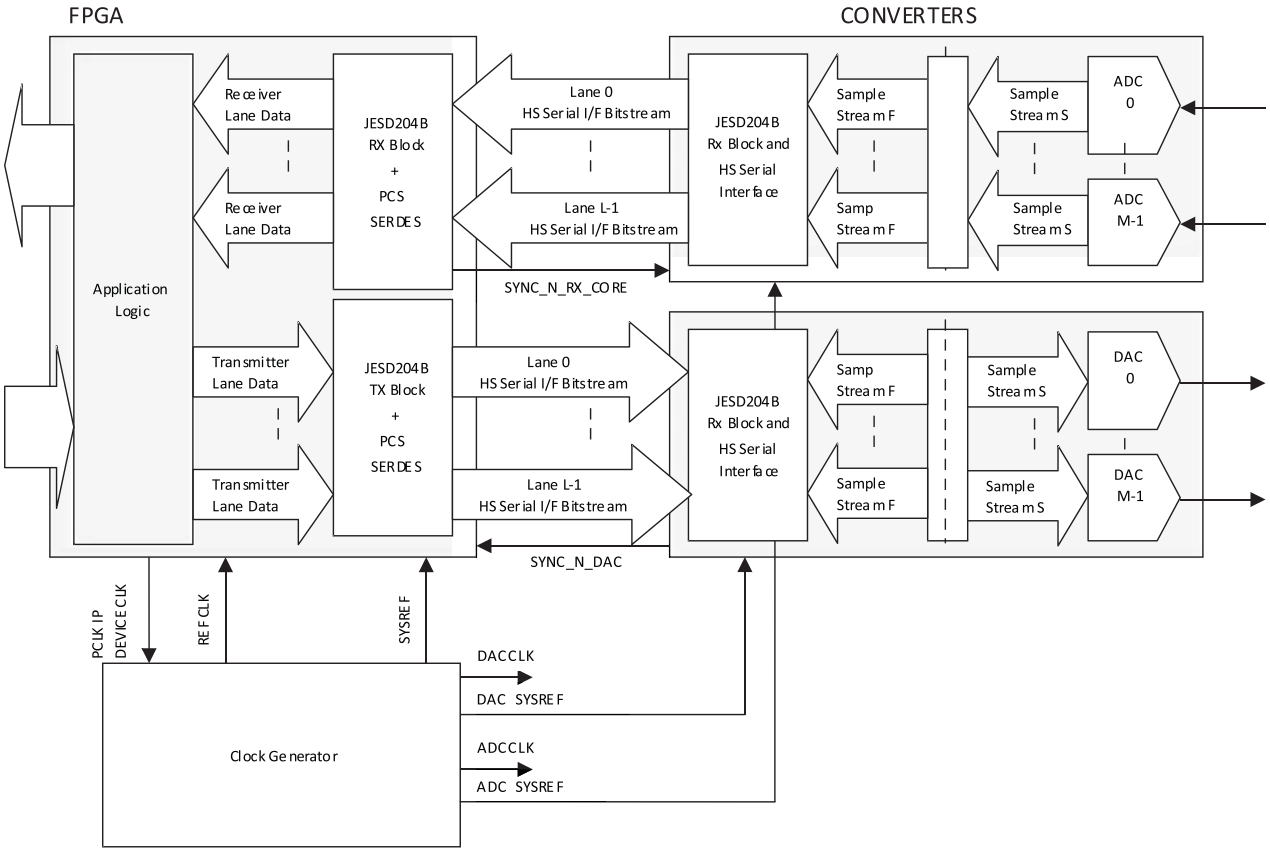

Figure 2.2. JESD204B Single Device Application

Figure 2.3 shows the functions performed internal to the Rx core. The Serial Bit Streams of the JESD204b links go through the PCS/SERDES block which performs 8b/10b decoding. The data is then passed to the Rx IP core which performs the synchronization and alignment functions, followed by descrambling. The IP core then sources the aligned data to the FPGA fabric where additional logic can perform the octet to frame stream conversion.

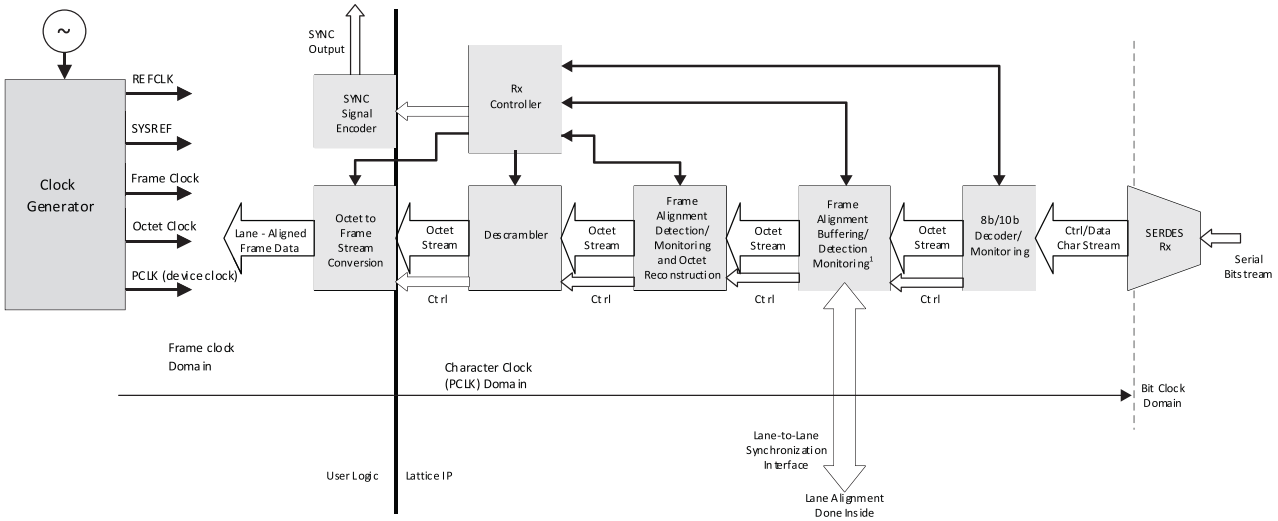

Figure 2.3. JESD204B IP Receiver Functions

Figure 2.4 shows the functions performed by the Rx IP core for a 2-lane configuration, although the following Discussion applies equally well to configurations with any number of lanes. As shown in the figure, two lanes of serial data are received by the PCS/SERDES which performs 8b/10b decoding and word alignment. The data is passed as two 22-bit buses to the Rx IP core which first performs code group synchronization as specified by Figure 44 of the JESD204B Standard. Each of the 18-bit buses per lane contains two 8-bit data words and two control signals two coding violation error signals and two disparity error signals. All the 22 bits bus data are dual-buffered in Tx core respectively for initial lane alignment and realignment to the LMFC boundary. The byte in the lower bit positions of the 2-byte wide bus is received on the serial link first.

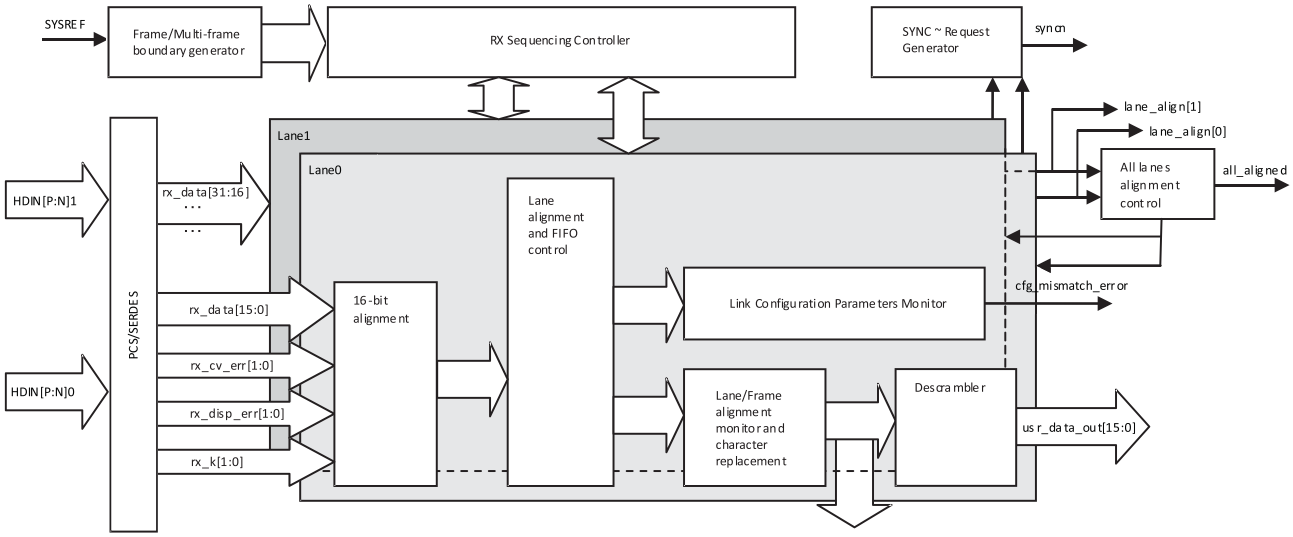

Figure 2.4. JESD204B Rx 3G IP Core Block Diagram

When power on or a loss of code group synchronization is detected on any lane, a sync_request is passed to the SYNC_gen block which asserts the SYNC output to the transmitting DAC device(s). You can also drive force_sync_n with a falling edge to start a new initialization sequence in case a realignment is necessary.

Lane alignment and FIFO control block in each lane buffers the input data and align it with the other lanes to the LMFC boundary with the assistance of the all lanes alignment control block in core level. A signal all_aligned will be driven high if all the lanes are aligned.

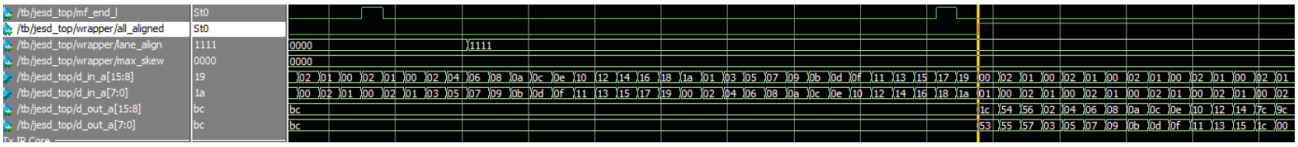

Figure 2.5. JESD204B Rx 3G IP Core Block Lane Alignment Status

Link configuration parameter monitor module monitors the configuration parameters transmitted by Tx Block (a JESD204B ADC) over the link during initial lane alignment and compares them with the corresponding parameters of Tx core itself. A signal cfg_mismatch_error will be driven to high if there is any inconsistence between them. Lane/Frame alignment monitor and character replacement block processes the received data based on rules defined in section 5.3.3.4 and 5.3.3.6 of the JESD204B standard. The monitor block in each lane can report an alignment error via signal unexpect_ctrl if a control character is captured but it does not fall on the end of a frame or multiframe. The current version of the Rx IP can monitor the data flow and replace the character back to the original data, but not do any alignment correction except driving high unxepect_ctrl pin. You can drive force_sync_n pin with a falling edge to restart an initialization lane alignment sequence in such case. A descrambler block with bypass control option is used to descramble the data flow in case the scramble option of ADC is enabled. The descrambler block can be configured to bypass when the scrambler in ADC is disabled. Following the descrambler, the data is output from the Rx core as a single bus having L*16 bits. Data for each lane occupies 16 bits of the overall bus. The upper 8 bits out of each group of 16 (high-byte) are the first byte of the sequence, the lower 8 bits of 16 (low-byte) are the next byte. A group of frame/multiframe boundary signals are provided to help you to process the received data. The same is true when using 5G IP Core where the bus width for each lane will increase to 32 bits. The boundary flags are designed to be 1 clock ahead of the received data.

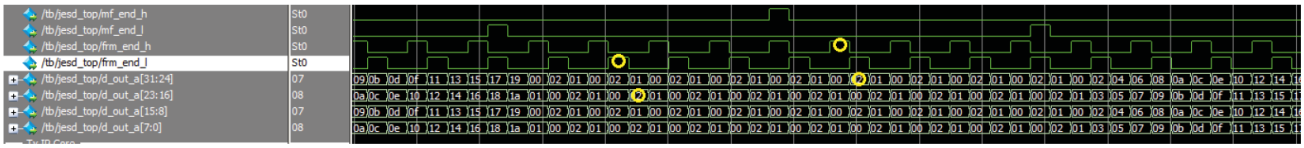

Figure 2.6. JESD204B Rx 3G IP Core User Interface

Table 2.2. Frame Boundary versus Octet Data at Rx User Interface for 3G IP Core (F=3)

frm_end_h	**1**	**0**	**0**	**1**
frm_end_l	**0**	**1**	**0**	**0**
d_out_a[31:24] - lane 1 (high byte)	byte 0	byte 2	byte 1	byte 0
d_out_a[23:16] - lane 1 (low byte)	byte 1	byte 0	byte 2	byte 1
d_out_a[15:8] - lane 0 (high byte)	byte 0	byte 2	byte 1	byte 0
d_out_a[7:0] - lane 0 (low byte)	byte 1	byte 0	byte 2	byte 1

Table 2.3. Frame Boundary versus Octet Data at Rx User Interface for 5G IP Core (F=3)

frm_end_h[1]	**0**	**0**	**1**	**0**
frm_end_l [1]	**0**	**1**	**0**	**0**
frm_end_h[0]	**0**	**1**	**0**	**0**
frm_end_l [0]	**1**	**0**	**0**	**1**
d_out_a[63:56] - lane 1 (high byte)	byte 2	byte 0	byte 1	byte 2
d_out_a[55:48] - lane 1 (low byte)	byte 0	byte 1	byte 2	byte 0
d_out_a[47:40] - lane 1 (high byte)	byte 0	byte 1	byte 2	byte 0
d_out_a[39:32] - lane 1 (low byte)	byte 1	byte 2	byte 0	byte 1
d_out_a[31:24] - lane 0 (high byte)	byte 2	byte 0	byte 1	byte 2
d_out_a[23:16] - lane 0 (low byte)	byte 0	byte 1	byte 2	byte 0
d_out_a[15:8] - lane 0 (high byte)	byte 0	byte 1	byte 2	byte 0
d_out_a[7:0] - lane 0 (low byte)	byte 1	byte 2	byte 0	byte 1

Logic external to the IP core will be needed to assemble the octets of the aligned data from the Rx IP core into the samples and control bits according to the user's sample stream mapping. This is the reason it is necessary for the Rx IP core to provide the boundary signals.

2.2. Tx Core

Figure 2.7 shows the I/O ports on the Tx IP core.

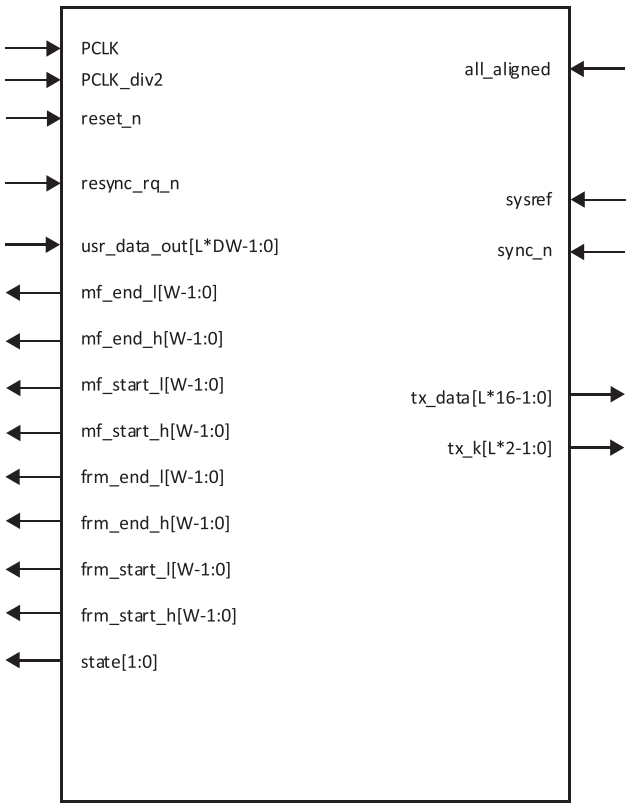

Figure 2.7. JESD204B Tx IP Core I/O Ports

Table 2.4. Tx IP Core Pin Function Description

Pin Name	Direction	Function Description
PCLK	Input	Clock for Tx core. See Note.
PCLK_div2	Input	Clock for Tx core configured to run at maximum of 5 Gb/s. 1:2 clock period ratio with PCLK. This clock is only available for 5G IP Core. See Note.
reset_n	Input	Active low core reset.
usr_data_in[L*DW-1]	Input	Aligned data input from user interface, DW = 16-bits per lane for 3G or 32-bits per lane for 5G.
mf_end_l[W-1:0]	Output	For 3G: W = 1. Multiframe boundary flag to indicate the current multiframe will end on bit[7:0] of the next aligned lane data. For 5G: W = 2. Multiframe boundary flag mf_end_l[0] to indicate the current multiframe will end on bit[7:0] or mf_end_l[1] to indicate the current multiframe will end on bit[23:16] of the next aligned lane data.

Table 2.4. Tx IP Core Pin Function Description *(continued)*

Pin Name	Direction	Function Description
mf_end_h[W-1:0]	Output	For 3G: W = 1. Multiframe boundary flag to indicate the current multiframe will end on bit[15:8] of the next aligned lane data. For 5G: W = 2. Multiframe boundary flag mf_end_h[0] to indicate the current multiframe will end on bit[15:8] or mf_end_h[1] to indicate the current multiframe will end on bit[31:24] of the next aligned lane data.
mf_start_l[W-1:0]	Output	For 3G: W = 1. Multiframe boundary flag to indicate the current multiframe will start on bit[7:0] of the next aligned lane data. For 5G: W = 2. Multiframe boundary flag mf_start_l[0] to indicate the current multiframe will start on bit[7:0] or mf_start_l[1] to indicate the current multiframe will start on bit[23:16] of the next aligned lane data.
mf_start_h[W-1:0]	Output	For 3G: W = 1. Multiframe boundary flag to indicate the current multiframe will start on bit[15:8] of the next aligned lane data. For 5G: W = 2. Multiframe boundary flag mf_start_h[0] to indicate the current multiframe will start on bit[15:8] or mf_start_h[1] to indicate the current multiframe will start on bit[31:24] of the next aligned lane data.
frm_end_l[W-1:0]	Output	For 3G: W = 1. Frame boundary flag to indicate the current frame will end on bit[7:0] of the next aligned lane data. For 5G: W = 2. Frame boundary flag frm_end_l[0] to indicate the current frame will end on bit[7:0] or frm_end_l[1] to indicate the current frame will end on bit[23:16] of the next aligned lane data.
frm_end_h[W-1:0]	Output	For 3G: W = 1. Frame boundary flag to indicate the current frame will end on bit[15:8] of the next aligned lane data. For 5G: W = 2. Frame boundary flag frm_end_h[0] to indicate the current frame will end on bit[15:8] or frm_end_h[1] to indicate the current frame will end on bit[31:24] of the next aligned lane data.

Table 2.4. Tx IP Core Pin Function Description *(continued)*

Pin Name	Direction	Function Description
frm_start_l[W-1:0]	Output	For 3G: W = 1. Frame boundary flag to indicate the current frame will start on bit[7:0] of the next aligned lane data. For 5G: W = 2. Frame boundary flag frm_start_l[0] to indicate the current frame will start on bit[7:0] or frm_start_l[1] to indicate the current frame will start on bit[23:16] of the next aligned lane data.
frm_start_h[W-1:0]	Output	For 3G: W = 1. Frame boundary flag to indicate the current frame will start on bit[15:8] of the next aligned lane data. For 5G: W = 2. Frame boundary flag frm_start_h[0] to indicate the current frame will start on bit[15:8] or frm_end_h[1] to indicate the current frame will end on bit[31:24] of the next aligned lane data.
resync_rq_n	Input	Falling edge of resync_rq_n forces Tx core back to CGS state to start a new CGS/ILAS.
state[1:0]	Output	State of Tx core control block.
mf_ila[7:0]	Input	Sets the number of multiframes in initialization lane alignment sequence from 4-256. Currently hardcoded to 4.
sysref	Input	Generated in system level to indicate the start of a multiframe.
sync_n	Input	Initial sequence request input from DAC.
tx_data[L*16-1:0]	Output	Transmitter data to PCS/SERDES, 16 bits per lane.
tx_k[L*2-1:0]	Output	Transmitter control character to PCS/SERDES, 2 bits per lane.

Note: Frame clock and octet clock used by framer/de-framer if necessary can be derived from the PCLK/PCLK_div2 as below:

- 3G: Frame clock = 2*PCLK/F(octets per frame), Octet clock = 2*PCLK=F*FCLK(Frame clock)
- 5G: Frame clock = PCLK/F(octets per frame), Octet clock = PCLK=F*FCLK(Frame clock)

Figure 2.8 shows a block diagram of the Tx IP core. Frame data from the user is put to the core at a rate of two octets per clock cycle per lane. The lane data then passes through the scrambler which can be configured to bypass. Then alignment and framing characters and configuration information for Lane/Frame alignment and ILA sequence are inserted into the data. A controller selects either frame data or initialization sequence data to send to the PCS/SERDES where 8b/10b encoding is performed.

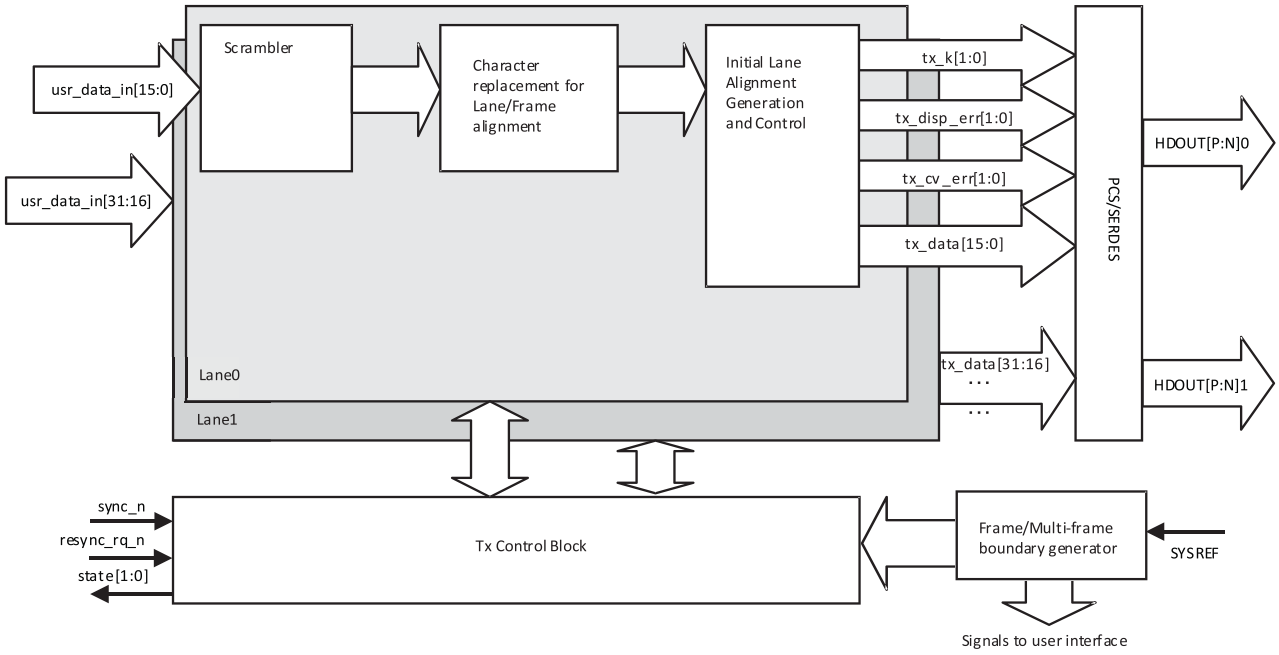

Figure 2.8. JESD204B Tx 5G IP Core Block Diagram

Frame data supplied by the user must be aligned to the octet counter outputs from the Tx IP core, similar to the way the Rx core outputs data aligned to the receive side octet counters. Table 2.5 and Table 2.6 show how the transmit data input to the Tx IP core by the user must be aligned to the frame/multiframe boundary flags. This example is for a 2- lane Tx IP core with F=3.

Table 2.5. Frame Boundary versus Octet Data at Rx User Interface for 3G IP Core (F=3)

frm_end_h	**1**	**0**	**0**	**1**
frm_end_l	**0**	**1**	**0**	**0**
d_in_a[31:24] - lane 1 (high byte)	byte 0	byte 2	byte 1	byte 0
d_in_a[23:16] - lane 1 (low byte)	byte 1	byte 0	byte 2	byte 1
d_in_a[15:8] - lane 0 (high byte)	byte 0	byte 2	byte 1	byte 0
d_in_a[7:0] - lane 0 (low byte)	byte 1	byte 0	byte 2	byte 1

Table 2.6. Frame Boundary versus Octet Data at Rx User Interface for 3G IP Core (F=3)

frm_end_h[1]	**0**	**0**	**1**	**0**
frm_end_l [1]	**0**	**1**	**0**	**0**
frm_end_h[0]	**0**	**1**	**0**	**0**
frm_end_l [0]	**1**	**0**	**0**	**1**
d_in_a[63:56] - lane 1 (high byte)	byte 2	byte 0	byte 1	byte 2
d_in_a[55:48] - lane 1 (low byte)	byte 0	byte 1	byte 2	byte 0
d_in_a[47:40] - lane 1 (high byte)	byte 0	byte 1	byte 2	byte 0
d_in_a[39:32] - lane 1 (low byte)	byte 1	byte 2	byte 0	byte 1
d_in_a[31:24] - lane 0 (high byte)	byte 2	byte 0	byte 1	byte 2
d_in_a[23:16] - lane 0 (low byte)	byte 0	byte 1	byte 2	byte 0
d_in_a[15:8] - lane 0 (high byte)	byte 0	byte 1	byte 2	byte 0
d_in_a[7:0] - lane 0 (low byte)	byte 1	byte 2	byte 0	byte 1

3. Parameter Settings

Table 3.1 shows the user parameters used to generate the Rx or Tx IP core. Parameters F, K, and L are the only parameters which affect the core generation. For the Tx core, all parameter values selected by the user in the user interface during Tx core generation are sent to the link configuration data fields during the initialization sequence. For the Rx core, all parameters received in the link configuration data fields are compared to the user selected parameter values after the initialization sequence is completed.

Table 3.1. JESD20B IP Core Parameters

Parameter	Effects Core Generation	Compared to Configuration Values Received During Link Initialization	Description
F	Yes	Yes	Number of octets per frame
K	Yes	Yes	Number of frames per multiframe
L	Yes	Yes	Number of lanes on the JESD204B interface
CF		Yes	Number of control words per frame clock cycle per link
CS		Yes	Number of control bits per conversion sample
HD		Yes	High-density mode, samples are allowed to be divided across multiple lanes
N		Yes	Number of bits per sample
N’		Yes	Number of bits per sample after padding to nibble groups
M		Yes	Number of converter devices (ADC)
S		Yes	Number of samples per converter per frame

3.1. Configuring JESD204B Core in IPexpress

When generating the JESD204B Core using IPexpress, the Global tab of the user interface shown in Table 3.1, allows the core type and F, K, or L parameters to be selected. You can select an Rx Only core if the core is configured to be connected with a JESD204B A/D converter. Tx Only needs to be selected if it is configured to be connected with a JESD204B D/A converter. Only combinations supported by the IP core are allowed to be entered. You can also disable the scrambler/descrambler when the core is configured.

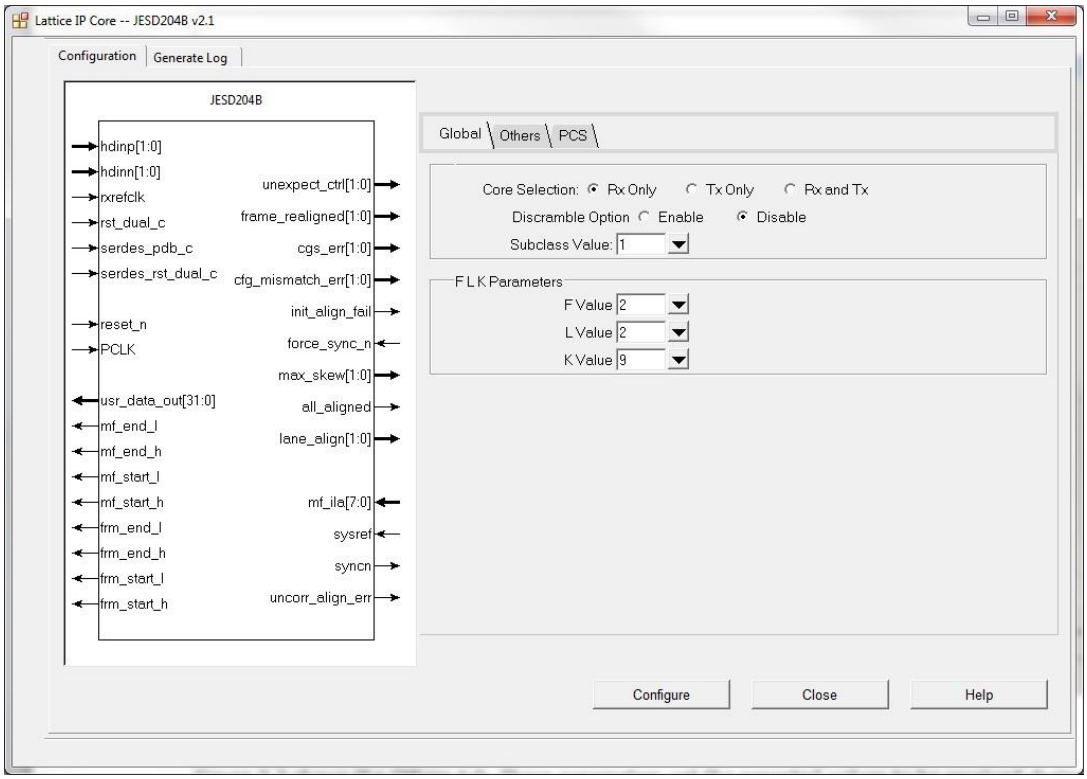

Figure 3.1. Global Tab

Figure 3.2 shows the Others tab. These parameters set the expected values to be received during link configuration. Mismatches result in the cfg_mismatch_err being asserted following initialization.

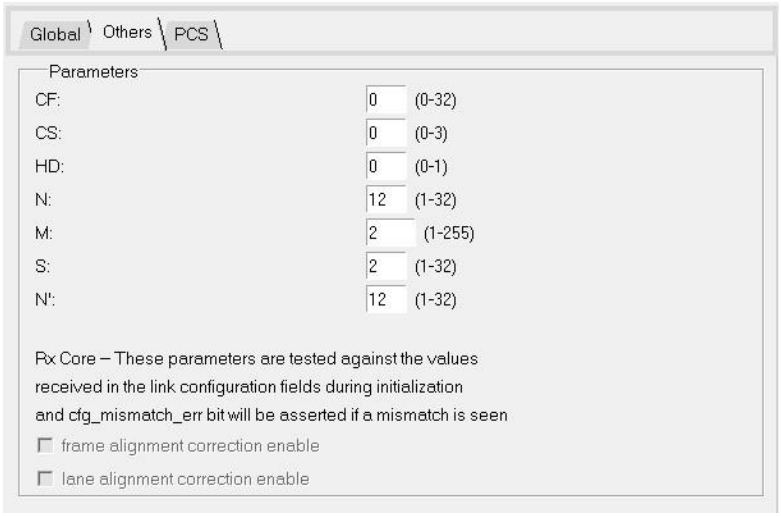

Figure 3.2. Others Tab

3.2. Configuring JESD204B Core in Clarity Designer

Clarity Designer requires the configured core to be connected with PCS/SERDES during core generation. For this purpose, a PCS tab is added to provide parameters required for PCS/SERDES as shown in Figure 3.3.

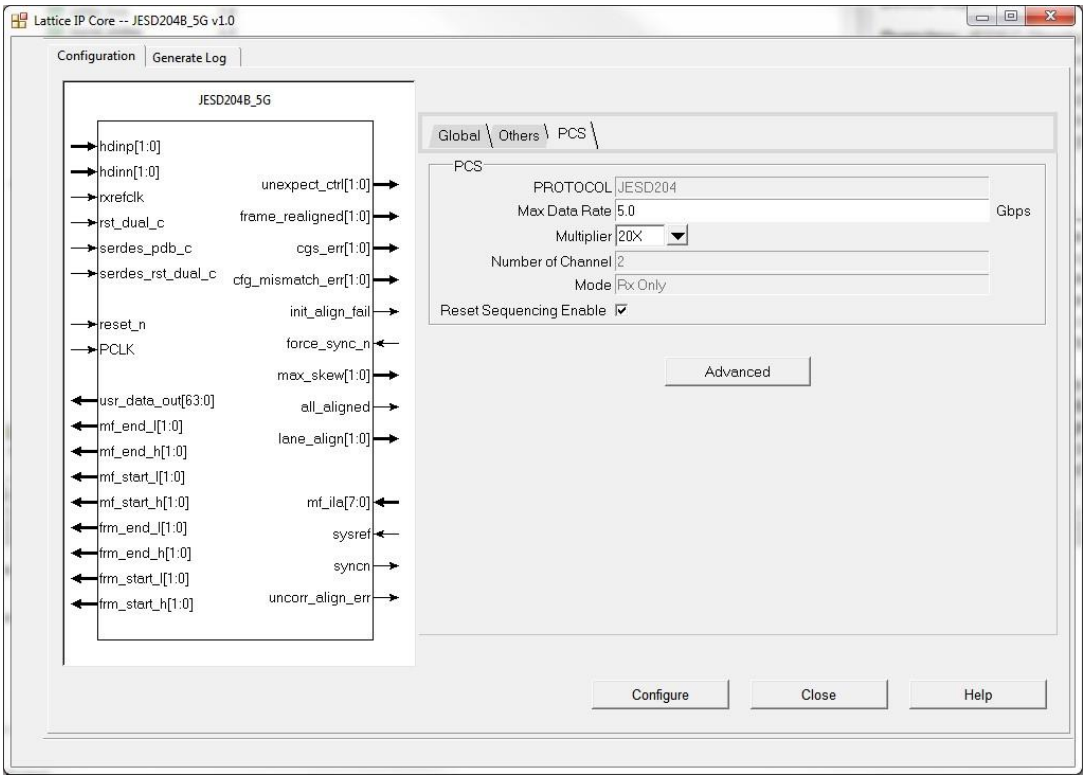

Figure 3.3. PCS Tab

4. IP Core Generation and Evaluation

This chapter provides information on how to generate the Lattice JESD204B IP core using the Diamond Clarity Designer or IPexpress tool, and how to include the core in a top-level design.

4.1. Licensing the IP

An IP core-specific license is required to enable full, unrestricted use of the JESD204B IP core in a complete, top- level design. You can get the license after the IP is purchased. Please find your local Lattice Sales Office at:

www.latticesemi.com/Buy/SalesLocator.aspx.

You may download and generate the JESD204B IP core and fully evaluate the core through functional simulation and implementation (synthesis, map, place and route) without an IP license. The JESD204B IP core also supports Lattice's IP hardware evaluation capability, which makes it possible to create versions of the IP core that operate in hardware for a limited time (approximately four hours) without requiring an IP license. See Hardware Evaluation for further details. However, a 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 JESD204B IP core 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 core is installed, the IP core will be available in the Clarity Designer user interface as shown in Figure 4.1.

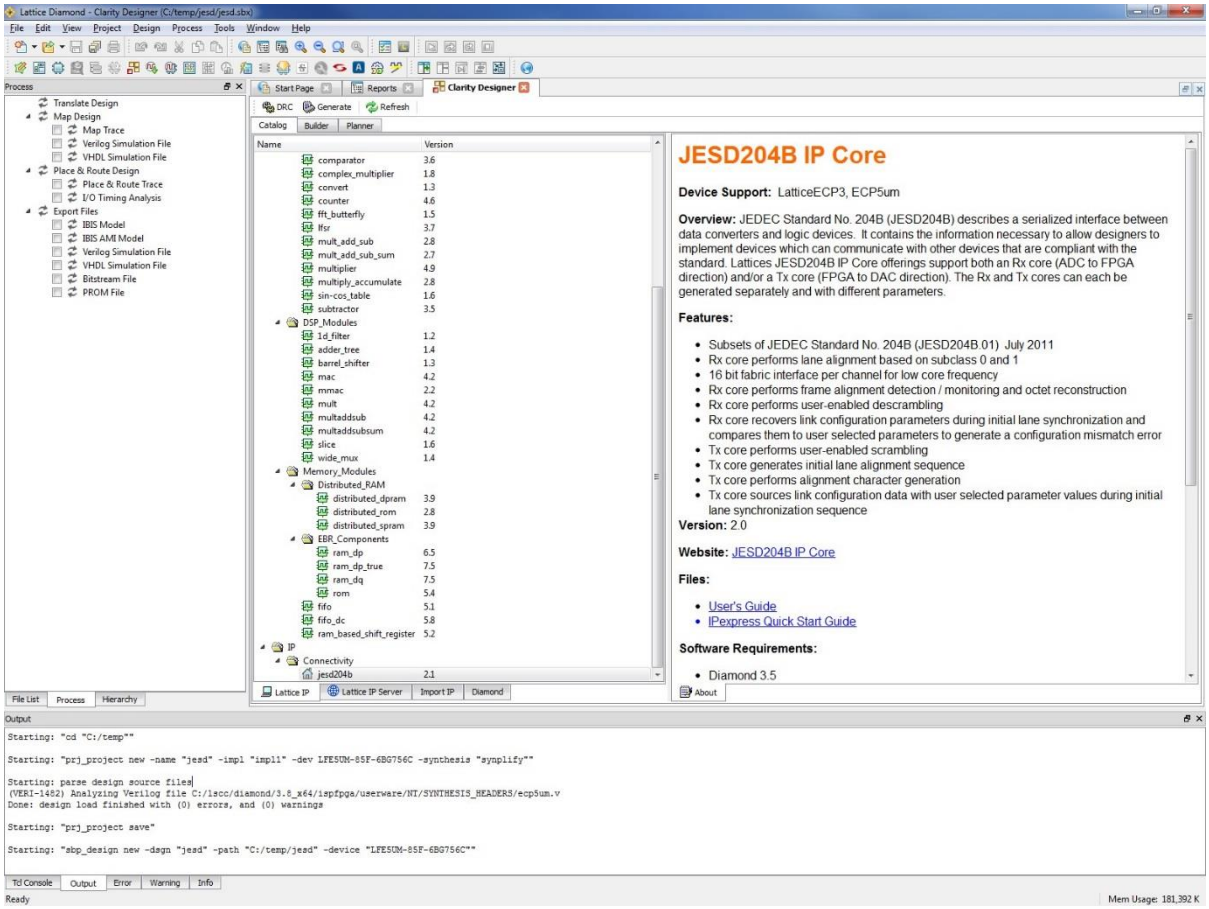

Figure 4.1. Clarity Designer Window

4.3. Creating IP in Clarity Designer

Clarity Designer is a tool used to customize modules and IPs and place them into the device's architecture. Besides configuration and generation of modules and IPs, Clarity Designer can also create a top module template in which all generated modules and IPs are instantiated.

The following describes the procedure for generating JESD204B IP in Clarity Designer.

Clarity Designer can be started from the Diamond design environment.

To start Clarity Designer:

1. Create a new empty Diamond project for ECP5UM/ECP5UM5G family devices.
2. From the Diamond main window, choose **Tools** > **Clarity Designer**, or click  in the Diamond toolbox. The Clarity Designer project dialog box is displayed.
3. Select and/or fill out the following items as shown in Figure 4.2:
   - **Create new Clarity design** — Choose to create a new Clarity Design project directory in which the JESD204B IP will be generated.
   - **Design Location** — Clarity Design project directory path.
   - **Design Name** — Clarity Design project name.
   - **HDL Output** — Hardware Description Language Output Format (Verilog).

   The Clarity Designer project dialog box also allows you to open an existing Clarity Designer project by selecting the following:
   - **Open Clarity design** — Open an existing Clarity Design project.
   - **Design File** — Name of existing Clarity Design project file with .sbx extension.

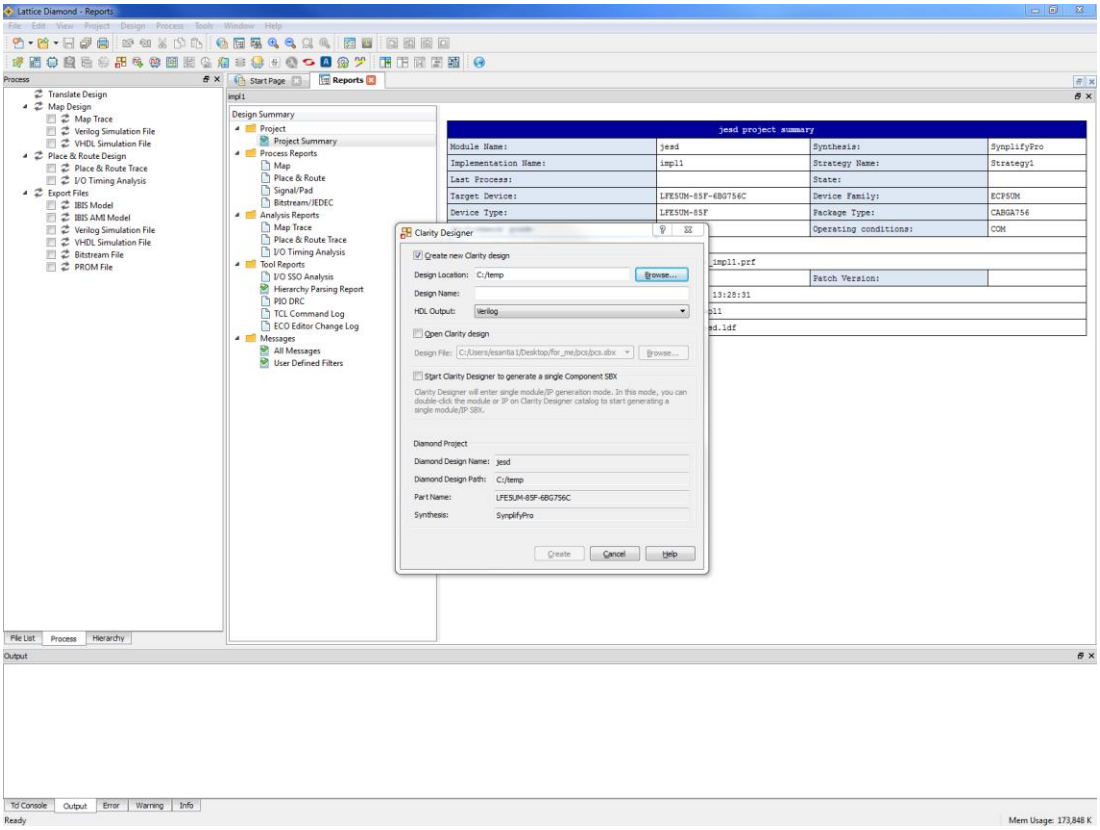

Figure 4.2. Starting Clarity Designer from Diamond Design Environment

4. Click the **Create** button. A new Clarity Designer project is created.

To configure the JESD204B IP in Clarity Designer:

1. Double-click **JESD204B 5G** in the IP list of the System/Planner view. The **jesd204b_5g** dialog box is displayed as shown in Figure 4.3.

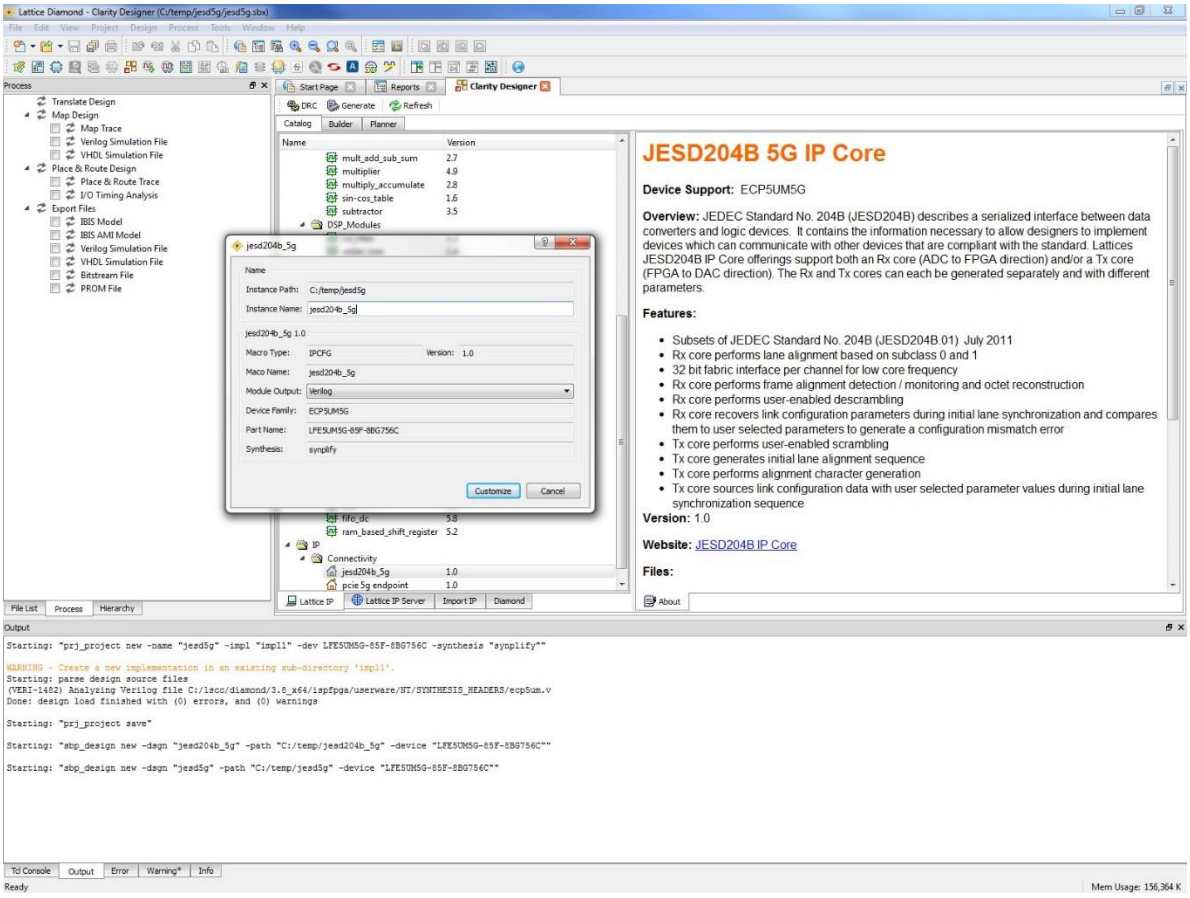

Figure 4.3. Configuring JESD204B IP in Clarity Designer

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

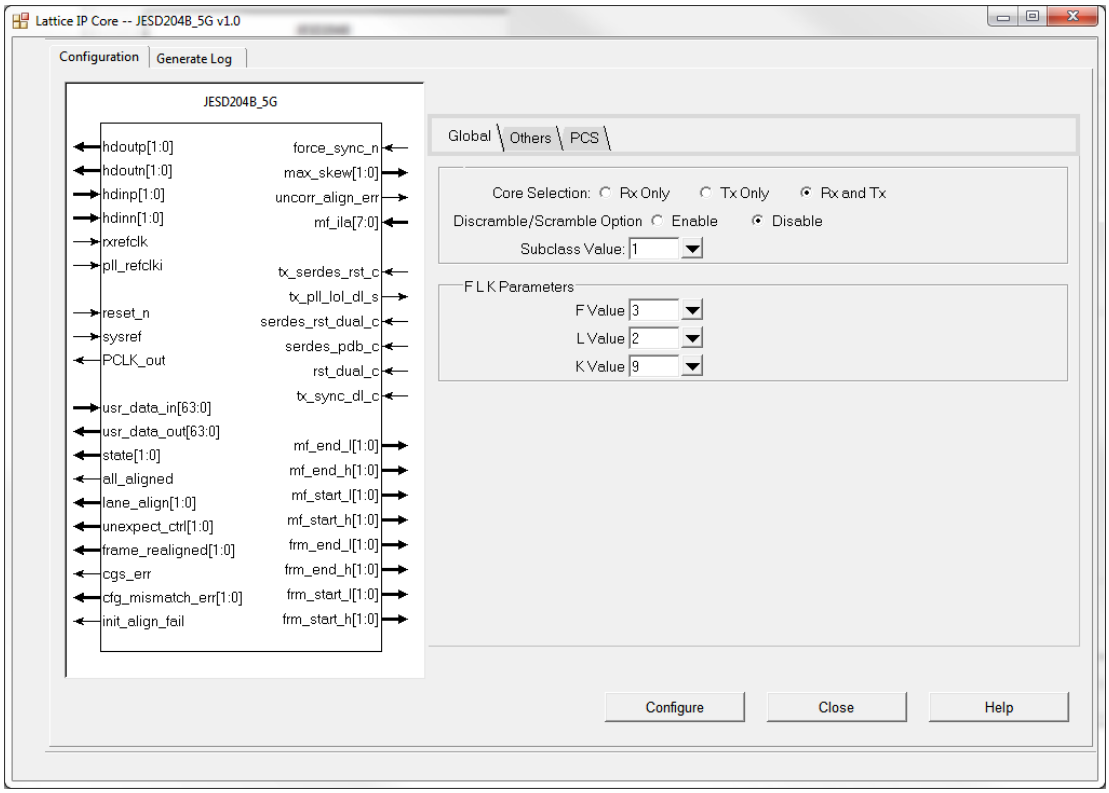

Figure 4.4. JESD204B 5G IP Configuration Interface in Clarity Designer

4. To configure the PCS for the IP, click the **Advanced** button in PCS tab as shown in Figure 4.5.

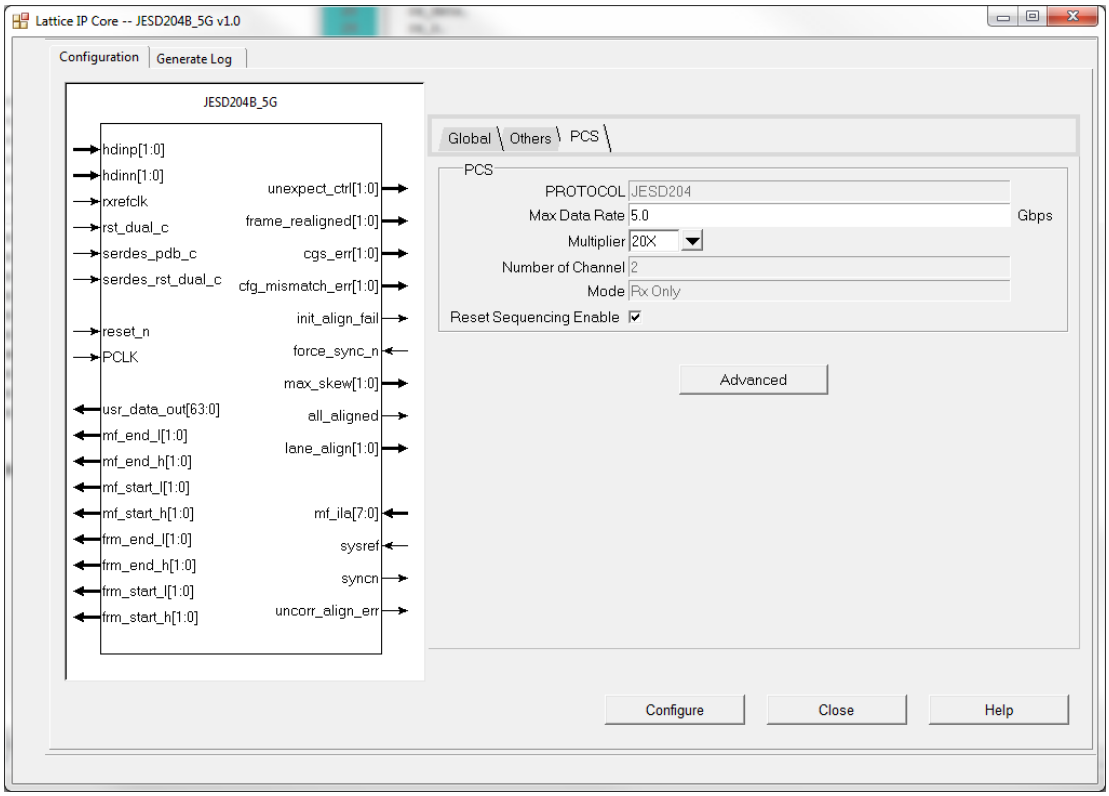

Figure 4.5. Configuring PCS in JESD204B 5G IP Configuration Interface in Clarity Designer

5. Click the **Configure** button after the required parameters are selected.
6. Click **Close**.
7. Click the **Planner** tab in the Clarity Designer user interface as shown in Figure 4.6. The configured IP instance is shown.

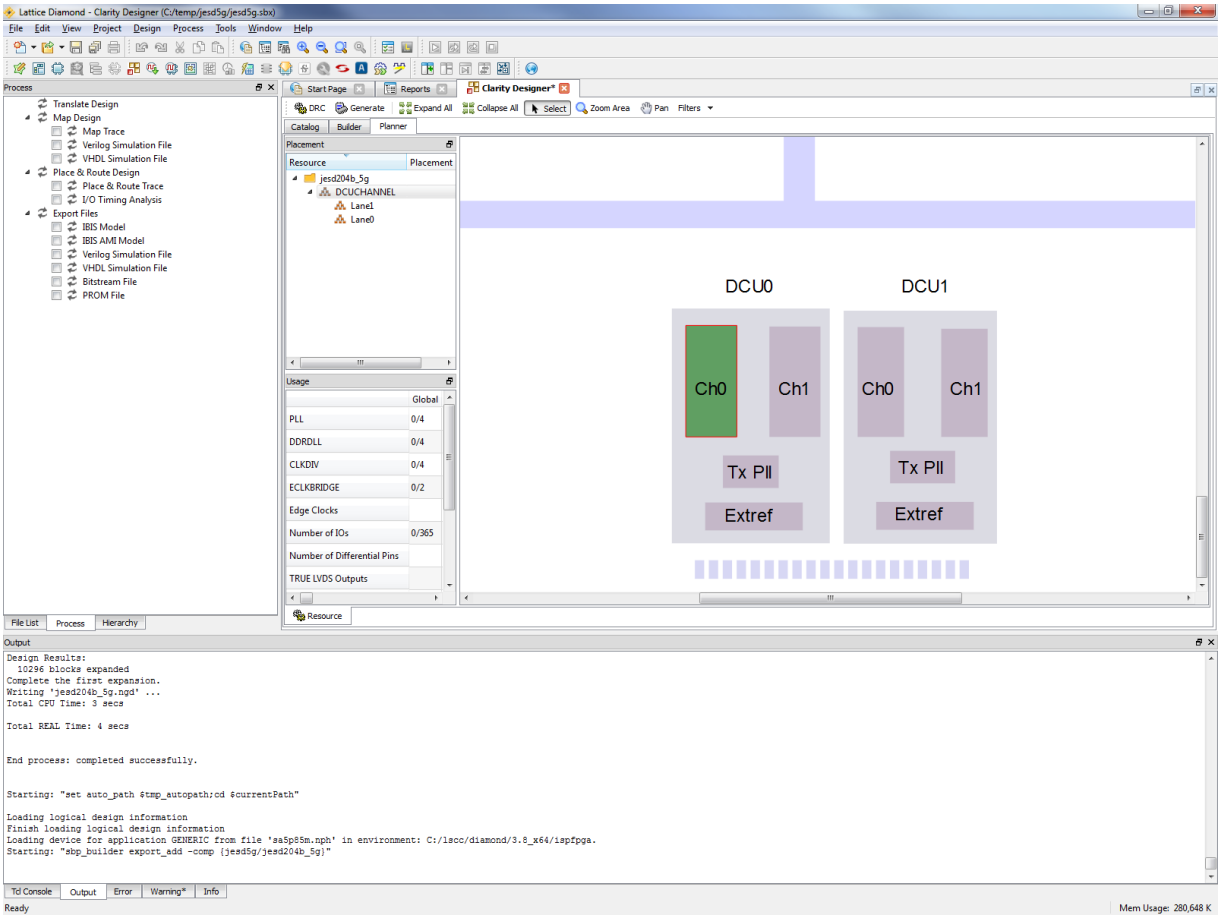

Figure 4.6. Place and Generate JESD204B IP in Clarity Designer

8. Drag the lanes of the IP instance and drop them to the DCU channel(s).
9. Click the **Catalog** tab to configure other required IPs/modules, such as an EXTREF module which can be used to configure the external reference clock connected to DCU.
10. After all required IPs/modules are configured and placed, click 

in the toolbox. Clarity Designer generates all the IPs and modules, and creates a top module to wrap them.

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

4.4. Generating IP in IPexpress

The IPexpress can be started from the Lattice Diamond accessories in start menu of Windows. It can be started from the command line in Linux. The initial window of IP Express is shown in Figure 4.2. To generate a specific IP core configuration, specify:

- **IP Name** – JESD204B, the core to be generated. It is automatically filled when the core is selected from the IP list on the left of the window. Note that JESD204B 5G IP Core does not support IPexpress generation.
- **Project Path** – Path to the directory where the generated IP files will be located.
- **File Name** – "username" designation given to the generated IP core and corresponding folders and files.
- **Module Output** – Verilog or VHDL.
- **Device Family** – Device family to which IP is to be targeted.
- **Part Name** – Specific targeted part within the selected device family

Note that if the IPexpress tool is called from within an existing project, Project Path, Module Output, Device Family and Part Name default to the specified project parameters. Refer to the IPexpress tool online help for further information.

To create a custom configuration, click the Customize button in the IPexpress dialog box to display the JESD204B IP core Configuration user interface, as shown in Figure 4.7. From this dialog box, you can select the IP parameter options specific to your application. Refer to the Parameter Settings chapter for more information on the JESD204B parameter settings.

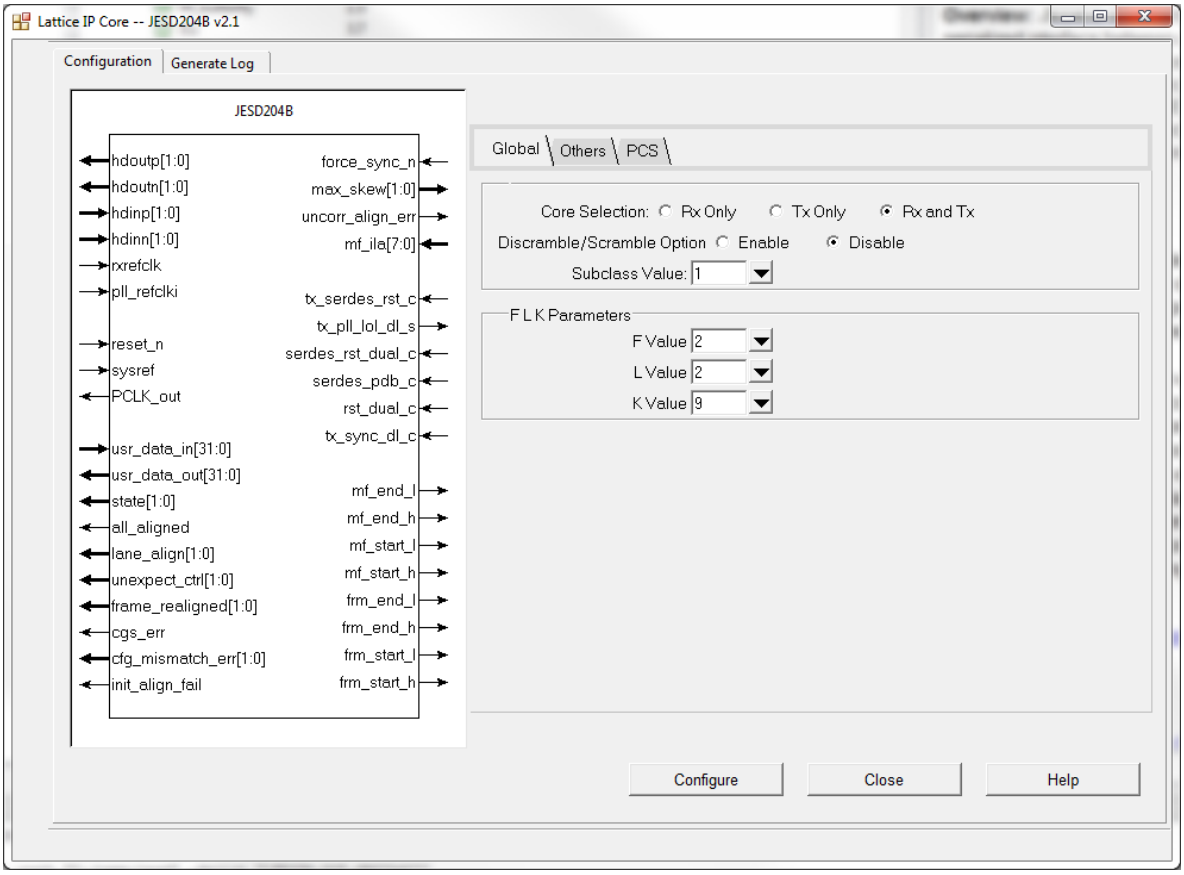

Figure 4.7. Configuration Interface

4.5. Generated IP Directory Structure and Files

The directory structure of the IP and supporting files generated in Clarity Designer is shown in Figure 4.8.

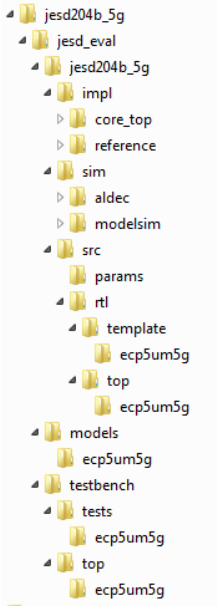

Figure 4.8. JESD204B Core Directory Structure

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 also generated based on your configuration and provided as examples to use or evaluate the IP core.

Table 4.1. Files Generated in Clarity Designer

File	Description
<username>*.v(vhd)	This file provides a wrapper of both core and PCS/SERDES module. Clarity Designer instantiates the file of each IP or module in the same design to create the top module to wrap them after they are generated together.
<username>_beh.v	This file provides the core simulation model with a top module to instantiate and configure the JESD204B IP core.
<username>_(tx/rx)_beh.v	This file provides a wrapper of Tx/Rx core and PCS/SERDES module when the core is configured in Rx only or Tx only mode. It's instantiated in test bench and used for simulation only.
<username>_core_bb.v	This file provides the synthesis black box of the PCS NGO for the user's synthesis.
<username>_phy_bb.v	This file provides the synthesis black box of the PCS NGO for the user's synthesis.
<username>_core.ngo	This file provides the user interface configured and synthesized IP core when it is being reconfigured.
<username>_phy.ngo	This file provides the user interface configured and synthesized PCS.
<username>.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.

***Note**: <username> is the IP instance name that you specified when you created the instance

Besides the files listed in the tables, most of the files required to evaluate the JESD204B IP core reside under the directory \<jesd_eval>. This includes the simulation model, testbench and simulation script files for running the simulation in Modelsim and Active HDL. Two Lattice Diamond project files are also included under the folder at \<jesd_eval>\<username>\impl\[core_only (core top) or reference].

The \<username> folder (jesd0 in Figure 4.8) contains files/folders with content specific to the <username> configuration. This directory is created by Clarity Designer or IPexpress each time the core is generated and regenerated with the same file name. A separate \<username> directory is generated for cores with different names, such as \<my_core_0>,\<my_core_1>, and others.

Different from the folder\<username>, the \jesd_eval and subtending directories provide files supporting JESD204B IP core evaluation that includes files/folders with content that is constant for all configurations of the JESD204B IP core. The \jesd_eval directory is created by Clarity Designer or IPexpress the first time the core is generated when multiple JESD204B cores are generated in the same root directory and updated each time the core is regenerated.

You can use the two prebuilt Diamond projects respectively at \<project_root>\jesd_eval\<username>\impl\[core_only (core top) or reference] to evaluate the implementation (synthesis, map, place and route) of the core in Lattice Diamond tool. The core_top directory exists only in Clarity Designer flow, implementations performed in the core_top directory contain the core and the PCS used by the core. The core_only directory exists only in IPexpress flow, implementations performed in the core_only directory contain only the core and can therefore be used to show the size of the core. The reference directory implementation option reveals a much larger design because it contains additional example user logic functions. The models directory provides library elements such as the PCS/SERDES.

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

Separate directories located at \<project_dir>\jesd_eval\<username>\sim\[Aldec or Modelsim]\rtl are provided and contain specific pre-built simulation script files. See the Running Functional Simulation section for more details.

4.6. Instantiating the IP Core

In Clarity Designer, the generated core is provided in two NGO files, <username>_core.ngo and <user- name>_phy.ngo. A <username>.v file which wraps the two NGO files is also generated. The <username>.v file of each IP in Clarity Designer is automatically instantiated in another wrapper file by Clarity Designer after they are generated based on the Clarity Designer project file.

For example, if the Clarity Designer project file is cdprj.sbx, the automatically generated wrapper file is cdprj.v(vhd) in which all generated IPs are instantiated. The user does not need to instantiate the IP instances one by one manually. The cdprj.v(vhd) is refreshed each time the IPs in the design are regenerated. This not recommended to be used as the top module of the whole design. It is expected that the user can create a top module in which the Clarity Design project, cdprj.sbx or cdprj.v(vhd), and user modules can be instantiated together.

In IPexpress, the generated core is provided in one NGO format, <username>.ngo. The corresponding black box Verilog file <username>_bb.v is also provided for the user to invoke the core to user design.

Unlike Clarity Designer, the IPexpress tool does not provide a <username>.v file to wrap the generated core. A <username>_inst.v file is provided to illustrate the instantiation of the core in a top-level design.

An example RTL top-level reference source file (<user_name>_reference_top.v), that can be used as an instantiation template for the IP core, is provided in <project_dir>\jesd_eval\<username>\src\rtl\top\[device]. You may also use this top-level reference as the starting template for the top-level for the complete design. An example RTL top-level source file (<user_name>_core_only_top.v) that illustrates the wrapping of the generated core is also pro- vided in the same directory.

The VHDL format of the above files <username>_inst.vhd, <username>_reference_top.vhd and <user-name>_core_only_top.vhd is generated in the same folder if “VHDL” is selected in Output Module when the IP is created.

4.7. Running Functional Simulation

Simulation support for the JESD204B IP core is provided for Active-HDL and ModelSim simulators. Figure 4.10 in the Simulation Strategies section shows the simulation architecture of the core. Both an Rx core and a Tx core are required to complete the simulation. The JESD204B IP core simulation model is generated from IPexpress with the name <username>_Ref_beh.v which instantiates both a Tx core and Rx core with the user-configured parameters and contains the obfuscated simulation model. An obfuscated simulation model is Lattice's unique IP protection technique which scrambles the Verilog HDL while maintaining logical equivalence. VHDL users use the same Verilog model for simulation.

4.7.1. Running Functional Simulation in ModelSim

If the core is generated with a VHDL output option, compile the VHDL library as described below before the simulation is started:

Open ModelSim and run the following commands in Modelsim "Transcript":

11.> cd x:/lscc/diamond/3.0(_x64)/cae_library/simulation/vhdl/<device_family>/mti

12.> source orc_cmpl.csh

To run the evaluation simulation:

1. Open ModelSim.
2. Under the File tab, select **Change Directory** and choose folder \<project_dir>\jesd_eval\<username>\sim\modelsim\rtl.
3. Under the Tools tab, select **Execute Macro** and execute the ModelSim "do" script with the name "[verilog|vhdl]_<username>_reference_eval_se.do".

4.7.2. Running Functional Simulation in Active HDL

The top-level file supporting Aldec Active-HDL simulation is provided in \<project_dir>\jesd_eval\<username>\sim\aldec\rtl. This FPGA top is instantiated in an evaluation testbench provided in \<project_dir>\jesd_eval\testbench\top\<device_family> that drives the Tx data with a simple incrementing count pattern.

To run the evaluation simulation:

1. Open Active-HDL.
2. Account Number 3749074379
3. Under the Tools tab, select **Execute Macro** and browse to \<project_dir>\jesd_eval\<username>\sim\aldec\rtl.
4. In the Open file dialogue box, select and execute the Active-HDL "do" script with the name "[verilog|vhdl]_<username> _reference_eval_oem.do".

The simulation waveform results are displayed in the Aldec Wave window.

4.8. Simulation Strategies

This section describes the simulation environment which demonstrates basic JESD204B functionality. A block diagram of the simulation environment is shown in Figure 4.9.

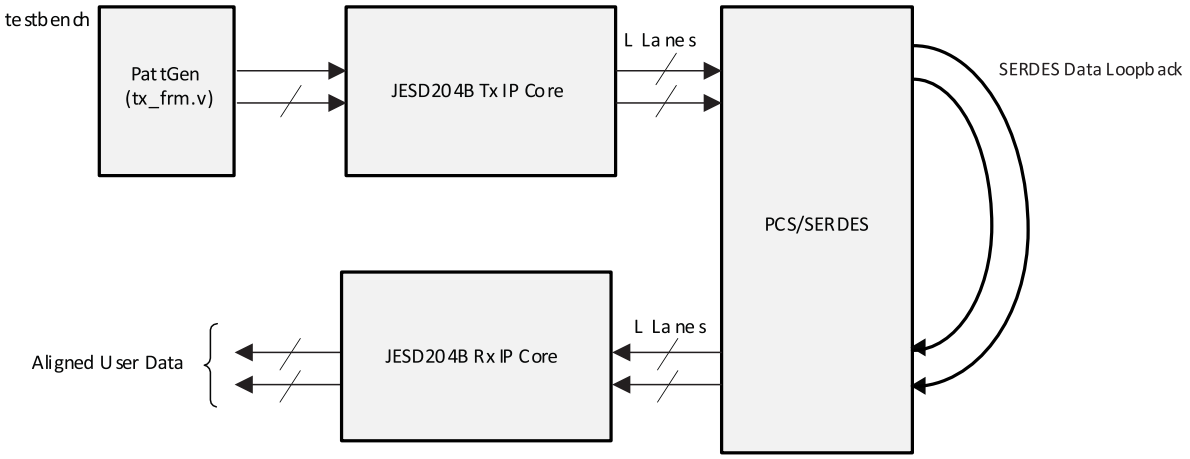

Figure 4.9. Simulation Environment Block Diagram

4.9. Simulation Environment

The simulation environment is made up of a Tx core and Rx core connected to each other by looping back the PCS/SERDES outputs to inputs. Both the Tx and Rx IP cores used in this simulation environment are user-configurable. Data is driven into the Tx core's user inputs from a simple pattern generator which creates an incrementing upcount. Periodically, the upcount is interrupted and the same value is repeated so that the Tx core can insert Frame and Alignment characters into the JESD204B link data stream when the scrambler is disabled. User data output from the Rx IP core is viewable in the simulator's wave window.

After the Rx core reports that all lanes are aligned (output signal all_aligned = 1), having an identical data on both lanes means that the lanes are aligned to the multi frame boundary of the Rx core. The periodic upcount pattern seen from the output of the Tx core clearly shows the multiframe boundary in the data flow. It should be consistent with multiframe boundary generated in Tx control logic. Figure 4.10 shows the results from a simulation. The data patterns highlighted show the incrementing patterns on the Rx outputs (0x12, 0x13, 0x14, …). Also shown is that periodically the pattern generator repeats the same value twice. If this were not done, the Tx IP core would never have the opportunity to insert /A/ alignment and /F/ framing characters into the JESD204B link data stream.

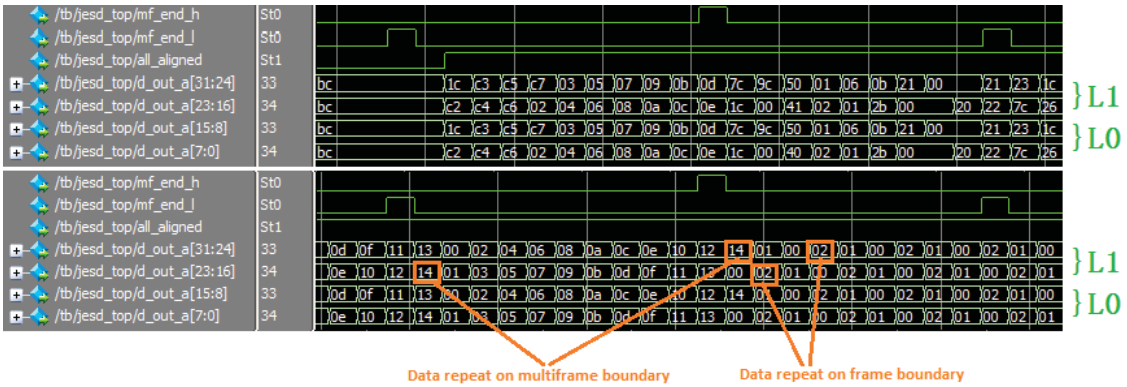

Figure 4.10. Data Patterns Seen on User Interface in Simulation

4.10. 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. All required files are invoked automatically. You can directly synthesize, map and place/par the design in the Diamond design environment after the cores are generated.

As mentioned above, the JESD204B IP core itself is synthesized and provided in NGO format with a black box Verilog source file(s) for synthesis. You can also directly use the NGO file and black box file by instantiating the core in their top-level design as described previously and then synthesizing the entire design with either Synplify or Precision (LatticeECP3 only) RTL Synthesis. That's the only way to use the core in IPexpress flow.

There are two examples of the top-level files in the generated core folder:

- <username>_core_only_top.v(vhd)
- <username>_reference_top.v(vhd)

These files are provided in \<project_dir>\jesd_eval\<username>\src\ rtl\top\<device_family> to support the ability to implement the JESD204B IP core in isolation.

Push-button implementation of the top-level design using either Synplify or Precision RTL Synthesis (LatticeECP3 family only) is supported through the following Diamond project files:

- <username>_core_only_eval.ldf located in the \<project_dir>\jesd_eval\<username>\impl\core_only\ directory
- <username>_reference_eval.ldf located in the \<project_dir>\jesd_eval\<username>\impl\reference\ directory

JESD204B IP Core generated in System Builder/Planner does not provide a core only implementation. A core top implementation with both core and PCS/SERDES wrapped is provided instead. The corresponding Diamond project file <username>_core_top_eval.ldf is located in the folder \<project_dir>\jesd_eval\<username>\impl\core_top\.

To use the project files in Diamond:

1. Choose **File > Open > Project**.
2. Browse to \<project_dir>\jesd_eval\<username>\impl\lifmd\[lse|synplify]\ in the Open Project dialog box.
3. Select and open <username>_[core_only|reference]_eval.ldf. At this point, all of the files needed to support top-level synthesis and implementation will be imported to the project.
4. Click the **Process** tab in the left-hand user interface window.
5. Implement the complete design via the standard Diamond user interface flow.

4.11. Hardware Evaluation

The JESD204B IP core 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.

4.11.1. Enabling Hardware Evaluation in Diamond

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

4.12. Updating/Regenerating the IP

The Clarity Designer interface allows you to update the local IPs from the Lattice IP server. You can use the updated IP to regenerate the IP in the design.

To change the parameters of the IP used in the design, the IP must also be regenerated.

4.12.1. 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 4.11.
2. The IP Configuration interface is displayed. Change the parameters as required and click the **Configure** button.

3. Update the pin connection in the Builder tab for configuration changes.
4. Click  in the toolbox.

Clarity Designer regenerates all the instances which are reconfigured.

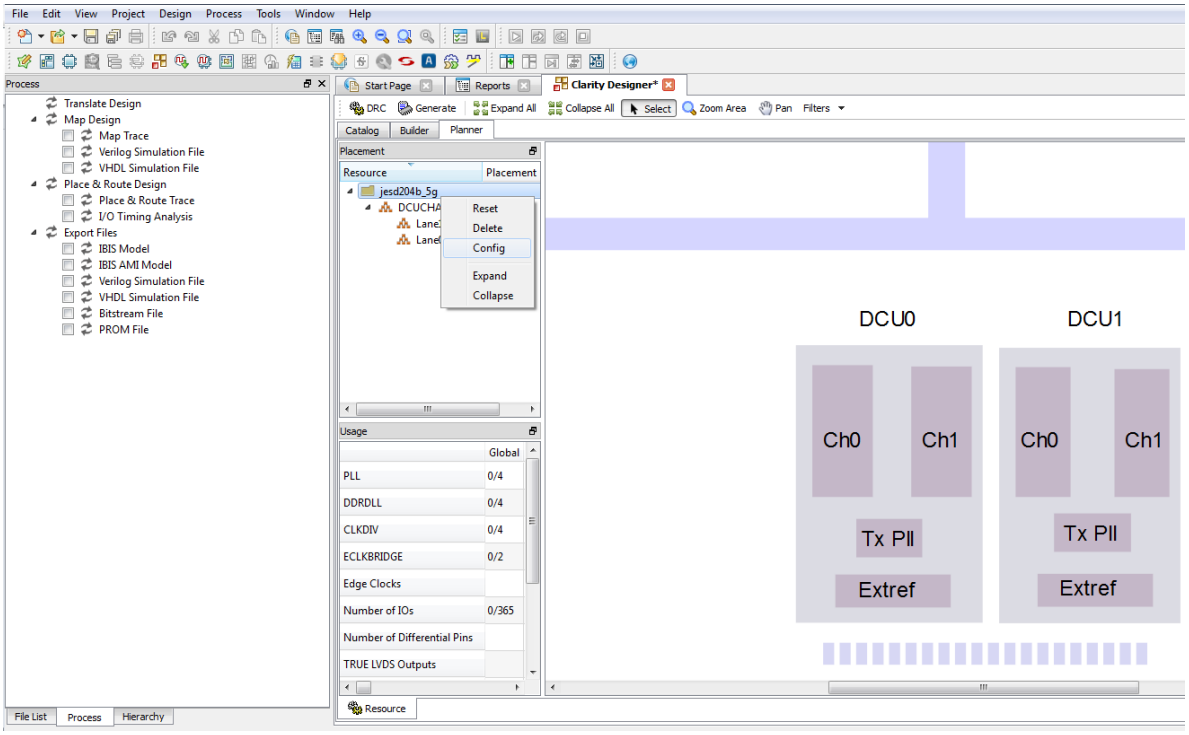

Figure 4.11. IP Regeneration in Clarity Designer

5. Application Support

The Kondor AX Advanced System Development Board from MikroProjekt can be used for hardware evaluation. The following figure is a high-level diagram using JESD204B IP core as described by the product website.

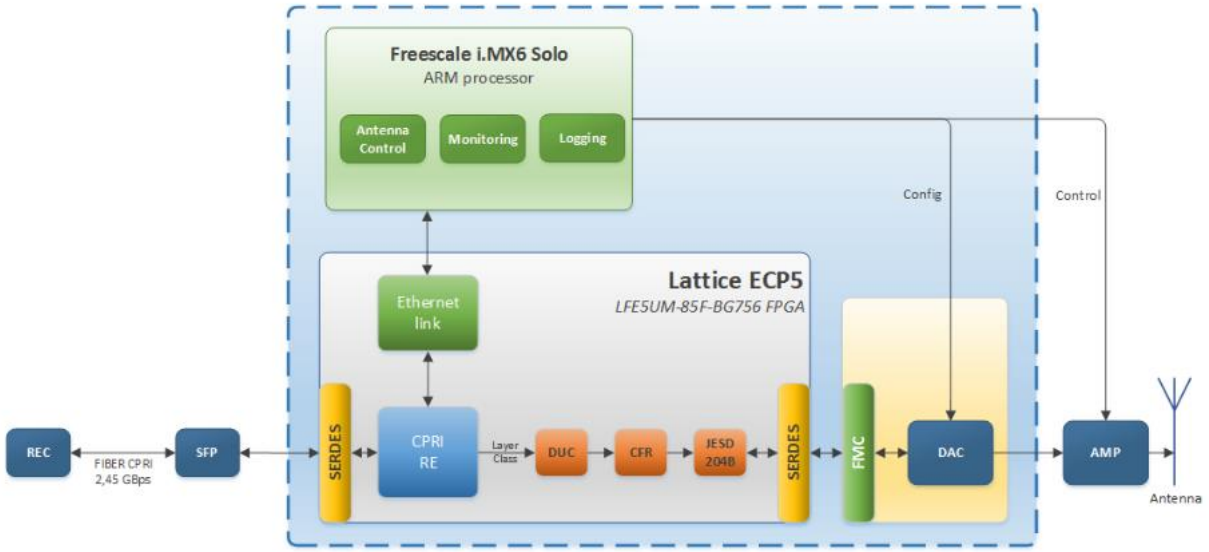

Figure 5.1. HetNet and Small Cells Application

For detailed information about the board, please refer to documentation located at http://www.mikroprojekt.hr/products/development-boards/kondor-ax.

References

For more information, refer to the following documents:

- FPGA-DS-02012 (previously DS1044), ECP5 and ECP5-5G Family Data Sheet
- TN1261, ECP5 and ECP5-5G SERDES/PCS Usage Guide
- DS1021, LatticeECP3 Family Data Sheet
- TN1176, LatticeECP3 SERDES/PCS Usage Guide
- JEDEC Standard, Serial Interface for Data Converters, JESD204B.01, July 2012, www.jedec.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 JESD204B IP core.

IPexpress is the Lattice IP configuration utility, and is included as a standard feature of the Diamond and ispLEVER design tools. Details regarding the usage of IPexpress can be found in the IPexpress and Diamond or ispLEVER help system. For more information on the Diamond design tools, visit the Lattice web site at www.latticesemi.com//Products/DesignSoftware.

LatticeECP3-70 Utilization

Table A.1 lists resource utilization for LatticeECP3-70 FPGAs using the JESD204B 3G IP core.

Table A.1. Resource Utilization*

IP User-Configurable Parameters	Slices	LUTs	Registers	sysMEM EBRs
F=3, L=2 ,K=9, Subclass1, Scram=0, Rx	2828	4886	2174	2
F=3, L=2 ,K=9, Subclass1, Scram=0, Tx	354	651	266	0

***Note:** Performance and utilization data target an LFE3-70EA-6FN672C device using Lattice Diamond 3.2 and Synplify Pro I-2013.09L software. Performance may vary when using a different software version or targeting a different device density or speed grade within the LatticeECP3 family.

LFE5UM-85 Utilization

Table A.2 lists resource utilization for LFE5UM-85 FPGAs using the JESD204B 3G IP core.

Table A.2. Resource Utilization*

IP User-Configurable Parameters	Slices	LUTs	Registers	sysMEM EBRs
F=3, L=2 ,K=9, Subclass1, Scram=0, Rx	1826	2276	2170	2
F=3, L=2 ,K=9, Subclass1, Scram=0, Tx	289	534	266	0

***Note:** Performance and utilization data target an LFE5UM-85F-8BG756C device using Lattice Diamond 3.2 and Synplify Pro I-2013.09L software. Performance may vary when using a different software version or targeting a different device density or speed grade within the ECP5 family.

Ordering Part Number

The Ordering Part Number (OPN) for JESD204B 3G IP core targeting ECP5 devices are JESD-204B-E5-U and JESD-204B-E5-UT.

LFE5UM5G-85 Utilization

Table A.3 lists resource utilization for LFE5UM5G-85 FPGAs using the JESD204B 5G IP core.

Table A.3. Resource Utilization*

IP User-Configurable Parameters	Slices	LUTs	Registers	sysMEM EBRs
F=3, L=2 ,K=9, Subclass1, Scram=0, Rx	2549	3468	2477	0
F=3, L=2 ,K=9, Subclass1, Scram=0, Tx	721	1001	621	0

***Note:** Performance and utilization data target an LFE5UM5G-85F-8BG756C device using Lattice Diamond 3.9 and Synopsys Synplify Pro L-2016.09L-1 software. Performance may vary when using a different software version or targeting a different device density or speed grade within the ECP5-5G family.

Ordering Part Number

The Ordering Part Number (OPN) for JESD204B 5G IP core targeting ECP5-5G devices are JESD204B-E5G-U and JESD204B-E5G-UT.

Appendix B. Limitations

JESD204B 5G IP Core

Table B.1 shows the minimum device supported by core and reference design generation based on default parameters.

Table B.1. Core and Reference Design Minimum Device Support*

IP User-Configurable Parameters	Device	Package	Core Top	Reference Top
F=3, L=2 ,K=9, Subclass1, Scram=0, Rx and Tx	LFEUM5G-25F	CSFBGA285	No	No
		CABGA381	No	Yes
	LFEUM5G-45F	CSFBGA285	No	No
		CABGA381	Yes	Yes
		CABGA554	Yes	Yes
	LFEUM5G-85F	CSFBGA285	No	No
		CABGA381	Yes	Yes
		CABGA554	Yes	Yes
		CABGA756	Yes	Yes

***Note:** Minimum device supported may vary when L parameter is increased due to increased pin count.

Revision History

Date	Document Version	IP Core Version	IP Core Max Rate	Change Summary
June 2017	2.3	1.0	5G	• Updated supported Diamond version in Table 1.2. JESD204B 3G IP Core Quick Facts. • Added Kondor AX development board in Application Support. • Updated the document number of ECP5 and ECP5-5G Family Data Sheet from DS1044 to FPGA-DS-02012. • Updated values of Slices, LUTs, and Registers in Table A.3 for ECP5-5G. Updated Diamond version in the footnote. • General update to the Limitations section.
October 2016	2.2	Beta	5G	Updated Ordering Part Numbers in LFE5UM5G-85 Utilization section.
				Corrected IP Core Version in Revision History.
	2.1	Beta	5G	Added support for ECP5-5G.
June 2014	2.0	2.0	3G	RTL update. subclass 0 Support.
June 2013	01.0	1.0	3G	Initial release.
	01.1			Customized version with LatticeECP3 information only.

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

X-ON Electronics

Largest Supplier of Electrical and Electronic Components

Click to view similar products for Development Software ***category:***

Click to view products by Lattice ***manufacturer:***

Other Similar products are found below :

RAPPID-560XBSW RAPPID-567XFSW DG-ACC-NET-CD SRP004001-01 SW006021-1NH SW163052 SYSWINEV21 Core429-SA SW500006-HPA CWP-BASIC-FL W128E13 CWP-PRO-FL SYSMACSE210L SYSMACSE203L AD-CCES-NODE-1 NT-ZJCAT1-EV4 CWA-BASIC-FL RAPPID-567XKSW CWA-STANDARD-R SW89CN0-ZCC CWA-LS-DVLPR-NL VDSP-21XX-PCFLOAT RAPPID-563XMSW IPS-EMBEDDED SWR-DRD-L-01 SDAWIR-4532-01 SYSMAC-SE201L MPROG-PRO535E AFLCF-08-LX-CE060-R21 WS02-CFSC1-EV3-UP SYSMAC-STUDIO-EIPCPLR LIB-PL-PC-N-1YR-DISKID SYSMACSE2XXL LS1043A-SWSP-PRM 1120270005 1120270006 MIKROBASIC PRO FOR FT90X (USB DONGLE) MIKROC PRO FOR AVR (USB DONGLE LICENSE) MIKROC PRO FOR FT90X (USB DONGLE) MIKROBASIC PRO FOR AVR (USB DONGLE LICEN MIKROBASIC PRO FOR FT90X MIKROC PRO FOR DSPIC30/33 (USB DONGLE LI MIKROC PRO FOR FT90X MIKROC PRO FOR PIC32 (USB DONGLE LICENSE 52202-588 MIKROPASCAL PRO FOR ARM (USB DONGLE LICE MIKROPASCAL PRO FOR FT90X MIKROPASCAL PRO FOR FT90X (USB DONGLE) MIKROPASCAL PRO FOR PIC32 (USB DONGLE LI SW006021-2H