HB0430
Handbook
CoreJESD204BRX v3.3

Microsemi Corporate Headquarters
One Enterprise, Aliso Viejo,
CA 92656 USA
Within the USA: +1 (800) 713-4113
Outside the USA: +1 (949) 380-6100
Sales: +1 (949) 380-6136
Fax: +1 (949) 215-4996
E-mail: sales.support@microsemi.com
www.microsemi.com

About Microsemi

Microsemi Corporation (Nasdaq: MSCC) offers a comprehensive portfolio of semiconductor and system solutions for aerospace & defense, communications, data center and industrial markets. Products include high-performance and radiation-hardened analog mixed-signal integrated circuits, FPGAs, SoCs and ASICs; power management products; timing and synchronization devices and precise time solutions, setting the world's standard for time; voice processing devices; RF solutions; discrete components; enterprise storage and communication solutions, security technologies and scalable anti-tamper products; Ethernet solutions; Power-over-Ethernet ICs and midspans; as well as custom design capabilities and services. Microsemi is headquartered in Aliso Viejo, Calif., and has approximately 4,800 employees globally. Learn more at www.microsemi.com.

50200430. 11 5/19

Contents

Figures

Tables

1 Revision History

The revision history describes the changes that were implemented in the document. The changes are listed by revision, starting with the most current publication.

1.1 Revision 11.0

Updated changes related to CoreJESD204BRX v3.3.

- Updated description for CF, HD, FAC_EN, and LCD_EN parameters
- Added new section Clocking and Reset

1.2 Revision 10.0

Updated changes related to CoreJESD204BRX v3.2.

- Table 8 EPCS_[n]_RX_VAL Description
- Table 8 LANE_CLK Description
- License section

1.3 Revision 9.0

Updated changes related to CoreJESD204BRX v3.1.

- Table 1 through Table 6
- Table 8 RESET_N Description
- SYNC Signal encoder section
- Data Link Layer section
- License section

1.4 Revision 8.0

Updated changes related to CoreJESD204BRX v3.0.

- SERDES Interface section
- Table 8 CoreJESD204BRX I/O Signal Description
- Replacement of latest figures
- Added a note in Table 7 CoreJESD204BRX Parameters and Generics Descriptions

1.5 Revision 7.0

Updated changes related to CoreJESD204BRX v3.0. User interface updates.

1.6 Revision 6.0

Updated changes related to CoreJESD204BRX v3.0.

1.7 Revision 5.0

- Updated changes related to CoreJESD204BRX v2.4.
- Added RTG4 information to Supported Microsemi FPGA Families section and to all Device Utilization tables.
- Update the details of all CoreJESD204BRX Blocks in the Introduction.
- Added parameter LCD_EN to Table 3.
- Added EPCS_RX_VAL signals and updated Figure 2, Figure 4 and Figure 5.

1.8 Revision 4.0

Updated changes related to CoreJESD204BRX v2.4.

1.9 Revision 3.0

Updated changes related to CoreJESD204BRX v2.3.

1.10 Revision 2.0

Updated changes related to CoreJESD204BRX v2.2.

1.11 Revision 1.0

Revision 1.0 was the first publication of this document. Created for CoreJESD204BRX v2.0.

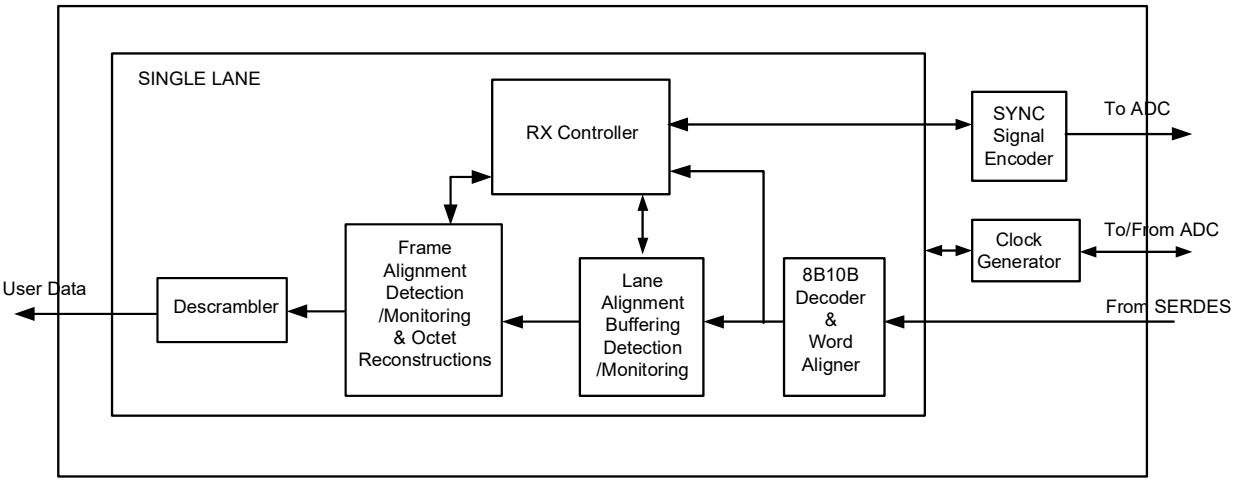
2 Introduction

2.1 Overview

CoreJESD204BRX is the receiver conforming to the JEDEC JESD204B standard. This specification describes a high speed serial interface for data converters.

Figure 1, page 3 shows the CoreJESD204BRX block diagram of a single lane.

Figure 1 • **CoreJESD204BRX Block Diagram**



2.1.1 CoreJESD204BRX Blocks

CoreJESD204BRX consists of the following blocks:

- Frame Alignment Detection/Monitoring and Octet Reconstruction
- Lane Alignment Buffering Detection/Monitoring
- 8B10B Decoder and Word Aligner
- SYNC Signal Encoder
- Clock Generator
- RX Controller
- Descrambler

2.1.1.1 Frame Alignment Detection/Monitoring and Octet Reconstruction

This block monitors the frame position and report any detected misalignment of a lane to the RX Controller. Character replacement of octets is performed by the transmitter and this block reconstructs the octets back into their original values.

After initialization this block monitors the frame position and asserts the SOF and SOMF outputs to indicate when the current position is at the start of a frame or multi-frame. It also monitors for the alignment character K28.7 (/F/) and K28.3 (/A/) which can only occur at the end of frame or multi-frame respectfully. End of frame detection monitors the position of the alignment character /F/. When two successive /F/ symbols do not arrive in the expected position and without receiving a /F/ symbol at expected position between the two unexpected /F/ symbols the receiver will re-align to the new position. End of multi-frame detection monitors the position of the alignment character /A/.

When two successive /A/ symbols do not arrive in the expected position and without receiving a /A/ symbol at expected position between the two unexpected /A/ symbols the receiver will re-align to the new position. The detection of unexpected alignment symbols is reported to the RX Controller which determines the adjustment required to realign the lane to the expected position. The output UCC_ERR will assert whenever an unexpected control character is detected.

The transmitter replaces certain octets with alignment characters which must be reconstructed back into the original octets when received by this block. How the octets are reconstructed depends on if scrambling is enabled or disabled. If scrambling is disabled, when an alignment character is received at the last octet of the current frame it is replaced with the last octet of the previous frame because alignment character replaced an octet with the same value. The last octet of the previous frame may have been decoded from a previous octet reconstruction. If scrambling is enabled, when an alignment character is received no replacement occurs because in this case the alignment character value equals the actual user data. This is because K28.3 /A/ bits are the same as D28.3 and K28.7 /F/ bits are the same as D28.7.

Note: Read section 5.3.3.4 of the JESD204B.01 specification for more details

2.1.1.2 Lane Alignment Buffering Detection/Monitoring

This block performs lane alignment, buffers data, re-aligns lanes, and monitors the link configurations parameters.

JESD204B supports multiple lanes that must stay aligned to each other. Because of different delays that can occur on any of the data path. The data for each lane can arrive at different times which mean they can be out of sync with each other. During synchronization request, this block monitors for the ILA sequence. Once the first octet of the ILA sequence is detected on a lane, the data gets buffered and will be held in the buffer until all lanes detect first octet of their ILA sequence. The first octet is the control symbol K28.0 (/R/) which indicates start of subsequence. When all lanes detect their first /R/ the buffer releases all the ILA sequences at the same time. The buffers will release all the ILA sequences at the start of the next frame (subclass 0) or the start of the next multi-frame (subclass 1 & 2) which will successfully align all the lanes. The ALIGNED output will then assert indicating that lane alignment was successful. If any of the lanes fail to detect the ILA sequence a re-initialization request will be sent to the RX Controller.

This block also monitors for the link configuration parameters which occur in the second multi-frame of the ILA sequence. The control symbol K28.4 (/Q/) in the ILA sequence indicates that the next octet is the start of the link configuration data. The parameters in the link configuration data are compared to the configuration parameters of the core. If there is any mismatch between the parameters the LINK_CD_ERR output will assert. If a mismatch is found LINK_CD_ERR assertion will not trigger a re-initialization request, it is up to the user to determine what actions to take. Although the JESD204B.01 specification indicates that the checksum (FCHK) in the link configuration mapping is the sum of all the fields and not the sum of all the octets created by the mapping some data converter companies have the checksum as the sum of all the octets. To cover both checksum types the parameter/generic FIELD_OCTET can select which type of checksum to calculate.

During normal operation if any of the lanes detect two or more alignment symbols in the same unexpected position a realignment request of that lane occur. Realignment will shift the data of the lane back into the expected position. If there is not enough data in the buffer to perform the realignment or if the adjustment will cause the buffer holding the data to overflow an invalid request occurs causing a re-initialization request to be sent to the RX Controller.

2.1.1.3 8B10B Decoder and Word Aligner

This block is only used when parameter/generic DECODER_EN = 1. It decodes data using 8B10B encoding techniques and it also supports word alignment for the comma character K28.5.

Word Alignment is performed prior to 8B10B decoding. A stream of bits are received and then de-serialized by the SERDES. The most significant bit (MSB) and the least significant bit (LSB) of the code could start at any point because the parallel data is continuously shifting by one bit which means the bit boundaries of the codes end up in a random position. During initialization the word aligner monitors for K28.5 comma characters. When the word aligner detects the bit boundaries of this comma character and it receive enough consecutively comma's it completes alignment by locking to this bit boundary and then tarts sending the received codes to the 8B10B decoder.

The 8B10B decoder is a marriage of two sub-blocks, the 6b5b and the 4b3b decoders. The purpose of the decoders is to convert 10-bit code into a 8-bit data. The 10-bit code contains an equal number of 0’s and 1’s. In addition, the code is built so that no more than five consecutive 0’s or 1’s is ever transmitted. If parameter/generic SERDES_MODE is set to 1 each lane has a 20-bit input which will decode two 10-

bit codes. If SERDES_MODE is set to 2 each lane has a 10-bit input which will decode one 10-bit. 8B10B running disparity error and not-in-table error detection can be monitored on the DISP_ERR and NIT_ERR output signals. Bit-1 assertion means error occurred in upper octet and bit-0 assertion means error occurred in lower octet. During normal operation if four or more 8B10B errors occur the RX Controller will issue a re-initialization request.

2.1.1.4 SYNC Signal Encoder

This block controls the synchronization request and internal error reporting by encoding the SYNC_N output signal.

Reset or power-on will trigger initialization and re-initialization is triggered by the RX Controller but both behave the same way. The behavior of SYNC_N changes based on subclass selected. SYNC_N will be asserted for a minimum of 5 frames + 9 octets during a synchronization request but the de-assertion of SYNC_N occurs at the start of the next frame for subclass 0 and start of next multi-frame for subclasses 1 and 2. During initialization SYNC_N will only de-assert when four or more consecutive K28.5 comma characters are received, however SYNC_N will re-assert if the ILA sequence is not seen after (F * K * 4) LANE_CLK clock cycles.

There are internal errors that can occur that don't require re-initialization. Reporting of these errors is supported for subclasses 1&2 through the SYNC_N output. If an internal error occurs that does not require re-initialization SYNC_N will report this error by asserting for the last two frames of a multi-frame. If the core is configured for subclass 0 or for JESD204A internal error reporting via the SYNC interface is disabled.

Note: Read section 5.3.3 for information about synchronization requests and section 7.6.4 in the JESD204B.01 specification for error reporting via the SYNC interface.

2.1.1.5 Clock Generator

This block generates the internal LMFC and frame clock. When subclass 1 is selected it also generates SYSREF_OUT output and monitors SYSREF_IN input for deterministic latency. When reporting errors via the SYNC interface is enabled (disabled for subclass 0 and JESD204A configuration) this block controls when to report the errors.

Note: More information can be found in the JESD204B.01 specification. See Annex G for clock generation information, section 6 for deterministic latency and section 7.6.4 for error reporting via the SYNC interface.

2.1.1.6 RX Controller

This block monitors the frame position and requests a realignment of a lane if misalignment is detected. It also requests re-initialization when an error that cannot be fixed internally occurs.

During a synchronization request this block performs Code Group Sync (CGS) detection. CGS_ERR output will be asserted indicating that CGS has not yet been achieved. When CGS_ERR is asserted a synchronization request is sent to the SYNC Signal Encoder block. When four or more consecutive K28.5 (/K/) characters are detected the receiver assumes CGS was successful and deasserts CGS_ERR. After CGS this block will monitor the error outputs from the 8B10B decoder. If four or more consecutive 8B10B errors occur, the receive assumes loss of CGS and CGS_ERR asserts triggering a re-initialization request.

This block also determines the adjustment required to realign a lane when a realignment request occurs. A realignment request with the adjustment information is sent to the Lane Alignment Buffering Detection/Monitoring block. If the request is valid realignment will occur but if it is an invalid realignment request re-initialization will occur instead. If there is not enough data in the buffer to perform the realignment or if the adjustment will cause the buffer to overflow they are considered an invalid realignment request.

2.1.1.7 Descrambler

This block descrambles data based on the polynomial 1 + x14 + x15. Descrambling is enabled by setting parameter/generic SCR to 1. Descrambling must only be enabled if the transmitter has scrambling enabled. Scrambling of data is used to avoid electromagnetic compatibility and interface problems in sensitive application. Spectral peaks are produced when the same data pattern is repeated over a period of time. This polynomial offers a period of 32,767 which is long enough to meet the spectral requirements of sensitive application. The scrambler is self-synchronous and also allows the descrambler in the receiver to synchronize within two octets. The reason scrambling is optional is because the digital operations of the interface cause switching noise which makes scrambling a disadvantage for some applications. Switching noise causes current spikes which causes sensitive analogue portions of the design to degrade or even malfunction.

More information can be found in the JESD204B.01 specification. Read Section 5.2 for information about scrambling and view Figure D.5, for the descrambler implementation.

2.2 Features

- Supports 1 to 8 lanes
- Supports Subclasses 0, 1, and 2
- Performs word alignment
- Sync~ signal encoding
- Performs user-enabled 8B/10B decoding
- Recovers link configuration parameters
- Performs lane alignment buffering, monitoring and correction
- Performs user-enabled frame alignment, monitoring, and correction
- Performs octet reconstruction
- Performs user-enabled descrambling
- Error detection
- Supports data width of 16, 32, or 64 bits

Note: Octet to Frame Stream conversion is not performed by the core.

2.3 Core Version

This handbook is for CoreJESD204BRX version 3.3.

2.4 Supported Families

- PolarFire®
- RTG4™
- SmartFusion®2
- IGLOO®2

2.5 Device Utilization and Performance

CoreJESD204BRX has been implemented in the following Microsemi device families. A summary of the implementation data for CoreJESD204BRX is provided in Table 1, .

Table 1 • **CoreJESD204BRx Utilization for Data Width 16 With Decoder**

	Logic Elements			Utilization		
Family	**Sequential**	**Combinatorial**	**Total**	**Device**	**Total %**	**Performance (MHz)**
RTG4	3550	5878	9428	RT4G150	3.10	115
SmartFusion2	3466	5108	8574	M2S150T	2.93	160
IGLOO2	3466	5108	8574	M2GL150T	2.93	160
PolarFire	3254	5240	8494	MPF500T	0.88	186

Notes:

- Data in this table were achieved using synthesis and layout settings optimized for speed. The following parameters/generics were set: L=3, F=2, K=9, RAM_SEL=1, DECODER_EN = 1 & D_WIDTH=16.
- Higher performance frequencies are achievable for other configurations.
- Performance results were achieved with speed grade -1 (up to 15% faster than STD). Over constraining clocks during synthesis is advised. Improved performance seen when FAC_EN=0, LCD_EN=0 and RAM_SEL=0.

Table 2 • **CoreJESD204BRx Utilization for Data Width 32 With Decoder**

	Logic Elements			Utilization		
Family	**Sequential**	**Combinatorial**	**Total**	**Device**	**Total %**	**Performance (MHz)**
RTG4	5801	8812	14613	RT4G150	4.81	110
SmartFusion2	5743	8034	13777	M2S150T	4.71	144
IGLOO2	5743	8034	13777	M2GL150T	4.71	144
PolarFire	5482	7601	13083	MPF500T	1.36	184

Notes:

- Data in this table were achieved using synthesis and layout settings optimized for speed. The following parameters/generics were set : L=3, F=2, K=9, RAM_SEL=1, DECODER_EN = 1 & D_WIDTH=32.
- Performance results were achieved with speed grade -1 (up to 15% faster than STD). Over constraining clocks during synthesis is advised. Improved performance seen when FAC_EN=0, LCD_EN=0 and RAM_SEL=0.

Table 3 • **CoreJESD204BRx Utilization for Data Width 64 With Decoder**

Family	Logic Elements			Utilization		Performance (MHz)
	Sequential	Combinatorial	Total	Device	Total %	
RTG4	10176	17378	27554	RT4G150	9.07	98
SmartFusion2	10647	16522	27169	M2S150T	9.30	120
IGLOO2	10647	16522	27169	M2GL150T	9.30	120
PolarFire	9596	15915	25511	MPF500T	2.65	122

Notes:

- Data in this table were achieved using synthesis and layout settings optimized for speed. The following parameters/generics were set : L=3, F=2, K=9, RAM_SEL=1, DECODER_EN = 1 & D_WIDTH=64.
- Performance results were achieved with speed grade -1 (up to 15% faster than STD). Over constraining clocks during synthesis is advised. Improved performance seen when FAC_EN=0, LCD_EN=0 and RAM_SEL=0.

Table 4 • **CoreJESD204BRx Utilization for Data Width 16 Without Decoder**

Family	Logic Elements			Utilization		Performance (MHz)
	Sequential	Combinatorial	Total	Device	Total %	
PolarFire	2283	2826	5109	MPF500T	0.53	197

Notes:

- Data in this table were achieved using default synthesis and layout settings. The following parameters/generics were set: L=3, F=2, K=9, RAM_SEL=1, DECODER _EN = 0 & D_WIDTH=16.
- Performance results were achieved with speed grade -1 (up to 15% faster than STD). Over constraining clocks during synthesis is advised. Improved performance seen when FAC_EN=0, LCD_EN=0 and RAM_SEL=0.

Table 5 • **CoreJESD204BRx Utilization for Data Width 32 Without Decoder**

Family	Logic Elements			Utilization		Performance (MHz)
	Sequential	Combinatorial	Total	Device	Total %	
PolarFire	3821	3979	7700	MPF500T	0.80	189

Notes:

- Data in this table were achieved using default synthesis and layout settings. The following parameters/generics were set: L=3, F=2, K=9, RAM_SEL=1, DECODER _EN = 0 & D_WIDTH=32.
- Performance results were achieved with speed grade -1 (up to 15% faster than STD). Over constraining clocks during synthesis is advised. Improved performance seen when FAC_EN=0, LCD_EN=0 and RAM_SEL=0.

Table 6 • **CoreJESD204BRx Utilization for Data Width 64 Without Decoder**

Family	Logic Elements			Utilization		Performance (MHz)
	Sequential	Combinatorial	Total	Device	Total %	
PolarFire	6866	7862	14728	MPF500T	1.53	162

Notes:

- Data in this table were achieved using default synthesis and layout settings. The following parameters/generics were set: L=3, F=2, K=9, RAM_SEL=1, DECODER _EN = 0 & D_WIDTH=64.
- Performance results were achieved with speed grade -1 (up to 15% faster than STD). Over constraining clocks during synthesis is advised. Improved performance seen when FAC_EN=0, LCD_EN=0 and RAM_SEL=0.

3 Functional Description

CoreJESD204BRX, shown in Figure 2, page 10, consists:

- Optional descrambling
- SYNC_N signal encoding
- Clock generation
- Lane alignment buffering detection and/or monitoring
- Frame alignment detection and/or monitoring
- Octet reconstruction
- Optional 8B10B decoding with word alignment

3.1 Data Link Layer

CoreJESD204BRX implements the JEDEC JESD204B receiver interface.

Figure 2 • **Data Link Layer**

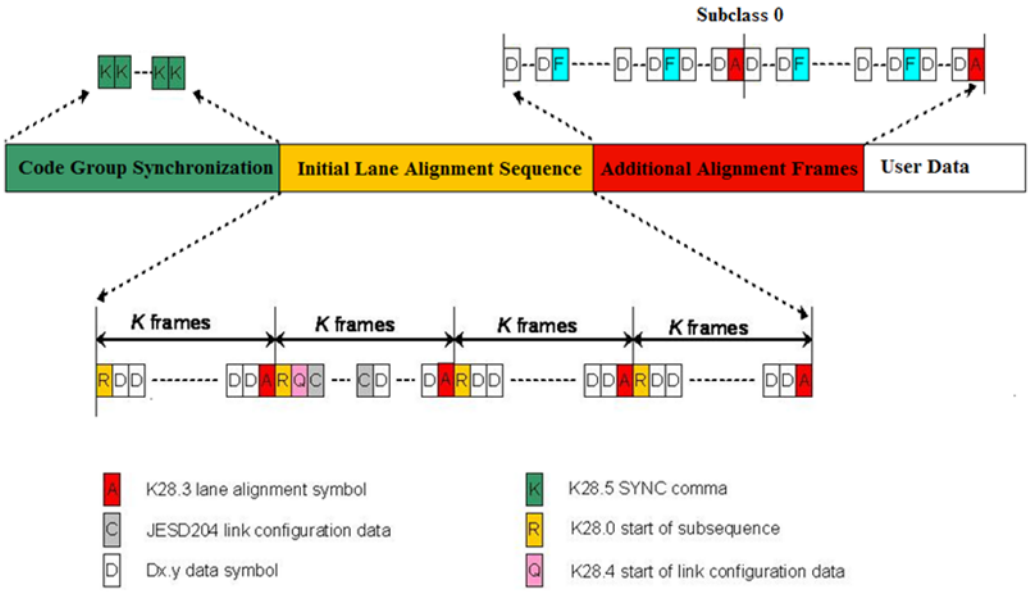


Figure 2, page 10 shows the data link layer of CoreJESD204BRX. Code Group Synchronization (CGS) is triggered when the receiver issues a synchronization request via the synchronization interface. During a synchronization request the transmitter emits a continuous stream of K28.5 (/K/) control symbols. The receiver will then align its bit boundaries to this symbol and will deactivate the synchronization request when it receives at least four successive /K/ symbols. Then the Initial Lane Alignment (ILA) sequence is transmitted, however if the ILA sequence is not seen in (F * K * 4) LANE_CLK clock cycles the synchronization request is reactivated. The first control symbol of every multi-frame in the ILA sequence is K28.0 (/R/) which indicates start of the subsequence.

The last control symbol of every frame in the ILA is K28.3 (/A/) which indicates the end of multi-frame. The receiver lane(s) align to the symbol during initialization. The second frame of the multi-frame has the link configuration data. The control symbol K28.4 (/Q/) indicates that the next octet is the start of the link configuration data. The receiver compares its parameters to the parameters received in the link configuration data and reports an error if there is a mismatch. Subclass 0 support addition alignment frames if more frames are required for alignment. The additional alignment frames will have K28.7 (/F/) symbols at the end of frame and /A/ symbols at the end of multi-frame. After alignment completes the user data can be transmitted.

3.2 JESD204B Sub-classes

CoreJESD204BRX supports sub-class 0, 1, and 2. Following is a summary of each sub-class.

3.2.1 Sub-class 0

Sub-class 0 allows the device to be backward compatible with the JESD204A standard. It offers no support for deterministic latency.

3.2.2 Sub-class 1

Sub-class 1 offers support for deterministic latency using SYSREF signaling.

During a synchronization request SYSREF_OUT will only assert at the start of a LMFC period and when SYNC_N is asserted. The assertion of SYSREF_OUT indicates to the transmitter when start of a LMFC period occurs. The transmitter has the option to shift the phase of its internal clocks based on SYSREF_OUT assertion. The minimum duration SYSREF_OUT will assert for is quarter of a LMFC period.

The assertion of SYSREF_IN input indicates the start of a LMFC period in the transmitter. During a synchronization request the receiver asserts SYNC_N and when SYSREF_IN asserts it will cause LMFC and frame clock to phase shift to sync to the transmitters LMFC period for deterministic latency support. When SYNC_N is de-asserted any assertion of SYSREF_IN will be ignored so that the LMFC and frame clock positions are locked.

3.2.3 Sub-class 2

Sub-class 2 is similar to sub-class 1 except it offers support for deterministic latency using SYNC~ sampling.

The de-assertion of SYNC_N is synchronous to the internal frame clock and will only occur at the LMFC boundary. This control means that the LMFC phase information is sent to the transmitter device through the SYNC interface. If the transmitter is a converter device it shall have the ability to adjust its LMFC and frame clock relative to the de-assertion of SYNC_N. If the transmitter is a logic device it captures the de-asserts of SYNC_N relative to its LMFC and the correction information shall be contained in PHADJ, ADJDIR and ADJCNT parameters on the link configuration data. The adjustment resolution is set in the transmitter and is stored in ADJCNT. When PHADJ is set (= 1) a phase adjust has been requested by the transmitter. ADJDIR value indicates an advance (= 0) or delay (= 1) adjustment. Since CoreJESD204BRx is on a logic device and there is no requirement for any adjustments to be made based on the adjustment information received in the link configuration data. The adjustment information is stored in CoreJESD204BRx and can currently only be viewed in simulation or on a logic analyzer.

Note: More information about deterministic latency can be found in section 6 of the JESD204B.01 specification.

3.3 SERDES Interface

For RTG4, IGLOO2, SmartFusion2 family devices CoreJESD204BRX is connected to the receiver lanes of the respected family's serializer/deserializer (SERDES) interface macro. The SERDES interface must be configured in EPCS mode as this macro is used as a standalone SERDES. EPCS mode has two default data rates, which are 1.25 Gbps and 2.5 Gbps per lane. This means 10 Gbps can be achieved across the 4 lanes of the CoreJESD204BRX for most configurations. The SERDES interface can be reconfigured through the advanced peripheral bus (APB) interface to allow EPCS mode to achieve a higher data rates.

For PolarFire family devices the SERDES has a hardened 8B10B decoder which means CoreJESD204BRX can be configured with its decoder removed (that is, DECODER_EN = 0). CoreJESD204BRx will connect to the 8B10B interface on the SERDES. The PolarFire SERDES can be configured up to 64bit data width (8 octets) and CoreJESD204BRx supports this data width which means up to 12.5Gbps data rate is achievable per lane.

4 Interface

4.1 Configuration Parameters

CoreJESD204BRX has parameters (Verilog) or generics (VHDL) for configuring the RTL code, described in Table 7, page 12. All parameters and generics are integer types.

Table 7 • **CoreJESD204BRX Parameters and Generics Descriptions**

Name	Range	Default Value	Description
CF	0 to 7	0	Number of control words per frame clock period per link. It controls, which lanes will carry control words. CF=0 means that no control words are used. Other allowed CF values are common sub-divisors of Number of lanes (L) and Number of converters (M).The L lanes are divided into CF groups of L/CF lanes. Each group of lanes transmits the samples of M/CF converters. After these samples a control word is inserted, containing in successive order the control bits belonging to these samples. If the control word fits on a single lane, it is not allowed to break it over a lane boundary.
CS	0 to 3	0	Number of control bits per sample
F	2 to 256	2	Number of octets per frame
HD	0 to 1	0	High density format. HD controls whether a sample may be divided over more lanes. In the Low Density mode (HD=0), partial conversion words at the end of a group of F octets are avoided by adding more tail bits (TT) after the last full nibble group (NG) in the group if necessary. In the High Density mode (HD=1), the conversion words may break at the lane boundary.
JESDV	0, 1	1	JESD204 version
			0: JESD204A
			1: JESD204B
K	4 to 32	9	Number of frames per multi-frame
L	0 to 7	0	Number of lanes per converter device (link) **Note:** L+1= No. of lanes
M	1 to 256	1	Number of converters per device
N	1 to 32	16	Converter resolution
N’	1 to 32	16	Total number of bits per sample
S	1 to 32	2	Number of samples per converter per frame cycle
SCR	0 to 1	1	Scrambling enabled
SUBCLASSV	0 to 2	0	Device subclass version
			0: Subclass 0
			1: Subclass 1
			2: Subclass 2

Table 7 • **CoreJESD204BRX Parameters and Generics Descriptions**

FAC_EN	0 to 1	1	Frame alignment correction enable. 0: Disabled 1: Enabled **Note:** User can disable the resynchronization , because without scrambling, certain types of periodic data may not produce enough alignment characters for reliable detection of frame misalignment.
RAM_SEL	0 to 2	0	RAM implementation parameter. 0: In FPGA fabric (only valid for K*F<=20) 1: In µSRAM 2: In LSRAM
FIELD_OCTET	0 to 1	1	Configuration Options parameter which selects how the FCHK(checksum) of the link configuration parameters will be calculated 0: Field addition 1: Octet addition
SERDES_MODE	1 to 2	1	Configuration Options parameter which selects the width of the input data 1 : ((D_WIDTH/8) * 10) 2 : 10bit (when DECODER_EN = 1), : 8bit (when DECODER_EN = 0) **Note:** When SERDES_MODE=2 LANE_CLK is a slower frequency than EPCS Rx Clock.
LCD_EN	0 to 1	1	Link Configuration Error Detection Enable. 0: Disabled 1: Enabled **Note:** Link configuration error may not always require detection or action in each application. When disabled, it improves resource utilization.
D_WIDTH	16, 32, or 64	16	Configuration Options parameter which selects the user data width. 0 : 16bit (2 octet) 1 : 32bit (4 octet) 2 : 64bit (8 octet)
DECODER_EN	0 to 1	1	Configuration Options parameter which includes/removes the decoder in the core. 0 : Removed 1 : Included
ILA_MFS	4 to 256	4	Initial lane alignment multi-frame size. This value must match the transmitters ILA multi-frame size (typical 4). **Note:** RX_STATE_* requires this to be set correctly to accurately detect transition from INIT_LANE_ST to DATA_DEC_ST.

Note: A multiframe is defined as a group of K successive frames, which are made up of F successive octets, where K is valid from 4 to 32 and F is valid from 2 and 256, such that the number of octets per multiframe is between 18 and 1024.

4.2 I/O Signals

The port signals for the CoreJESD204BRX macro are shown in Figure 3, page 14 and Figure 4, page 14.

Figure 3 • **CoreJESD204BRX I/O Signal Diagram (4 lanes) with Decoder**

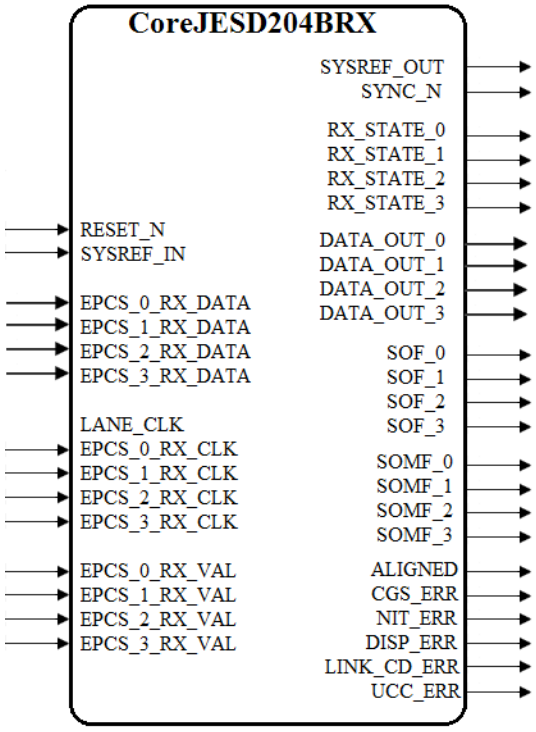


Figure 4 • **CoreJESD204BRX I/O Signal Diagram(4 lanes) without Decoder**

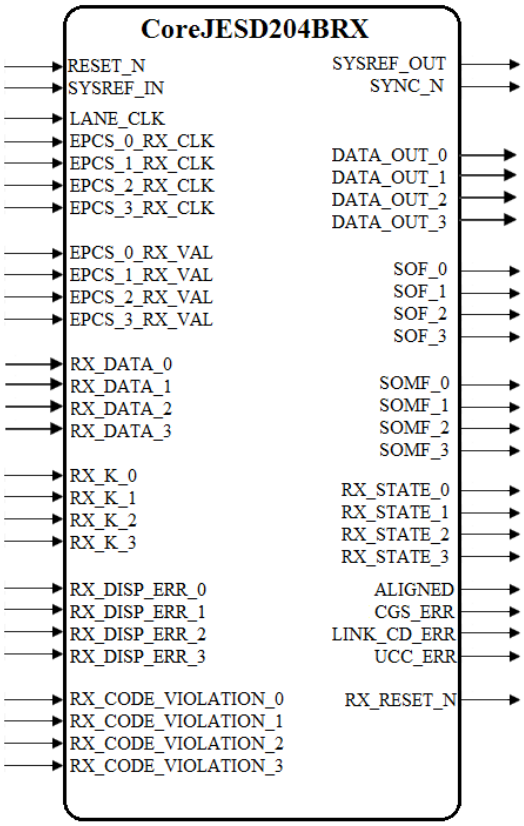


Table 8 • **CoreJESD204BRX I/O Signal Description**

Name	Width	Direction	Description
RESET_N	1	In	This active low reset will reset the core to its initial state. This IP core has internal reset synchronizers per clock domain. For RTG4 family devices, RESET_N is used as a synchronous reset and to guarantee successful synchronization this signal should be asserted for a minimum of three clock cycles of LANE_CLK frequency.
External PCS Receive Interface			
EPCS_[n]_RX_DATA	10, ((D_WIDTH/8)*10)	In	EPCS receiver channel [n] data. This input should match with the width of the EPCS receiver channel on the SERDES. If SERDES_MODE = 2 then Width = 10 If SERDES_MODE = 1 then Width = (D_WIDTH/8)*10 When DECODER_EN=0 this signal is tied low. **Note:** [n] = 0 to 7
EPCS_[n]_RX_CLK	1	In	EPCS receiver channel [n] clock. EPCS receiver channel 0 clock must be connected to a clock source because it is used for the JESD lanes internally. **Note:** [n] = 0 to 7
EPCS_[n]_RX_VAL	1	In	EPCS receive channel [n] valid data signal. This is used along with RESET_N to reset the IP core. This IP core has internal Reset synchronizers per clock domain. **Note:** [n] = 0 to 7
8B10B Decoder Interface			
RX_DATA_[n]	8, D_WIDTH	In	8b10b data for channel[n] data. This input should match with the width of the 8b10b decoder on the SERDES. If SERDES_MODE = 2 then Width = 8 If SERDES_MODE = 1 then Width = D_WIDTH When DECODER_EN=1 this signal is tied low. **Note:** [n] = 0 to 7
RX_K_[n]	1, (D_WIDTH/8)	In	8b10b K control for channel[n] data. If SERDES_MODE = 2 then Width = 1 If SERDES_MODE = 1 then Width = D_WIDTH/8 When DECODER_EN=1 this signal is tied low. **Note:** [n] = 0 to 7
RX_RESET_N	1	Out	Receiver channel reset. This signal is only required for when parameter/generic DECODER_EN=0 and it should be tied to all active receiver channel resets.

RX_DISP_ERR_[n]	1, (D_WIDTH/8)	In	8b10b Disparity error for channel [n]. The received code group exists in the 8B/10B decoding table, but is not found in the proper column according to the current running disparity. **Note:** [n] = 0 to 7 **Note:** When DECODER_EN=1 this signal is tied low.
RX_CODE_VIOLATION_[n]	1, (D_WIDTH/8)	In	8b10b Not In table error for channel [n]. The received code group is not found in the 8B/10B decoding table for either disparity. **Note:** [n] = 0 to 7 **Note:** When DECODER_EN=1 this signal is tied low.
JESD204B Rx Interface			
LANE_CLK	1	In	This is the clock all lanes will sync to internally in the IP core. When SERDES_MODE = 2, EPCS_*_RX_CLK to LANE_CLK ratio's are as follows. 2:1 when D_WIDTH = 16bit 4:1 when D_WIDTH = 32bit 8:1 when D_WIDTH = 64bit When SERDES_MODE = 1, EPCS_*_RX_CLK to LANE_CLK ratio is 1:1 **Note:** It is recommended to generate this clock using Clock Conditioning Circuitry (CCC). Input clock to the CCC should be EPCS_0_RX_CLK.
SYSREF_IN	1	In	This is the SYSREF input signal used by JESD204B sub-class 1.
SYSREF_OUT	1	Out	This is the SYSREF output signal generated from inside the core. This or an external SYSREF signal is tied to SYSREF of the receiver or it can be used to loopback into the SYSREF_IN of the transmitter on a number of factors (sub-class 1 only).
DATA_OUT_[n]	D_WIDTH	Out	This is the data output for channel[n]. **Note:** [n] = 0 to 7
ALIGNED	1	Out	All lanes are aligned where this signal is asserted.
SYNC_N	1	Out	Asserts when an initialization/re-initiation request occurs and also indicates loss-of-sync.. **Note:** The "_N" in the name represents active low and not the negative of a differential. When connecting to a differential output this signal must be connected to the positive rail or to the D input of an OUTBUF_DIFF macro.
DISP_ERR	L+1	Out	Disparity error. The received code group exists in the 8B/10B decoding table, but is not found in the proper column according to the current running disparity. **Note:** Required only when DECODER_EN=1

NIT_ERR	L+1	Out	Not In table error. The received code group is not found in the 8B/10B decoding table for either disparity. **Note:** Required only when DECODER_EN=1
UCC_ERR	L+1	Out	Unexpected Control Character Error: A control character is received that is not expected at the given character position.
CGS_ERR	L+1	Out	Code Group Sync Error: The state machine for code group synchronization has returned to the CS_INIT state.
LINK_CD_ERR	L+1	Out	Link Configuration Data Mismatch Error.
SOF_[n]	D_WIDTH/8	Out	Monitors when start-of-frame occurs on channel[n]. **Note:** [n] = 0 to 7
SOMF_[n]	D_WIDTH/8	Out	Monitors when start-of-multi-frame occurs on channel[n]. **Note:** [n] = 0 to 7
RX_STATE_[n]	2	Out	Monitors the current state of receiver channel [n] 00: SYNC_ST 01: CHECK_ST 10: INIT_LANE_ST 11: DATA_DEC_ST

5 Clocking and Reset

This section describes the options available for clocking and reset the IP core.

5.1 Clocking

Following clocks are used in the IP core.

5.1.1 EPCS_[n]_RX_CLK

EPCS receiver channel [n] clock. This clock is used to sample EPCS receiver channel [n] data.

Note: [n] = 0 to 7

5.1.2 LANE_CLK

This is the clock to which all lanes are synchronized internally in the IP Core. This clock must be generated using EPCS_0_RX_CLK. For frequency relation of the LANE_CLK to EPCS_[n]_RX_CLK, please refer LANE_CLK description in Table 8 on page 15.

Note: [n] = 0 to 7

5.2 Reset

Following resets are used in the IP core.

5.2.1 RESET_N

For RTG4 family, active low RESET_N is used as synchronous reset to reset the IP core. To guarantee successful synchronization, this signal should be asserted for a minimum of three clock cycles of LANE_CLK frequency.

For PolarFire/SmartFusion2/IGLOO2 families, active low RESET_N is used as asynchronous reset to reset the IP core.

5.2.2 EPCS_[n]_RX_VAL

Similar to RESET_N, EPCS_[n]_RX_VAL is used as synchronous reset to reset the IP core for RTG4 family and it is used as asynchronous reset for PolarFire/SmartFusion2/IGLOO2 families. This IP core has internal synchronizers per clock domain.

Note: [n] = 0 to 7

Following figure shows the reset synchronizer for RTG4 family.

Figure 5 • **Reset Synchronizer for EPCS_[n]_RX_CLK**

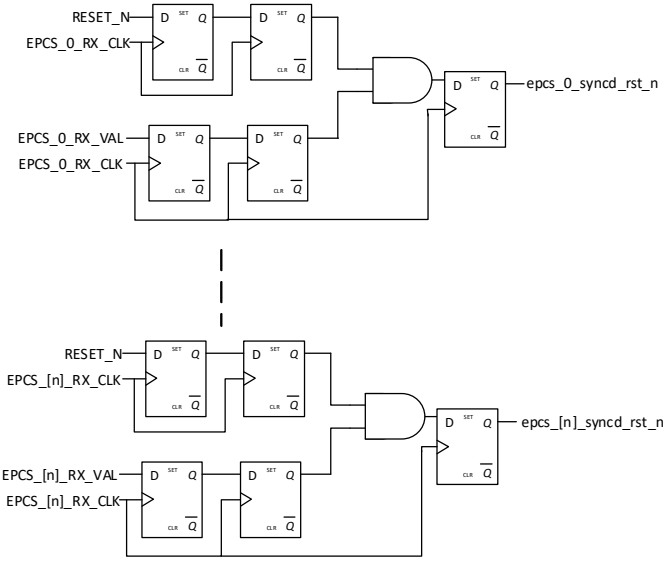
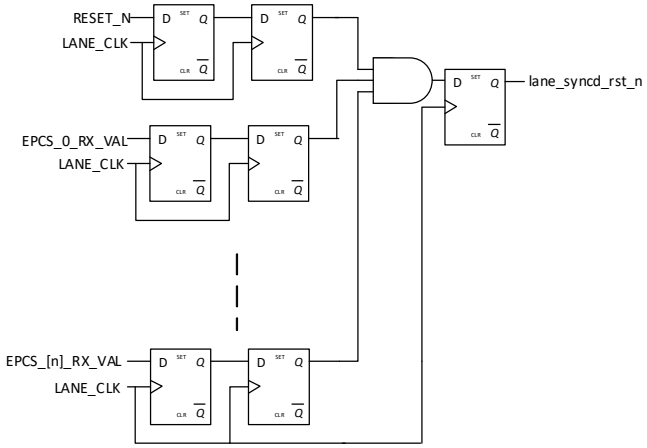


Figure 6 • **Reset Synchronizer for LANE_CLK**



Following figure shows the reset synchronizer for PolarFire, SmartFusion2, and IGLOO2 families.

Figure 7 • **Reset Synchronizer for EPCS_[n]_RX_CLK**

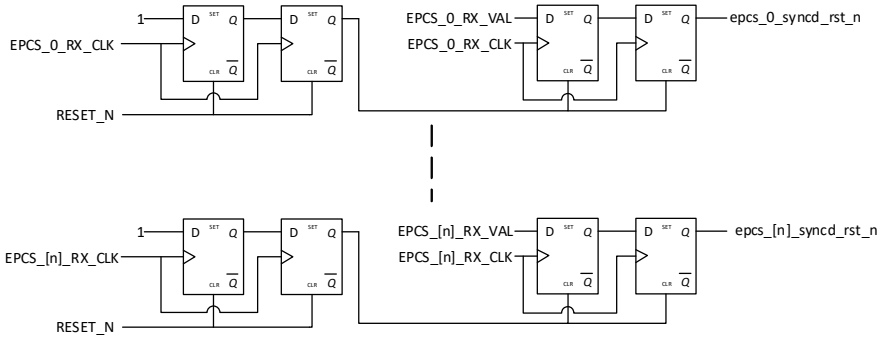
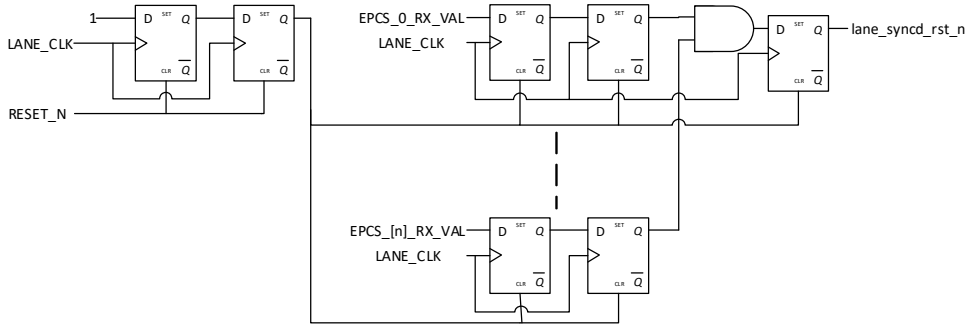


Figure 8 • **Reset Synchronizer for LANE_CLK**



Note: epcs_[n]_syncd_rst_n and lane_syncd_rst_n are used as active low syncronous reset for RTG4 family and for PolarFire, SmartFusion2, and IGLOO2 families these are used as active low asynchronous reset into their respective clock domain.

6 Tool Flow

6.1 License

This core will support generation of un-obfuscated Verilog and VHDL versions of the core. The un-obfuscated Verilog and VHDL versions will be license locked at the time of packaging. The core will be included in the Libero SoC IP bundle with Gold and Platinum licenses in clear RTL form.

6.2 SmartDesign

CoreJESD204BRX is pre-installed in the SmartDesign IP deployment design environment.

The core should be configured using the configuration GUI within the SmartDesign tool, as shown in Figure 11. For information on using SmartDesign to instantiate and generate cores, refer to Libero SoC online help.

Figure 9 • **CoreJESD204BRX Full I/O View (4 lanes) with Decoder**

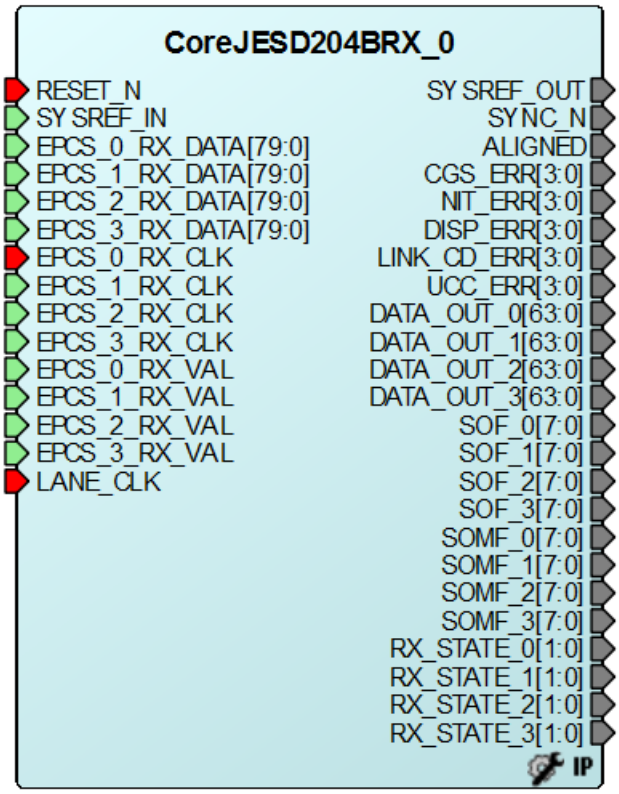
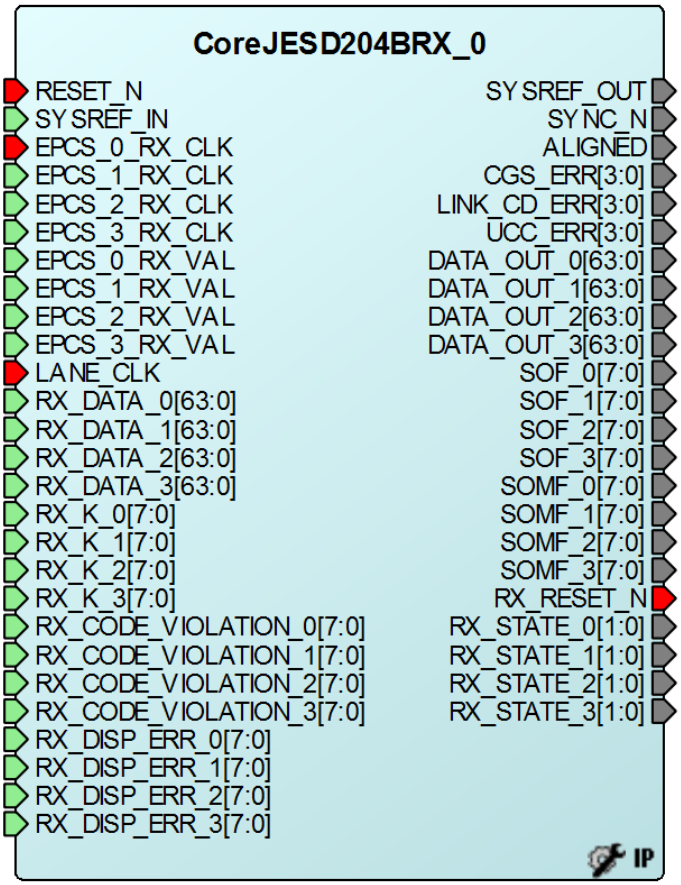


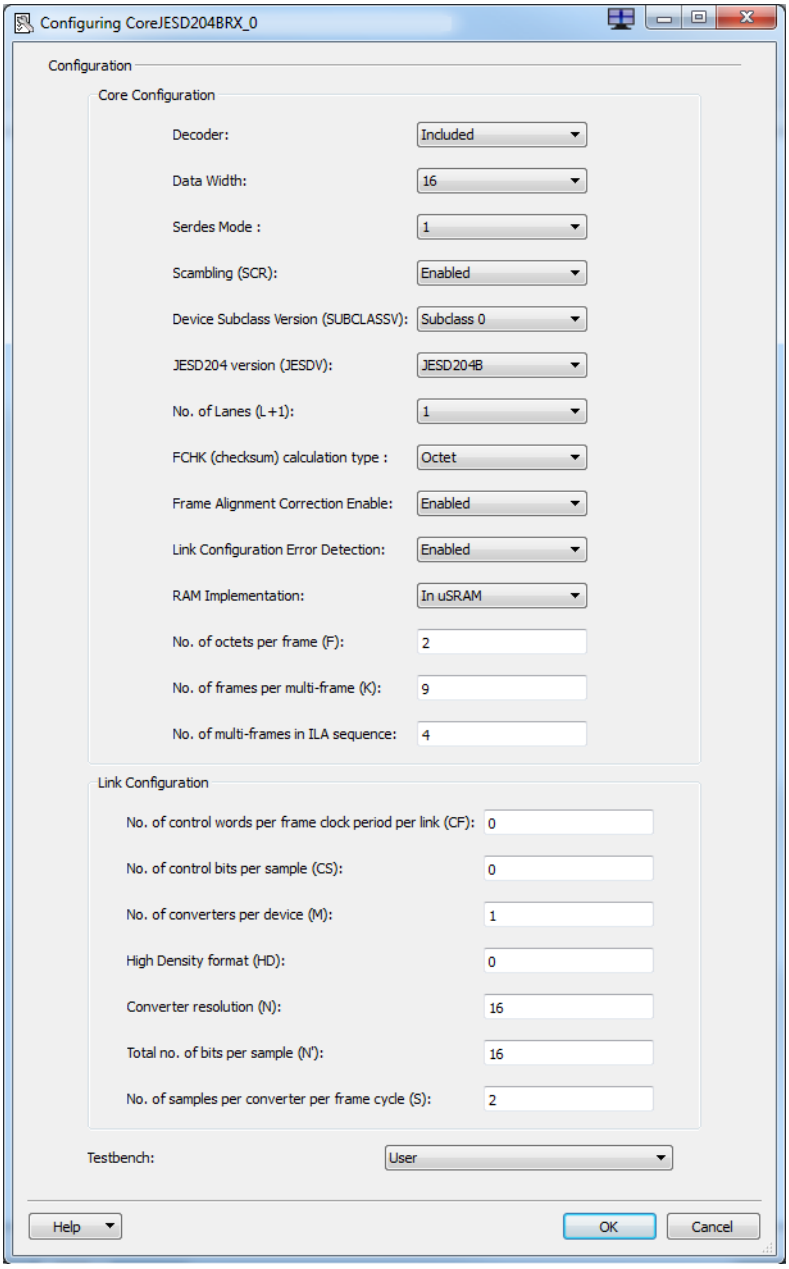
Figure 10 • **CoreJESD204BRX Full I/O View (4 lanes) without Decoder**



6.3 Configuring CoreJESD204BRX in SmartDesign

The core can be configured using the configuration GUI within SmartDesign. An example of the GUI for the SmartFusion2 family is shown in the following figure.

Figure 11 • **CoreJESD204BRX SmartDesign Configuration GUI**



6.4 Simulation Flows

The User Testbench for CoreJESD204BRX is included in all releases.

To run simulations, select the User Testbench flow within the SmartDesign CoreJESD204BRX configuration GUI, right-click the canvas, and select **Generate Design**.

When SmartDesign generates the design files, it installs the user testbench files.

To run the user testbench, set the design root to the CoreJESD204BRX instantiation in the Libero SoC design hierarchy pane and click **Simulation** in the **Libero SoC Design Flow** window. This invokes ModelSim® and automatically runs the simulation.

6.5 Synthesis in Libero

After setting the design root appropriately for your design, click **Synthesis** in the Libero SoC software. The **Synthesis** window appears, displaying the Synplicity® project. Set Synplicity to VHDL 2008 standard if VHDL is being used. To run Synthesis, click **Run**.

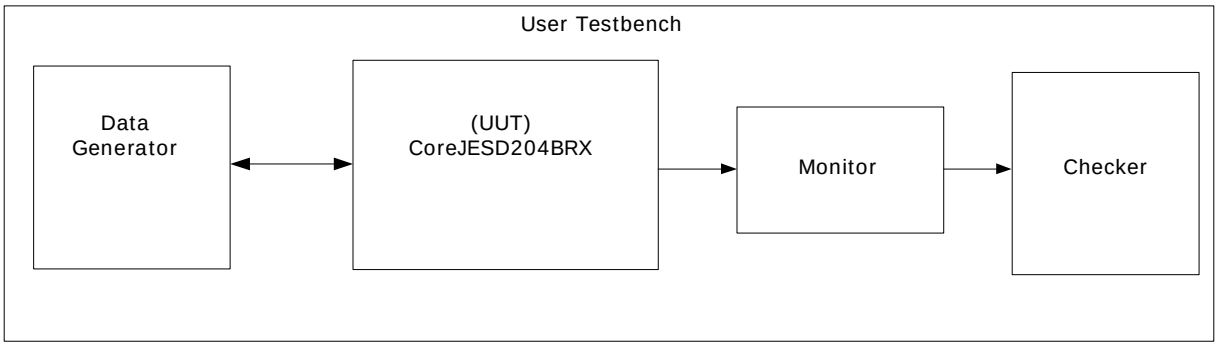
6.6 Place-and-Route in Libero

After setting the design root appropriately for your design, and running Synthesis, click **Layout** in the Libero SoC software to invoke Designer. CoreJESD204BRX requires no special place-and-route settings.

7 Testbench

CoreJESD204BRX user testbench gives an example of how to use the core.

Figure 12 • **CoreJESD204BRX User Testbench**



The simulation testbench includes an instantiation of the CoreJESD204BRX macro, data generation, data monitor, and checker. Because the data on the input of the CoreJESD204BRX core is 8B10B encoded and scrambled, the user testbench is only used to test the core in a fixed configuration for each sub-class. The purpose of the testbench is to test the functionality of the core by inputting known data, monitoring the output, and checking for expected results.