PolarFire Family Transceiver User Guide

Introduction (Ask a Question)

The PolarFire® family of FPGAs include multiple embedded low-power, performance-optimized transceivers. Each transceiver has both the Physical Medium Attachment (PMA), protocol Physical Coding Sub-Layer (PCS) logic, and interfaces to the FPGA fabric. The transceiver has a multi-lane architecture with each lane natively supporting serial data transmission rates from 500 Mbps (250 Mbps with interpolation) to 12.7 Gbps.

This user guide describes the transceiver block available in the PolarFire family. The FPGA fabric is common to the PolarFire family, which consists of the following FPGA devices.

PolarFire FPGAs	Microchip's PolarFire FPGAs are the fifth-generation family of non-volatile FPGA devices, built on state-of-the-art 28 nm non-volatile process technology. PolarFire FPGAs deliver the lowest power at mid-range densities. PolarFire FPGAs lower the cost of mid-range FPGAs by integrating the industry's lowest power FPGA fabric, lowest power 12.7 Gbps transceiver lane, built-in low power dual PCI Express Gen2 (EP/RP), and, on select data security (S) devices, an integrated low-power crypto co-processor.
PolarFire SoC FPGAs	Microchip's PolarFire SoC FPGAs are the fifth-generation family of non-volatile SoC FPGA devices, built on state-of-the-art 28 nm non-volatile process technology. The PolarFire SoC family offers industry's first RISC-V based SoC FPGAs capable of running Linux®. It combines a powerful 64-bit 5x core RISC-V Microprocessor Subsystem (MSS), based on SiFive's U54-MC family, with the PolarFire FPGA fabric in a single device.
RT PolarFire FPGAs	Microchip's RT PolarFire FPGAs combine our 60 years of space flight heritage with the industry's lowest-power PolarFire FPGA family to enable new capabilities for space and mission-critical applications. RT PolarFire FPGA family includes RTPF500T, RTPF500TL, RTPF500TS, RTPF500TLS, RTPF500ZT, RTPF500ZTL, RTPF500ZTS, and RTPF500ZTLS devices.

The following table summarizes the transceiver components available in the PolarFire family.

Table 1. Transceiver Components

Components	PolarFire FPGA (MPF)	RT PolarFire FPGA (RTPF)	PolarFire SoC FPGA (MPFS)
PMA/PCS	✓	✓	✓
TXPLL	✓	✓	✓
REF_CLK	✓	✓	✓

The transceiver includes all required analog functions for high-speed data transmission between devices over printed circuit boards (PCB) and high-quality cables. The transceiver is suitable for a variety of device-to-device communication protocols, as listed in the following table.

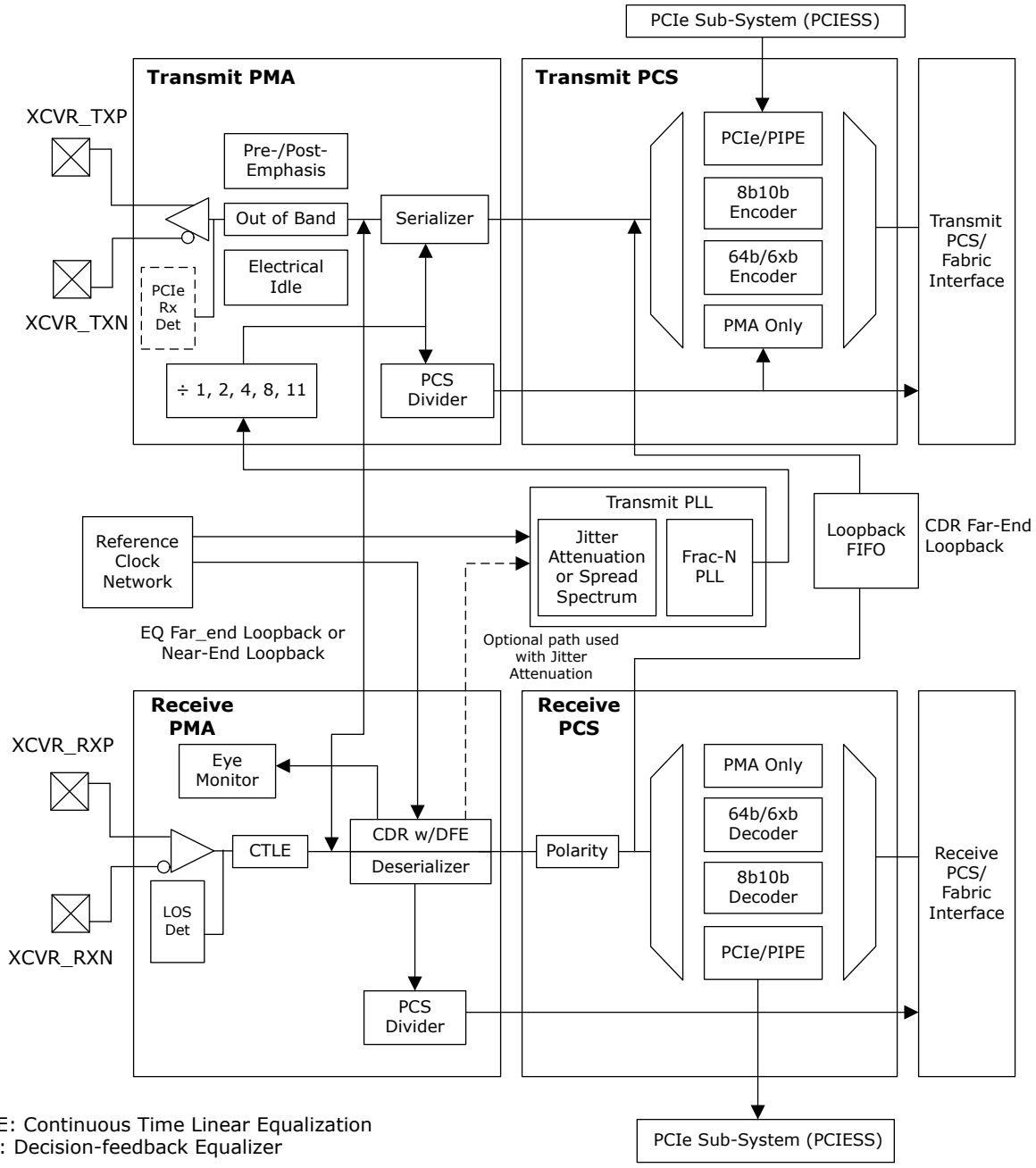
Table 2. Supported Serial Protocols

Protocol	Standard/Version	Protocol	Standard/Version
PCIe®	Gen1	CPRI [1]	CPRI-1, 2, 3, 4, 5, 6, 7, 7A, 8, 9
PCIe	Gen2	SATA [2]	1.0a
XAUI	IEEE® 802.3	SATA [2]	2.0
Interlaken [3]	10.3125	SATA [2]	3.0
Interlaken [3]	6.375	SDI-SD	SMPTE 259M
10GBASE-R	IEEE 802.3	SDI-HD	SMPTE 292M
10GBASE-KR [4]	IEEE 802.3	SDI-3G	SMPTE 424M
Fibre Channel	1GFC	SGMII	—
Fibre Channel	2GFC	1000BASE-X	IEEE 802.3
Fibre Channel	4GFC	LiteFast	Proprietary Lightweight Serial Protocol Interface (up to 12.7 Gbps, 8b10b mode only.)
Fibre Channel	8GFC	RXAUI	N/A
JESD204B	LV-OIF-SxI5	QSGMII	—
JESD204B	LV-OIF-6G-SR	SLVS-EC	—
JESD204B	LV-OIF-11G-SR	DisplayPort	—
USXGMII	—	CoaXPress	—

Important:

1. Common Public Radio Interface (CPRI) Specification V7.0 (2015-10-09).
2. SATA is supported using PMA only mode using customer provided RTL IP.
3. Interlaken Protocol Specification, v1.2.
4. PMA supports 10GBASE-KR operation with -1 SPD only.

Figure 1. Transceiver Lane Overview



Important: Transmit/receive fabric interfaces are specified in the associated PCS pin lists, that is, 8b10b, 64b6xb, PIPE, and PMA Only. For more information on PCIESS, see PolarFire Family PCI Express User Guide.

Features (Ask a Question)

Transceiver enables users to quickly build high-speed links that support many standard protocols with the features listed:

- Supports data rates from 500 Mbps (250 Mbps with interpolation) up to 12.7 Gbps.
  RT PolarFire supports up to 10.3125 Gbps
- Serialization/deserialization width at FPGA fabric interface—8, 10, 16, 20, 32, 40, 64, and 80 bits.
- Differential output termination from 85Ω, 100Ω, and 150Ω.
- Low-power modes.
- Receivers are compatible with CML and LVDS I/O Standards.
- Transmitters are compatible with CML, LVDS, and LVPECL I/O Standards.
- Configurable transmit pre- and post-tap de-emphasis controls.
- Configurable amplitude control from 250 mV to 1V differential.
- Receivers detect circuitry for use with PCIe.
- Out-of-band (OOB), electrical idle signaling capability for
  - Serial-attached SCSI: small computer system interface (SAS).
  - Serial advanced technology attachment (SATA).
  - Peripheral component interconnect express (PCIe).
- Spread-spectrum generation built into the transmit Phase-Locked Loop (PLL).
- 1 Gb and 10 Gb SyncE compatible Jitter attenuation available in the transmit PLL for loop timing applications.
- Continuous time linear equalizer (CTLE) with optional auto-calibration to improve received signal integrity.
- 5-tap decision feedback equalizer (DFE) with auto-calibration to compensate for high-frequency losses.
- Receive data eye monitor for link analysis.
- Configurable peak detector/signal detect.
- Polarity inversion (receiver).
- Diagnostic loopback modes.
- Embedded pseudo-random binary sequence (PRBS) test pattern generators/checkers—available PRBS polynomials (2^n), where n = 7, 9, 15, 23, and 31.
- AC JTAG (IEEE® 1149.6) and DC JTAG (IEEE 1149.1) transmitter and receiver.
- IBIS-AMI support of transceiver inputs and outputs.
- Supports AC and DC coupling modes with configurable transmit common-mode voltage.
- Embedded PCS:
  - 8b10b—encoding/decoding is provided.
  - 64b6xb—64b/66b or 64/67b encoding/decoding with gearbox logic is provided.
  - PIPE—PHY interface for the PCI Express Rev 3.0 supporting PCIe Gen1/2.
  - PMA only—direct access to the PMA without any encoding.
  - PCIe—fully embedded PCIe Gen1/Gen2 root-port or endpoint subsystem (PCIESS) with AXI4 user interface with built-in DMA. The embedded PCIe controller subsystem is available only within Quad0. See PolarFire Family PCI Express User Guide for more information on the embedded PCIE capabilities and its usage.

The Microchip Libero® SoC software supports configuration for the various modes of transceiver operations. Table 2 shows which of these configurations support industry-standard protocols and user-defined custom protocols. The Libero SoC software design tools allow designers to set the configuration needed for a specific operational mode for each transceiver lane. The software correctly provisions and generates all the required

programming and configuration data used to initialize and bring the transceiver into operation. The transceiver configuration registers are set automatically by the Libero SoC transceiver configurator. These registers must be left at their default values set by the configurator, except for use cases that explicitly request different values.

References (Ask a Question)

- For information about PCIESS, see PolarFire Family PCI Express User Guide.
- For information about regional clock resources and connectivity of the transceivers to the global clock network, see PolarFire Family Clocking Resources User Guide.
- For information about available I/O standards, see PolarFire FPGA and PolarFire SoC FPGA User I/O User Guide.
- For information about Optimized DFE feature, see PolarFire FPGA SmartDebug User Guide.
- For information about implementing transceiver designs on printed circuit boards, see respective UG0726: PolarFire FPGA Board Design User Guide, RT PolarFire FPGA Board Design User Guide, or PolarFire SoC FPGA Board Design Guidelines User Guide.

Table of Contents

1. Functional Description (Ask a Question)

The transceiver (Figure 1) is divided into four distinct transmit (Tx) and receive (Rx) blocks:

- PMA
- PCS interface block, including a dedicated PCIe PCS
- Transmit PLL (Tx PLL)
- Reference clock inputs

The high-speed PMA blocks connect to the FPGA fabric through the PCS block. The PMA generates the required clocks and converts the transmit data from parallel to serial, and receive data from serial to parallel. Each PMA block includes a connection to a PCS block and associated interface to the FPGA fabric making up a transceiver lane. The PCS interface block provides several industry-standard interfaces for use in protocol-specific designs.

A group of four transceiver lanes is called a quad. Each quad has a local transmit PLL used exclusively within the four transceiver lanes. Additional transmit PLLs are shared between quads.

In addition to the 8b10b, 64b6xb, PIPE, and PMA only blocks, two PCIe PCS logic blocks are included in each device. These blocks include hard embedded logic that provides full-featured PCIe endpoint/root port sub-system. These PCIe sub-systems (PCIESS) have hard connections to multiple transceiver lanes, providing flexibility for ×1, ×2, and ×4 width links. See PolarFire Family PCI Express User Guide for additional information pertaining to PCIe.

1.1. PMA (Ask a Question)

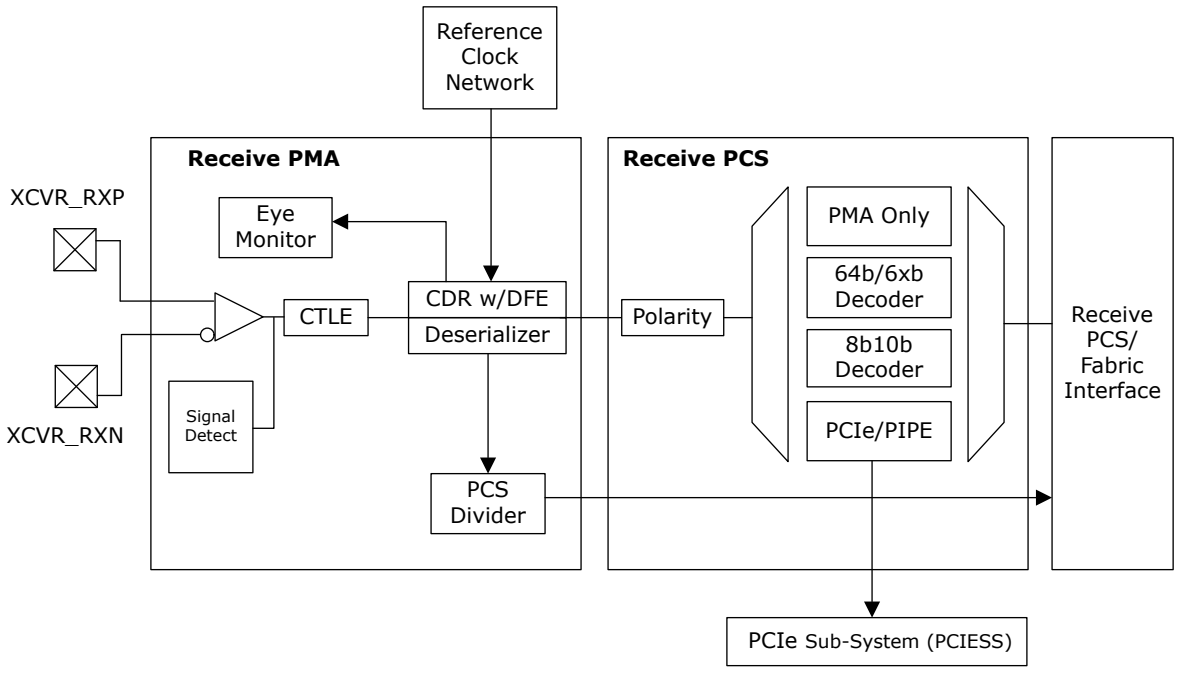
The transceiver lanes include PMA receiver and transmitter sub-modules. These PMA sub-modules include the input and output buffers, signal conditioning circuits, CDRs, and transceiver. The PMA architecture allows the receive and transmit portions of each lane to operate independently. The PMA features are initialized at power-up and can also be altered during device operation using an APB dynamic reconfiguration interface (DRI).

The SmartDebug tool set provides access to dynamic changes of PMA features, including transmit and receive tuning, and receive eye monitoring capabilities.

1.1.1. Receiver (Ask a Question)

The receiver deserializes high-speed serial data received through the input buffer by creating a parallel data stream for the FPGA fabric and recovering the clock information from the received data. The receiver portion of the PMA includes the receiver buffer, the clock and data recovery (CDR) unit, and the deserializer. The deserializer within the receive PMA passes deserialized data to the PCS block across a data bus up to 40-bits wide of the PMA-PCS interface, which provides the data path to the gearing logic before the data is passed to the FPGA fabric.

Figure 1-1. Transceiver Receiver

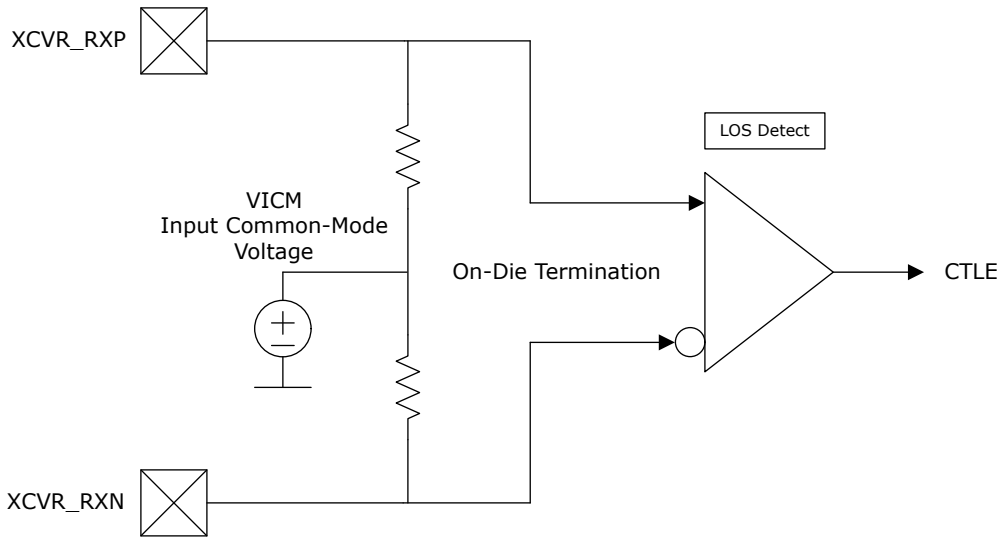


1.1.1.1. Receive Input Buffer (Ask a Question)

The receiver provides an external input interface through differential pins, XCVR_RXP/N, as shown in the following figure. The receiver includes a current mode logic (CML) input buffer with programmable DC restoration that is used in either AC- or DC-coupled applications.

The receive buffer provides an on-die differential termination scheme which can be programmed to 85Ω, 100Ω, or 150Ω. The receiver buffer also includes a high-impedance (high-Z) mode for hot-swap capability when the device is powered OFF. Additionally, the receiver input supports logical swapping of the polarity of the P and N pins for added flexibility.

Figure 1-2. Receiver Input Buffer



Important: VICM is connected to VDDA when AC-coupled link is configured. DC-coupled configurations effectively disconnects VICM source. See AC/DC Coupled Connection.

Each receiver lane includes optional signal threshold detection circuitry that users can select according to protocol or application requirements. This feature identifies whether the signal level present at the receiver input buffer is above the signal detect threshold voltage needed to trip or activate the receiver input, which prevents false activity on the receiver path. The signal detection has both a high and low signal detector. The Libero SoC software configurator provides the correct setting based on protocol or customization.

Important: The user can also use a JTAG-based interface from SmartDebug to experiment with receiver settings.

1.1.1.2. Loss of Signal Detect (LOS) (Ask a Question)

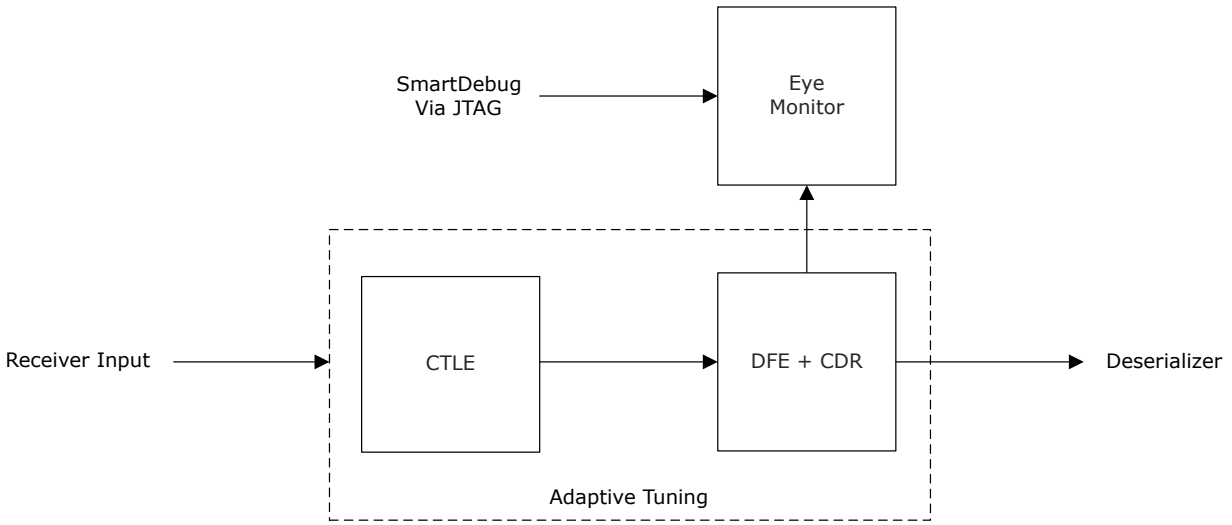
Loss of signal (LOS) detection is included within the receiver path. The LOS circuitry detects the initial incoming signal determining a valid input (electrical RX_IDLE=0) for clock-data recovery operations. The LOS peak detection captures the most positive and negative points of the input signal and compares the amplitude to a limit set by the user. See Loss-of-Signal Detector. The performance of the physical peak detector is limited by the bandwidth of the input signal. The LOS detection works for rates 5 Gbps and less and may not be suitable for all protocols or data patterns. For conditions outside the range of the LOS operation, see Enhanced Receiver Management.

1.1.1.3. Continuous-Time Linear Equalizers (CTLE) (Ask a Question)

The CTLEs equalize a lane's low-pass response to compensate for high-frequency losses in that lane, thereby improving the quality of the received signal. This circuit can be adjusted to compensate for any physical lane mismatches.

There are two transparent stages of CTLE and a separate pair of stages for the decision feedback equalizer (DFE)/eye monitor receive path. The input signal path (Figure 1-3) is conditioned by tuning the incoming signal allowing the user to observe the effects of the tuning. The DC gain and peak bandwidth of each stage is selected with Libero. CTLE settings can be selected based on DC gain, peaking frequency, and AC gain or with an auto adaptive setting through the Libero transceiver interface configurator. The automatic or adaptive mode uses internally generated settings to the physical channels for lane optimization.

Figure 1-3. Input Signal Path



1.1.1.4. Decision Feedback Equalizer (Ask a Question)

In the receiver front end, an optionally enabled 5-tap decision feedback equalizer (DFE) is available to equalize the lane response in conjunction with the CTLE. The DFE allows better compensation of transmission channel losses than a linear equalizer of CTLE, by providing a closer adjustment of filter parameters. The tap values of the DFE are the coefficients of this filter that are set by the adaptive algorithm.

The DFE mitigates lane noise or inter-symbol interference (ISI) caused by reflections or cross-talk without amplifying the high-frequency noise within the data. The DFE-based operation uses current bit information to cancel ISI for the following bit through a feedback mechanism, allowing the following bits to be correctly sampled. Using taps to delay and multiply the symbols, the DFE effectively cancels out interference on the analog signal. Similar to the CTLE operation, the DFE has an automatic mode. When the DFE is used in automatic mode, the CTLE can be in automatic mode as-well.

The operation is nonlinear, allowing it to overcome the notch response that the CTLE cannot perform. The DFE also includes an automatic calibration that finds the best possible tuning to match the transceiver lane to the system channel.

1.1.1.5. Eye Monitor (Ask a Question)

The eye monitor is on-device circuitry to visualize the post-equalization signal quality in the receive path while the data path is still active in the system. The non-destructive eye monitor runs a separate sampler in parallel with the CDR and DFE data sampler. This permits the system to remain operational while the eye monitor is functioning.

The eye monitor systematically adjusts the offsets across the complete eye, calculates the bit-error rate (BER) for each offset setting, then correlates the BER and offset to statistically rebuild the eye diagram. Eye diagram statistics can be read and reconstructed using the Libero SmartDebug tools, which permits access through a JTAG interface for transceiver debugging and test access. For more information, see Eye Monitoring.

1.1.1.6. Receive Clock and Data Recovery (Ask a Question)

The receive CDR circuit follows the CTLE and works in tandem with the DFE. The receive CDR PLL can lock onto the input reference clock or the incoming data stream to be able to re-time the incoming data. The deserializer is closely coupled with the CDR, and translates the data from a serial to a parallel stream.

1.1.1.6.1. CDR Options (Ask a Question)

The PMA of each lane includes a PLL used for the receiver CDR. The CDR PLL supports lock-to-reference and lock-to-data modes, which allows customization of the CDR options best suited for the application. It also includes a Burst-mode receiver option, which can switch between both options that are selectable through the Libero transceiver configurator.

Lock-to-Reference: The phase frequency detector (PFD) in the CDR tracks the receiver input reference clock. The PFD controls the charge pump that tunes the VCO in the CDR. The LOCK status signal is asserted high to indicate that the CDR has locked to the phase and frequency of the receiver input reference clock regardless of the data phase detector (PD). Lock-to-reference is used to lock the transceiver CDR to the reference clock rather than the incoming data when the receiver is used as a simple over-sampler, or when the CDR must be locked to a local oscillator.

Lock-to-Data: The CDR must use the lock-to-data mode to recover the clock and data from the incoming serial data. In this mode, the data phase detector of the CDR tracks the incoming serial data at the receiver input. Depending on the phase difference between the incoming data and the CDR output clock, the PD controls the CDR charge pump that adjusts the VCO. The LOCK status signal is asserted when the CDR finds valid data. The actual lock time depends on the incoming data stream's transition density.

Burst Mode Receiver: The transceiver CDR circuit has enhanced capabilities to support burst mode receivers (BMR). BMR is used in NGPON2 and 10GEPON passive optical network applications for fast and bounded lock times. The BMR option is used to implement the fast clock-data recovery when the conventional bang-bang phase detector PLLs cannot meet the stringent lock times required by the passive optical network (PON) applications.

Note: DFE auto-calibration is not available when the transceiver is configured for burst mode (BMR).

When BMR mode is selected, LANE_X_CDR_LOCKMODE[1:0] are exposed for CDR mode control. The following table lists the value and the description of the CDR lock mode control bits.

Table 1-1. CDR Lock Mode Values

LANE_X_CDR_LOCKMODE[1:0] Values	Mode
2'b00	Not used
2'b01	High gain[1]
2'b10	Lock to reference
2'b11	Normal Mode[2]

Notes:

1. High Gain mode is used during Preamble/delimiter detection phase to fast phase lock to the incoming RX data. This mode might generate additional clock jitter on the recovered clock. Once the preamble/de-limiter is detected, it is recommended to switch to normal CDR mode to minimize jitter.
2. This is used when payload is received from Burst mode receiver.

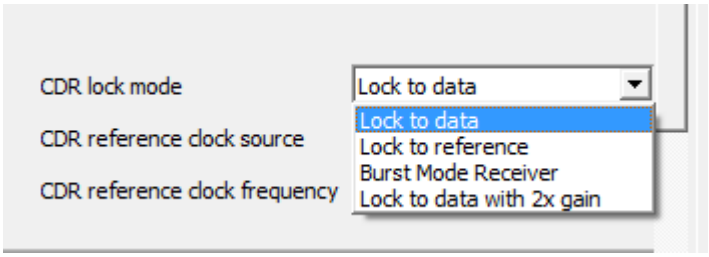
For detailed CDR specifications, see respective PolarFire FPGA Datasheet and PolarFire SoC Datasheet. For application example and implementation details, see PolarFire Burst Mode Receiver Demo Guide.

Lock to Data with 2X Gain: This transceiver option implements the fast clock-data recovery when incoming data streams such as stressful SDI patterns require high gain to quickly phase lock to the incoming pattern. This mode produces faster lock times than that of normal Lock to data mode.

Important: When CDR PLL Lock Mode is set to **Lock to Data with 2X Gain**, users must go to **I/O Editor** > **XCVR View** > **Signal Integrity View** and ensure that the `CDR_GAIN` parameter is not set to "Low".

The following figure shows the CDR Lock mode options.

Figure 1-4. CDR Lock Mode Options



1.1.1.7. Bit Slip (Ask a Question)

The deserializer has a bit-slip feature for word alignment. In this mode, the CDR slips to the next bit from the deserializer. This feature helps with building word-alignment logic in the fabric. It is not used with the built-in 8b10b PCS core but is available for PMA only applications using fabric-based alignment. This feature adjusts the alignment of the deserialized word by 1-bit in either direction when the bit-slip feature is active, reducing the uncertainty by ensuring deterministic latency. This feature is supported by the transceiver configurator. The configurator enables this RX_SLIP input port. This port requests the transceiver CDR lane slip the parallel boundary by 1-bit.

In PMA mode applications, the RX_BIT_SLIP port is exposed on the block for the fabric to access. The RX_BIT_SLIP rising edge requests Rx data path slip relative to the RX_CLK by 1-bit (UI) using handshaking between RX_BIT_SLIP and RX_VAL. The handshake works as follows:

1. Fabric must wait for RX_VAL = 1. Then RX_BIT_SLIP may be asserted to initiate slip.
2. XCVR responds by lowering RX_VAL to 0.
3. Fabric then lowers RX_BIT_SLIP.
4. XCVR completes the slip and the RX_VAL is assigned to 1 synchronously with respect to RX_CLK rising edge.

1.1.1.8. Receive PCS Divider (Ask a Question)

The PCS divider divides the bit-rate clock from the CDR PLL to a lower rate for use in the receive PCS. The PMA sends parallel data from the de-serializer up to 40-bits wide. This divider also sets the width of the parallel data provided to the PCS to 8, 10, 16, 20, 32, 40, 64, or 80 bits. The Libero transceiver configurator sets the divider based on the data rate and ultimate fabric interface width.

1.1.1.9. Receiver Calibration (Ask a Question)

The XCVR receivers include both analog and digital blocks that require calibration to compensate for process, voltage, and temperature (PVT) variations in conjunction with signal integrity. The embedded calibration block of transceiver performs calibration operations that optimize the performance of the transceiver interconnection. It includes an adaptive deserializer calibration algorithm to correct for lossy channels.

Libero selects CTLE only and CTLE/DFE modes based on the system data rate and channel loss needs. The pre-determined settings provide the starting point for the design. These settings are configured to the appropriate calibration requirements based on the targeted requirements. Three types of calibrations are carried out within the Rx:

- CTLE DC-offset

- CTLE Frequency Response
- DFE

1.1.1.9.1. CTLE DC-Offset Calibration (Ask a Question)

Process, voltage, and temperature (PVT) variations result in a DC-offset of the receiver front-end amplifiers, that is, the output is different from zero when the input is zero. This limits the sensitivity of the receiver and therefore the signal-to-noise ratio (SNR). It also limits the performance of the other calibration mechanisms.

The CTLE DC-offset calibration circuitry calibrates the DC-offset by zeroing the input and adding an offset. This offset is dynamically determined by a binary search in response to the logic output of the amplifier.

1.1.1.9.2. CTLE Frequency Response Calibration (Ask a Question)

The CTLE frequency response can be set to a few discrete values, therefore calibration depends on searching for the settings that result in the largest eye area.

CTLE DC_offset and CTLE Frequency Response calibration together make up the CTLE solution. For the most lossy and disruptive channels, many or all CTLE settings combinations can result in a zero eye-opening area. In these scenarios, DFE can sometimes allow for a non-zero area, which would otherwise be impossible with CTLE alone.

1.1.1.9.3. DFE Calibration (Ask a Question)

DFE calibration is carried out by an embedded sequence function, which is optimized to avoid local minima, achieve predictable results, allow for low area, and operate at high clock speeds. It adjusts the feedback coefficients in response to the eye-area. The sequence of the function is used to determine the width, height, and center of the eye opening. DFE Calibration is carried out by a algorithm that adjusts the feedback coefficients (from H1 to H5) by trial-and-error in response to the eye-area of the eye_monitor. The algorithm operates on one dimension (a single coefficient) at a time. It takes a step of size 1 in the positive direction and then the negative direction that is H1+1 and H1-1. If the area improves on either step, it continues to take another step in the same direction. If both directions yield a lower area, it continues to the next coefficient with the same step size. After failing to improve the area on all coefficients, it increases the step size and continue. If the area is improved, the step size immediately reduces to 1. For more information, see PolarFire FPGA Transceiver Decision Feedback Equalization Application Note.

Dependent on the specific design targets chosen through Libero, the design can be configured in one of two modes that require calibration of the receiver.

In CTLE only mode, CTLE solution is executed to optimize gain/frequency settings using CTLE Frequency Response and DC Offset calibration.

In CTLE/DFE mode, CTLE calibration is first run to optimize gain/frequency settings using CTLE Frequency Response and DC Offset calibration. DFE calibration is then run for centering and coefficients.

DFE is optionally programmed to be used in several operations. Full DFE calibration autonomously calibrates to the best found DFE coefficients for optimized data eye centering. These are controlled by Libero specified options.

DFE can also be set in static mode where the user can specify the exact DFE coefficients required by the design. DFE Coefficients are set through PDC commands (see DFE Coefficients) can be used from the register rather than from calibration. This mode does not expose the CALIB_REQ pin or any of the pins to trigger auto-calibration or incremental calibration.

Incremental DFE is another option of calibration to incrementally improve the performance of the DFE path. A specific PF_XCVR_ERM core is generated by Libero which, exposes the required pins to trigger the incremental calibration.

Two algorithms are available for re-calibration:

- Data Eye clock centering Re-calibration
- DFE Coefficient Re-calibration.

Both algorithms do eye-centering, however, they are independent operations. Full calibration and Static calibration are mutually exclusive and Incremental calibration of any of the two algorithms can only be applied after at least one 'Full calibration'.

Users can enable one or both choices in PF_XCVR_ERM configurator depending on the mode of operation and the receiver calibration options selected.

The calibration blocks are used at power-up calibration and user demanded recalibration. These are Libero configured. The following table lists the summary of mode of operations.

Table 1-2. Mode of Operations

CDR Mode (Data Rate ≤ 10312.5 Mbps)	Incrementally Re-calibrate Data Eye	Incrementally Re-calibrate DFE Co-efficients
None_CDR	Not supported	Not supported
On Demand	Supported	Not supported
On Demand and First Lock	Supported	Not supported
None_DFE	Not supported	Not supported

Examples of the specific types of calibration are:

ON_DEMAND

The receiver does not calibrate automatically. The user must initiate an on-demand calibration using either the wires on the XCVR interface or over the DRI. If the specific design performs dynamic reconfiguration using DRI, then the user routine must perform a recalibration each time the XCVR locks to a new data rate and/or data pattern.

In CDR modes where data rate ≤ 10312.5 Mbps, the device performs DC offset calibration of the CDR and the CTLE calibration when the user toggles the CALIB_REQ port. When calibration is completed, the best DC offset and RX CTLE settings are applied to the receiver.

In DFE modes where data rate > 10312.5 Mbps, the device performs the same optimization as in CDR mode with the addition on performing full DFE calibration of the DFE coefficients. When calibration is completed the best DC offset, RX CTLE, and DFE coefficient settings are applied to the receiver.

To successfully complete the RX (CTLE) calibration process, the reference clocks must be stable and free running at device power-up and valid data must be present at the transceiver Rx input buffers. The data should be approximately the actual data that is received but does not need to be any particular data pattern. However, for DFE, calibration can be dependent on the data pattern used at DFE calibration. For example, JESD204B startups with a continuous K28.5 stream, then later shifts to actual 8b10b data. This is a change in data pattern and may impact calibrated DFE coefficients.

The transceiver component is generated by the Libero software to include enhanced receiver management logic to control the proper calibration of the receiver, see Enhanced Receiver Management. The ERM manages calibration providing the user design a streamlined procedure to initiate and monitor calibration from the fabric interface. These calibration modes have higher power than using the NONE selection as the EYE MONITOR circuitry is active during these modes.

Incrementally Recalibrate Data Eye

This is a method to improve the performance of the DFE path after an initial calibration is performed. This recalibration is intended to improve the data eye for most gradients that typically occur due to temperature or voltage changes within the system.

The recalibration is performed by using the DC offset values for the DFE path that were determined by the prior offset calibration as the initial values and then perform the clock phase centering function. The calibration is marked as complete, when the area compute function is completed in the silicon/FPGA. This is indicated by driving the output signal LANE#_DATA_EYE_CALIBRATION_DONE to high.

Incrementally Recalibrate DFE Co-efficients

In this recalibration, the DFE co-efficients are recomputed in an incremental manner when an initial calibration is performed (on-demand or on initial power up).

The recalibration is performed by using the DC-offset values for the DFE path that were determined during a prior offset calibration as the initial value. The prior computed DFE coefficient values (H1-H5) are used as the starting coefficients for the DFE calibration. This reduces the DFE computation time. The calibration is marked as complete when the DFE calibration is completed in the silicon/FPGA. The output signal LANE#_DFE_COEFF_RECALIBRATION_DONE is driven high when the calibration is completed.

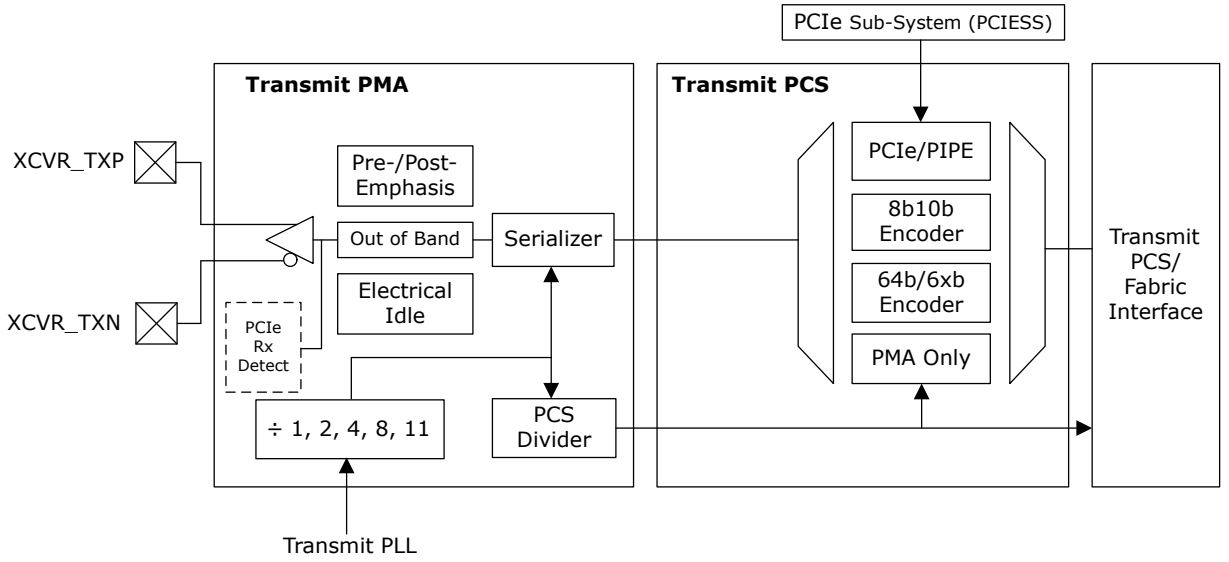
ON_DEMAND_AND_FIRST_LOCK

It is same as On_Demand with the addition of auto calibration. Auto calibration occurs automatically the first time the CDR locks to data. The user can also on-demand initiate a calibration event using wires on the XCVR interface or over the DRI.

1.1.2. Transmitter (Ask a Question)

The transmitter takes parallel data from the FPGA fabric through the PCS-fabric interface block and gearing logic. The data passes through the PMA-PCS interface to the serializer to create a high-speed serial data stream using the serial clock provided from the transmit PLL. The transmitter portion of the PMA includes the transmitter serializer and the transmitter buffer as shown in the following figure.

Figure 1-5. Transceiver Transmitter



1.1.2.1. Serializer (Ask a Question)

The serializer provides the link between the high-speed interface and the transmit PCS by performing a parallel-to-serial conversion. Each lane has up to 40-bit data bus to the transmit PCS block and a separate post-divider for a divide by 1, 2, 4, 8, or 11. The post dividers are provided to divide the high-speed clock from the TxPLL to exactly what the serializer requires for the data rate.

This allows sharing of a high-speed TxPLL by adjusting the local data rate within the transceiver lane. The glitch-free post-divider also allows for dynamic switching between dividers and data rates using the APB DRI.

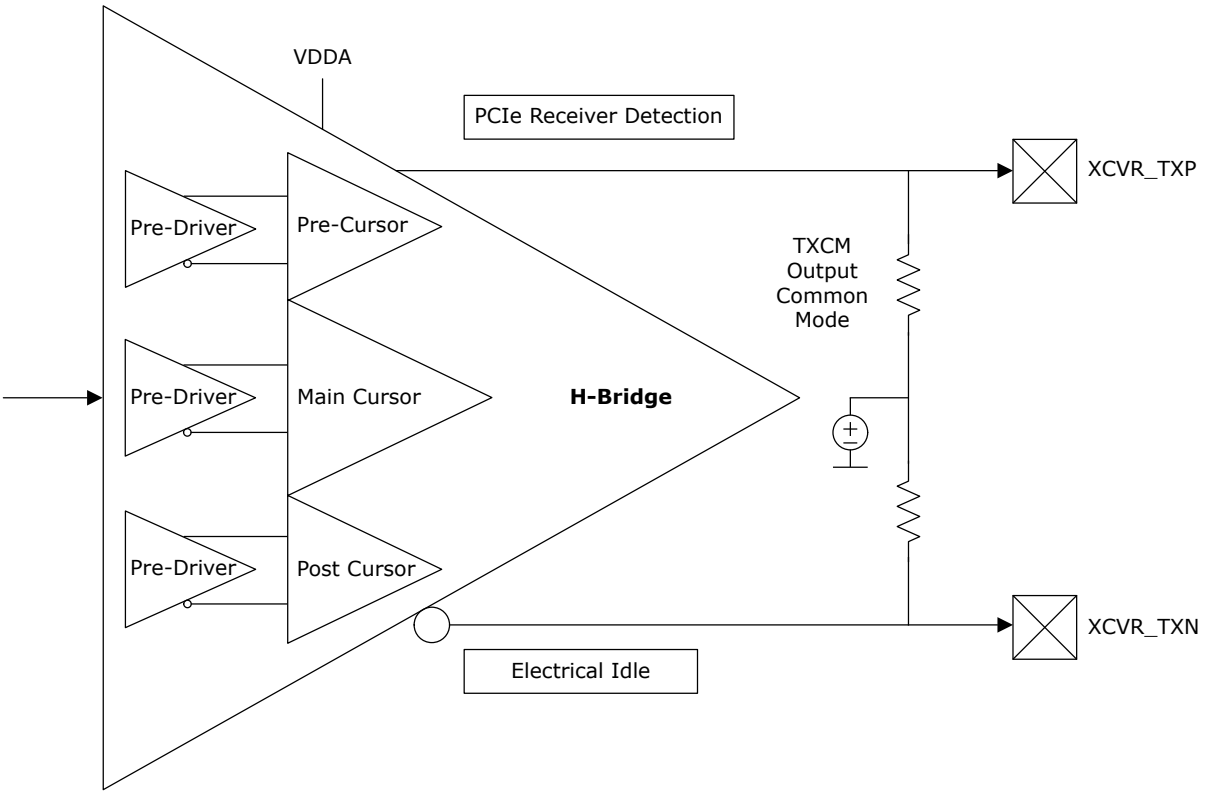
1.1.2.2. Transmit PCS Divider (Ask a Question)

The PCS divider divides the bit-rate clock from the transmit PLL to a lower rate TX_CLK clock for use in the fabric. The PMA receives parallel data in the serializer up to 40 bits wide. This divider also sets the width of the parallel data received from the PCS to 8, 10, 16, 20, 32, 40, 64, and 80 bits. The specific ratio is a function of the parallel-to-serial or serial-to-parallel conversion in the PMA.

1.1.2.3. Transmit Output Buffer (Ask a Question)

The following figure shows the transmit output buffer of the transceiver lane, which connects to the PCB through the XCVR_TXP/N output pins. The low-power H-bridge differential output buffer includes a configurable driver for amplitude on the 100Ω differential load up to a maximum swing of 1V peak to peak. In addition, selectable levels of transmit common-mode voltage (Tx VCM) are available besides the swing amplitude control for the output driver segments to control the amount of emphasis. The transmit output buffer settings are accessible in real-time for adjustment through the JTAG interface using SmartDebug. It includes a receiver detection to recognize the presence of a physical link. The output buffer also has electrical idle capabilities used to orderly quiet the link transmitters.

Figure 1-6. Transmit Output Driver



Transmit common-mode voltage levels for the drivers are selected by the user during the configuration of the PMA when using the transceiver configurator in the Libero software. Reducing the transmit output amplitude lowers the overall transceiver power consumption.

Note: The user can also use a JTAG-based interface from SmartDebug to experiment with transmit settings.

1.2. Enhanced Receiver Management (Ask a Question)

Enhanced Receiver Management (ERM) is implemented in FPGA logic inside the XCVR component. The ERM adds DFE/CDR calibration management, and lock-to-data lock detection capabilities of the XCVR component. The RTL is automatically generated by Libero software. The generated blocks manage the start-up/on-demand CDR/DFE calibration and fine-grain lock detector using PMA, 8b10b, and 64b6xb modes of the XCVR component. The ERM enables the Rx Lock detection logic to handle advanced capabilities of Delayed Traffic and Cable Pull during operation without further interaction from other user logic.

The enhanced receiver management feature provides the following functions.

- Manages receiver lock-to-reference versus lock-to-data operation modes of XCVR component.
- Manages receiver calibration, see Receiver Calibration.
  - Provides optional automatic calibration upon determining first valid receiver bit-lock.
  - Provides optional support for on-demand requested calibration.
  - Provides optional support for Data Eye clock centering recalibration.
  - Provides optional support for DFE Coefficient recalibration.
- Provides indicator when receiver completes calibration.
- Provides LANEx_LOS input, which may be asserted as a means of holding lock management in lock-to-reference. This is useful when interfacing to an optical interface, which provides a loss-of-signal indicator such as SFP.

Enhanced receiver management is recommended to optionally improve the link management for high data rates but can be used to improve link reliability with lower data rates while using the automatic calibration features of ERM. Enhanced receiver management is not included for PCI Express, PIPE, or Burst Mode Receiver solutions of PF_XCVR when generated by Libero software. Generally, the advanced features from ERM are already managed by these protocol layers.

ERM is utilized to augment the Rx_IDLE peak detector logic, which is only valid for a limited minimum density of transitions on the Rx data or high bit rates. Other lower density patterns such as SDI at 270 Mbps or 10G-KR Auto-negotiation data are good examples of data patterns that cannot use the peak detector and thus other system-dependent methods to keep the Rx PLL in lock-to-reference mode in the absence of incoming data are required.

For example, some protocols such as PCI-Express, exchange specific training patterns to establish and tune the respective links. These protocols does not use the ERM as it can interfere with these kinds of negotiation operations. Other applications such as CPRI goes through a sophisticated process by stepping through multiple data rates during its startup. The ERM must not be used in this type of application. ERM must also not be used with Rx-only modes such as DisplayPort where the RxPLL is locked to a lower rate data stream.

ERM can be used with JESD204b, Interlaken, and fixed rate CPRI applications. ERM is not recommended for data rates of 6 Gbps or lower for 8b10b balanced data streams. These data rates are supported natively using the built-in signal detection circuitry not requiring the ERM. Unnecessary use of the ERM for slower data rates can interfere with proper operation of the native clock-data recovery of the transceiver.

The ERM uses minimal logic resources, but does extend the lock time of the CDR by approximately three times versus when the ERM is not included.

The ERM ports are exposed by the transceiver configurator dependent on the selected calibration when enabled (default) in the GUI, see Figure 1-7. If the XCVR module is generated without selecting the ERM option in the Libero SoC configurator, the module will still contain ERM in the component name, however, the ERM functionality is removed. The following table lists the ports required or used in conjunction with the ERM module.

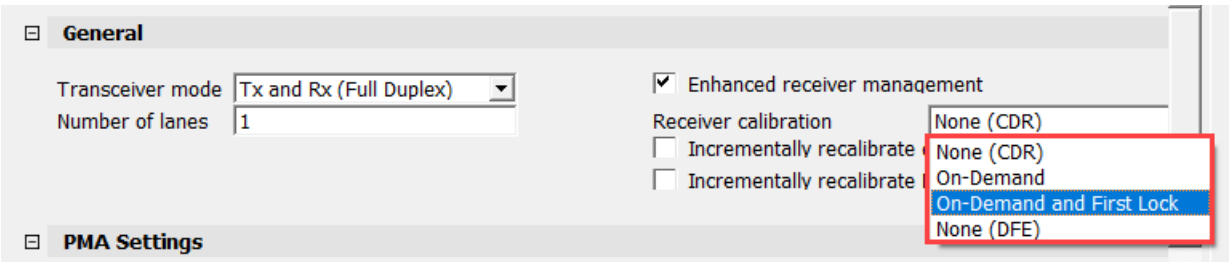
Table 1-3. ERM Ports

Name	Direction	Description
CTRL_CLK	Input	40 MHz clock for the enhanced receiver management logic. The CTRL_CLK input clock can be sourced from the divided output of the PF_OSC (RCOSC_160MHZ_CLK_DIV).
CTRL_ARST_N	Input	Input signal needed to reset ERM. User must drive this input from the XCVR_INIT_DONE signal of the PF_INIT_MONITOR component.
LANE#_LOS	Input	LANEx_LOS input which may be asserted from an external source such as optical SFP during no-signal condition as a means of preventing entry to lock-to-data. This input must be used to control application scenarios where the incoming data stream has enough activity to trigger the LANE#_RX_IDLE but lacks enough transitions to lock the RXPLL. LOS=0 (de-assertion) must occur when valid serial data is applied at the RxP/N inputs of the receiver. LOS=1 assertion must happen prior to the ERM's entry to lock-to-data, otherwise the assertion does nothing.
LANE#_CALIB_REQ	Input	Active-high input signal used to request an On-Demand calibration. LANE#_CALIB_REQ is edge triggered not level. User must clear and re-assert the CALIB_REQ for trigger on-demand calibration request.
LANE#_CALIBRATING	Output	Output signal that will go HIGH to indicate that the DFE/CDR is calibrating.
LANE#_RX_VAL	Output	Indicates CDR fine lock and ERM managed operations complete. See Transceiver PCS Interface Modes for more information. 1 – indicates Fine Lock is asserted, and recovered data is valid. 0 – indicates Fine Lock is de-asserted and recovered data is invalid.
LANE#_RX_IDLE	Output	Indicates activity on the receiver inputs (RXD[P:N]). For ≤ 5 Gbps, active-low (RX_IDLE=0 indicates activity). For >5G bit rates, this signal may not be accurate indicator of data activity. It may toggle or be HIGH although valid signal is applied at the receiver. This signal is exposed for debugging purposes only, for example, detection of signal when RxPLL is in lock2ref mode. See Figure 1-11, Figure 1-12, Figure 1-13, and Figure 1-14.
LANE#_RX_READY	Output	Indicates CDR fine lock completion. See Transceiver PCS Interface Modes for more information. 1 – Fine lock is asserted (that is, RxPLL is locked to incoming data within ± 4000 ppm of the LANE#_TX_CLK_{G,R} frequency). 0 – Fine lock is de-asserted (that is. recovered clock is outside the ± 4000 ppm of the REFCLK frequency).
LANE#_RXD[P:N]	Input	Differential pair of serial data inputs.
LANE#_DATA_EYE_ CALIBRATION	Input	Active-high input signal (Asynchronous signal) to request Data Eye clock centering recalibration.
LANE#_DFE_COEFF_CALIBRATION	Input	Active -high input signal (Asynchronous signal) to request Incremental DFE Coefficient recalibration.
LANE#_DATA_EYE_ CALIBRATION_DONE	Output	Data Eye clock centering recalibration request handshake signal. This signal goes high when the Data Eye clock centering calibration is done.
LANE#_DFE_COEFF_CALIBRATION_DONE	Output	Incremental DFE coefficient recalibration request handshake signal. This signal goes high when the Incremental DFE coefficient recalibration is done.

Figure 1-7. Enhanced Receiver Management in XCVR Configurator

The operation ERM relies on the transceiver configurator settings to determine the selected calibration requirements, as shown in the following figure. The transceiver lane calibration options are selected with the transceiver configurator. See Receiver Calibration for more information.

Figure 1-8. Calibration Options for Enhanced Receiver Management Operations



The following receiver calibration options are provided for the ERM operation:

- **None (CDR)**: Select if the XCVR is configured as CDR and no CTLE auto-calibration is performed. Static settings are configured by Libero based on data rate and backplane model. This receiver calibration mode uses Lock2Reference of the Rx PLL.
- **On-Demand and First Lock**: Select to perform calibration on first lock (after PoR) and on-demand. This option is available for both CDR and DFE configuration of the XCVR. You can trigger calibration on-demand using CALIB_REQ port. The CALIBRATING signal is asserted upon CALIB_REQ assertion and de-asserted when the calibration is completed.
- **On-Demand**: Select to perform calibration on-demand. This option is available for both CDR and DFE configuration of the XCVR. You can trigger calibration on-demand using CALIB_REQ port. The CALIBRATING signal is asserted upon CALIB_REQ assertion and de-asserted when the calibration is completed.
- **None (Static DFE)**: DC Offset Calibration of the CDR is performed, however, the DFE Coefficients are set through PDC commands used from the register rather than from automatic DFE calibration operation. See Physical Constraints. Static_DFE does not use the DFE calibration routine and requires the user to carefully select DFE coefficient values. These values can be gathered by the SmartDebug tool or by simulation.

There are two potential ways to incrementally improve the performance of the DFE path when an initial calibration is completed.

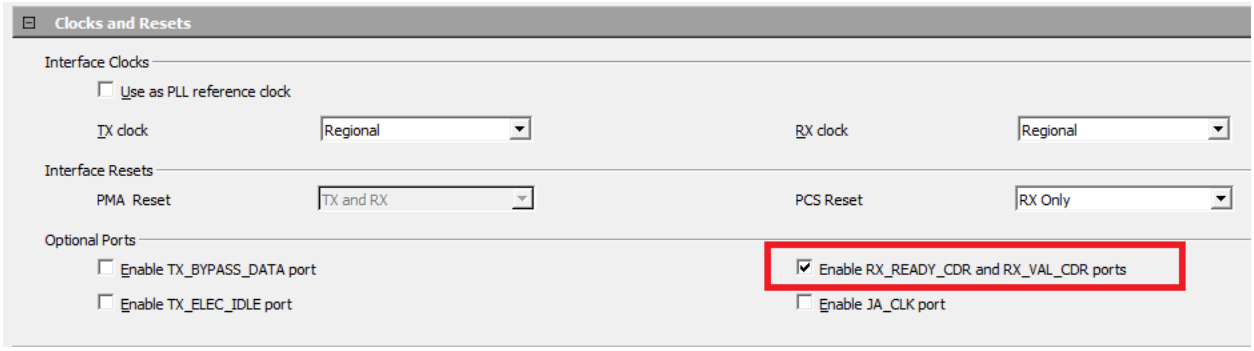
- **Incrementally Recalibrate Data Eye**: This recalibration should improve the data eye for most gradients that typically occur from temperature or voltage changes within the system.
- **Incrementally Recalibrate DFE Coefficient**: This recalibration performs the DFE calibration in incremental method. The initially calculated DFE coefficient values are used as the starting values for this algorithm. This results in the reduction of the Calibration time by reducing the number of DFE coefficients that requires recalibration.

Important: Full calibration is always done for DFE. You must select one of the two options—On-Demand and First Lock or On-Demand—if the transceiver is configured in DFE mode.

Enable LANE#_RX_READY_CDR and LANE#_RX_VAL_CDR ports are optionally exposed by selecting the associated checkbox for the ERM solution. The XCVR component provides optional RX_READY_CDR and RX_VAL_CDR ports for the datapath and variation of the LANE#_RX_READY and LANE#_RX_VAL ports that are always exposed with the ERM. These additional ports are provided to monitor the CDR lock signal. These ports can be used for rate change from >5G to <5G and 10GBASE-KR auto-negotiation support.

XCVR designs implementing protocols with auto-negotiation logic typically negotiates link rates over DRI. In these use cases, the ERM will interfere with the auto-negotiation and can cause incorrect behavior.

Figure 1-9. Exposing RX_READY_CDR and RX_VAL_CDR Pins



The ERM manages the device behavior at first power-up or release from device reset (DEVRSTn). The ERM managed calibration begins after the device complete its initial startup configuration and the assertion of XCVR_INIT_DONE de-asserts the CTRL_ARST_N. This requires a valid serial data stream to be applied to the RXD input pins at the time of de-assertion of the CTRL_ARST_N signal. At that time, the ERM performs the desired user selected calibration operation as shown in the following figure. The LANE#_RX_IDLE activity shows the detection of incoming data moving the ERM to place the CDR into LOCK2DATA operation while it completes calibration and fine lock operation. The operation is indicated by the LANE#_CALIBRATING and RX_READY status signals. The LANE#_RX_VAL with output high upon completion of the calibration and locking routine.

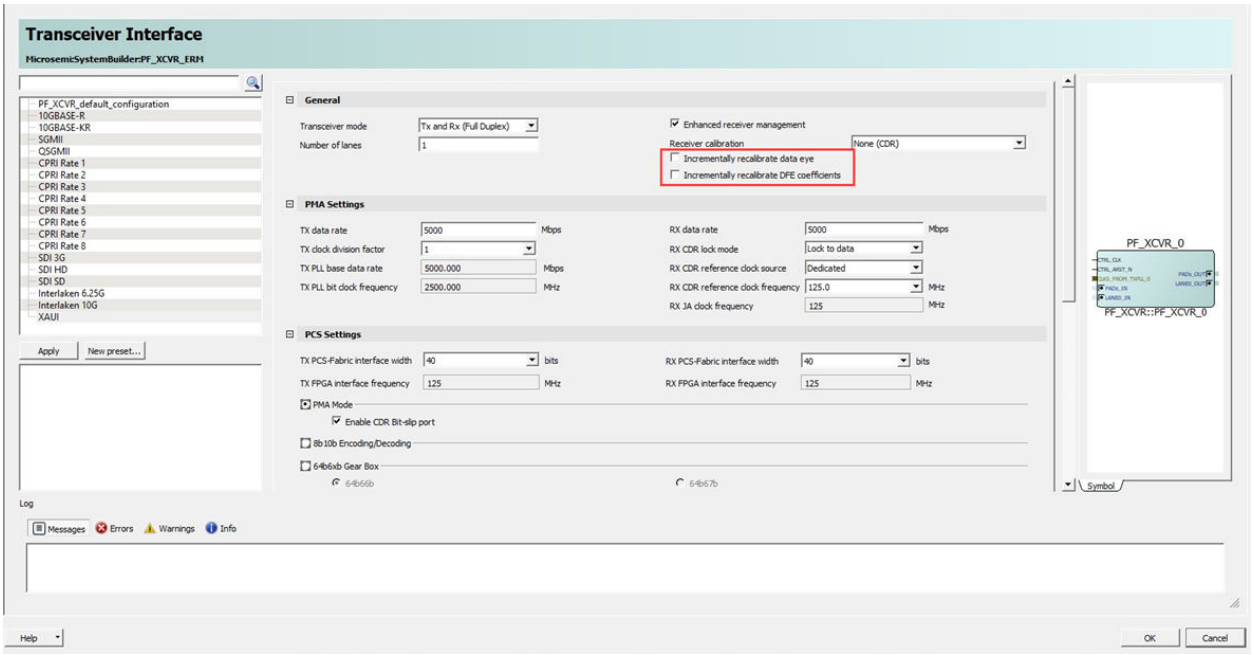
Important: When the ERM logic drives the XCVR lane into Lock-to-Reference mode, the corresponding RX_CLK_R/G is held LOW. In contrast, the XCVR Burst Mode Receiver dynamically switches between Lock-to-Reference and Lock-to-Data modes while continuing to output a clock on RX_CLK_R/G.

The following table lists the ports exposed based on DFE options.

Table 1-4. DFE Options

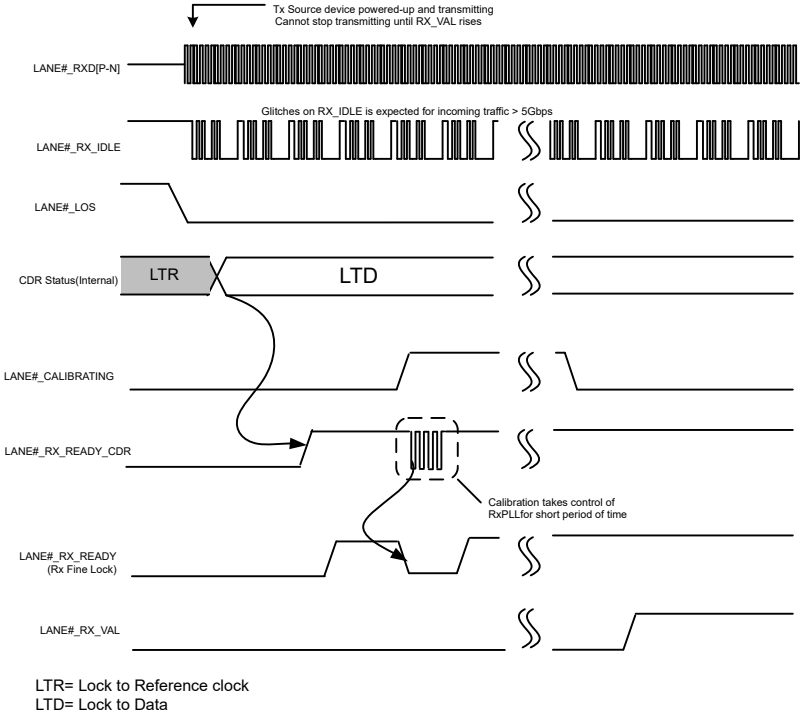
Options	Input Port	Output Port
Incrementally recalibrate data eye	LANE#_DATA_EYE_CALIBRATION	LANE#_DATA_EYE_CALIBRATION_DONE
Incrementally recalibrate DFE coefficients	LANE#_DFE_COEFF_CALIBRATION	LANE#_DFE_COEFF_CALIBRATION_DONE

Figure 1-10. DFE Options



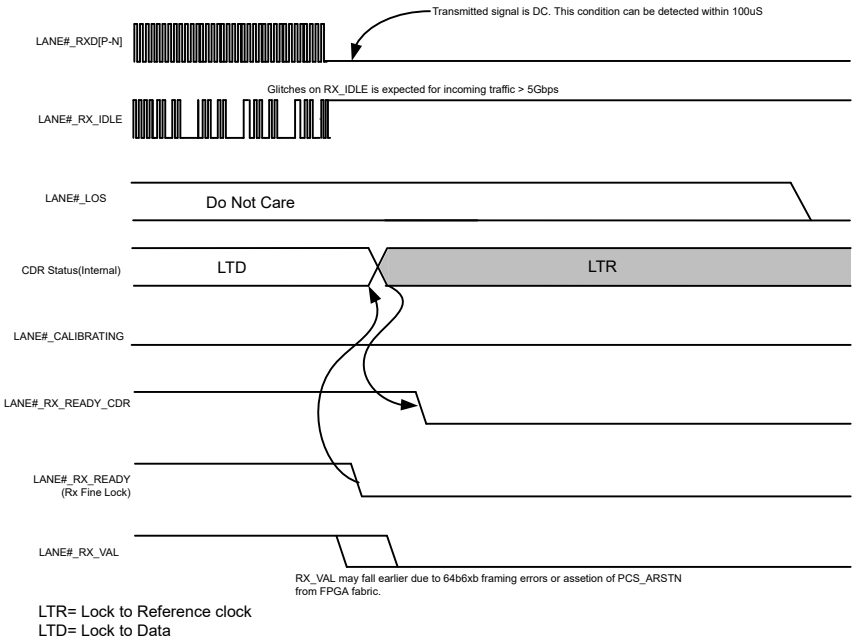
The following waveforms show the behavior of the XCVR with ERM optionally included in the Libero generated component. XCVR configurations without ERM operates with reduced functionality and the LANE#_RX_READY behavior is identical to the LANE#_RX_READY_CDR waveform since the ERM no longer manages the lane.

Figure 1-11. First Lock Calibration Waveform



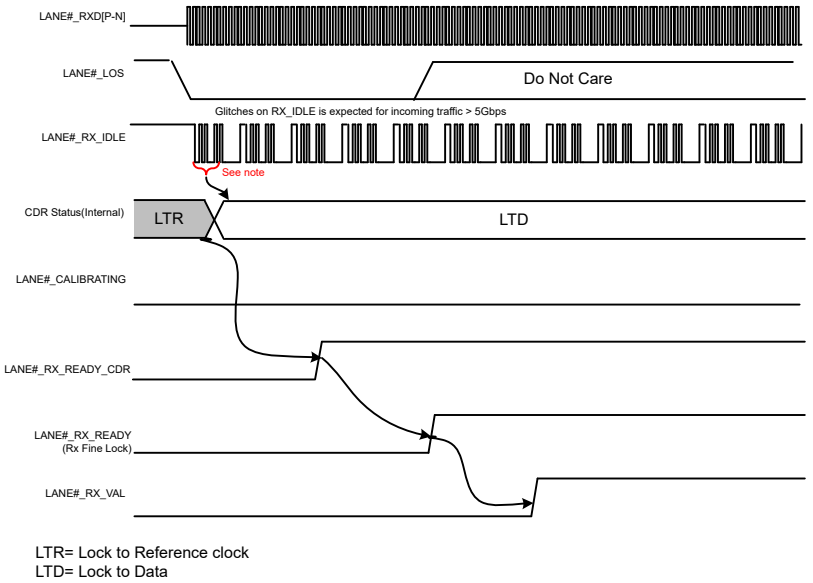
During data transmission if the serial link is disrupted due to lack of activity detected by the Fine Lock LANE#_RX_READY = 0 and toggling of the LANE#_RX_IDLE output signal, the ERM handles a switch over of the CDR from LOCK2DATA to LOCK2REF as shown in the following figure. The ERM manages this mode until the data stream is re-established or the transceiver is placed into reset.

Figure 1-12. Disruption of Serial Rx Data Stream



The ERM manages the re-establishment of a link after a break or disruption. A restart begins when a valid data stream is re-applied to the receiver inputs. This starts the operation of the ERM to systematically control the CDR to re-initialize the link. As shown in the following figure, the ERM properly re-establishes a CDR fine lock after assuring the link is stable. When the link polling is complete, the ERM transitions the transceiver back to proper operation. This is completed without changing any of the CDR or DFE calibration settings and returns calibration setting to those used prior to the disruption or break in the serial data stream.

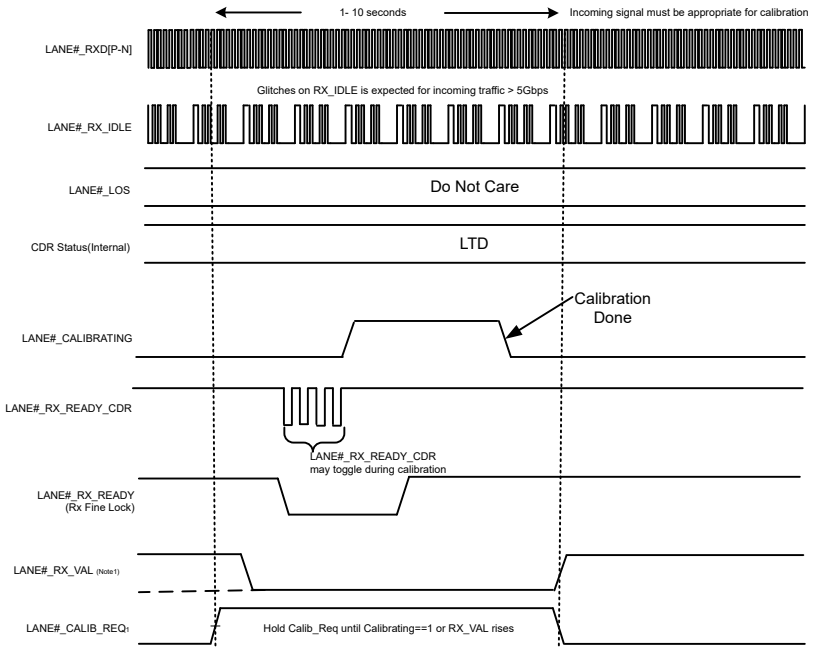
Figure 1-13. Restart after Initialization



Important: RX_IDLE integrator detects threshold for signal-presence.

The user can also optionally invoke a calibration on-demand. This would be needed if the interconnection is changed such as swapping cables or other interface media. In this case, the user can request calibration only after a valid data stream is applied to the RXD inputs. As shown in the following figure, assertion of the LANE#_CALIB_REQ initiates the ERM to invoke a CDR/DFE calibration during which the CDR remains LOCK2DATA. The LANE#_CALIB_REQ is required to stay asserted by the user logic until LANE#_CALIBRATING goes HIGH.

Figure 1-14. On-Demand Calibration Waveform



LTD = Lock to Data

Note 1 - starting state of LANE#_RX_VAL might be 0 or 1

Important: At any time when the Rx PLL is not locked: Once a contiguous Rx data stream of > 25 us occurs, no transmit data gaps of non-standard-compliant data of > 5000 UI are allowed until LANE#_RX_VAL = 1. All non-compliant data gaps of any length must be followed by 5000 UI of compliant data until LANE#_RX_VAL = 1. If a violation to this requirement is seen, receiver calibration may not occur properly and the device may require the DRI bus to command a register access to cycle the PMA_LANE\DES_RSTPD\RXPD = 1'b1 -> 1'b0 in order to continue operating correctly. Assertion of PMA_ARST should occur at approximately the same time as the RXPD = 1 is written through DRI. Then hold RXPD = 1 for 5 us before writing RXPD = 0 followed by deassertion of PMA_ARST. For DRI specific information, see the respective PolarFire Device Register Map or PolarFire SoC Register Map. This scenario can also be controlled by using the LANE#_LOS input to hold off ERM startup until the RX data stream is properly settled.

The ERM can be optionally not included at design creation using the Libero transceiver configurator. Transceiver designs that do not include the ERM have to manage the before mentioned considerations and should only be not used after carefully understanding the system requirements. Without the ERM, the transceiver LANEx_RX_READY pin may toggle when the Rx signal is open or disconnected, or while an out-of-range condition occurs. For example, incorrect Rx serial data rates, with serial input data >1.17% away, is considered out of range.

Initially, the Rx CDR lock may not lock with missing or bad data stream. The following conditions prevent the incorrect behavior.

- The Rx data rate is <±300 ppm of the Libero configured rate.
- Rx data is present when PMA_ARST is de-asserted.
- Data stream must not stop once locked or the CDR Lock circuitry might not properly indicate the status of the CDR.

- PMA_ARST can be used to restart with any data stream disruptions.

1.3. Transceiver PCS Interface Modes (Ask a Question)

The transceiver PMA connects with the fabric using four PCS interface modes. PMA-PCS gearing is used in conjunction with the interface clock. The TX_CLK and RX_CLK frequency is equal to the FPGA interface based on the data rate/(PMA-PCS width × PCS gearing). The PCS interface instantiates the embedded transceiver and RTL blocks when the user customizes and generates the block. These pre-defined protocol interfaces provide data, control, and status signaling to the user logic in the FPGA fabric, including support for the following modes:

- 8b10b: encoding/decoding and word aligner.
- 64b6xb: 64b/66b or 64/67b encoding/decoding with gearbox logic.
- PIPE: a PHY interface for PCI Express (PIPE) supporting PCIe Gen2. Used with the embedded PCIe core or with the soft-IP hosted in the fabric. See PolarFire Family PCI Express User Guide for details about the embedded PCIe core solution. This interface is transparent with the PCIE (PCIESS) core.
- PMA only: direct access to the PMA without any encoding or decoding.

1.3.1. 8b10b (Ask a Question)

The 8b10b mode supports the encoder and decoder only for interface widths of 16, 32, and 64 bits at the PMA.

The following features are supported in the 8b10b:

- Transmit encoding.
- Transmit disparity forcing.
- Transmit disparity adaptation, which operates in two sub-modes—1 Gbps IEEE 802.3 mode and Fiber Channel mode.
- Receiver symbol alignment using CDR slip mechanism.
- Deterministic latency through the 8B10B transmitter and receiver data paths.
- Receiver decoding.
- Electable fabric width (on transceiver level) of two, four, or eight octets per clock beat.

The 8b10b trans-coder is protocol independent, in other words, it does not include a protocol-specific word aligner or word alignment state machine. Comma-detection is supported in this mode. The serial data must be aligned to comma-alignment boundaries before being used as parallel data. Without proper alignment, the incoming 8b10b data does not decode correctly. The comma character (K28.5) is usually used for alignment purposes as its 10-bit code is guaranteed not to occur elsewhere in the encoded bit stream.

1.3.1.1. Word Alignment (Byte Boundary or Comma Detect) (Ask a Question)

The 8b10b PCS block performs the comma code-word detection and alignment operation. The comma character is used by the receive logic to align the incoming data stream into 10-bit words. The alignment comma descriptions (K28.1, K28.5, and K28.7) are defined in section 36.2.4.9 of the IEEE 802.3.2002.

A comma is identified when there is a match across any eight consecutive bits to {00111110} or {11000001} patterns. The only legal 10b characters, which contain series of bits are K28.1, K28.5, and K28.7. In 802.3 specification definition, there is no occurrence of two legal 10b characters sent in a sequence containing the comma pattern, which drastically reduces the chance that a symbol aligner can falsely lock. Alignment status per lane is indicated by the LANE#_RX_VAL output pin going to high only after the PMA CDR locks onto an incoming data stream.

Word Aligner can lock onto an incorrect alignment causing disparity errors and/or code violations from the 8b10b decoder. In this case, the word aligner needs to be reset to find a new

alignment. This can be done by using the PCS_ARST_N reset. The fabric logic needs to monitor the LANE#_RX_CODE_VIOLATION and LANE#_RX_DISPERROR to determine when to issue a PCS_ARST_N and find a new alignment.

Note: Every serial protocol has a specification on how to use disparity errors and code violations to reset the word aligner. In the XCVR 8b10b mode without a soft IP, the user must implement some type of monitoring scheme to identify false alignments.

1.3.1.2. 8b10b Data Path Interface (Ask a Question)

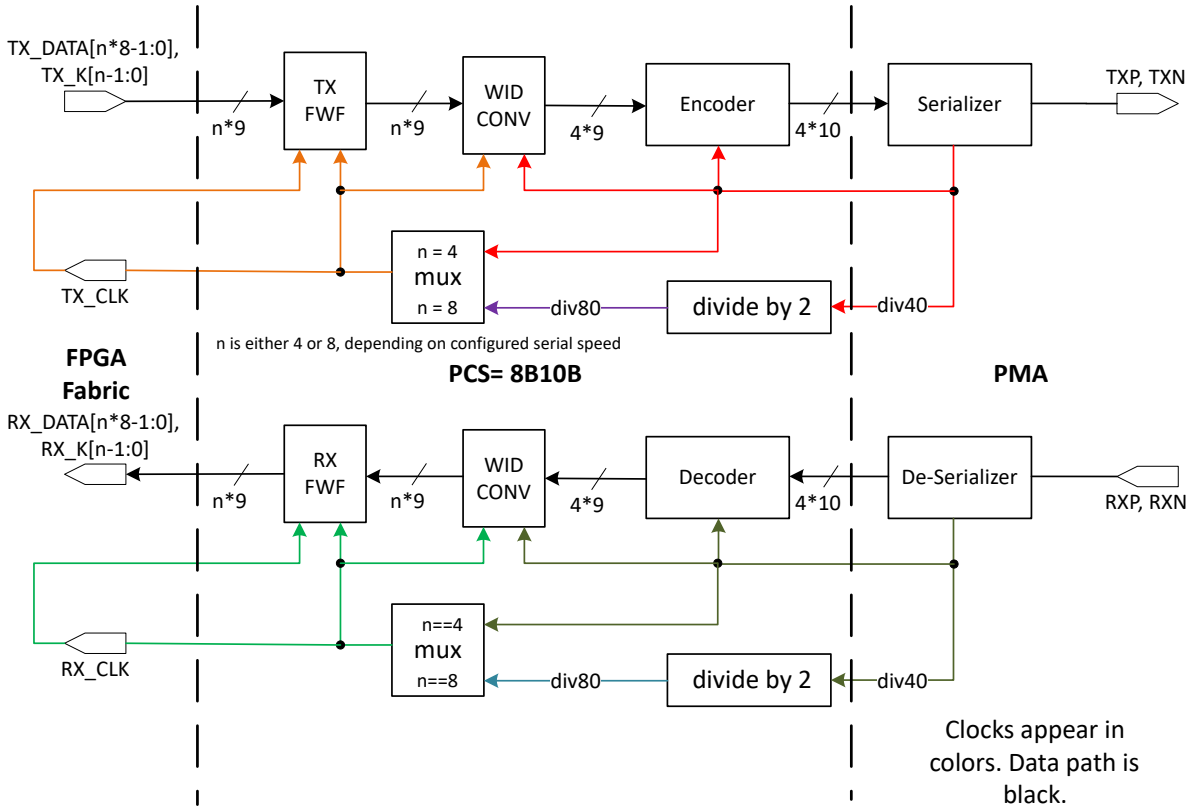
An 8b10b lane data path has the following interfaces:

- Fabric interface – a data path interface with soft-logic
- Internal clocks and resets interface
- PMA interface
  - Parallel transmit data to the serializer
  - Parallel receive data from the de-serializer
- Tx and Rx Fly-wheel FIFO (FWF)
- System registers interface – controlling modes and options

The following figure shows an overview of the 8b10b data path within the 8b10b lane. The figure does not show specific 8b10b functional blocks (that is, encoder, aligner, decoder, and so on). The diagram is intended to show the relative data and clock paths from the serial to fabric interface and vice-versa.

The fabric TX_DATA and RX_DATA ports are allocated to pin functions as described in the Table 1-6. In addition to the fabric data pins, there are additional signals described in 8b10b port list (Table 1-6). on the fabric interface for 8B10B mode.

Figure 1-15. 8b10b Data Path



1.3.1.3. 8b10b Bit and Octet Sequencing (Ask a Question)

Fabric octet bit order is defined in similar way on the Tx and Rx interfaces. In 8b10b notation, each bit of decoded 8-bit octet is assigned a capital letter name of the set A to H. The fabric bus definition is tx_data[7:0] == {H,G,F,E,D,C,B,A}. Thus for the common Dx.y (and Kx.y) character notation, x = tx_data[4:0] and y = tx_data[7:5].

The fabric octet sequence is little-endian. In other words, the first octet to be encoded per fabric clock is tx_data[7:0]. The first octet at the receiver serial input stream per fabric clock is rx_data[7:0].

The data sent out by the 8b10b lane to the PMA follows the standard convention for the XCVR module.

- Bit 0 is the first bit to be output by the serializer for the 40-bit word, pma_txdata[39:0].
- Bit 0 is the first bit that is seen at the serial inputs by the deserializer in the SerDes for the 40-bit word, pma_rxdata[39:0].

The 8b10b 10-bit code-group notation where lower-case letters a to j represent bits of the encoded character relate to the tx_data[9:0] = {j,h,g,f,i,e,d,c,b,a}. The 'a' bit of each code-group is the first bit transmitted and/or received as per 8b10b protocol standards.

Depending on configuration, the 8b10b transmitter allows two, four, or eight octets to be transferred from the fabric per clock beat. This variable transfer rate implies the internal PCS clock domain is either: (a) two times the frequency of pcs_tx_clk; (b) the same frequency as pcs_tx_clk; or (c) half the frequency of pcs_tx_clk. The domains are always synchronous even when they are at different frequencies.

1.3.1.4. 8b10b System Registers (Ask a Question)

There are specific registers used for configuring the 8b10b lane function options in the respective PolarFire Device Register Map or PolarFire SoC Register Map. Other fields are required to properly program the clocks, resets, XCVR, and lane overlay blocks and data path steering. Required system register field setting combinations required for enabling 8b10b lane usage.

Table 1-5. System Registers Affecting 8b10b Data Path

Register Page xls	Register Name	Field Name	Description	Required Value
pcslane	L8_R0	L8_TXENCSWAPSEL	Selects between 1000BASE-X/T and Fibre Channel octet-swapping modes.	Optional: 0=1000BASE-X/T, 1=Fibre Channel
		L8_GEARMODE[1:0]	Sets data path width of FWF interfaces.	Must be consistent with clock selections for txfwf_rclk and rxfwf_wclk.
	LOVR_R0	FAB_IFC_MODE[3:0]	Selects path through fabric and FWF overlay blocks.	Must be set to the 8b10b value (3'd4).
		PCSPMA_IFC_MODE[3:0]	Selects lane mode for driving data into the SerDes serializer.	
	LCLK_R0	LCLK_EPCS_RX_CLK_SEL [1:0]	Chooses which clock is sent to fabric on epcs_rx_clk port.	Usually this must be set to 2'd1 so that the frequency of the fabric is the same as the internal side of the FWF. However variations are possible if the use of the Rx FWF synchronous enable will be employed. See FWF description for further information.
pcslane	LCLK_R0	LCLK_EPCS_TX_CLK_SEL [1:0]	Chooses which clock is sent to fabric on epcs_tx_clk port.	
		LCLK_PCS_RX_CLK_SEL [1:0]	Defines clock module's source for pcs_rx_clk.	Must be set to 2'd3 for all applications using 8B10B function.
		LCLK_PCS_TX_CLK_SEL [1:0]	Defines clock module's source for pcs_rx_clk.	
		LCLK_RXFWF_WCLK_SEL [1:0]	Defines clock module's source for rxfwf_wclk.	Must be consistent with L8_GEARMODE setting.
		LCLK_TXFWF_RCLK_SEL [1:0]	Defines clock module's source for txfwf_rclk.	
		LCLK_RXFWF_WCLK_PIPE	Defines whether Rx FWF is clocked by Tx side clocks or Rx side clocks.	Must be set to 1'd0 for 8B10B functionality.
	LCLK_R1	LCLK_ENA_8B10B_RX_CLK	Instructs clock module to drive 8B10B pcs_rx_clk.	Must be set to 1'd1 for 8B10B operation.
		LCLK_ENA_8B10B_RXFWF_WCLK	Instructs clock module to drive 8B10B rxfwf_wclk.	
		LCLK_ENA_8B10B_TX_CLK	Instructs clock module to drive 8B10B pcs_tx_clk.	
		LCLK_ENA_8B10B_TXFWF_WCLK	Instructs clock module to drive 8B10B txfwf_rclk.	

Table 1-5. System Registers Affecting 8b10b Data Path (continued)

Register Page xls	Register Name	Field Name	Description	Required Value
pma_lane	DES_CLK_CTRL	DESMODE[2:0]	Selects parallel bus width of deserializer interface.	Must select the 40-bit wide bus mode for 8b10b functionality (3'd7).
	SER_CLK_CTRL	SERMODE[2:0]	Selects parallel bus width of serializer interface.	
	DES_CDR_CTRL3	SLIP_DES_CDR_SEL	Selects source of CDR slip control.	Must be 1'd0 so that the fabric can control the symbol alignment.
		SLIP_DES_CDR_EN	Optionally turns slip control off.	Must be set to 1'd1.

The following table lists the port names and description for the 8b10b mode of PCS module.

Table 1-6. 8b10b Port List

Port Name	Direction	Clock	Description
LANE#_CDR_REF_CLK_#/ LANE#_CDR_REF_CLK_FAB	Input	—	Reference clock to lane CDR. Can be sourced from either an FPGA clock or from a XCVR_#[A,B,C]REFCLK_P/N pin.
LANE#_REF_CLK	Input	—	This port is exposed to user with Half-Duplex option. LANE#_REF_CLK must be connected by the user to a stable clock with same clock frequency as Recovered clock such as the local clock. **Note:** LANE#_REF_CLK can be provided by a separate PLL or CCC, however the LANE#_REF_CLK frequency must be within ± 300 ppm with respect to LANE#_RX_CLK_R.
LANE#_TX_PLL_REF_CLK_#	Input	—	Input clock from TX_PLL REF_CLK_TO_LANE output pin. Included in CLKS_FROM_TXPLL_# BIF (bus interface).
LANE#_TX_BIT_CLK_0	Input	—	Clock from BIT_CLK of the XCVR TxPLL. Included in CLKS_FROM_TXPLL_# BIF (bus interface).
LANE#_TX_PLL_LOCK_#	Input	—	Input lock status from TX_PLL LOCK output pin. Included in CLKS_FROM_TXPLL_# BIF (bus interface).
LANE#_TX_DISPFNC[N:0][1]	Input	TX_CLK_[R:G]	The TX_DISPFNC is a 2-bit encoded setting per octet where bit[1:0] is the lowest octet. The TX_DISPFNC port size is 4-bit, 8-bit, or 16-bit respective to 16-bit, 32-bit, or 64-bit PCS-Fabric interface widths. The TX_DISPFNC encoding is as follows for each octet per IEEE specification Clause 36 (802.3). The octet swap feature is designed such that the fabric marks the swap indicator on any octet of the interface. It is not necessary to align the K28.5 to octet 0 or octet 2. None - 2'b00 - Normally encode the octet with the encoder's current running disparity. Swap - 2'b01 - Search for tx_dispfnc = 1, swap next octet when running disparity prior to ordered set is '+'. ForcePlus - 2'b11 - Replace running disparity from encoder with '+' when encoding associated octet. This tx_dispfnc occurs on any octet. ForceMinus - 2'b10 - Replace running disparity from encoder with '–' when encoding associated octet. This tx_dispfnc occurs on any octet.
LANE#_8B10B_TX_K[N:0][2]	Input	TX_CLK_[R:G]	Active-high signal indicating that TX_DATA contains k-character information. This indicates that the input is a k-character byte, not a data byte.
LANE#_TX_DATA[N:0]	Input	TX_CLK_[R:G]	Encoded user data from the fabric. The send/receive order is low to high byte.

Table 1-6. 8b10b Port List (continued)

Port Name	Direction	Clock	Description
LANE#_PCS_ARST_N	Input	—	Asynchronous active-low reset for the PCS lane. This reset is responsible for the reset of the 8b10b logic and COMMA word aligner. The RX_SLIP is internally used to align the parallel word on the fabric interface, but does not reset the word aligner.
LANE#_PMA_ARST_N	Input	—	Asynchronous active-low reset for the PMA lane.
LANE#_RXD_N	Input	—	Transceiver receiver differential input.
LANE#_RXD_P	Input	—	Transceiver receiver differential input.
LANE#_8B10B_RX_K[N:0][2]	Output	RX_CLK_[R:G]	Active-high output from the decoder to the receiver indicating that the received data is a K character. The order of character bits within an octet, in 8b10b mode, is least to most-significant bit as defined by 8b10b code notation. [0]: K decoded data on RX_DATA[7:0]. [1]: K decoded data on RX_DATA[15:8].
LANE#_RX_DISPARITY_ERROR[N:0][2]	Output	RX_CLK_[R:G]	Active-high output indicates when the received code group exists in the 8b10b decoding table but is not found in the proper column according to the current running disparity.
LANE#_RX_CODE_VIOLATION[N:0][2]	Output	RX_CLK_[R:G]	Active-high signal indicating that the decoder has detected an error in the received data.
LANE#_RX_DATA[N:0][3]	Output	RX_CLK_[R:G]	Decoded user data to fabric. The send/receive data order is low to high byte meaning the octet order is least to most-significant. Data[7:0] = First octet Data[15:8] = Second octet
LANE#_RX_VAL	Output	Synchronous	LANE#_RX_VAL indicates that the XCVR data path is initialized. The parallel bus of LANE#_RX_DATA[N:0] contains actual data recovered from the serial stream when LANE#_RX_VAL = 1. In 8b10b mode, the Rx PCS logic self-resets when the CDR is not locked. In this mode, LANE#_RX_VAL rises just after LANE#_RX_READY rises. Always pulse LANE#_PCS_ARST_N=0 using a clock independent from the LANE#_RX_CLK_R transceiver output. Asserting and holding LANE#_PCS_ARST_N=0 can prevent LANE#_PCS_RX_READY from rising and hold LANE#_RX_CLK_R at a static level. In 8b10b mode, the LANE#_RX_VAL is qualified when XCVR receiver calibration completes, included with the enhanced receiver management completion, and the CDR locks and the initial comma/word alignment occurs.
LANE#_RX_READY	Output	Asynchronous	Rises when the enhanced receiver management and CDR completes a fine lock detection to the incoming data transitions and the de-serializer is powered-up. If there is no incoming data to the CDR then the RX_READY is low. The primary purpose of this pin is to let the fabric know the CDR is locked to serial input data and is producing valid clocking. Note: In a loopback case where looping the local transmitter output to the receiver input, it is necessary to take the Tx out of reset to ensure valid serial transitions, allowing the Rx CDR to lock. If the user waits till Rx CDR locks before releasing Tx from reset, in such a case, the system deadlocks.
LANE#_RX_IDLE	Output	Asynchronous	Receive Electrical Idle (EI) detection flag. Flag is 1 when EI is valid. LANE#_Rx_IDLE peak detector logic is only valid for a limited minimum density of transitions on the Rx data and not to be used in applications above 5Gbps.
LANE#_TX_CLK_STABLE	Output	—	Transmit transceiver/PCS lane ready flag. This flag is 1 when the transmit PLL is locked to the reference clock.
LANE#_RX_CLK_[R:G]	Output	—	Global or regional receive clock to the fabric.[4]
LANE#_TX_CLK_[R:G][5]	Output	—	Global or regional transmit clock to the fabric.[4]

Table 1-6. 8b10b Port List (continued)

Port Name	Direction	Clock	Description
LANE#_TXD_N	Output	—	Transceiver transmitter differential output.
LANE#_TXD_P	Output	—	Transceiver transmitter differential output.

Notes:

1. N can be 1, 3, 7, and 15.
2. N can be 1, 3, and 7.
3. N can be 15, 31, and 63.
4. [R:G] naming is generated based on the use of regional or global resources that are selected with Libero. LANE# can be 0, 1, 2, and 3.
5. In PolarFire FPGA MPF500 devices, RT PolarFire RTPF500 devices, and PolarFire SoC FPGA devices, the TX_CLK_R and RX_CLK_R pins of XCVR lanes placed in the PCIESS (Q0) and GPSS1 (Q1) quads cannot drive I/Os.

1.3.2. 64b66b/64b67b (Ask a Question)

The 64b66b/64b67b (64b6xb) interface modes are used mainly for 10 Gbps-based protocols, 10G base interface over Ethernet (10GBASE-R/KR), common public radio interface (CPRI) rates of 9.830 Gbps, and 40GBASE-R standards. The 64b/66b encoder is used to achieve DC balance and sufficient data transitions for clock recovery. It encodes 64-bit XGMII data and 8-bit XGMII control into 10GBASE-R
66-bit control or data blocks in accordance with Clause 49 of the IEEE802.3-2008 specification.

The following features are supported in the 64b6xb:

- Fabric width is selectable. Four and eight byte widths are available.
- Gearbox functions are set for 67-bit or 66-bit block size. The gearbox functions can be bypassed.
- Optional scrambler and descrambler 64b66b data.
- Optional test pattern generate/compare mode.
- Optional PRBS generate/compare mode.
- Optional disparity generate/check (applies to 64b67b data).
- Receiver block lock state-machine controlling the hunt for synchronization header boundaries is available in two forms.
  - The IEEE 802.3 Clause 49 state-machine loses lock when 16 invalid headers are observed within a contiguous set of 64 headers.
  - The IEEE 802.3 Clause 82 state-machine loses lock when 65 invalid headers are observed within a contiguous set of 1024 headers.
    Both block lock state-machines require an initial set of 64 contiguous valid headers to gain block lock. The Clause 82 state-machine is specified for use with 40 G and 100 G links. One of these state-machines must be enabled in order for the receiver to locate and lock onto the block boundaries.
- Optional receiver IEEE 802.3 bit error rate monitor function.
- Optional data path delay status monitors for transmit and receive.

Note: Forward Error Correction (FEC) option of IEEE 802.3 10GBASE-KR is not supported by the 64b6xb PCS.

The encoder uses per-lane block interfaces with a fabric interface to receive and transmit encoded data and a PMA interface to send parallel data to the transceiver. This mode also supports the Interlaken protocol by providing 64b67b with optional embedded gearing logic.

1.3.2.1. 64b6xb Data Path Interface (Ask a Question)

An 64b6xb lane data path has the following interfaces:

- Fabric interface – a data path interface with soft-logic
- Internal clocks and resets interface
- PMA interface
  - Parallel transmit data to the serializer
  - Parallel receive data from the de-serializer
- Tx and Rx Fly-wheel FIFO(FWF)
- 64b6xb Transmit Data Path Blocks, Fabric to PMA Order
- 64b6xb Receive Data Path Blocks, PMA to Fabric Order
- System registers interface – controlling modes and options

The following figure shows an overview of the 64b6xb data path within the 64b6xb lane. The diagram is intended to show the relative data and clock paths from the serial to fabric interface and vice-versa.

The fabric TX_DATA and RX_DATA ports are allocated to pin functions as described in 64B6xB Port List Table 1-10. In addition to the fabric data pins, there are additional signals described in 64B6xB portlist (Table 1-10) on the fabric interface for 64B6xB mode.

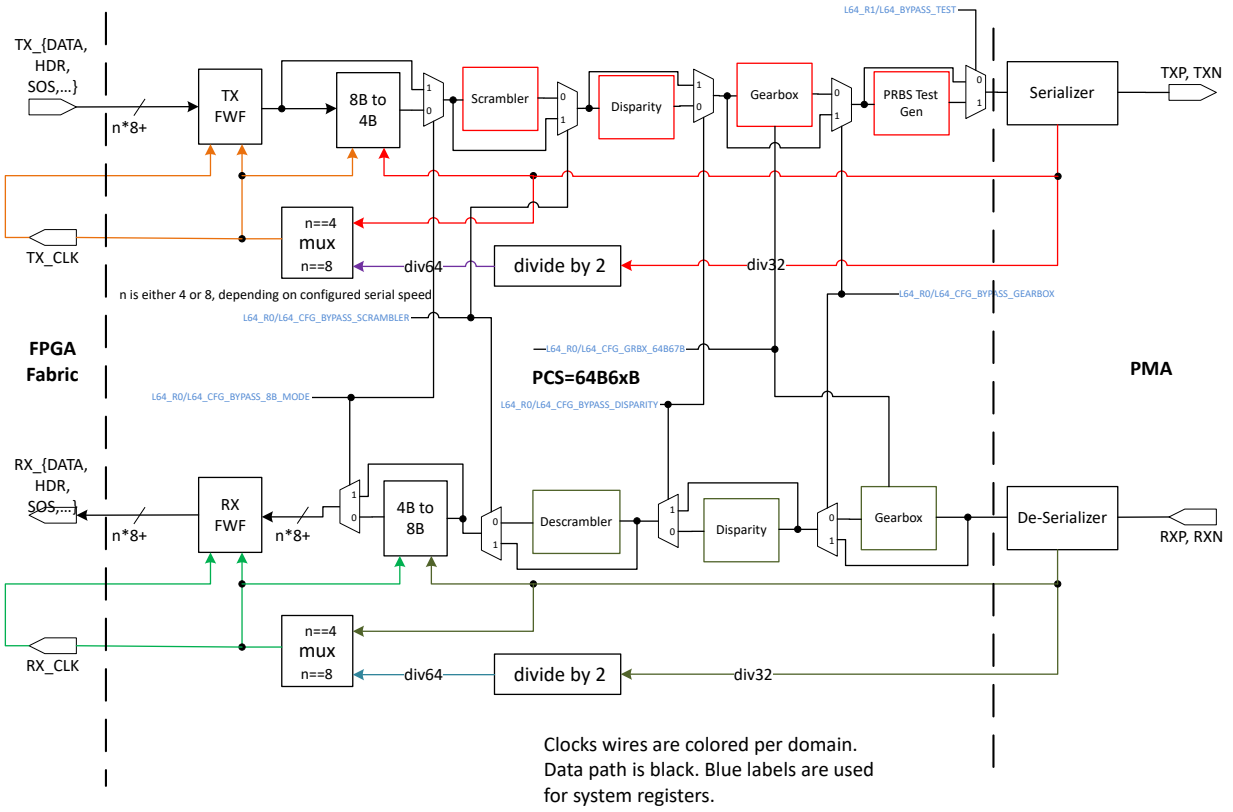
Table 1-7. 64b6xb Transmit Data Path Blocks, Fabric to PMA Order

Tx Block	Purpose
8B_to_4B	Optionally compresses transmit bus width from 64-bits to 32-bits.
Tx Scrambler	Implements IEEE 802.3 Clause 49 data scrambling. Same scrambling also specified for use in Interlaken.
Tx Disparity	Implements inversion of 67-bit symbol. Only for use in Interlaken.
Tx Gearbox	Outputs continuous stream of 32-bits per clock beat to the PMA given an input consisting of 64-bit block symbols, which has cyclic gaps in its active clock beats. This block inserts the 2 or 3 bits of header on each block.
Tx Test Gen	Optionally replaces the output stream with PRBS or square-wave pattern (1111_0000).

Table 1-8. 64b6xb Receive Data Path Blocks, PMA to Fabric Order

Rx Block	Purpose
Rx Gearbox	Locates block boundaries (configurable to 66-bit or 67-bit) in continuous stream of 32-bits data per clock beat from PMA. Removes the 2 or 3 bit headers and plays out the 64-bit data blocks in cyclic pattern of 32-bit clock beats with some inactive cycles.
Rx Disparity	Inverts block based on header bit used in Interlaken.
Rx De-scrambler	Implements IEEE 802.3 Clause 49 data de-scrambling. Same de-scrambling also specified for use in Interlaken.
Rx 4B_to_8B	Optionally widens the data path to 64-bits per clock beat.

Figure 1-16. 64b6xb Data Path



1.3.2.2. 64b6xb System Registers (Ask a Question)

There are specific registers used for configuring the 64b6xb lane function options in the respective PolarFire Device Register Map or PolarFire SoC Register Map. Other fields are required to properly program the clocks, resets, XCVR, and lane overlay blocks and data path steering. System register field setting combinations is required for enabling 64b6xb lane usage.

Table 1-9. System Registers Affecting 64b6xb Data Path

Register Page xls	Register Name	Field Name	Description
pcslane	L64_R0	L64_CFG_BYPASS_GEARBOX	When = 1, both transmit and receive gearbox functions are bypassed.
		L64_CFG_BYPASS_SCRAMBLER	When = 1, both the scrambler and the descrambler are bypassed.
		L64_CFG_BYPASS_DISPARITY	Enable the Interlaken block disparity generation and checking blocks by setting cfg_bypass_disparity=0.
		L64_CFG_BYPASS_8B_MOD	When = 0 to obtain 64-bit fabric interfaces. Set cfg_bypass_8B_mode=1 for 32-bit fabric interfaces.
		L64_CFG_GRBX_64B67B	When = 1 for 67-bit blocks (Interlaken).
		L64_CFG_GRBX_SM_C49	When = 1 to enable IEEE 802.3 Clause 49 receiver lock state machine. Value of L64_CFG_GRBX_SM_C82 must be set as inverse of this bit.
		L64_CFG_GRBX_SM_C82	When = 1 to enable IEEE 802.2 Clause 82 receiver lock state machine behavior (40G link version). Value of L64_CFG_GRBX_SM_C49 must be set as inverse of this bit.
		L64_CFG_BER_MON_EN.	When = 1 to enable the BER Monitor in the receiver. The BER Monitor is useful in CPRI applications.
		L64_CFG_BER_1US_TIMER_VAL	This register must be set to the number of clock beats which most closely represents one microsecond. This field is used by the BER Monitor to time the 125 microsecond interval. Examples: CPRI rate 7A setting is 0x18B; CPRI rate 8 setting is 0x140; CPRI rate 9 setting is 0x107.

1.3.2.3. 64b66b Receiver (Ask a Question)

The receiver takes 32-bits or 64-bits of data from the PMA's CDR on each clock beat into the gearbox. The gearbox frames the data into 66-bit symbols by searching for valid values on the sync header bits as per IEEE 802.3 Clause 49.

Figure 1-17. 64b66b Receive Sequence For 32-Bit Interface

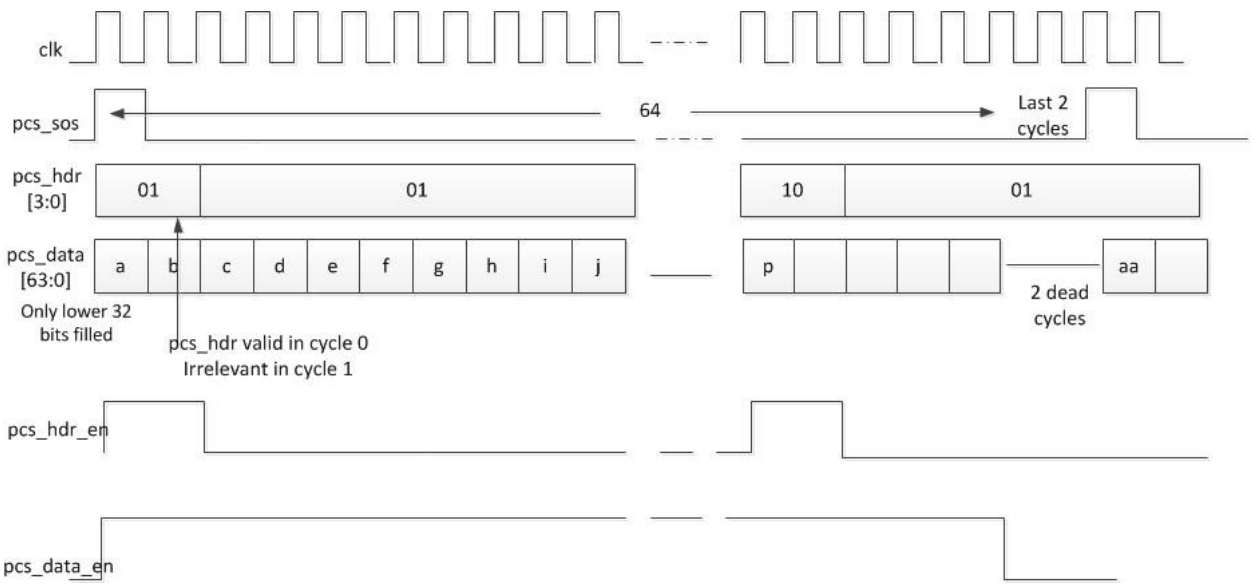
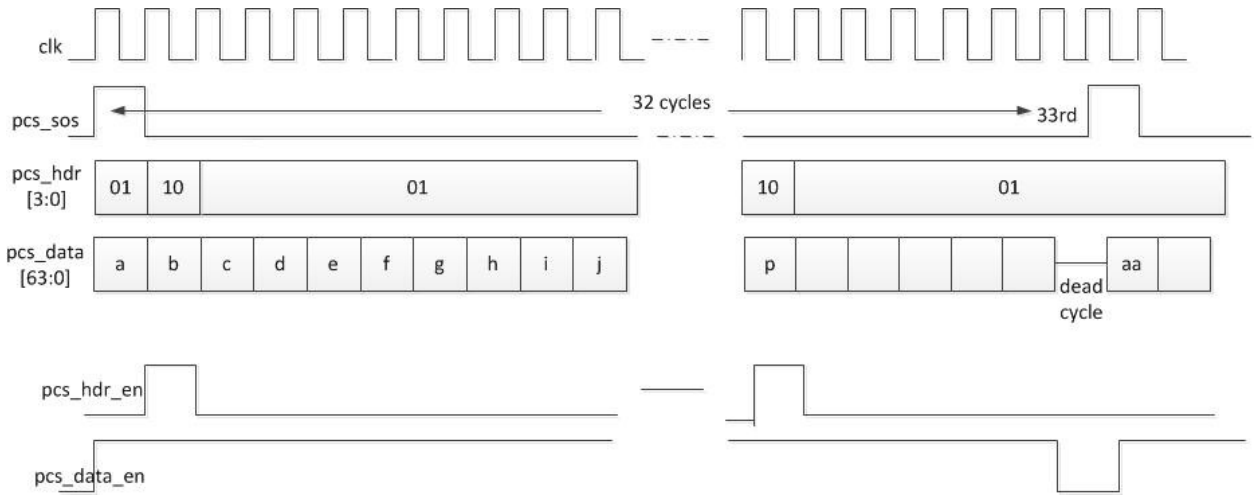


Figure 1-18. 64b66b Receive Sequence For 64-Bit Interface



1.3.2.4. 64b66b Transmit (Ask a Question)

In the transmit direction, the encoded 64-bit blocks are applied to PCS data from the fabric along with the sync headers. When the 64-bit symbol is a control block, like Idle, Start or Terminate, then pcs_hdr[1:0]=0b10. Otherwise, pcs_hdr[1:0]=0b01. The timing signal pcs_sos, indicates the sequence boundaries. The sequence boundaries differ for 32-bit and 64-bit interfacing as shown in the following figures. The configuration includes the scrambling of the data within the 64-bit blocks. Scrambling is done in accordance with IEEE 802.3.

The transmit gearbox produces 32-bits to the PMA on each clock beat where the data is converted to serial form and sent out to the link partner.

Figure 1-19. 64b66b Transmit Sequence For 64-Bit Interface

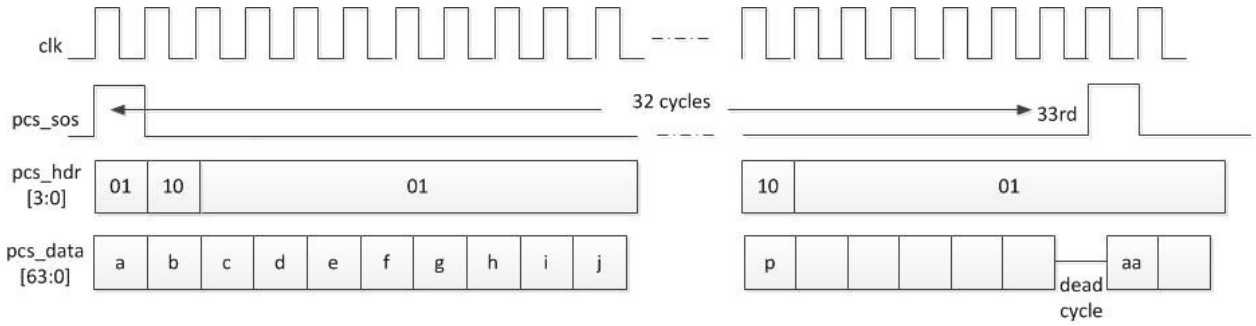
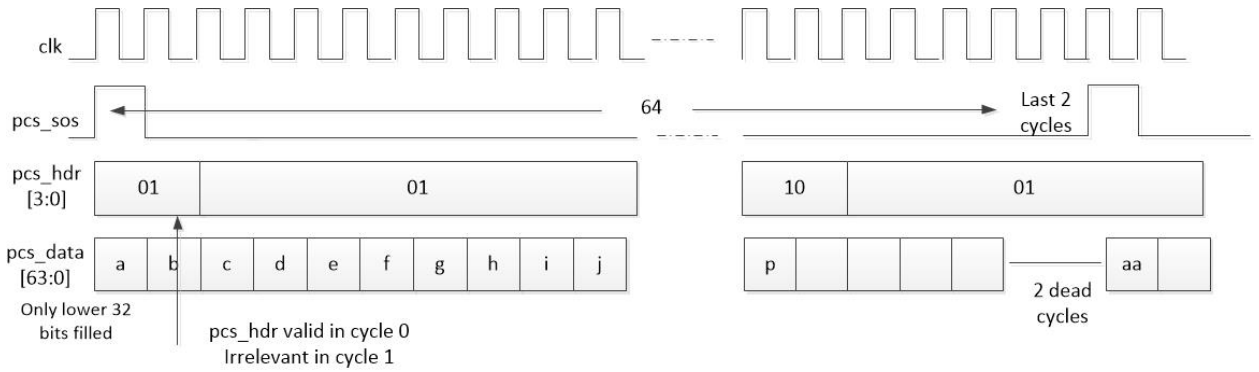


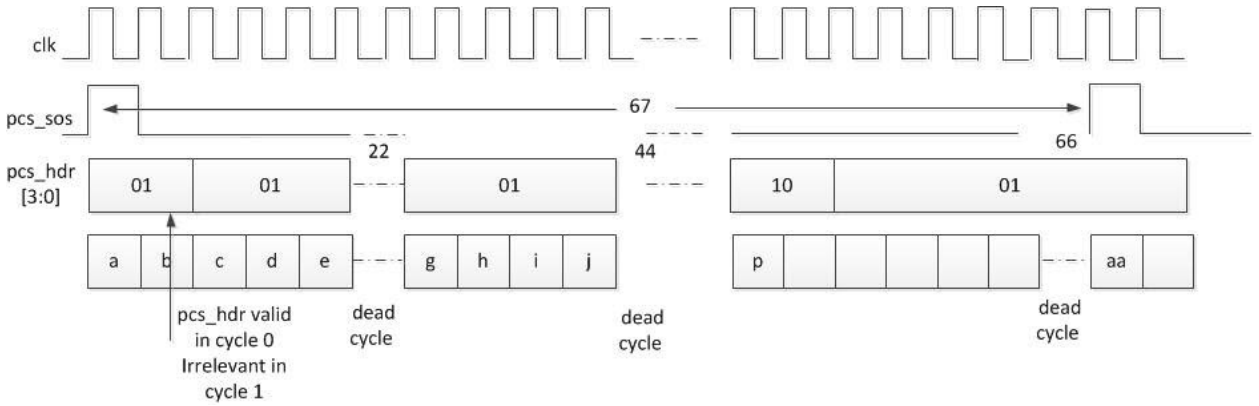
Figure 1-20. 64b66b Transmit Sequence For 32-Bit Interface



1.3.2.5. 64b67b Transmit (Ask a Question)

In the recommended configuration for 64b67b, encoded and scrambled data is presented from the fabric into TX_DATA along with sync headers on TX_HDR[3:0]. The data from the fabric must conform to the expected sequence of clock beats according to the fabric interface width.

Figure 1-21. 64b67b Transmit Sequence For 32-Bit Interface



The 64-bit interface option has dead cycles on clock beats 44, 45, 88, 89, 132, and 133.

The transmit data first enters the PCS Timing control block, which generates the appropriate markers for 64b67b blocks. The data then goes to the disparity calculation block. The total number of ones and zeros are balanced by inverting blocks as necessary. When a block is inverted, sync header bit 2 is set to mark the inversion. Bits [1:0] of the sync header have the same meaning as they do in 64b66b blocks.

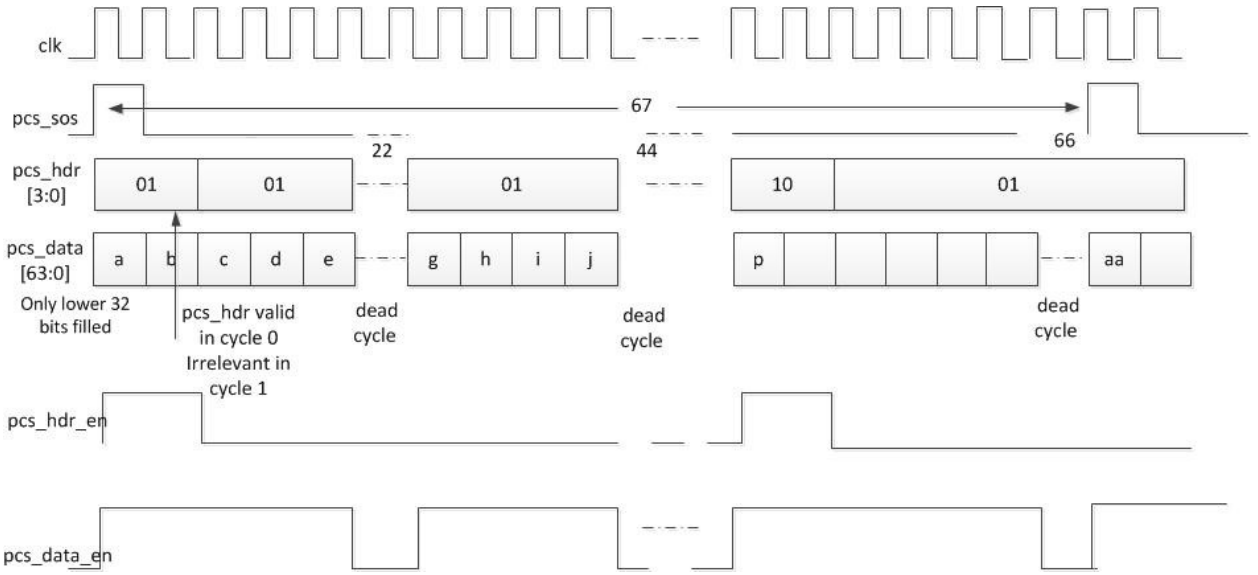
Data from the disparity block then goes to the transmit gearbox where it is output in 32-bit chunks per clock beat and sent to the PMA serializer.

1.3.2.6. 64b67b Receive (Ask a Question)

Data comes into the 64b67b receiver from the PMA 32-bits per clock beat from the de-serializer. The data is qualified by RX_READY=1, which is set by the greater PMA logic when CDR achieves bit lock and is producing a stable clock.

The receive gearbox frames the 67-bit blocks by examining sync header bits [1:0] in the same method as described for 64b66b. The difference is in the span of bits between sync headers. The receive disparity block invert blocks as indicated by sync header bit [2]. The receive data is then sent to the fabric in 64-bit or 32-bit clock beat sequences. The 32-bit sequence is shown in the following figure where two clock beats transfer one symbol. In the 32-bit width, the sequence is 134 clock beats long and contains blank symbol transfers for clock beats 44-45, 88-89 and 132-133.

Figure 1-22. 64b67b Receive Sequence For 32-Bit Interface



Sequence timing for the 64-bit interface consists of 67 clock beats with dead cycles at beats 22, 44, and 66.

The following table lists the port names and description for the 64b66b/64b67b mode of the PCS module. See section 49.2.4 of the IEEE® 802.3ae specification for more information.

Table 1-10. 64b66b/64b67b Port List

Port Name	Direction	Clock	Description
LANE#_CDR_REF_CLK_#/ LANE#_CDR_REF_CLK_FAB	Input	—	Reference clock to lane CDR. Can be sourced from either the FPGA clock or from a XCVR_#[A,B,C]REFCLK_P/N pin.
LANE#_REF_CLK	Input	—	This port is exposed to user with Half-Duplex option. LANE#_REF_CLK must be connected by the user to a stable clock with same clock frequency as Recovered clock such as the local clock. **Note:** LANE#_REF_CLK can be provided by a separate PLL or CCC, however the LANE#_REF_CLK frequency must be within ± 300 ppm with respect to LANE#_RX_CLK_R.
LANE#_TX_PLL_REF_CLK_#	Input	—	Input clock from the TX_PLL REF_CLK_TO_LANE output pin. Included in CLKS_FROM_TXPLL_# BIF (bus interface).
LANE#_TX_BIT_CLK_0	Input	—	Clock from BIT_CLK of the XCVR TxPLL. Included in CLKS_FROM_TXPLL_# BIF (bus interface).
LANE#_TX_PLL_LOCK_#	Input	—	Input lock status from the TX_PLL LOCK output pin. Included in CLKS_FROM_TXPLL_# BIF (bus interface).
LANE#_TX_SOS	Input	TX_CLK_[R:G]	Start-of-sequence pulse for a super frame, the length of which varies with the mode.
LANE#_TX_HDR[3:0]	Input	TX_CLK_[R:G]	Sync header corresponding to different encoding types.
LANE#_TX_DATA[63:0]	Input	TX_CLK_[R:G]	Input encoded data from fabric. PF_XCVR sends/receives bytes in high to low byte order only in the 64B6xB mode.
LANE#_PCS_ARST_N	Input	TX_CLK_[R:G]	Asynchronous active-low reset for PCS lane.
LANE#_PMA_ARST_N	Input	TX_CLK_[R:G]	Asynchronous active-low reset for the PMA lane.
LANE#_TX_ELEC_IDLE	Input	Asynchronous	Transceiver configurator allows pin to be exposed on component. This input forces XCVR_P/N transmit output pad pair to a common-mode voltage. It is used for low-frequency out-of-band signaling, or to signal entry into a low-power state to the link partner.

Table 1-10. 64b66b/64b67b Port List (continued)

Port Name	Direction	Clock	Description
LANE#_TX_BYPASS_DATA	Input	Asynchronous	Transceiver configurator allows pin to be exposed on component. When LANE#_TX_BYPASS_DATA=1, the data can be driven into the transmit pads instead of the normal serializer data. The bypass is asynchronous and this signal does not transit through the FWF. This pin must be tied low if exposed to fabric and not used.
LANE#_RXD_N	Input	—	Transceiver receiver differential input.
LANE#_RXD_P	Input	—	Transceiver receiver differential input.
LANE#_RX_HDR_VAL	Output	RX_CLK_[R:G]	Enable for header data in
LANE#_RX_SOS	Output	RX_CLK_[R:G]	Start-of-sequence pulse for a super frame, the length of which varies based on the mode. High output indicates start of sequence.
LANE#_RX_DATA_VAL	Output	RX_CLK_[R:G]	Valid when there is data on RX_DATA.
LANE#_RX_HDR[3:0]	Output	RX_CLK_[R:G]	Sync header corresponding to different encoding types.
LANE#_RX_DATA[63:0]	Output	RX_CLK_[R:G]	Receive encoded data from 64b6b to the fabric. PF_XCVR sends/ receives bytes in high to low byte order only in the 64B6xB mode. Bit 31 or bit 63 arrives first in the serial data.
LANE#_RX_VAL	Output	RX_CLK_[R:G]	LANE#_RX_VAL indicates that the XCVR data path is initialized. The parallel bus of LANE#_RX_DATA[N:0] contains actual data recovered from the serial stream when LANE#_RX_VAL = 1. In 64b66b/64b67b mode, the Rx PCS logic self-resets when the CDR is not locked. In this mode, LANE#_RX_VAL rises just after LANE#_RX_READY rises. If you want to control Rx PCS reset, hold LANE#_PCS_ARST_N in reset when LANE#_RX_READY is low and release, when LANE#_RX_READY goes high. Once LANE#_PCS_ARST_N is released from reset, LANE#_RX_VAL rises to indicate the Rx parallel data is valid. In 64b6xb mode, the RX_VAL is qualified when the XCVR receiver calibration completes, included with the enhanced receiver management completion, and the CDR locks.
LANE#_RX_READY	Output	—	Rises when the enhanced receiver management and CDR completes a fine lock detection to the incoming data transitions and the de-serializer is powered-up. If there is no incoming data to the CDR then the RX_READY is low. The primary purpose of this pin is communicating to fabric that the CDR is locked to serial input data and is producing valid clocking. Note: In a loopback case while looping the local transmitter output to the receiver input, it is necessary to take the Tx out of reset to ensure valid serial transitions, allowing the Rx CDR to lock. The system deadlocks, if the user waits till Rx CDR locks before the Tx is released from reset.
LANE#_RX_IDLE	Output	—	Receive electrical-idle detection flag. LANE#_Rx_IDLE peak detector logic is only valid for a limited minimum density of transitions on the Rx data and not to be used in applications above 5 Gbps.
LANE#_TX_CLK_STABLE	Output	—	Transmit transceiver/PCS lane ready flag.
LANE#_RX_CLK_[R:G]	Output	—	Global or regional receive clock to the fabric for the receiver.
LANE#_TX_CLK_[R:G][1]	Output	—	Global or regional transmit clock to the fabric for the transmitter.
LANE#_STATUS_HI_BER	Output	RX_CLK_[R:G]	From the bit error rate monitor, which counts bad sync header values (0b00 or 0b11). Occurs over 125 µs interval.
LANE#_STATUS_LOCK	Output	RX_CLK_[R:G]	From the selected receive sync lock state-machine (Clause 49 or Clause 82). This is 0 when the sync header boundary is not locked, and it is 1 when sync lock is achieved. See IEEE 802.3 Clause 49 and Clause 82 for block lock state machine.
LANE#_TXD_N	Output	—	Transceiver transmitter differential output.
LANE#_TXD_P	Output	—	Transceiver transmitter differential output.

Important:

1. In PolarFire FPGA MPF500 devices, RT PolarFire RTPF500 devices, and PolarFire SoC FPGA Devices, the TX_CLK_R and RX_CLK_R pins of XCVR lanes placed in the PCIESS(Q0) and GPSS1(Q1) quads cannot drive I/Os.
2. LANE# can be 0, 1, 2, and 3. [R:G] naming is generated based on the use of regional or global resources that are selected using Libero.

1.3.3. PIPE (Ask a Question)

The transceiver PMA interface to PCIe is based on a standard PIPE interface logic. It provides a standard interface between the PMA lane and the higher link-level of the PHY. The logic handles the following functions:

- 8b10b encoding/decoding logic.
- Lane polarity requirements.
- Elastic FIFO and SKP character logic.
- Hot-plug insertion logic.
- PMA controls required by PCIe standard.
- PCIe detection of remote receiver, power state change, and so on. Provides translation of MACs LTSSM states into PMA power states.

The PIPE interface is used by the embedded PCIESS or can be used with a soft PCIe IP in the FPGA fabric. The embedded PCIESS is accessed through a dedicated interface to the PIPE interface mode, which ties the PCIESS to the PIPE without additional fabric logic. All PHY interface signals are synchronous to the PIPE CLOCK. The PIPE PCS is used as the interconnection between either the embedded PCIe block or used with a fabric-based soft-IP connected to the transceiver PMA. The port list differs slightly based on whether the PIPE interface is configured in PCIe mode. See the PHY Interface for the PCI Express.

Table 1-11. PIPE Port List

Port Name	Direction	Clock	Description
LANE#_CDR_REF_CLK_#/ LANE#_CDR_REF_CLK_FAB	Input	—	Reference clock to lane CDR. Can be sourced from either FPGA clock or from a XCVR_#[A,B,C]REFCLK_P/N pin.
LANE#_REF_CLK	Input	—	This port is exposed to user with Half-Duplex option. LANE#_REF_CLK must be connected by the user to a stable clock with same clock frequency as Recovered clock such as the local clock. **Note:** LANE#_REF_CLK can be provided by a separate PLL or CCC, however the LANE#_REF_CLK frequency must be within ± 300 ppm with respect to LANE#_RX_CLK_R.
LANE#_TX_PLL_REF_CLK_#	Input	—	Input clock from TX_PLL REF_CLK_TO_LANE output pin. Included in CLKS_FROM_TXPLL_# BIF (bus interface).
LANE#_TX_PLL_LOCK_#	Input	—	Input lock status from TX_PLL LOCK output pin. Included in CLKS_FROM_TXPLL_# BIF (bus interface).
LANE#_TX_BIT_CLK_#	Input	—	Clock from BIT_CLK of the XCVR TXPLL. Included in CLKS_FROM_TXPLL_# BIF (bus interface).
LANE#_TX_DATA[39:0]	Input	—	Parallel data bus to the PCS from the fabric. The PF_XCVR send/ receive order is low to high byte. When the hard PCIESS is used, received data is always 32 bits wide. The upper bits [39:32] are ignored. When the soft PCIe IP design is used, then TXDATA is 40 bits wide; otherwise, it is 32 bits wide (the upper byte is ignored).

Table 1-11. PIPE Port List (continued)

Port Name	Direction	Clock	Description
LANE#_TXDATAK[3:0]	Input	TX_CLK_[R:G]	Indicates the type of characters on the TXDATA bus. A 0 indicates a data character, while 1 indicates a control character. If the soft PCIe IP is used, this signal is ignored.
LANE#_TXDETECTRX_LOOPBACK	Input	TX_CLK_[R:G]	High instructs the PHY to begin the receive detect process or external loopback as described in the PCI Express specification. As with many PIPE signals, the meaning of this signal depends on which power state the PHY is in. When in P1 state, this signal informs the PHY to perform a receive detect, but when in P0 state, this signal informs the device to go into loopback mode.
TXCOMPLIANCE	Input	TX_CLK_[R:G]	When asserted, this ordinarily forces the currently running disparity to negative. As the name implies, this is useful in conjunction with the transmission of the compliance pattern to generate test data. It is also used in a multi-lane implementation to turn off any unused lanes, for example, when a ×4 link must operate as a ×1. When both TXCOMPLIANCE and TXELECIDLE are asserted, the affected lane turns off to conserve as much power as possible.
POWERDOWN[1:0]	Input	TX_CLK_[R:G]	These inputs place the transceiver into one of four power states: P0: normal operational mode. P0s: PCLK remains on, but the receiver conserves power; entered when the receiver detects electrical idle. Corresponds to link state L0s. P1: PCLK remains on; both the receiver and transmitter are in electrical idle. Corresponds to link state L1. P2: PCLK is off. The PHY must minimize power consumption as it must operate within the VAUX limits. Corresponds to link state L2. See the PHY Interface for the PCI Express for more details on PHY power management.
PCIE_RATE	Input	TX_CLK_[R:G]	Controls the link signaling rate. In PCIe mode: 0: Gen1 (2.5 Gbps) 1: Gen2 (5.0 Gbps)
TXMARGIN[2:0]	Input	TX_CLK_[R:G]	Selects transmitter voltage levels: 000: TxMargin value 0. Normal operating range 001: TxMargin value 1. 800 mV–1200 mV for full swing or 400 mV –700 mV for half swing 010: TxMargin value 2 (required). Vendor defined 011: TxMargin value 3 (required). Vendor defined 100: TxMargin value 4 (required). 200 mV–400 mV for full-swing or 100 mV–200 mV for half-swing 101: TxMargin value 5 (optional). 200 mV–400 mV for full-swing or 100 mV–200 mV for half-swing 110: TxMargin value 6 (optional). 200 mV–400 mV for full swing or 100 mV–200 mV for half-swing 111: TxMargin value 7 (optional). 200 mV–400 mV for full swing or 100 mV–200 mV for half-swing
TXDEEMPH	Input	TX_CLK_[R:G]	Selects transmitter de-emphasis: 0: 6 dB de-emphasis at 5 Gbps 1: 3.5 dB de-emphasis at 5 Gbps PIPE implementations that only support 2.5 Gbps signaling rate do not implement this signal.

Table 1-11. PIPE Port List (continued)

Port Name	Direction	Clock	Description
TXSWING	Input	TX_CLK_[R:G]	Controls transmitter voltage swing: 0: Full swing 1: Half swing
LANE#_RXPOLARITY	Input	TX_CLK_[R:G]	This active-high signal indicates the PHY to do a polarity inversion on the received data.
LANE#_RXSTANDBY	Input	TX_CLK_[R:G]	Used to set the RxStandby state: 0: Active 1: Standby
LANE#_TXELECIDLE	Input	TX_CLK_[R:G]	When this signal is asserted high, it forces the transmitter to the electrical idle state regardless of power states. When this signal is de-asserted, valid data from TXDATA and TXDATAK are transmitted in the P0 state. If the PHY is in the P2 state, a beacon must be transmitted when TXELECIDLE is de-asserted. In the P0s and P1 states, TXELECIDLE must be asserted. The use of this signal is also affected by the PHY power state since there are some states in which the transmitter must be electrically idle. See the PHY Interface for the PCI Express for more detail.
LANE#_PCS_ARST_N	Input	—	Asynchronous active-low reset for PCS lane.
LANE#_PMA_ARST_N	Input	—	Asynchronous active-low reset for PMA lane.
LANE#_RXD_N	Input	—	Transceiver receiver differential input.
LANE#_RXD_P	Input	—	Transceiver receiver differential input.
LANE#_RXELECIDLE	Output	RX_CLK_[R:G]	The PCIe receiver pins detect an electrical idle state on the link. High indicates receiver detection of electrical idle, and low indicates beacon signaling when in P2.
LANE#_RXSTANDBYSTATUS	Output	RX_CLK_[R:G]	The PHY uses this signal to indicate its Rx Standby state. 0: Active 1: Standby
LANE#_RX_DATA[39:0]	Output	RX_CLK_[R:G]	Parallel data bus from the PCS to the fabric. The PF_XCVR send/receive order is low to high byte. When the soft PCIe IP design is used, then RXDATA is 40 bits wide; otherwise, it is 32 bits wide (the upper byte is ignored).
LANE#_RXDATAK[3:0]	Output	RX_CLK_[R:G]	The data#/control indicator(s) for the received symbols on the RXDATA bus: 0: RXDATA contains data 1: RXDATA contains ordered sets If the soft PCIe IP is used, then RXDATAK is ignored.
LANE#_RXVALID	Output	RX_CLK_[R:G]	Qualifies the data on RXDATA and RXDATAK. When this signal is asserted, the data on the receive data bus is valid, and the PHY has achieved symbol lock.
LANE#_RXSTATUS[2:0]	Output	RX_CLK_[R:G]	Delivers receiver status and error codes for the received data and receiver detect status from the PHY to the MAC; for example, SKP symbol added or removed, disparity error, elastic buffer overflow or underflow, 8b10b decode error, and so on. 0 0 0: Received data OK 0 0 1 1: SKP added 0 1 0 1: SKP removed 0 1 1: Receiver detected 1 0 0: Code error 1 0 1: Elastic buffer overflow 1 1 0: Elastic buffer underflow 1 1 1: Receive disparity error

Table 1-11. PIPE Port List (continued)

Port Name	Direction	Clock	Description
LANE#_RX_BYPASS_DATA	Output	Async	RX_BYPASS_DATA output is a low-speed bypass of the differential receiver that is used for the receive pads. This is a a low-frequency out-of-band debug signal.
LANE#_PHYSTATUS	Output	RX_CLK_[R:G]	Signals that the PHY has completed its setup and is ready for data traffic. It is also used to indicate successful transition from power management state, rate change, and receive detection.
LANE#_RX_VAL	Output	RX_CLK_[R:G]	LANE#_RX_VAL indicates that the XCVR data path is initialized. The parallel bus of LANE#_RX_DATA[N:0] contains actual data recovered from the serial stream when LANE#_RX_VAL = 1. In PIPIE mode, the Rx PCS logic self-resets when the CDR is not locked. In this mode, LANE#_RX_VAL rises just after LANE#_RX_READY rises. If you want to control Rx PCS reset, hold Rx PCS in reset when LANE#_RX_READY is low and release, when LANE#_RX_READY goes high. Once Rx PCS is released from reset, LANE#_RX_VAL rises to indicate the Rx parallel data is valid. In PIPE mode, the LANE#_RX_VAL is qualified when the CDR locks and the initial comma/word alignment occurs.
LANE#_RX_READY	Output	—	Rises when the CDR is phase-locked to the incoming data transitions and the de-serializer is powered-up. If there is no incoming data to the CDR then the RX_READY is low. The primary purpose of this pin is communicating to fabric that the CDR is locked to serial input data and is producing valid clocking. Note: In a loopback case while looping the local transmitter output to the receiver input, it is necessary to take the Tx out of reset to ensure valid serial transitions, allowing the Rx CDR to lock. The system deadlocks, if the user waits till Rx CDR locks before the Tx is released from reset.
LANE#_RX_IDLE	Output	—	Receive electrical-idle detection flag. Asserts asynchronously, but de-asserts synchronously to the RX_CLK_OUT rising edge.
LANE#_TX_CLK_STABLE	Output	—	Transmit transceiver/PCS lane ready flag. This flag is 1 when the transmit PLL is locked to the reference clock.
LANE#_TX_CLK_[R:G][1]	Output	—	Global or regional transmit clock to fabric.
PIPE_CLOCK	Output	—	In PCIe Gen1/Gen2 modes of Soft PIPE PCS setting, there is only one TX_CLK_[R:G] for all the 4 lanes that could be configured in single XCVR configurator instance, that is, LANE0_TX_CLK_[R:G] (renamed as PIPE_CLOCK). This PIPE_CLOCK is commonly used for all the lanes. Internally, both RX_FWF_CLK and TX_FWF_CLK (internal ports on the XCVR macro) of each lane are connected to the PIPE_CLOCK (LANE0_TX_CLK_R/G) through RCLKINT/CLKINT.
LANE#_TXD_N	Output	—	Transceiver transmitter differential output.
LANE#_TXD_P	Output	—	Transceiver transmitter differential output.

(1) In PolarFire FPGA MPF500 devices, RT PolarFire RTPF500 devices, and PolarFire SoC FPGA, the TX_CLK_R and RX_CLK_R pins of XCVR lanes placed in the PCIESS(Q0) and GPSS1(Q1) quads cannot drive I/Os.

Important: LANE# can be 0, 1, 2, and 3. [R:G] naming is generated based on the use of regional or global resources that are selected using Libero.

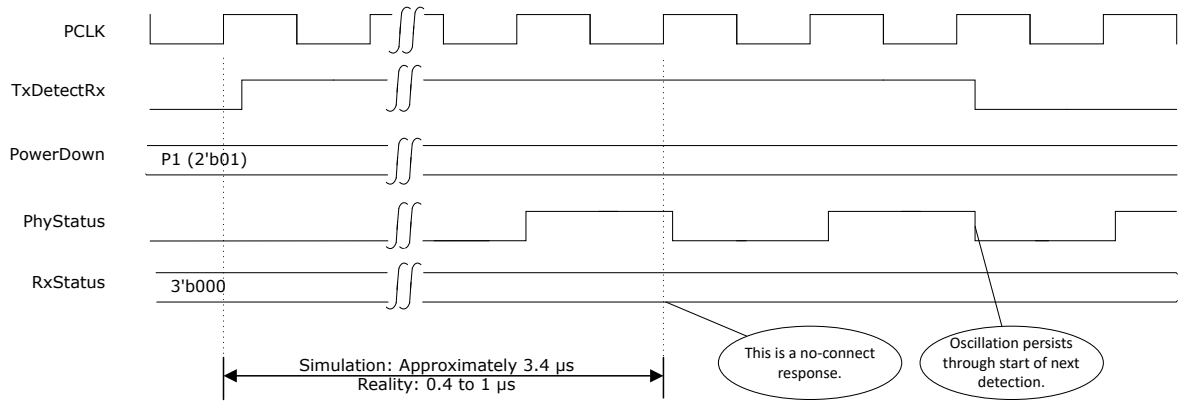
1.3.4. PIPE Interface Compliance Exceptions (Ask a Question)

The PIPE Interface complies with Intel's Specification version 4.0 with the following exceptions.

- P1 to P0 power-state, PhyStatus pulse response is approximately 30 microseconds delayed from the request to enter P0. The long delay is caused by waiting for the Rx PLL to spin-up and stabilize. The P0-entry is observed externally as a long delay before sending training patterns. The training pattern delay is a fatal protocol error.
- PCI-Express receiver detection has the following issues when used with soft PIPE interface.
  - A receiver detection returning a Receiver-Not-Present status does so with an oscillating PhyStatus pulse train. MAC-side LTSSM logic must ignore all but the first PhyStatus pulse of the oscillation.
  - A subsequent receiver detection operation N, begins a receiver detect, but immediately returns the status of detection N-1 through PhyStatus and RxStatus when N > 0. The very first receiver detection (N = 0) waits until the conclusion of the receiver detect activity and returns its status through PhyStatus and RxStatus.

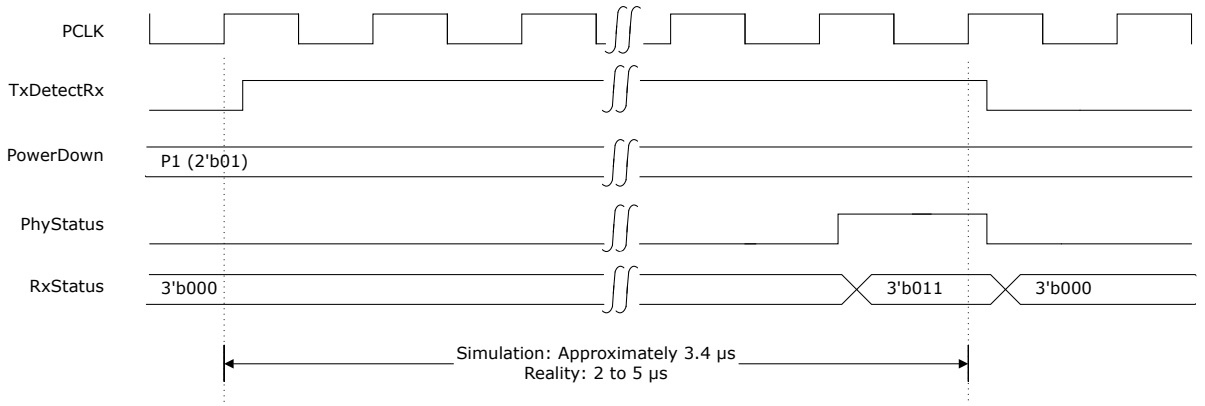
The following figure shows how PIPE interface signals behave when the first detection finds no connected receiver. The response time in simulation is fixed for a given reference clock frequency due to the means of modeling the XCVR. The XCVR model has a fixed reflection time of the common-mode voltage step. In real silicon, the reflection time depends upon the termination network, and this also determines when PIPE PhyStatus pulse occurs.

Figure 1-23. Initial Receiver Detection Response For Receiver-Not-Present



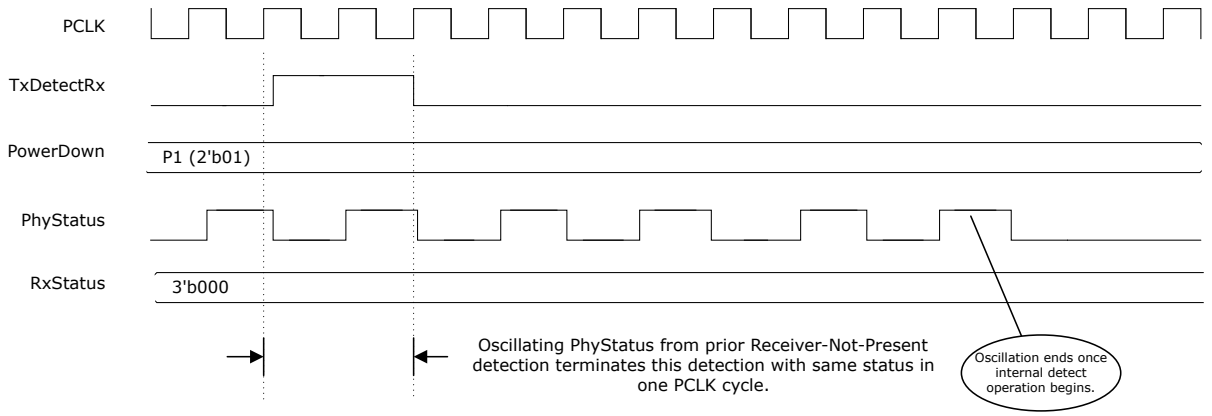
The following figure shows how the soft PIPE interface signals behave when the first detection finds a connected receiver. The difference between receiver presence and absence is the value of RxStatus in simulation, but in real silicon, the delay to the PhyStatus pulse can be different. In real silicon, the Receiver-Not-Present response occurs earlier than a Receiver-Present response.

Figure 1-24. Initial Receiver Detection For Receiver-Present



The following figure shows how PIPE interface responds to a successive receiver detection when the prior detection result is Receiver-Not-Present.

Figure 1-25. Subsequent Receiver Detection Where Prior Status Was Receiver-Not-Present



1.3.5. PMA Only (Ask a Question)

This interface uses the transceiver PMA as a conduit to receive and transmit data to or from the fabric boundary. Serial data is deserialized and sent to the receive FWF before the data is sent to the parallel FPGA fabric bus. Similarly, parallel data from the fabric goes through a transmit FWF prior to entering the parallel interface of the serializer.

The following features are supported in the PMA only mode:

- Flexible PMA interface width options of [8, 10, 16, 20, 32, 40] bits.
- Flexible fabric interface width options of [8, 10, 16, 20, 32, 40, 64, 80] bits per clock beat.
- Fabric control of CDR bit-slip for optional symbol alignment purposes.
- Fabric control of transmitter electrical-idle.
- Fabric control override of CDR lock behavior enabling Burst Mode Receiver support.
- Fabric side interface for transmitter operates at half the frequency of the transmit PMA interface.
- Fabric side interface for receiver operates at half the frequency of the receiver PMA interface.

The PMA interface bypasses any PCS encoding and decoding logic and is used with customized PCS functionality implemented in the FPGA fabric. The fabric-hosted soft-IP can be customer supplied or soft-IP provided through 3rd-party of Microsemi direct cores. The transceiver PMA mode is useful

in supporting protocols such as SDI-HD. The PMA Only mode is also used for 1GbE interfaces. The CoreTSE suite of 1GbE IPs contain a soft 8b10b encoder/decoder, which allows the use of either the transceiver or the I/O CDR to implement this standard.

Figure 1-26. PMA-Bus Waveform

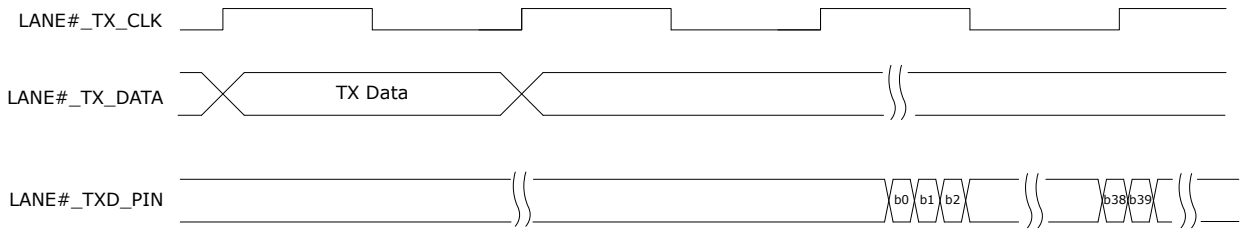


Figure 1-27. PMA Only Data Path – 80-bits

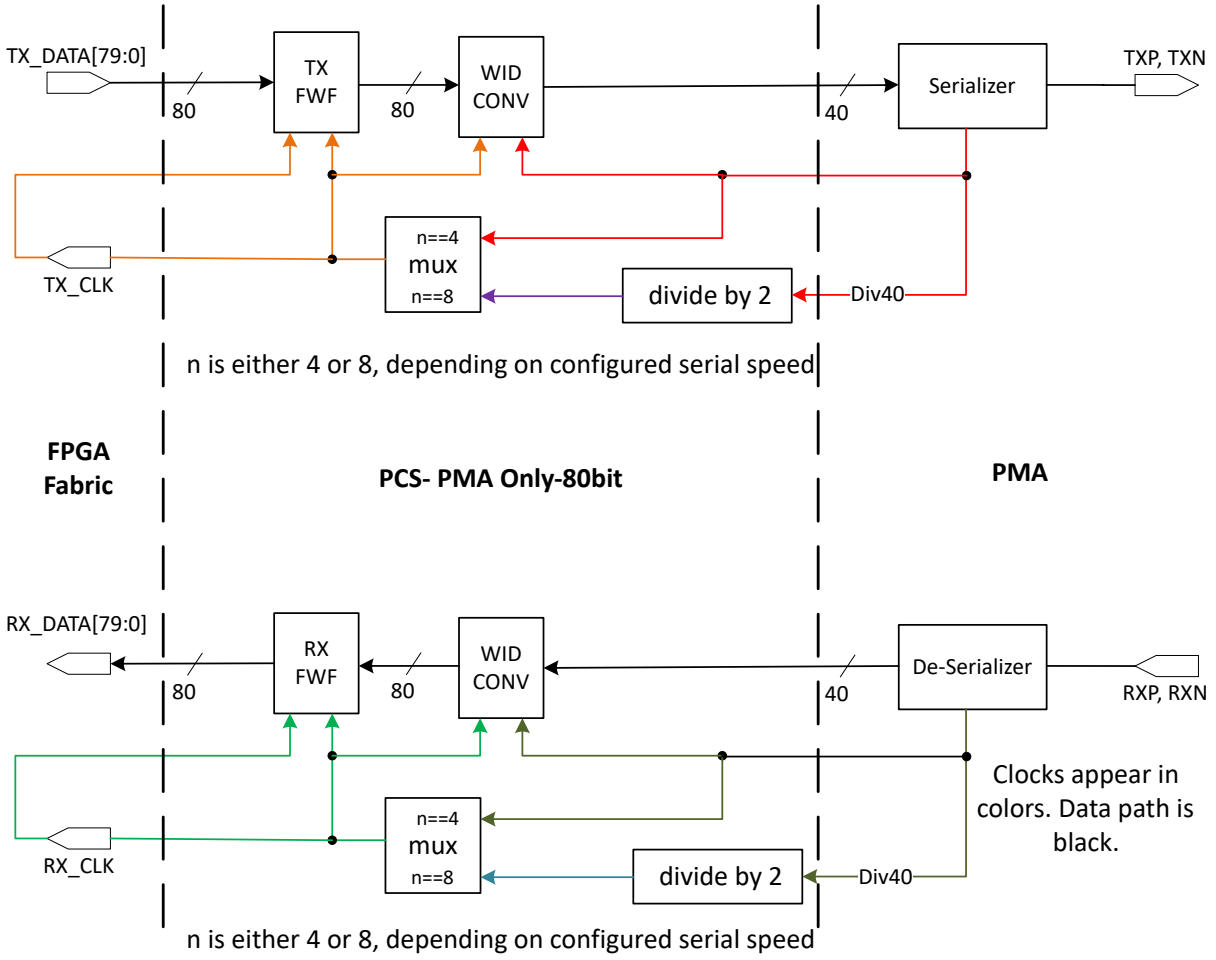
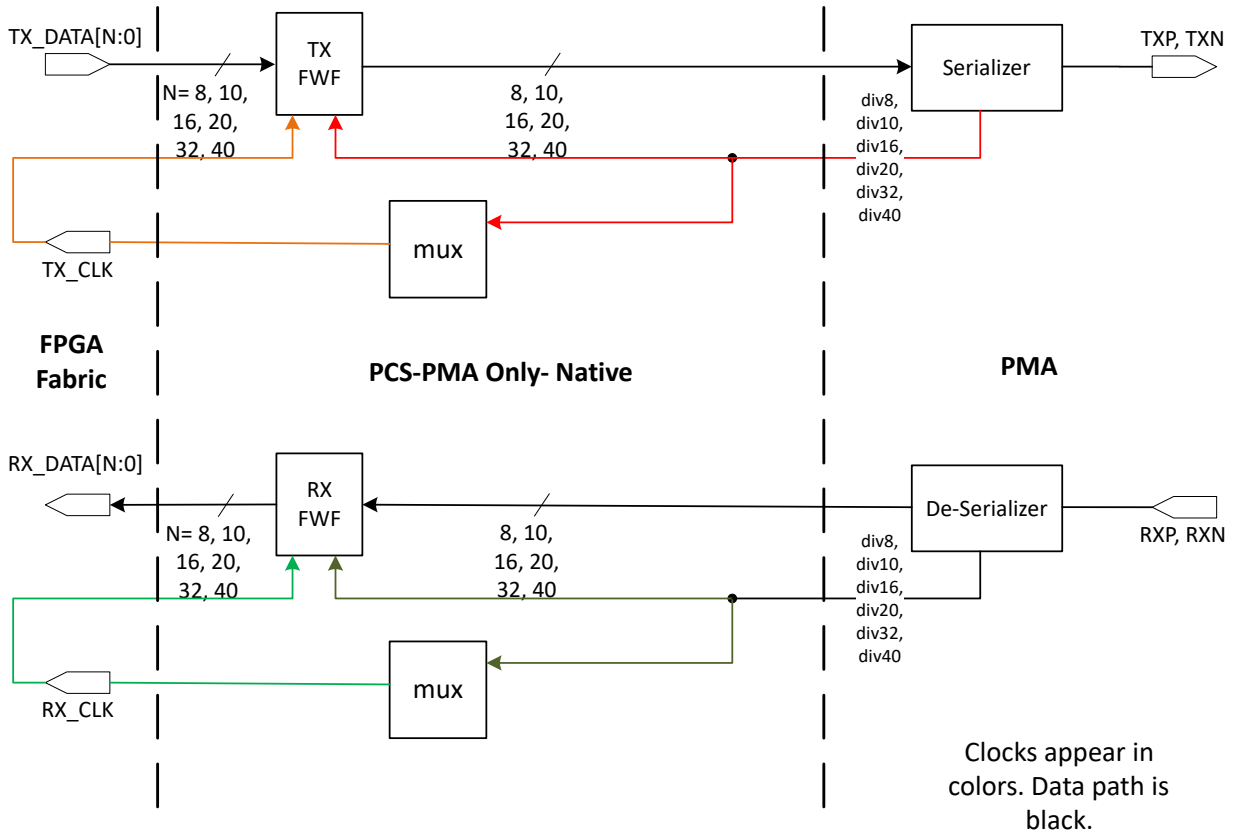


Figure 1-28. PMA Only Data Path – Less Than or Equal to 40-bits



The following table lists the port names and description for the PMA mode module.

Table 1-12. PMA Port List

Port Name	Direction	Clock	Description
LANE#_CDR_REF_CLK_#/ LANE#_CDR_REF_CLK_FAB	Input	—	Reference clock to lane CDR. Can be sourced from either FPGA clock or from a XCVR_#[A,B,C]REFCLK_P/N pin.
LANE#_REF_CLK	Input	—	This port is exposed to user with Half-Duplex option. LANE#_REF_CLK must be connected by the user to a stable clock with same clock frequency as Recovered clock such as the local clock. **Note:** LANE#_REF_CLK can be provided by a separate PLL or CCC, however the LANE#_REF_CLK frequency must be within ± 300 ppm with respect to LANE#_RX_CLK_R.
LANE#_TX_BIT_CLK_[1:0][2]	Input	—	Clock from BIT_CLK of the XCVR TxPLL. Included in CLKS_FROM_TXPLL_# BIF (bus interface).
LANE#_TX_PLL_REF_CLK_#	Input	—	Input clock from TX_PLL REF_CLK_TO_LANE output pin. Included in CLKS_FROM_TXPLL_# BIF (bus interface).
LANE#_TX_PLL_LOCK_#	Input	—	Input lock status from TX_PLL LOCK output pin. Included in CLKS_FROM_TXPLL_# BIF (bus interface).
LANE#_TX_DATA[N:0][1]	Input	TX_CLK_[R:G]	Transmits data. The PF_XCVR send/receive order is low to high byte. The first serial bit appears in bus bit0. Switching transmission order in PMA mode is accomplished by reversing the bus connections in the fabric.

Table 1-12. PMA Port List (continued)

Port Name	Direction	Clock	Description
LANE#_TX_ELEC_IDLE	Input	Asynchronous	Transceiver configurator allows pin to be exposed on component. This input forces XCVR_P/N transmit output pad pair to a common-mode voltage. This can be used for low-frequency out-of-band signaling, or to signal entry into a low-power state to the link partner.
LANE#_TX_BYPASS_DATA	Input	Asynchronous	Transceiver configurator allows pin to be exposed on component. When LANE#_TX_BYPASS_DATA=1, the data can be driven into the transmit pads instead of the normal serializer data. The bypass is asynchronous and this signal does not transit through the FWF. This pin must be tied low if exposed to fabric and not used.
LANE#_RX_SLIP	Input	RX_CLK_[R:G]	LANE#_RX_SLIP assertion resets the Rx_FWF causing LANE#_RX_VAL to momentarily deassert(=0). Rising-edge requests that the transceiver lane CDR slip the parallel boundary by one bit. The direction of slip is different for 8b parallel word modes than it is for 10b parallel word modes. See Bit Slip.
LANE#_PCS_ARST_N[2]	Input	—	Asynchronous active-low reset for the PCS lane.
LANE#_PMA_ARST_N[2]	Input	—	Asynchronous active-low reset for the PMA lane.
LANE#_RXD_N[2]	Input	—	Transceiver receiver differential input.
LANE#_RXD_P[2]	Input	—	Transceiver receiver differential input.
LANE#_RX_DATA[N:0][1]	Output	RX_CLK_[R:G]	Receives data. The PF_XCVR send/receive order is low to high byte. The first serial bit appears in bus bit0.
LANE#_TX_CLK_STABLE[2]	Output	—	Transmits transceiver/PCS lane ready flag. This flag is 1 when the transmit PLL is locked to the reference clock.
LANE#_RX_VAL[2]	Output	—	LANE#_RX_VAL indicates that the XCVR data path is initialized. The parallel bus of LANE#_RX_DATA[N:0] contains actual data recovered from the serial stream when LANE#_RX_VAL = 1. LANE#_RX_VAL is qualified when the XCVR receiver calibration completes, included with the enhanced receiver management completion, and the CDR locks. LANE#_RX_VAL pulsing low while RX_READY = 1 does not indicate that the clocking is unstable. It means that the LANE#_RX_DATA output is all-zeros temporarily because the Rx datapath is in reset. If that condition does not need to be detected for a specific application, then LANE#_ RX_VAL can be ignored. Lane#_RX_VAL = 1, indicates the PCS is out of reset, which implies LANE#_RX_READY = 1 because PCS Rx is normally being held in reset while LANE#_RX_READY = 0
LANE#_RX_READY[2]	Output	—	Lane#_ RX_READY = 1 means the Rx PLL is locked. LANE#_RX_READY rises when the enhanced receiver management and CDR completes a fine lock detection to the incoming data transitions and the de-serializer is powered-up. If there is no incoming data to the CDR then the RX_READY is low. The primary purpose of this pin is communicating to fabric that the CDR is locked to serial input data and is producing valid clocking. Note: In a loopback case while looping the local transmitter output to the receiver input, it is necessary to take the Tx out of reset to ensure valid serial transitions, allowing the Rx CDR to lock. The system deadlocks, if the user waits till Rx CDR locks before the Tx is released from reset.
LANE#_RX_IDLE[2]	Output	—	Receives electrical-idle detection flag. LANE#_Rx_IDLE peak detector logic is only valid for a limited minimum density of transitions on the Rx data and not to be used in applications above 5Gbps. Enhanced Receiver Management can provide a reliable mechanism for higher data rates.
LANE#_RX_CLK_[R:G][2]	Output	—	Global or regional receive clock to fabric.
LANE#_TX_CLK_[R:G][2, 3]	Output	—	Global or regional transmit clock to fabric.

Table 1-12. PMA Port List (continued)

Port Name	Direction	Clock	Description
LANE#_TXD_N[2]	Output	—	Transceiver transmitter differential output.
LANE#_TXD_P[2]	Output	—	Transceiver transmitter differential output.

(1) N can be 7, 9, 15,19, 31, 39, 63, and 79.
(2) LANE# can be 0, 1, 2, and 3.
(3) In PolarFire FPGA MPF500 devices, RT PolarFire RTPF500 devices, and PolarFire SoC FPGA devices, the TX_CLK_R and RX_CLK_R pins of XCVR lanes placed in the PCIESS(Q0) and GPSS1(Q1) quads cannot drive I/Os.

Important: [R:G] naming is generated based on the use of regional or global resources that are selected with Libero.

1.4. PCS/FPGA Fabric Interface (Ask a Question)

The FPGA fabric-transceiver interface includes both clock and data signals. This interface provides clock interconnects using the global or regional clock networks in the FPGA fabric. Based on the transceiver lane configuration, the receive parallel output clock is recovered from either the receive serial data or the rate matched clock generated by the embedded clock domain crossing logic. Transmit output is always from the TxPLL. The PCS/FPGA fabric can gear the fabric interface by an additional 1:2 ratio provided by the PCS dividers (see Transmit PCS Divider). The modes where this can occur are 8b10b using 8-octet interfacing, 64b6xb using 8-byte interfacing, PMA mode 64-bit, and 80-bit interfacing.

Local clock outputs of the PF_XCVR are programmed to track with the additional gearing done in the PCS. If the fabric interface width is geared by the PCS and global clocks are the source of the interface timing, then the global clock output must be divide-by-two.

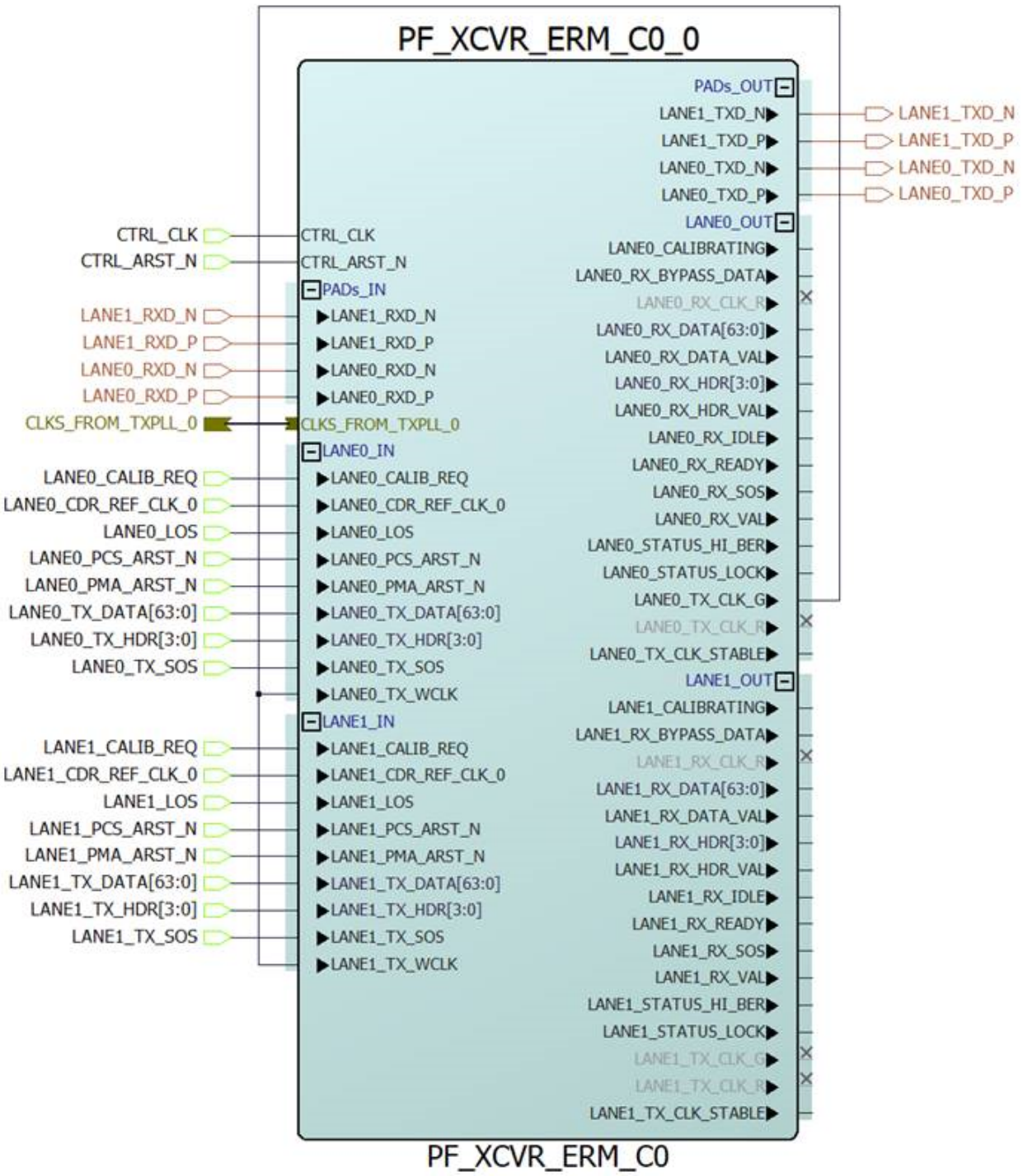
There are four clocking resources from PCS to the fabric-Global, Regional, Regional (Deterministic), and Global-Shared.

- **Global**: These clocks have a dedicated interconnection specifically designed to reach throughout the device from the transceiver fabric interface onto dedicated FPGA fabric global clock network. Global clocks are designed to have low skew and low duty cycle distortion, low power, and improved jitter tolerance and support for very high-frequency signals.
- **Global-Shared**: These clocks are similar to Globals but have resources to allow sharing between the TxPLLs. The sharing of lane clocking resources results in equal latency in the transmitter phase compensation FIFO of all shared lanes.

Important: Global-Shared mode allows the global clock output from one lane to be used for other lanes. This clocking mode is used when several lanes are part of the same protocol using the same clock.

When Global-Shared mode is enabled, all the lanes include a global clock output port, but only one lane can be used as the master port. All the lanes include a TX_WCLK input port. This port must be connected to the global clock output of the master lane. The fly-wheel FIFO absorbs the phase difference between the lanes and the master lane's global clock. See the following SmartDesign figure for an example.

Figure 1-29. Global-shared Clocking Example



- **Regional**: These clocks use local resources to interconnect with the FPGA fabric with a FIFO in the data path at the fabric boundary.
- **Regional (Deterministic)**: These clocks use local resources to interconnect with the FPGA fabric with zero-cycle data path at the fabric boundary providing low-latency.
  Microchip IP and solutions use these resources to optimize the FPGA architecture to provide robust use cases. See Table 1-16.

1.4.1. Non-Deterministic Interface (Ask a Question)

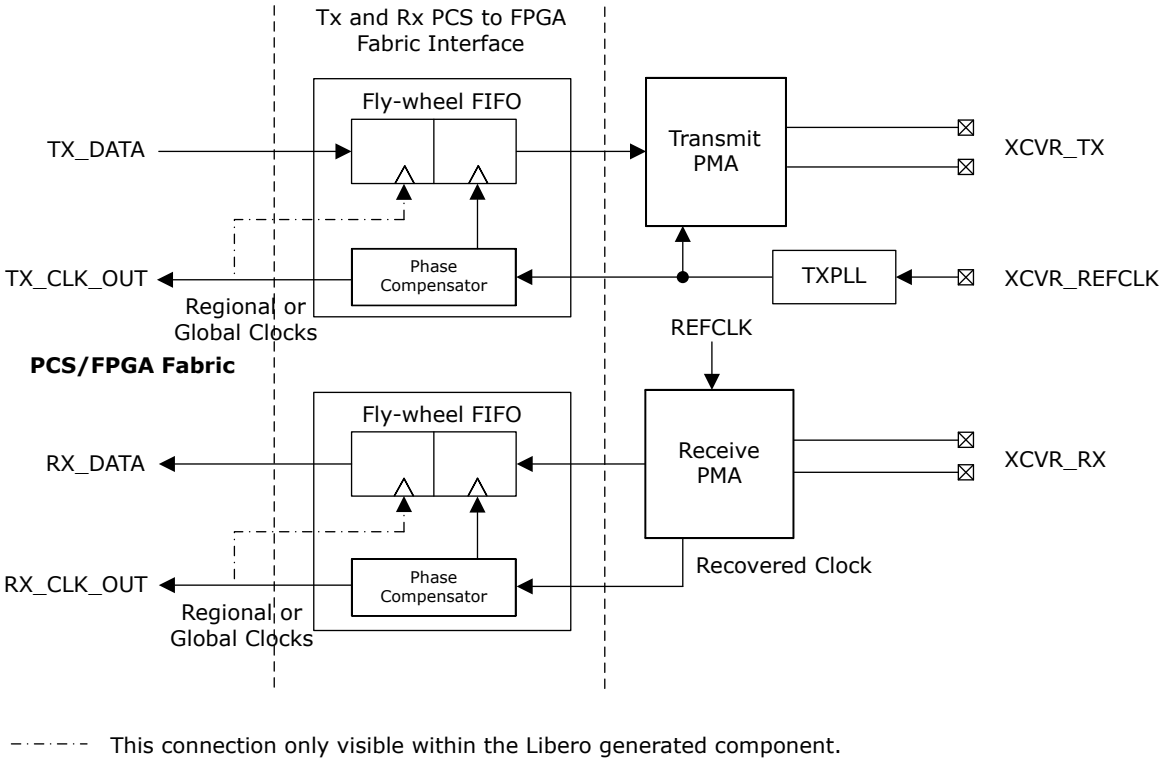
The transceiver PMA to FPGA fabric/PCS data path includes a fly-wheel FIFO (FWF) interface, which is used to transfer data between clock domains. This interface is included within the protocol-specific PCS HDL modules or user FPGA fabric logic.

FWF is a simple form of FIFO where its write and read clocks are known to be at the same nominal frequency, with some allowed phase difference and jitter that is compensated within the block.

The interface between the PMA and fabric cannot be throttled. However, the addition of the FWF block in the data path handles the phase crossing of every PMA lane, ensuring that timing is met across this interface. The received data, along with the recovered parallel clock, is passed to the FWF, which synchronizes the data and clock for either regional or global clock routing.

In the transmit direction, data from the fabric or PCS is passed through the FWF with a clock from the FWF ensuring synchronous clock and data relationships passing to the PMA interface. The FWF is optionally selected in the Libero Transceiver Configurator by choosing the correct global or regional interface clock option, see Table 2-8.

Figure 1-30. Non-Deterministic Interface With FWF



The FWF provides clock domain crossing functionality to manage clock and data setup and hold. The following figures show the timing relationships of transmit and receive data paths at the fabric interface.

Figure 1-31. Non-Deterministic Interface Transmit Timing Waveform

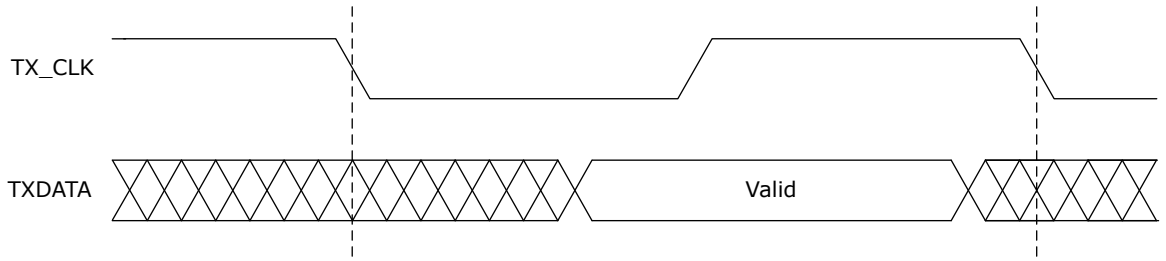
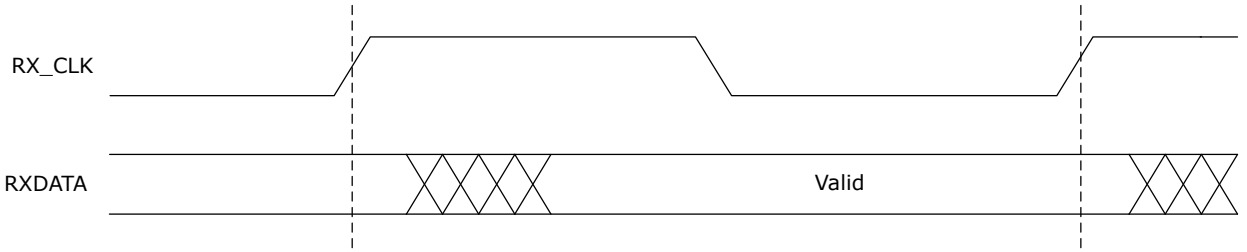


Figure 1-32. Non-Deterministic Transceiver Receive Timing Waveform



1.4.2. Deterministic Interface (Ask a Question)

Low-latency regional clocks with a specific mode of the FWF are used when a zero-cycle path is required. For example, by protocols such as CPRI and JESD204B that require both receive and transmit paths have a fixed deterministic latency as expressed in number of clock cycles. In this case, data is interfaced directly to capture registers while the clock is routed on regional clock resources. The regional clock does not have the large clock insertion delay as the global clock network. A regional clock can easily achieve timing closure to the fabric with this small amount of clock delay. Deterministic timing is optionally selected in the Libero Transceiver configurator by choosing the deterministic regional options, see Table 2-8.

Figure 1-33. Deterministic Timing Interface

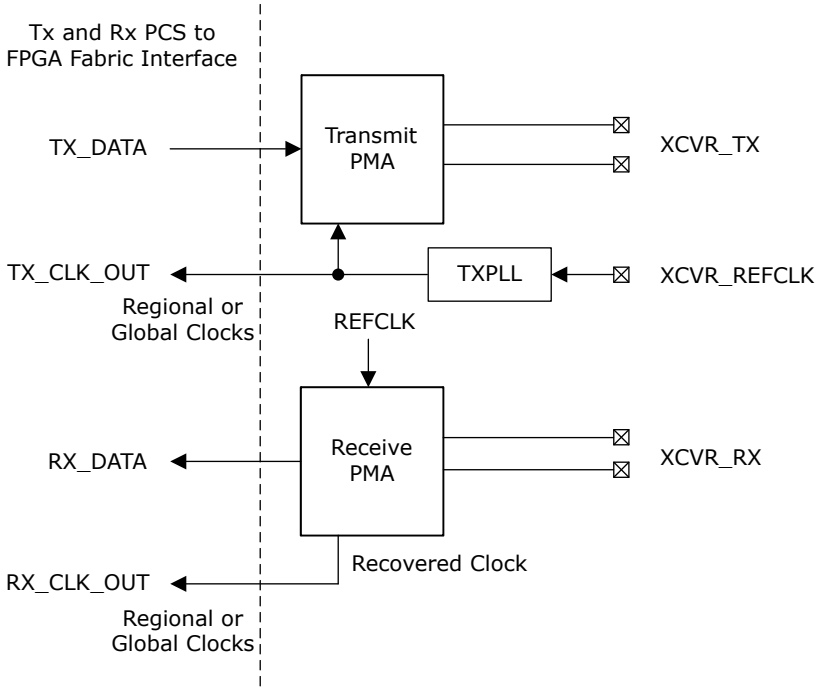
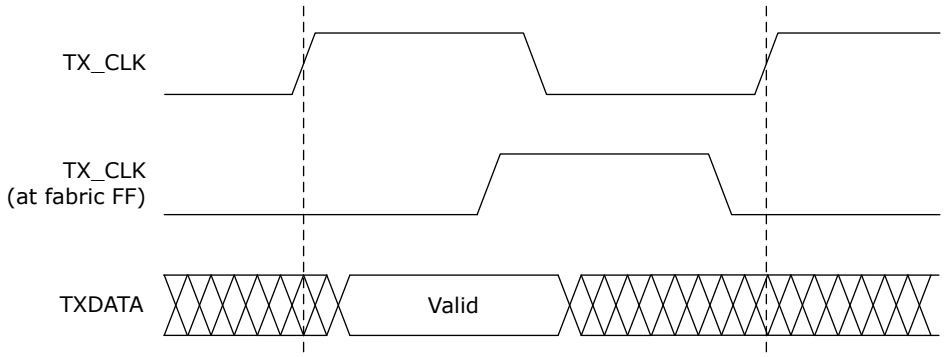
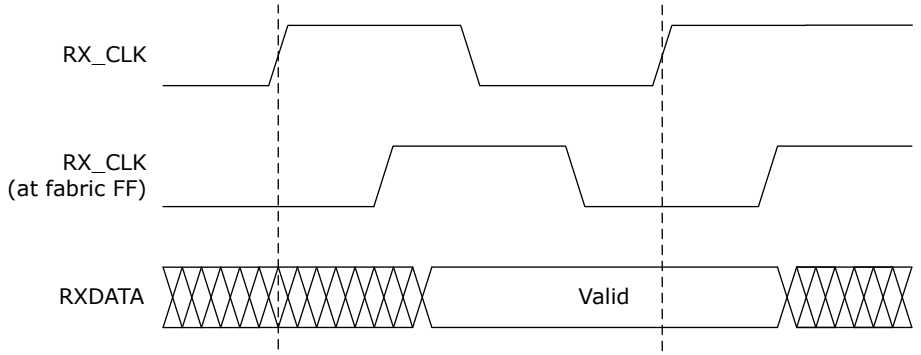


Figure 1-34. Deterministic Transceiver Transmit Timing Waveform



Note: TXCLK_FABRIC at PCS I/F after Regional clock route.

Figure 1-35. Deterministic Transceiver Receive Timing Waveform



Note: RXCLK_FABRIC at PCS I/F after Regional clock route.

1.4.3. Interface Latency (Ask a Question)

Latency of a FWF is half of its depth with an uncertainty of 1. The Deterministic mode for the FWF has a pipeline stage added between the fabric and the FWF due to which the latency is higher for this mode. This means that the Non-Deterministic (phase-compensating) mode has less latency because it bypasses the extra pipeline stage.

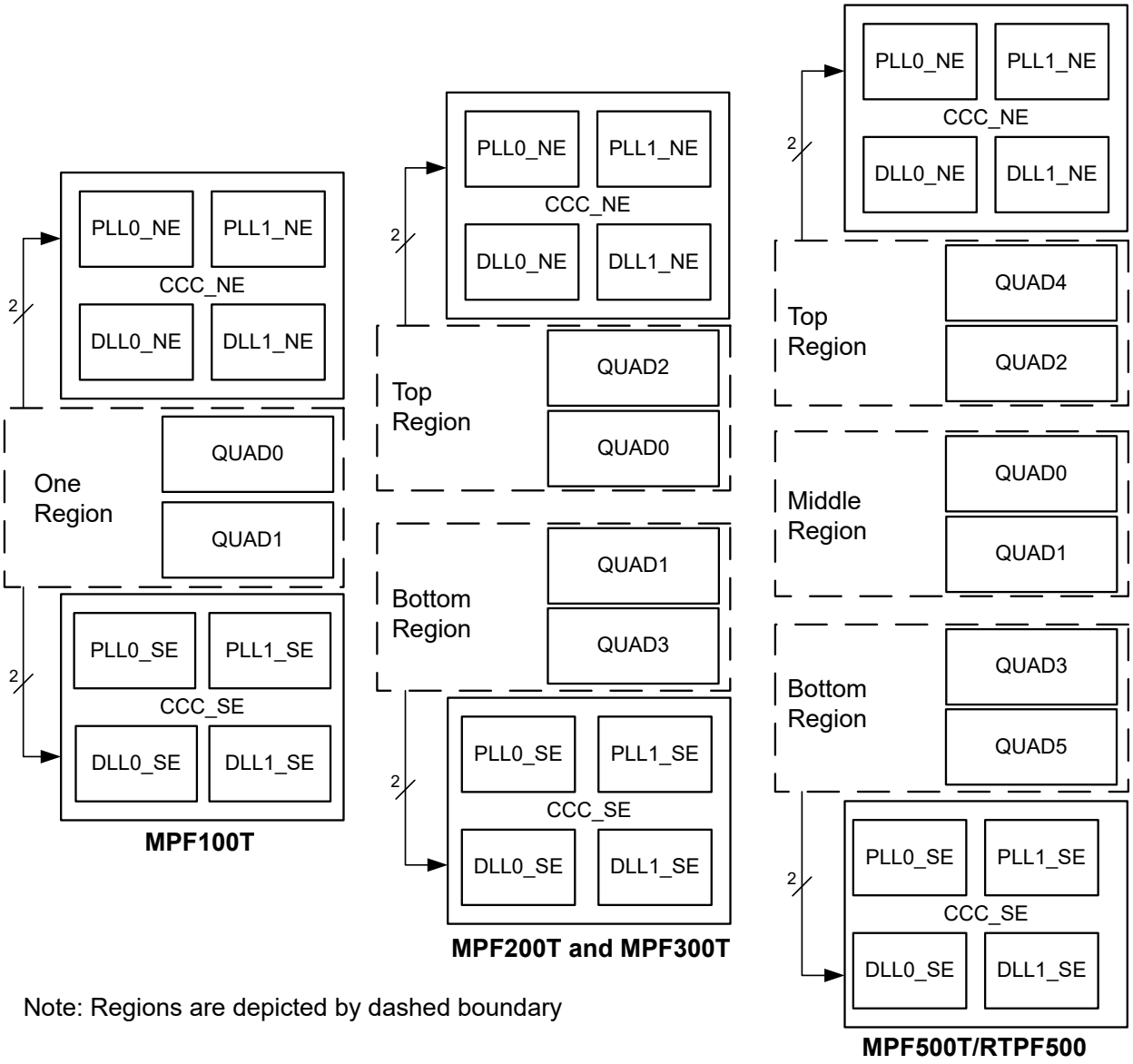
FWF is 8-deep for both Deterministic (± 1) and Non-Deterministic modes. The Non-Deterministic mode is controlled by the pcslane/lclk_r0/lclk_txfwf_tmg_mode register selected between Deterministic mode and the Phase Compensating mode. Deterministic mode is enabled when lclk_txfwf_tmg_mode is equal to 1.

Libero SoC software allows to select the Deterministic mode options within the Transceiver Interface configurator when selecting Tx and Rx clock options.

1.4.4. Transceiver Clock Regions (Ask a Question)

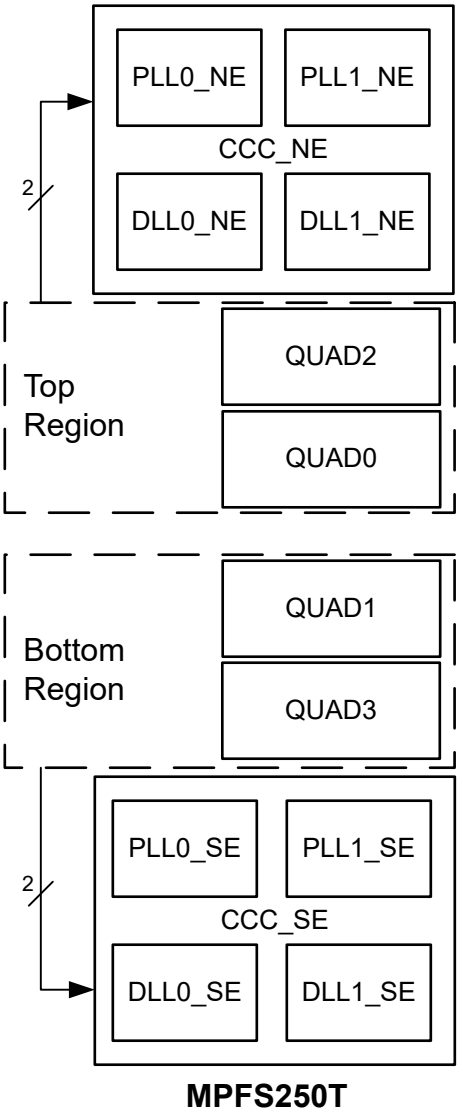
Two regional clock buffers per transceiver lane (eight per transceiver quad) come from the transceiver. These interconnections are the basis for specific regions that the particular quads can drive. These regions vary from device to device within the family. These regions have a predetermined connectivity to fabric resources, CCC, and I/Os.

Figure 1-36. PolarFire® and RT PolarFire FPGA Transceiver Clock Regions



Note: Regions are depicted by dashed boundary

Figure 1-37. PolarFire® SoC FPGA Transceiver Clock Regions



Note: Regions are depicted by dashed boundary.

Users need to understand the regional clock implications when targeting designs that may migrate to different device sizes. The user must also use this in pin planning of boards when desiring to drive I/O from the transceiver clocks.

Table 1-13. PolarFire® and RT PolarFire FPGA Clock Region Connectivity

Device	Region	FPGA Fabric Resource Count						
		IO	4LUT	DFF	MATH	SRAM	URAM	CCC (PLL/DLL) (CCC_NE, CCC_SE)
MPF100T	One Only	120	49260	49260	152	160	456	4(PLL/DLL) (CCC_NE, CCC_SE)
MPF200T	Top	60	49980	49980	152	160	456	2- CCC (PLL/DLL) (CCC_SE)
	Bottom	60	37296	37296	114	120	342	2- CCC (PLL/DLL) (CCC_SE)

Table 1-13. PolarFire® and RT PolarFire FPGA Clock Region Connectivity (continued)

Device	Region	FPGA Fabric Resource Count						
MPF300T	Top	96	80652	80652	248	256	744	2- CCC (PLL/DLL) (CCC_NE)
	Bottom	96	60192	60192	186	192	558	2- CCC (PLL/DLL) (CCC_SE)
MPF500T/RTPF500	Top	120	75468	75468	234	240	702	2- CCC (PLL/DLL) (CCC_NE)
	Middle	0	102516	102516	312	320	936	0
	Bottom	120	75468	75468	234	240	702	2- CCC (PLL/DLL) (CCC_SE)

Table 1-14. PolarFire® SoC FPGA Clock Region Connectivity

Device	Region	FPGA Fabric Resource Count						
		IO	4LUT	DFF	MATH	SRAM	URAM	CCC (PLL/DLL) (CCC_NE, CCC_SE)
MPFS250T	Top	96	80652	80652	248	256	744	2- CCC (PLL/DLL) (CCC_NE)
	Bottom	96	60192	60192	186	192	558	2- CCC (PLL/DLL) (CCC_SE)

See PolarFire Family Clocking Resources User Guide for more information about regional clock resource regions.

1.4.5. Transceiver Data Path Latency (Ask a Question)

Transceiver data path latency is defined, when the earliest serial bit appears in bus bit 0 for both transmit and receive interfaces of PMA mode. The PMA mode data path can be setup for different transmit and receive widths as needed. Clock phase latency is not considered.

Table 1-15. Transceiver Data Path Latency

PCS Mode	PCS Fabric Width	PMA PCS Width	Transmit Latency (UI)	Receive Latency (UI)
PMA	8	8	82	58.5
PMA	10	10	100	73.4
PMA	16	16	148	122.4
PMA	20	20	184	153.6
PMA	32	32[1]	292	250.5
PMA	40	40	364	313.5
PMA	64	32[1]	452	410.5
PMA	80	40	564	513.5
8b10b	40	40	404	314
8b10b	80	40	564	513.5

Note:

1. Use 32-bit PMA latency for minimum 64b66b latency. Maximum 64b66b latency calculation is described in 64b66b Latency Considerations.

Important: 8b10b mode uses 20, 40, and 80 PmaPcsWidth. PIPE modes uses 40 PmaPcsWidth.

1.4.5.1. 64b66b Latency Considerations (Ask a Question)

When integrating systems with the 64b66b encoding protocol, it is important to consider the potential for increased variable latency. The 64b66b gearbox, which is used to convert between the 64-bit data words used by the Physical Media Attachment (PMA) and the 66-bit encoded words, can introduce a variable latency of up to one fabric tx/rx clock period in both transmit and receive directions. This variability in latency is a result of the operation of the gearbox, which must handle the alignment and conversion between the differing word sizes.

The 64-bit interface of the PMA does not directly match the 66-bit blocks of the encoded data stream, necessitating this conversion process and resulting in the potential for one clock cycle of latency variation. The following examples can be used when using the 64b66b PCS XCVR interface:

- Rx latency delay $= \frac{250.5}{10.3125\ Gbps} \pm \left(\frac{1}{161.13\ MHz}\right)$
- Tx latency delay $= \frac{292}{10.3125\ Gbps} \pm \left(\frac{1}{161.13\ MHz}\right)$

Important: For example, 10GBASE-R uses 32-bit fabric width. 161.13 MHz is used for fabric interface clock, which is 66-bit bus width.

1.4.6. Transceiver Clocking Use Cases (Ask a Question)

Each transceiver quad can source a global clock directly. Transceiver designs should use regional clocks for the interface logic when possible. This reduces over use of global clocks. In many cases, transceiver designs can share global clocks when multiple interfaces are used, depending on protocol requirements. Only one global clock is supported per transceiver quad. See respective PolarFire FPGA Datasheet, RT PolarFire FPGA Datasheet, or PolarFire SoC Datasheet for AC performance information. For information about connectivity of the transceivers to the global clock network, see PolarFire Family Clocking Resources User Guide.

The following table lists the transceiver interface clocking use cases in the Libero SoC software, which uses presets per protocol. See PCS/FPGA Fabric Interface for explanation of system clock source modes.

Table 1-16. Transceiver Interface Clocking Use Cases

Preset	Width	Rx	Tx[1]	System Clock Source[1]
10GBASE-R	x1	Regional	Global	Global from XCVR Tx
10GBASE-R	Multiple	Regional	Global shared	Global from XCVR Tx shared
10GBASE-KR	x1 and Multiple	Regional	Regional	
SGMII/1000BASE	x1	Regional	Regional	Global from XCVR Tx
SGMII/1000BASE	Multiple	Regional	Global shared	Global from XCVR Tx shared
JESD204B	x1	Regional	Global	Global from XCVR Tx shared
JESD204B	xN	Regional	Global shared	Global from XCVR Tx shared
CPRI	x1	Regional	Regional	Global
CPRI	xN	Regional	Regional	Global shared
Interlaken	xN	Regional ≥ Global[2]	Global shared	Global from XCVR Tx shared
XAUI	x4	Regional	Global shared	Global from XCVR Tx shared
RXAUI	x2	Regional	Global shared	Global from XCVR Tx shared
SDI[3]	x1	Global	Global	Global
SDI[3]	Multiple	Global	Global shared	Global
LiteFast[3]	x1	Global	Global	Global from XCVR Tx and Rx
LiteFast[3]	xN	Regional ≥ Global[2]	Global shared	Global from XCVR Tx (shared) and Rx
LiteFast[3]	x1 and Multiple	Global	Global shared	Global from XCVR Tx (shared) and Rx (per interface)
QSGMII	x1	Regional	Regional	Global 125 MHz
QSGMII	Multiple	Regional	Global shared	Global 125 MHz Shared
SATA[3]	x1	Global Tx	Global Tx	Global Tx
SATA[3]	Multiple	Global Tx	Global Tx	Global Tx (not shared)
SRIO	x1	Regional	Global	Global

Table 1-16. Transceiver Interface Clocking Use Cases (continued)

Preset	Width	Rx	Tx[1]	System Clock Source[1]
SRIO	xN	Regional	Global shared	Global
SRIO	Multiple	Regional	Global (per interface)	Global
Fiber Channel	x1	Regional	Global Tx	Global Tx
Fiber Channel	Multiple	Regional	Global Tx shared	Global Tx Shared

(1) Shared implies that multiple lanes use common clock resources.
(2) Uses regional clock and moves to global clock resources in the FPGA fabric.
(3) Typically, these interfaces are implemented uni-directional. For full duplex, the RX interface and TX interface clock can not be global or global (Shared) at the same time as only one global clock is supported per transceiver quad. If using the RX and TX both as global is required by design, the design must use two separate XCVR configurations (instances), one in RX half duplex mode and other in TX half duplex mode.

1.5. Transceiver Clocking (Ask a Question)

Clocking of transceivers use several dedicated resources that are embedded in each device. All XCVR designs require a XCVR_REF_CLK to provide and clock input to a XCVR_TXPLL, which provides the necessary clocks for the XCVR_LANE or XCVR_LANES. The XCVR_TXPLL synthesizes the input reference clock to generate the high-speed serial clock used in the transmitter PMA. XCVR_REF_CLKs and XCVR_TXPLLs are shared and used for several high-speed serial protocols.

1.5.1. Transmit PLL (Ask a Question)

Two variations of the transmit PLLs embedded within the transceiver lanes are available based on protocol requirements. Both TxPLLs use ring VCO-based PLLs. Using a combination of TxPLL ranges and post-dividers produce frequencies across the entire supported range of the device. The transmit PLLs include one type with spread-spectrum (SSCG) generation modulation capabilities (TXPLL_SSC) and another type without SSCG capabilities (TxPLL). Both types of transmit PLLs use the same half-rate, fractional-N type (Frac-N) architecture design, thereby relaxing the speed requirements of the phase detector and frequency dividers. This consequently expands the VCO tuning range and enhances the phase noise performance while having a significant impact on the total power. The transmit PLL phase detector provides a valid output while driving a full-rate random data stream on both edges using the half-rate clock. All transmit PLLs support a jitter-attenuator option. The jitter attenuator is used to track the data rate of any noisy reference clock with a clean input reference clock to provide a 0 ppm offset from the noisy reference clock while providing a jitter-cleaned output.

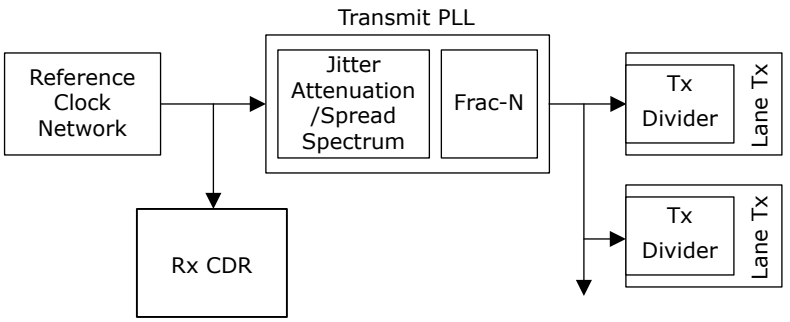
Each transceiver lane can select a transmit clock from the transmit PLLs that are close enough to drive their half rate clock (Figure 1-38). The PLL uses the input reference clock to generate a serial bit clock (at half the rate). The transmit PLL detects and signals a loss of lock in the event that the reference clock stops toggling or when the reference clock transitions to an incorrect frequency.

There are also instances that include additional transmit PLLs, which can be used by the local transceiver quad and in a subset of lanes of adjacent quads.

The output frequency of each transmit PLL is derived automatically from the reference clock frequency and the settings for the PLL multipliers. Each transmit lane can then divide this base transmit PLL rate per lane using the post-divider by 1, 2, 4, 8, or 11. The resulting frequency is half the bit rate based on the transmit half-rate architecture. For example, a 2.5 GHz clock is used for a 5 Gbps transmit transceiver line rate. The programmable multipliers are defined and programmed by the Libero transceiver interface configurator as per the desired protocol.

In addition, the transmit PLL can also provide the system clock for the FPGA logic.

Figure 1-38. Transmit PLL



Two different types of transmit PLLs can be used with the transceiver based on half-rate architecture. Both PLL types have identical analog portions with only digital logic differences, therefore each PLL type has identical performance.

Q#_TXPLL_SSC: This PLL operates in the 1.6 GHz – 6.4 GHz frequency range and can provide a transmit bit clock to a transceiver quad. The TxPLL_SSC supports jitter attenuation for loop-time applications. Unique to this PLL is the spread-spectrum clocking (SSC) generation support, which can generate a saw-tooth clock with various options.

For each quad, there is one TxPLL_SSC that can only be used by lanes within that quad.

Q#_TXPLLn: There are two of these PLLs within the transceiver quad location, TxPLL0 and TxPLL1. This type of PLL also supports the full 1.6 GHz – 6.4 GHz frequency range and can drive the transmit bit clock pair of adjacent transmit lanes both above and below the PLL. This PLL also supports jitter attenuation, but does not provide SSC support. There are also transmit PLLs, which can be used by the local transceiver quad and in a subset of lanes of adjacent quads. See Figure 1-56 and Figure 1-57 for more information about PolarFire FPGA TxPLL sharing. See Figure 1-54 for more information about PolarFire SoC FPGA TxPLL sharing.

Note: Both types of TxPLLs accept spread-spectrum input.

The jitter attenuation feature uses digital filtering within the transmit PLL to remove the unwanted noise of a reference clock across a wide frequency band. The low-jitter output is sent to an oscillator that is numerically controlled to adjust the phase and frequency relationships to achieve a 0 ppm offset from the original noisy reference clock.

Table 1-17. Transmit PLL

PLL Type	Rate	Details
Q#_TXPLL_SSC[1]	1.6 GHz – 6.4 GHz	PLL is used within the quad only. This PLL supports jitter attenuation and SSC.
Q#_TXPLL0[1] Q#_TXPLL1[1]	1.6 GHz – 6.4 GHz	PLL can be used by a pair of adjacent transmit lanes within each of the immediately adjacent transceiver quads (a total of four lanes). This PLL does not have SSC capability, but does support jitter attenuation.

(1) Q# = Transceiver quad identifier (Q0, Q1, and so on.)

1.5.2. Spread Spectrum Clocking (Ask a Question)

TxPLL produces spread spectrum clock generation (SSCG). SSCG uses a modulated output clock signal to reduce peak EMI. The lowering of peak EMI enables significant reduction in expensive shielding cost or reduce interference with other sensitive circuits. By modulating the PLL, the resulting spectrum at each clock harmonic is made broad-band or flattened, and reduced in amplitude from 10 db to 20 dB, depending on frequency and modulation amplitude.

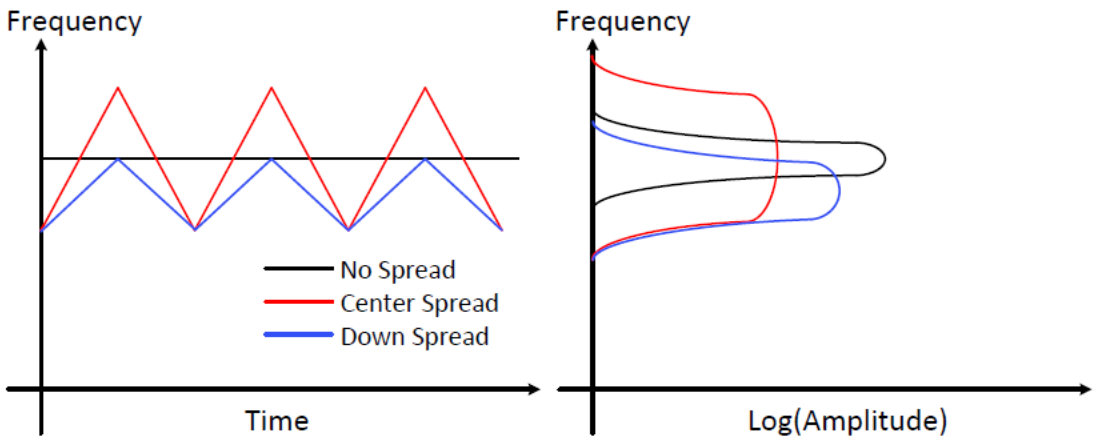
TxPLL is configured in Fractional-N mode to preserve the accuracy of modulation to the targeted modulation frequency. The SSCG works by modulating the feedback divider value (Divval) of the TxPLL, thus modulating the PLLs output frequency and introducing a noise source or wave table.

Setting the modulation mode (Center versus Down) and modulation amplitude depend on the amount of EMI reduction desired and the timing margin for circuits running on the spread clock domain.

SSCG is optionally implemented using the Libero TxPLL configurator by setting the modulation frequency, modulator spread mode (Center and Down spread), Spread/Divval, and Wave table. See Libero Configurators.

The following figure shows the frequency versus time and the resulting amplitude in the frequency domain.

Figure 1-39. Spread Spectrum Clocking Modulation Mode



1.5.3. Transmit Lane Alignment (Ask a Question)

Applications like Serial RapidIO, XAUI, DisplayPort, Interlaken, and JESD204B need transmit alignment across multiple lanes. Transmit lane alignment depends on the number of lanes, total skew, fabric clock frequency relative to the line rate, and number of TX PLLs. The method of alignment involves launching a reset from the shared PLL to each TX lane after the PLLs are locked.

The transceiver PMA quad supports transmit lane alignment for upto four lanes using the following quad-based reset methods:

- Q#_TxPLL_SSC PLLs and its four lanes (quad) as shown in Figure 1-40.
- Q#_TxPLL[1:0] and two lanes each from the adjacent quads as shown Figure 1-41.

The transmit clock for each transceiver lane can be driven by an external or a quad PLL. The reset for the TX lanes is based on the PLL selected. This reset is connected within the transceiver block to travel with the transmit clock. All of the TX lanes for a given TX clock are reset by asserting the internally connected signal TX PLL CLKRESET. To ensure a proper reset, alignment circuitry stops the Tx clock for four clock cycles. With this quad-based reset method, very low skew is achieved. In this scenario, all lanes are within the same QUAD using the same TXPLL. The TXPLL initially align the lanes internally with PLL lock. This occurs prior to the assertion of the PLL_LOCK output pin of the TXPLL. Any subsequent LANE#_PMA_ARSTN assertions will reset the post-dividers for the lane, which can cause a misalignment after the initial alignment that was completed by the CLK_RESET from the TXPLL.

The reinitialization of this lock and alignment mechanism can also be initiated through DRI register control, depending on TXPLL type. Register toggling (1=>0=>1) after initial power-up of PMA_CMN/TXPLL_CTRL/TXPLL_CLKRESET or EXTPLL/EXTPLL_CTRL/EXTPLL_CLKRESET is required. This reset operation requires PMA_CMN/TXPLL_CTRL/TXPLL_CLKRESETEN or EXTPLL/EXTPLL_CTRL/EXTPLL_CLKRESETEN = 1. This action disrupts the TxPLL and re-align the Tx lanes.

Libero SoC software includes support to automatically perform Tx Lane Alignment at start-up. The option is included in the TX PLL configurator to enable or disable (by default, it is disabled) Tx Lane alignment. Enabling this feature automatically generates the initializing instructions to perform alignment after XCVR_INIT_DONE is asserted from the PF_INIT_MONITOR. The Tx Lane Alignment option can be enabled only when Tx PLL is configured in Normal mode. PCIe lanes do not require Tx lane alignment, and it is limited to XCVR lanes configured in any other PCS mode.

Figure 1-40. Using TXPLL_SSC Upto Four Lanes

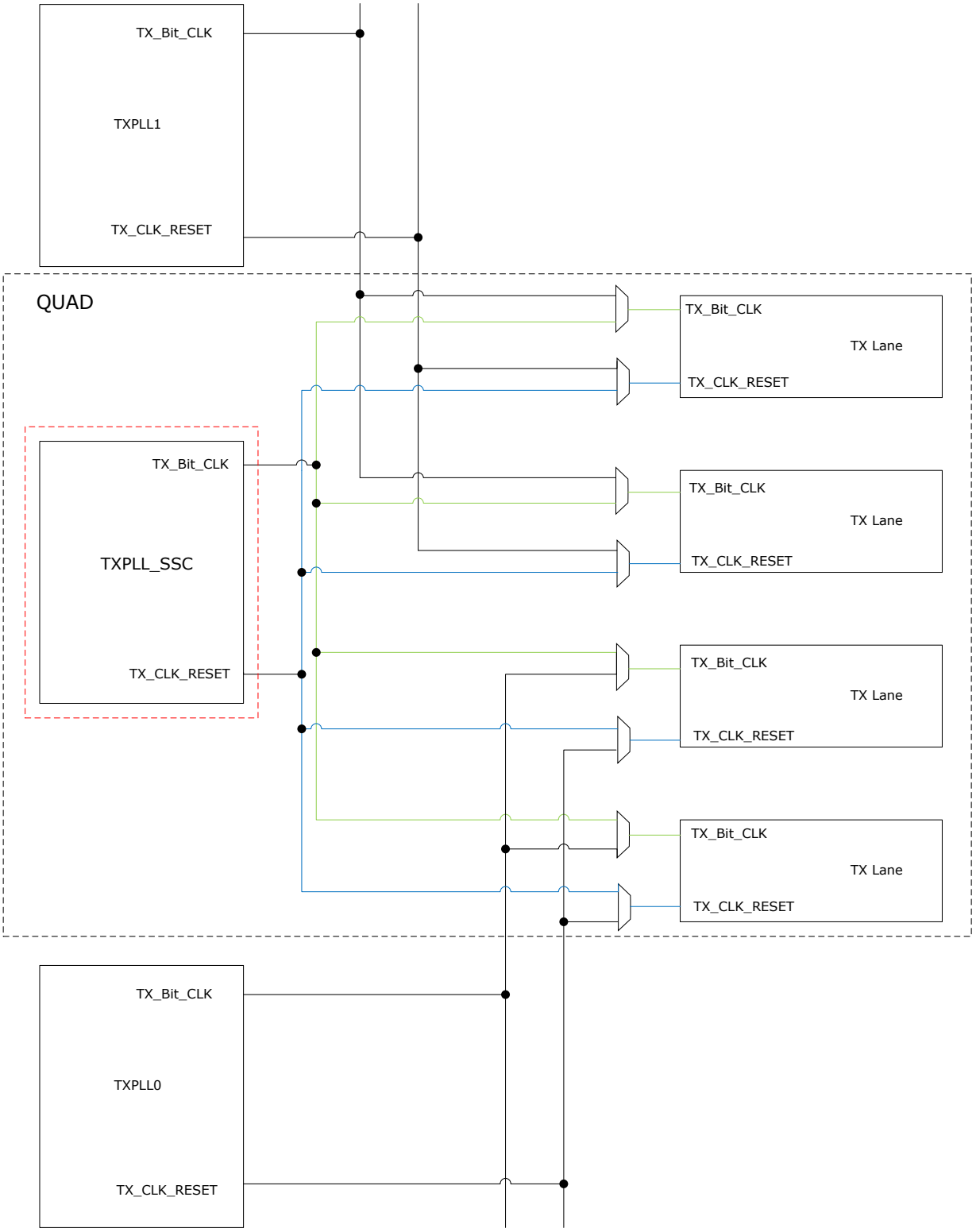
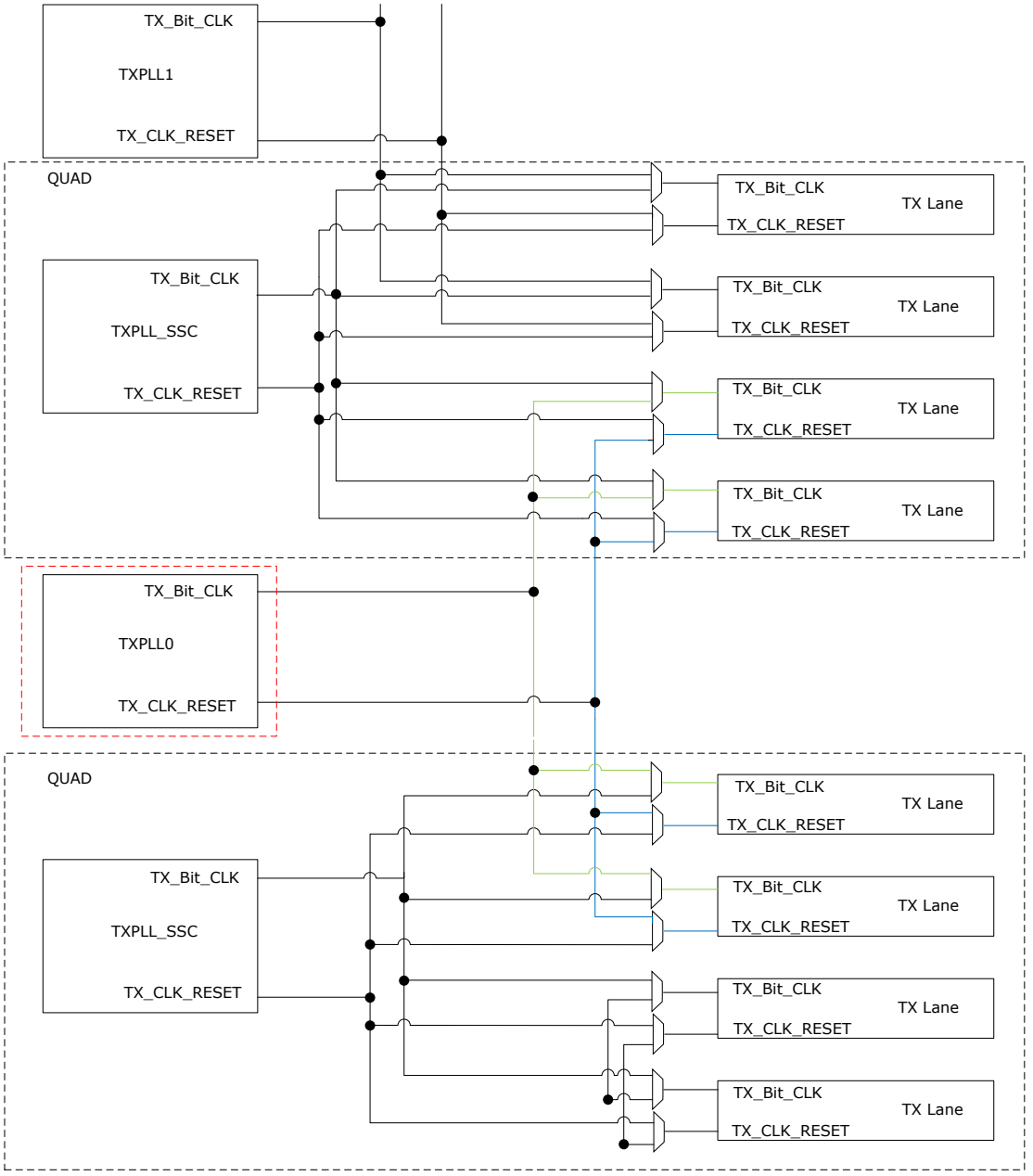


Figure 1-41. Using TXPLLs Upto Four Lanes

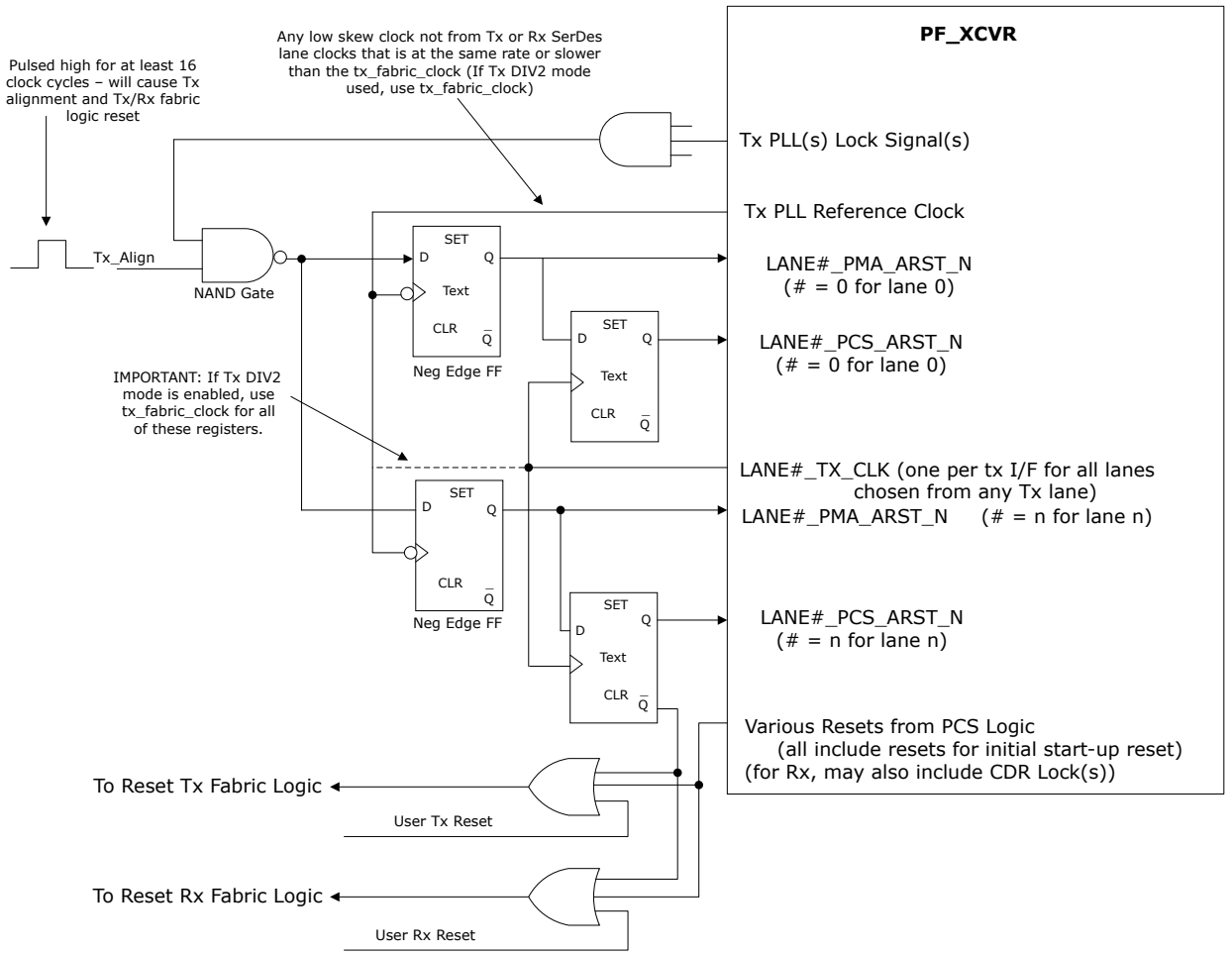


In the following scenarios, the quad based reset of TX lanes cannot be used:

- When one transmit clock is used for multiple lanes that do not belong to the same quad. In this scenario, per lane reset of TX lanes must be used because the reset operation on one lane does not affect the other lanes.
- When an arbitrary number of lanes more than four (5 to 8) need the transmit alignment.

In these scenarios, the transceiver uses two PLLs with the same reference clock and a separate fabric logic for reset of the TX lanes as shown in the following figure.

Figure 1-42. FPGA Logic For TX Alignment (5 to 8 Lanes)



The fabric reset logic per lane is provided by asserting the LANE#_PMA_ARST_N signal (where # represents the lane to be reset). The RESETSEREN# register must be set to reset each lane. To achieve proper skew alignment, the rising and falling edges of the RESETSER# signal must arrive at the transceiver PMA lane pins within two high-speed bit clocks. This can be implemented by a known low-skew internal clock signal using the global or local FPGA routing clocks to Flip-Flop inside the PCS. Any arbitrary low-skew clock can be used as a reference clock to the transceiver Tx PLLs. The Tx_Align pulse needs to be asserted for 16 clock cycles after the PLLs are locked and transmit lanes are programmed with the same post-PLL divide factors. If this requirement is met, all of the TX lanes used for the given transceiver port are aligned within 2-bit clock user interfaces.

1.5.4. Transceiver Clocks (Ask a Question)

The transceiver transmitters have high-performance bit clocks running at half the line rate of the fastest transmit lane driven by the clock. The transmit PLLs generate these clocks based on a transmit reference clock, with the configuration set in the Libero design software.

Within each lane, the transmit bit-rate clock (Figure 1-5) is divided by 1 (full rate), 2, 4, 8, or 11 to set the transmit rate of a given lane. The resulting clock is further divided by 8, 10, 16, 20, 32, 40, 64, and 80 to generate a parallel transmit word clock to the fabric.

The transceiver receivers have their own per-lane receive PLL built into the CDR to generate a per-lane receive clock supporting asynchronous data in that lane. Generally, the CDR is used in lock-to-data mode. The receive CDR PLL initially spins up to approximate the correct frequency

to lock to the incoming data by first locking to an input receive reference clock that is near the incoming data rate. Once that is achieved, it then switches to clock recovery mode where it locks to the incoming data and then extracts the clock from the incoming data (which is also a half-rate bit clock running at half the speed of the received data rate). Lock-to-reference is also available for customized protocols. The CDR PLL locks to the local input reference clock and spins to the desired frequency without performing phase compensation or clock recovery functions. These applications pass the data directly to fabric where it can be used for custom over-sampling and synchronizing processing.

The per-lane CDR extracts a clock from the incoming data stream and then generates a receive parallel word clock that is divided by 8, 10, 16, 20, 32, 40, 64, or 80 from the bit rate of the given lane.

1.5.4.1. Transceiver Reference Clock Interface (Ask a Question)

The reference clock interface block to the transceiver provides multiple options for supplying the reference clock input to the transceiver transmit and receive PLLs (Figure 1-43). Various sources can provide the reference clock interface through REFCLK0 and REFCLK1 ports.

- **Differential dedicated input pad**: allows a direct clock input of a low-jitter reference clock through a LVDS/HCSL input pins REFCLK_P/N.
- **Single-ended dedicated input pad(s)**: allows the selection from two different single-ended clock inputs, enabling the transmit PLL to select from two different clock sources. Separate single-ended clock inputs allow unrelated transmit and receive clock sources to be sourced to the transceiver.

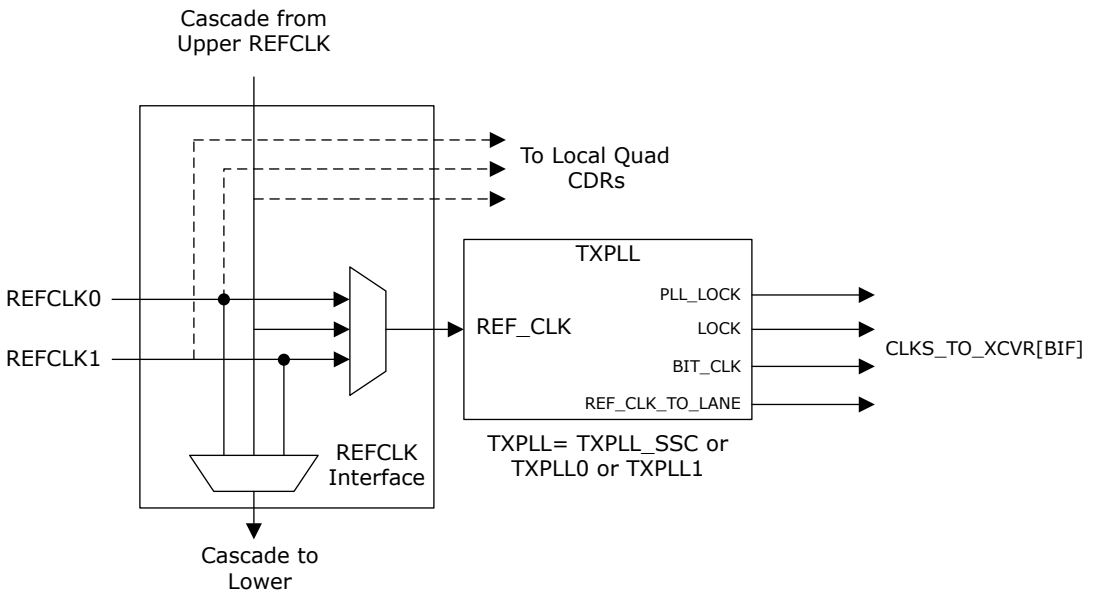
Important: Two separate single-ended inputs allow one for the transmit reference clock and a second for the receive reference clock.

- **Cascade of a reference clock**: the clock received on the external pins of a quad can be driven to the quad below. The reference clock interface provides cascading of an input reference clock path from the REFCLK pins of one transceiver quad to the TxPLLs and receiver CDRs of other transceivers. In designs that have lanes spanning different transceiver quads, the cascading clocks eliminate the need to connect the on-board reference clock sources to the REFCLK pin of each transceiver quad. The reference clock interface also drives a clock signal on the REFCLK pins to the clock logic in the FPGA fabric.
- **Recovered clock**: allows for the reference clock to be sourced by the recovered clock from local quad (through JA_REF_CLK). Within a Quad, the recovered clock used as JA_REF_CLK has dedicated connections between the lanes without any additional jitter.
  For inter-quad or using the recovered clock from one Quad to the JA_REF_CLK of another Quad, this has added jitter from using routing of the FPGA fabric. Since this clock is from the noisy digital VDD/VSS domain on the device, the reference clock jitter is higher than the dedicated inputs. Generally used with the jitter attenuation feature.

Important: The reference clock driving the transmit PLL REF_CLK also serves each lane for the receive CDR. Since each receive lane has an independent PLL, it is possible to use an independent reference clock per receive lane. PLL for the CDR is per transceiver PMA lane.

The PF_XCVR_REF_CLK does allow the FAB_REF_CLK output pin to drive a fabric global resource. This connection is limited to one per quad and if used, it prevents using the global of a XCVR lane in the same quad. However, you can connect XCVR_ FAB_REF_CLK output pin to PLL and one lane to global at the same time. For information about implementation of this connection, see Transceiver Reference Clock Configurator.

Figure 1-43. Reference Clock (REFCLK) Interface to Transmit PLL



The following table lists the transmit PLL pins.

Table 1-18. Transmit PLL Pin List[1]

Name	Direction	Description
REF_CLK	Input	Transmit PLL input clock from reference clock interface (Figure 1-43).
PLL_LOCK	Output	PLL_LOCK is the PLL lock indicator signal that can be used to drive logic in the fabric. It is a fabric routed signal. (High = LOCK).
CLKS_TO_XCVR	Output	CLKS_TO_XCVR is a bus interface port (BIF) with three outputs that are required to be interconnected between the TXPLL and the XCVR components. The BIF includes BIT_CLK, LOCK, and REF_CLK_TO_LANE outputs.
BIT_CLK[2]	Output	High-speed clock to lane.
LOCK[2]	Output	Connected to the TX_PLL_LOCK_0/1 input port, which is a part of the CLKS_FROM_TXPLL_0/1 BIF on the XCVR block. It is a hardwired connection.
REF_CLK_TO_LANE[2]	Output	Connected to the TX_PLL_REF_CLK_0/1 input port, which is a part of the CLKS_FROM_TXPLL_0/1 BIF on the XCVR block. It is a hardwired connection. TXPLL reference clock that is passed to the XCVR lane clock (used for simulation only)
CLK_125	Output	Exposed directly on the TXPLL block. Its frequency is fixed at 125 MHz. It is exposed only when the TXPLL BIT_CLK is 2500 Mbps, and must be used for a PCIe use case.

(1) Pin list for both Q#_TxPLL[1:0] and Q#_TxPLL_SSC PLLs.
(2) Port is a part of the bus interface port (BIF) CLKS_TO_XCVR.

There are some use-cases which can allow a XCVR quad's global output to be used to broadcast the SerDes REFCLK, if that XCVR is configured to use regional clocks for the TX/RX clocks allowing these specific use-cases to broadcast the SerDes REFCLK into the FPGA fabric with a more predictable amount of clock jitter.

1.5.4.2. Broadcasting REFCLK to the FPGA Fabric (Ask a Question)

In the PolarFire family devices, you can broadcast the external XCVR reference clock (REFCLK) signal into the FPGA fabric using a hardwired global routing resource. This is possible when the following conditions are met:

1. The selected XCVR quad (Q#) is not configured to use global output clocks. This ensures that the global output is available for broadcasting the REFCLK. Use regional clock outputs for XCVR related clock outputs from the selected quad.
2. The external REFCLK enters the device using a XCVR_#A_REFCLK_P/N input pin (not XCVR_#B_REFCLK nor XCVR#C_REFCLK pins)
   - Assigning the REFCLK input to the XCVR_#A_REFCLK_P/N input ensures that the selected XCVR_REF_CLK corresponds to a location associated with the TXPLL_SSC in the XCVR quad.
   - The XCVR_REF_CLK associated with a quad's TXPLL_SSC has additional connectivity to the fabric global clocking resources through the east Interface Clock Block (ICB), as described in Reference Clock Input Pins and in the "Global Clock Network" section of the PolarFire Family Clocking Resources User Guide.
   - The XCVR_REF_CLK can simultaneously broadcast the REFCLK to the FPGA fabric and drive either the selected quad's TXPLL_SSC or a Q#_TXPLL#, per allowable placements in the Libero SoC I/O Editor XCVR View.
3. Instantiate and connect a CLKINT_PRESERVE macro to the XCVR_REF_CLK hardwired output port REF_CLK. This ensures that the hardwired connection is preserved during Synthesis optimizations.

The following table lists the conditions that are listed in the preceding list and the resulting Global Net Report.

Table 1-19. Selecting XCVR_#A_REFCLK_P/N Inputs on a XCVR QUAD# Using Package Pin Assignment Table

Package Pins	MPF300T/MPF300TS-FCG1152 Pin Names
AF29	XCVR_2B_REFCLK_P
AF30	XCVR_2B_REFCLK_N
AE27	XCVR_2A_REFCLK_P
AE28	XCVR_2A_REFCLK_N
AC27	XCVR_2C_REFCLK_P
AC28	XCVR_2C_REFCLK_N

The following figures show the conditions that are listed in the preceding list and the resulting Global Net Report.

Figure 1-44. Using Libero SoC I/O Editor XCVR View to Place the REFCLK, TXPLL, and XCVR Q#

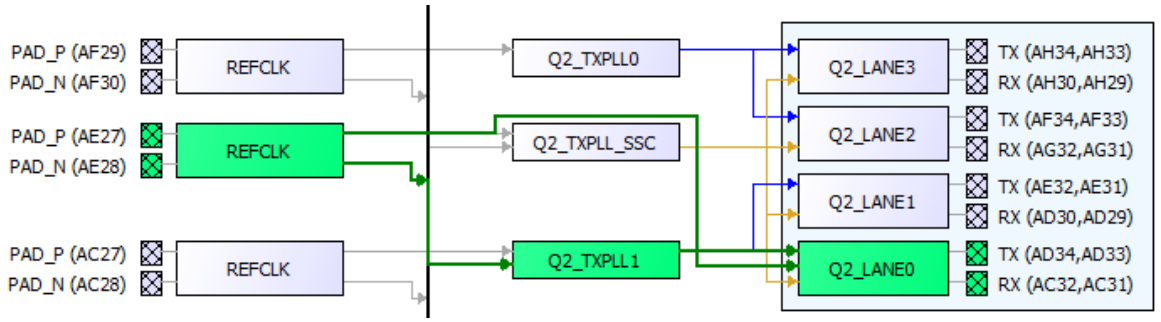


Figure 1-45. Schematic View Showing XCVR_REF_CLOCK Driving XCVR/TXPLL and Fabric Logic (through CLKINT_PRESERVE)

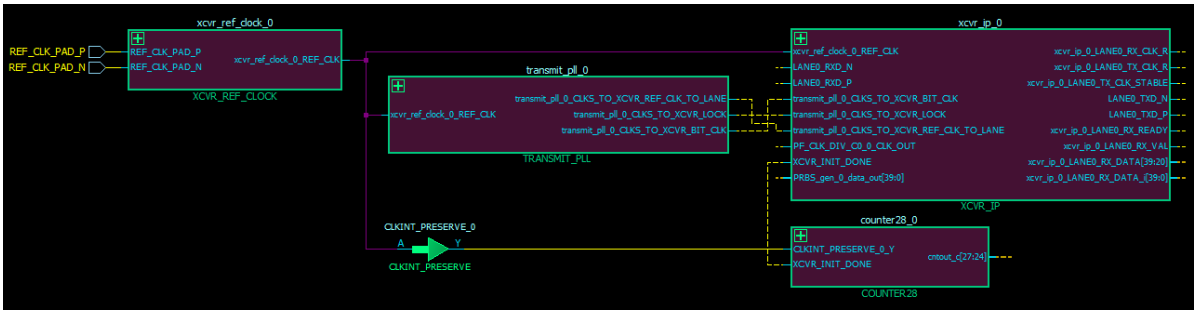


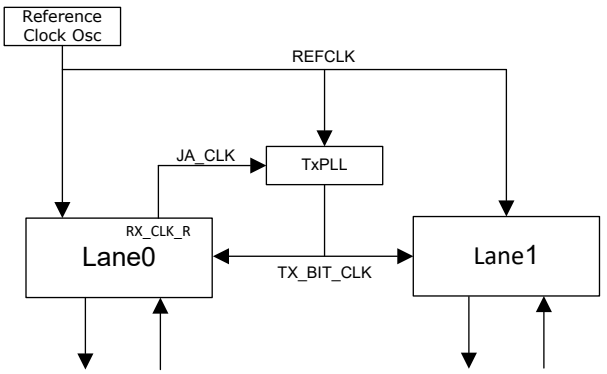
Table 1-20. Libero SoC Global Net Report Confirming the Hardwired Connection from XCVR REFCLK to Fabric CLKINT

From	From Location	To	Net Name	Net Type	Fanout
OSC_0_0/OSC_0_0/ I_OSC_160:CLK	(512, 2)	PF_CLK_DIV_C0_0/ PF_CLK_DIV_C0_0/ I_CD_RNIF7Q3/U0	OSC_0_0_RCOSC_160MHZ_CLK_DIV	HARDWIRED	2
xcvr_ref_clock_0/ xcvr_ref_clock_0/ I_IO:REFCLK_0	(2466, 371)	CLKINT_PRESERVE_0/U0	xcvr_ref_clock_0_REF_CLK	HARDWIRED	3
OSC_0_0/OSC_0_0/ I_OSC_160:CLK	(512, 2)	OSC_0_0/OSC_0_0/ I_OSC_160_INT/U0	OSC_0_0_RCOSC_160MHZ_CLK_DIV	HARDWIRED	2

1.5.4.3. Jitter Attenuator (Ask a Question)

All transmit PLLs support a jitter-attenuator option. The jitter attenuator is used to track the data rate of any noisy reference clock with a clean input reference clock to provide a 0 ppm offset from the noisy reference clock while providing a jitter-cleaned output. It is used in designs requiring loop-timing where the recovered clock is used as the transmit clock requiring low jitter performance. This is used in applications such as Synchronous Ethernet (SyncE) providing low jitter synchronization to an Ethernet network. The following figure illustrates the application using jitter attenuation.

Figure 1-46. Typical Jitter Attenuator Application Scheme



The TXPLL in the transceiver can provide jitter attenuation. Its design attenuates input jitter by using Libero configured settings to adjust the loop bandwidth and damping factors. The TXPLL input from the recovered source of the RX_CLK is cleaned from jitter components in the JA_CLK input path of

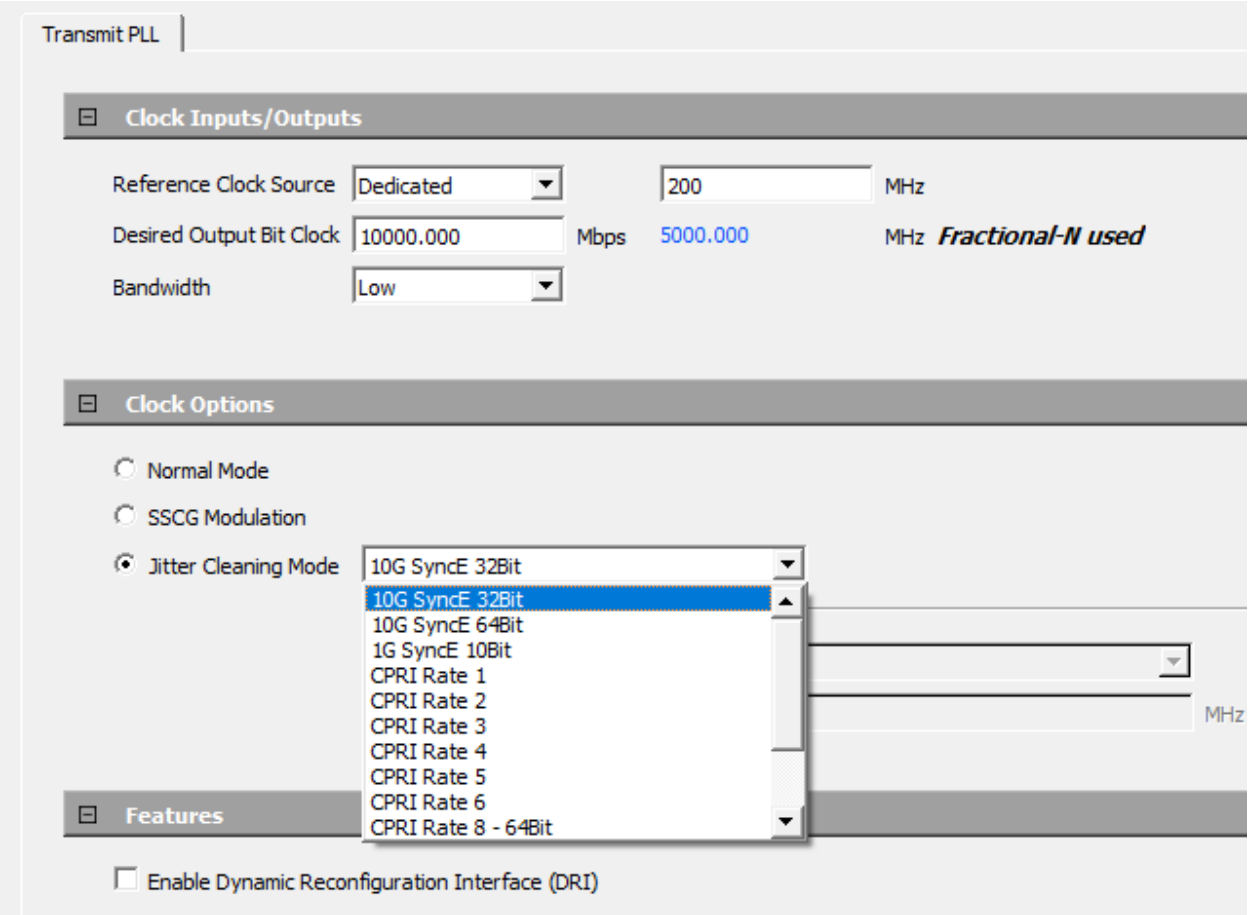
the TXPLL. The TXPLL configurator allows pre-determined configurations of the attenuation feature based on protocol including the following protocols.

- 10G SyncE (32-bit, 64-bit)
- 1G SyncE 10-bit
- CPRI Rate [1-8]
- SDI (3G, HD, SD)

JA_PLL demonstrates compliance with the 10G SyncE the ITU-T G.8261 and G.8262 standards. A 10G SyncE Characterization Report and Libero Reference Design solution is available.

Select **Jitter Cleaning Mode** under the **Clock Options** and choose the targeted protocol templates from the drop-down as shown in the following figure.

Figure 1-47. Jitter Attenuation TXPLL



Jitter attenuation PLL presets can be selected with TX PLL configurator (and the enable JA_CLK port in XCVR configurator). Jitter attenuation presets must follow the predefined requirements for the application design.

For example, if **SDI 3G** is selected in the TX PLL configurator, then set the PCS-FABRIC width in XCVR configurator to 20-bit and use the defined reference clock frequency.

Table 1-21. Jitter Attenuation PLL Presets

Jitter Cleaning Mode Selection	Data Rate (Mbps)	Reference Clock Frequency (MHz)	PCS - Fabric bit Width	Mode
10G SyncE 32Bit	10312.5	156.25	32	64b66b

Table 1-21. Jitter Attenuation PLL Presets (continued)

Jitter Cleaning Mode Selection	Data Rate (Mbps)	Reference Clock Frequency (MHz)	PCS - Fabric bit Width	Mode
10G SyncE 64Bit	10312.5	156.25	64	64b66b
1G SyncE 10Bit	1250	125	10	PMA
CPRI Rate 1	614.4	122.88	32	8b10b
CPRI Rate 2	1228.8	122.88	32	8b10b
CPRI Rate 3	2457.6	122.88	32	8b10b
CPRI Rate 4	3072	122.88	32	8b10b
CPRI Rate 5	4915.2	122.88	32	8b10b
CPRI Rate 6	6144	122.88	32	8b10b
CPRI Rate 8 64-Bit	10137.6	122.88	64	64b66b
SDI 3G	2970	148.5	20	PMA
SDI HD	1485	148.5	10	PMA
SDI SD	270	148.5	10	PMA
Custom Protocol Settings	See the following details.			

1.5.4.3.1. Custom Protocol Settings (Ask a Question)

Full duplex jitter attenuation solution is available for custom protocols in Libero SoC v12.3 or later. This support is for one refclk and one data rate allowing user the flexibility to create designs requiring customized settings not already supported by the presets.

The user must enter the reference clock frequency and clock source information for the correct generation of the jitter cleaning configuration. The reference clock source selection determines the input clock ports exposed on the component (See Figure 1-49). Users are required to connect the ports in the top-level to the correct XCVR lane. The use of the JAPLL with custom settings require the user to use the JA clock frequency for the associated lane as it appears in the RX JA clock frequency dialogue box of the XCVR configurator (See Figure 1-50).

Figure 1-48. JAPLL Custom Protocol Settings

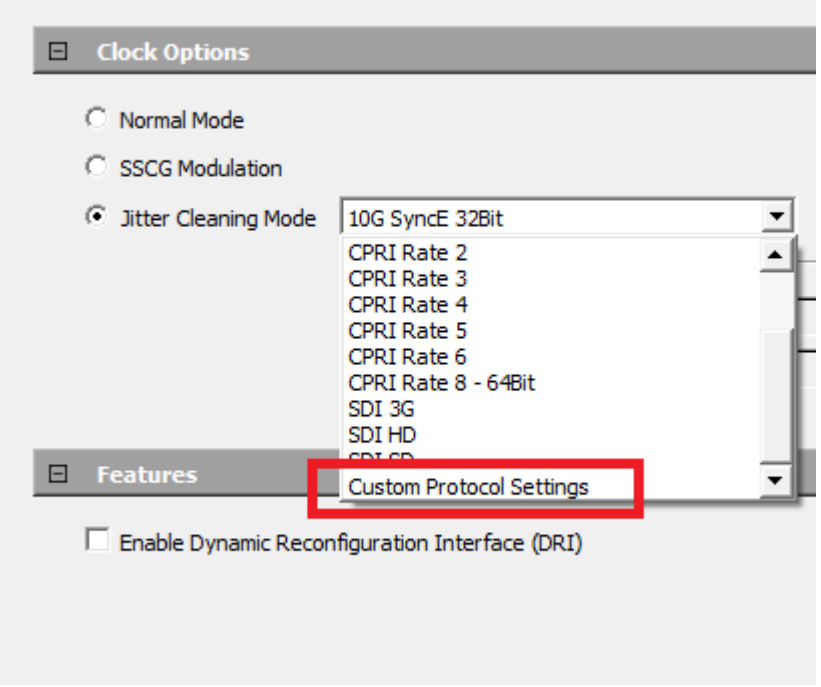


Figure 1-49. Reference Clock Source Options

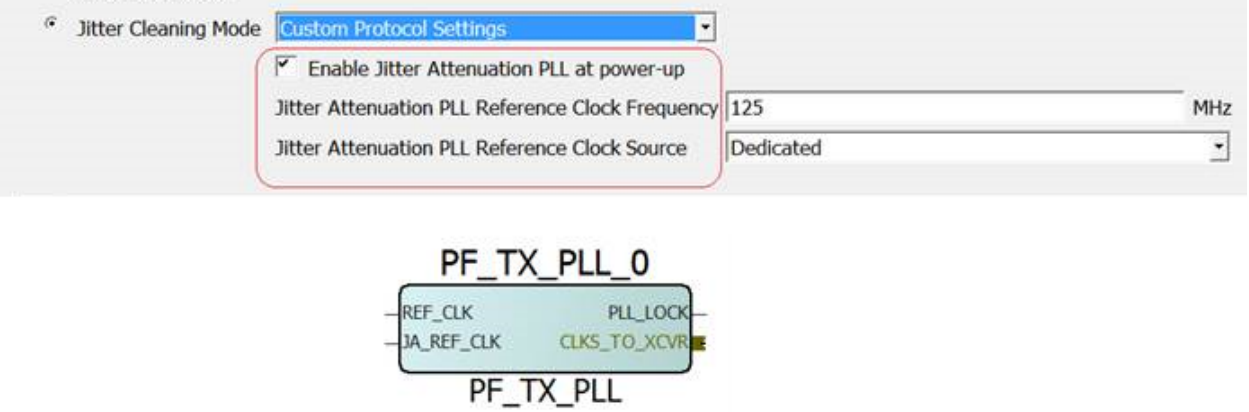
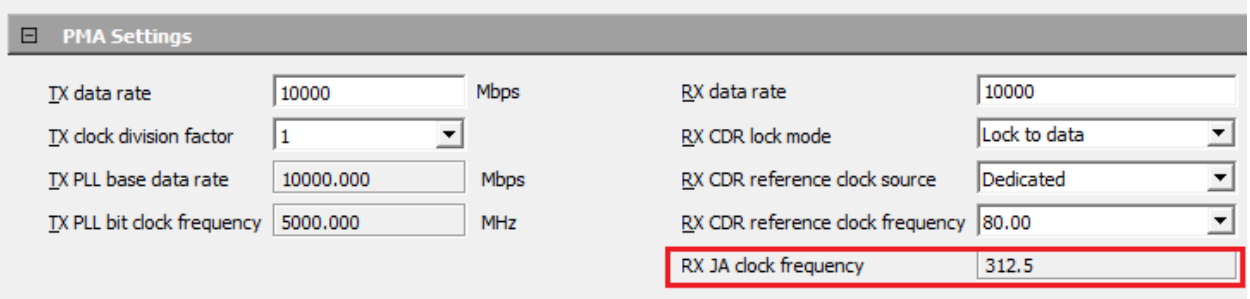


Figure 1-50. RX JA Clock Frequency (XCVR Configurator)

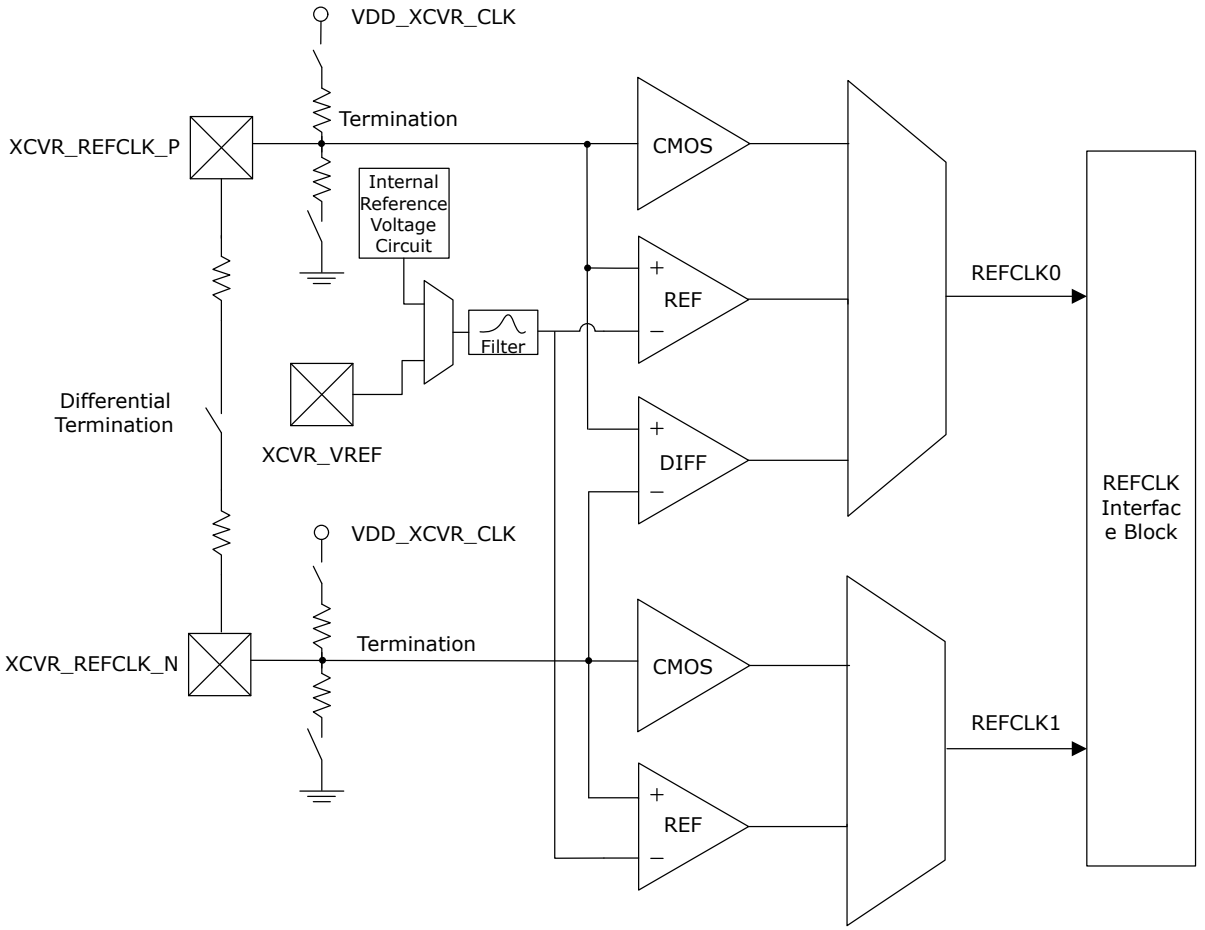


1.5.4.4. Dedicated Reference Clock Input Pins (Ask a Question)

For every transmit PLL within the transceiver PMA, there is a reference input pin pair for an external input of reference clocks to the device, as shown in the following figure. The reference clock inputs provide flexibility to interface with both single-ended and differential clocks and can drive up to two independent clocks per transceiver quad. The reference clock inputs has a single power supply (VDD_XCVR_CLK) that is shared across all reference clock buffers. These reference clocks can also be sourced for the global and regional clock networks in the FPGA fabric of the devices.

The following figure provides a detailed view of the dedicated reference clock.

Figure 1-51. Dedicated Transceiver Reference Clock Inputs



Note: For more information on reference clock interface block, see Figure 1-43.

1.5.4.4.1. Differential Input (Ask a Question)

This mode supports differential inputs such as LVDS/HCSL. The differential reference clock is available on REFCLK0 (REFCLK1 is not available with differential input clock mode). The inputs include an optional on-die 100Ω differential termination resistor. By default, the differential input termination resistance is set in high-Z mode until programmed with Libero software. XCVR_VREF is not used for differential reference clock input signaling. XCVR_VREF is used for single-ended signals requiring a voltage reference, such as SSTL.

1.5.4.4.2. Reference Voltage Input (Ask a Question)

In this mode, the input interface supports two single-ended modes:

- **Local reference input**: A reference voltage is connected to the XCVR_REFCLK_N input, which is used for the clock connected to the XCVR_REFCLK_P input. The resulting reference clock is available on the REFCLK0 output (REFCLK1 output is unavailable with local reference input mode).
- **Global reference input**: This mode allows for two separate reference clock sources, where the XCVR_REFCLK_P input is compared with a global transceiver reference called XCVR_VREF and is output on REFCLK0. The XCVR_REFCLK_N input is simultaneously compared to the same XCVR_VREF reference signal, and the result is output on REFCLK1. In this scenario, each XCVR_REFCLK_P and XCVR_REFCLK_N pins can accept a single-ended clock source. The resulting clock signals are available on REFCLK0 and REFCLK1 outputs, respectively. Internal voltage reference circuitry is included when a reference voltage input is used. No external voltage is required to the reference voltage input (XCVR_VREF) pin, which is internally connected.

Note: There is one XCVR_VREF signal per device that can be sourced from an external pin or an internal reference voltage circuit. An internally generated voltage reference (VREF) is default. A user can optionally use an externally supplied VREF if desired.

1.5.4.4.3. Single-Ended CMOS Input (Ask a Question)

In addition, the XCVR_REFCLK_P/N pins can connect a single-ended CMOS clock signal to the REFCLK0 and REFCLK1 inputs. This allows two independent reference clocks to be applied to a XCVR REFCLK inputs. XCVR_VREF is not used for single-ended CMOS reference clock input signaling.

1.5.4.4.4. XCVR REFCLK Usage (Ask a Question)

Transceiver Reference clock input standards will default to the following configurations dependent on the selection of LVCMOS, Voltage Reference, or Differential selection in the REFCLK configurator.

Table 1-22. XCVR REFCLK Defaults

Reference Clock Mode	I/O Std	Resistor Pull	Schmitt Trigger	ODT	VDDI
LVCMOS	LVCMOS25	None	OFF	0	2.5
Voltage Reference	SSTL25I	None	OFF	0	2.5
Differential	LVDS25	None	OFF	100	2.5

The default can be changed using I/O PDC constraints (not available in IOEditor).

```
-io_std <iostd>
-ODT_VALUE <odt>
-RES_PULL <res_pull>
-SCHMITT_TRIGGER <schmitt>
-USE_EXTERNAL_VREF <true, false>
-POWER_SUPPLY <power Supply for all Ports>
-EXTERNAL_VREF <true/false>
```

Important:

- By default, the VDD_XCVR_CLK power supply is set to 2.5V. If 3.3V is used, add all ports in the PDC. If a port is not specified in the PDC, it takes the default settings. All REFCLK ports must be specified in the PDC to specify their location else the flow stops.
- The new options are case sensitive, the values are not.
- To turn OFF the odt, set the ODT_VALUE to 0.
- SSTL18I, SSTL18II, SSTL25I, and SSTL25II inputs optionally have a VREF pin to set.
- The PDC option is called USE_EXTERNAL_VREF <true/false> where the default is false to use the internal VREF pin.

Table 1-23. Reference Clock Input Buffer Standards[1]

Single-ended	Differential[2]	Reference Voltage (Not supported for ES/XT devices).
LVCMOS18 (VDDI = 2.5)	HCSL25[3]	HSUL18I (VDDI = 2.5)
LVCMOS25 (VDDI = 2.5)	LVDS25	HSUL18II (VDDI = 2.5)
LVCMOS33 (VDDI = 3.3)	LVPECL33 (VDDI = 3.3)	SSTL18I (VDDI = 2.5)
LVTTL (VDDI = 3.3)	MINILVDS25	SSTL18II (VDDI = 2.5)
—	MIPI25	SSTL25I (VDDI = 2.5)
—	MLVDS25	SSTL25II (VDDI = 2.5)
—	PPDS25	—
—	RSDS25	—

Table 1-23. Reference Clock Input Buffer Standards[1] (continued)

Single-ended	Differential[2]	Reference Voltage (Not supported for ES/XT devices).
—	SLVS25	—
—	SUBLVDS25	—
—	LVDS33[4]	—

(1) VDDI = VDD_XCVR_CLK
(2) Differential inputs do not include internal voltage-bias circuitry.
(3) LP-HCSL is supported with HCSL25 reference clock setting without 100Ω differential ODT.
(4) LVDS33 is supported in device with VDDI= 3.3 V and supported in Libero by selecting LVDS25 IO Standard configuration.

Users need to be aware of power supply requirements and voltage reference requirements. See PolarFire FPGA and PolarFire SoC FPGA User I/O User Guide for more information.

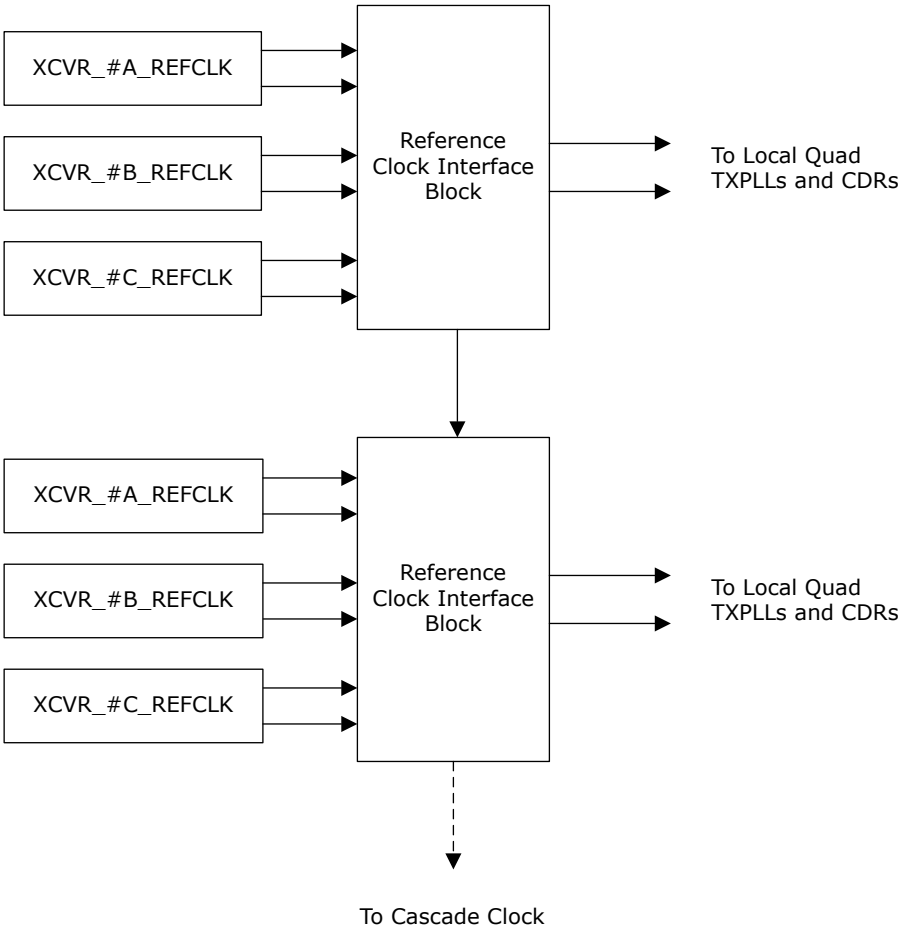
1.5.4.4.5. Reference Clock Input Pins (Ask a Question)

The input pins XCVR_REFCLK_P/N are assigned through the Libero transceiver configurator based on the targeted transceiver quad. The pins are identified based on quad. For example, where there are three transmit PLLs:

- XCVR_#A_REFCLK_P/N—This REFCLK input is associated with the connection dedicated to the TXPLL_SSC. Also, this input has connectivity to the Global clock resource.
- XCVR_#B_REFCLK_P/N
- XCVR_#C_REFCLK_P/N (This input is only available in a subset of transceiver quads per device as shown in Figure 1-56 and Figure 1-57)

The following figure shows REFCLK input pins. These pins drive into the reference clock interface block to the TxPLLs and CDRs per quad or cascaded among several quads as required by the user design.

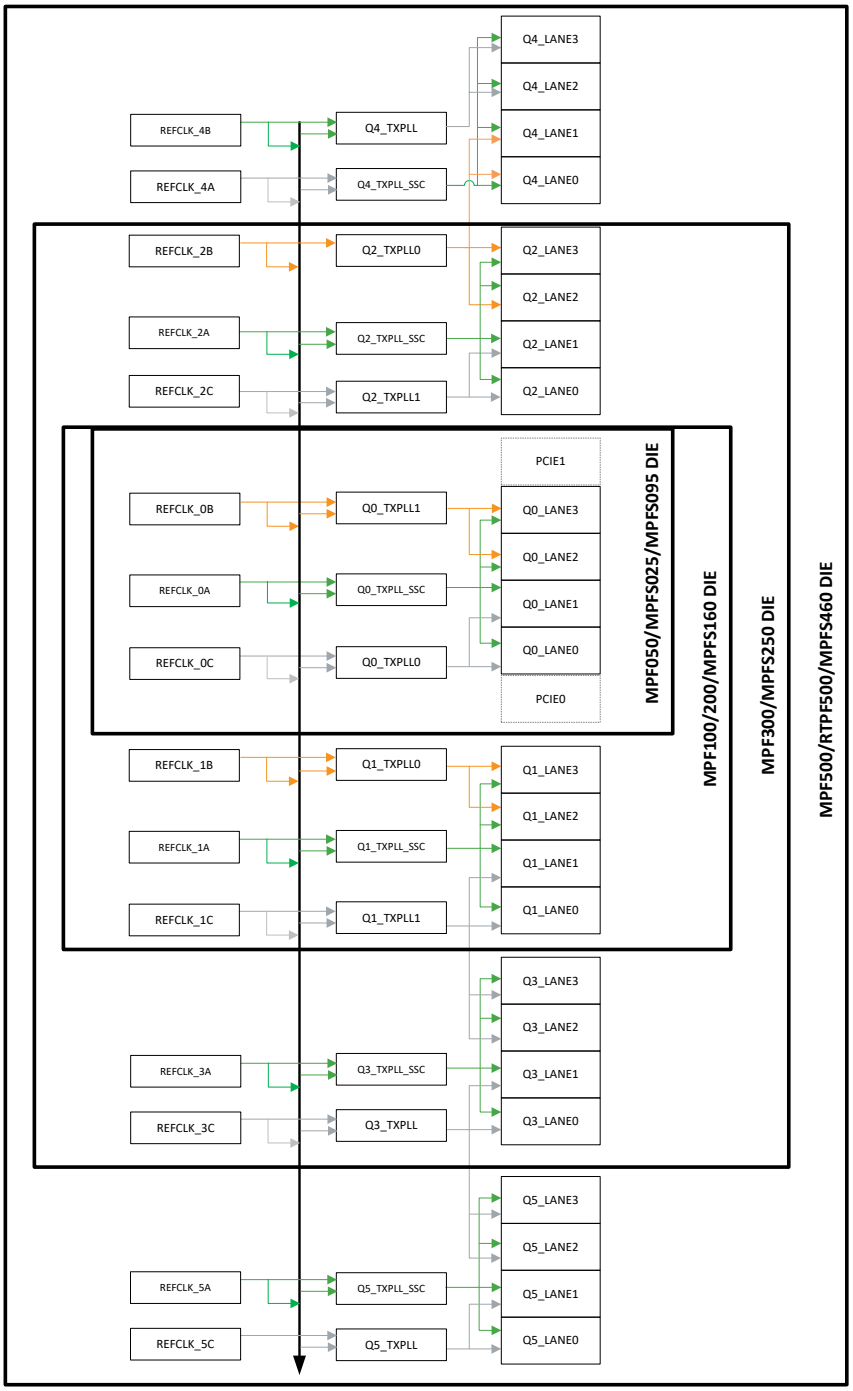
Figure 1-52. REFCLK Input Pin Diagram



Important: The XCVR_#[ABC]_REFCLK pins are available per quad. The REFCLK input to the reference clock interface (see Figure 1-43) connects the REFCLK source to the associated TxPLL.

The following figure shows the transceiver reference clock connectivity from the dedicated reference clock input pins to the cascaded clock routing to the TXPLLs.

Figure 1-53. Transceiver Reference Clock Interface



Important:

- MPFS460 does not have Quad5.
- XCVR_REF_CLK_A inputs to the quads have the greatest reach when cascading reference clocks to TXPLLs and XCVR Lanes.
- See the device specific PPAT tables for die and package restrictions.

For more information on reference clock interface, see Transceiver Reference Clock Interface.

1.5.4.5. Reference Clock Disruptions (Ask a Question)

Reference clocks are required to be applied and stable to the PLLs for proper transceiver operations for Lock to Data or Lock to Reference applications. TXPLL and RXCDR PLL require a guaranteed reference clock to ensure proper operation. Reference clock inputs does not include any “signal detection”. It is the responsibility of the user design to monitor and handle losses or switching of the system level reference clocks. Reference clock disruptions can cause the lock control circuitry to misbehave and possibly be stuck into whatever state it is in when the reference clock is lost.

1.5.4.6. Transceiver Resource Layout (Ask a Question)

For the PolarFire family of devices, there are between one to six transceiver quads with four transceivers each, for a total of 4 to 24 full-duplex transceiver lanes. There are up to three external reference clock inputs per quad, which can be used with every transmit PLL and transceiver lane CDRs. See Transceiver Clocks for more information.

The transmit PLL output clocks can be used by one or more lanes within a quad, or shared with adjacent quads.

Each receive lane CDR has its own PLL; therefore, all receive rates can run at independent frequencies. For the transmit lanes, one base rate can be created by a transmit PLL and driven to each of the transmit lanes it can connect to. Each lane can select between the base rate of two different transmit PLLs or a divided version (Div2, Div4, Div8, or Div11), which can be selected per transmit lane.

The reference clock to the CDR can be sourced from the FPGA fabric thereby allowing more reference clock sources within a design. The inherent noise from the fabric is tolerated by the CDR.

The following tables list the number of transceiver resources available for each family device.

Table 1-24. PolarFire, PolarFire SoC, and RT PolarFire FPGA Transceiver Resources

Device	XCVR Lanes	TxPLLs	Reference Clock Input Pins
MPF050/MPFS025/ MPFS095	4	3	6 single-ended/3 differential
MPF100/MPFS160	8	6	12 single-ended/6 differential
MPF200 [1]	16	11	22 single-ended/11 differential
MPF300/MPFS250 [2]	16	11	22 single-ended/11 differential
MPF500/RTPF500/ MPSC460 [3]	24	15	30 single-ended/15 differential

Notes:

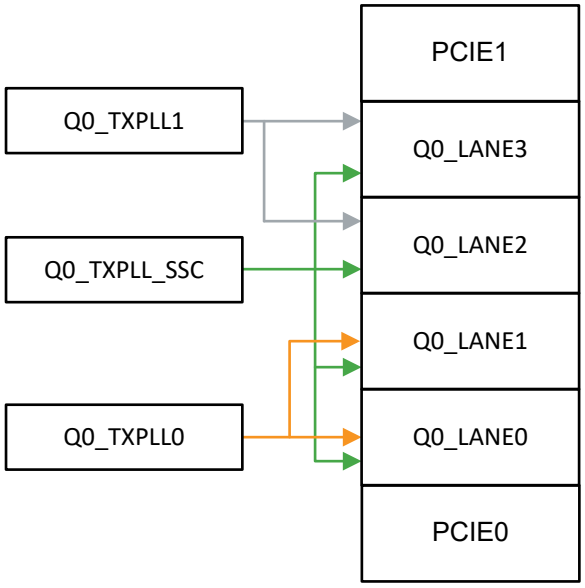
1. MPF200-FCG484 and FCVG484 packages only support up to eight XCVR lanes and six TXPLLs.
2. MPFS250-FCVG484 supports four XCVR lanes (Quad 0 only) and three TXPLLS.
3. MPFS460 does not have Quad 5.
4. The FCSG536 package supports four XCVR lanes (Quad 0 only) and three TXPLLS for MPF200T/ MPF300T/MPFS095/MPFS160T/MPFS250T.

Figure 1-55, Figure 1-56, and Figure 1-57 show the arrangement of the transceiver quads, the connectivity of lanes, transmit PLLs, and embedded PCIe blocks for the MPF100, MPF200, MPF300, MPF500, and RTPF500 device. This arrangement ensures package compatibility for all of the devices in the PolarFire FPGA family. For example, if a package supports all of the devices of the PolarFire family and a PCIe block is used on the smallest device, then the same PCIe block is available on the same package pins for all other devices in the family.

Figure 1-54 shows the arrangement of the transceiver quads, the connectivity of lanes, transmit PLLs, and embedded PCIe blocks for the MPFS250T device. This arrangement ensures package compatibility for all of the devices in the PolarFire SoC FPGA family. For example, if a package

supports all of the devices of the PolarFire SoC family and a PCIe block is used on the smallest device, then the same PCIe block is available on the same package pins for all other devices in the family.

Figure 1-54. MPF050/MPFS025/MPFS095/MPFS250-FCVG484 and MPF200T/MPF300T/MPFS095/MPFS160T/MPFS250T-FCSG536



Important: The FCSG536 package supports four XCVR lanes (Quad 0 only) and three TXPLLS for MPF200T/MPF300T/MPFS095/MPFS160T/MPFS250T.

Figure 1-55. MPF100/MPF200/MPFS160 Transceiver and Transmit PLL Layout

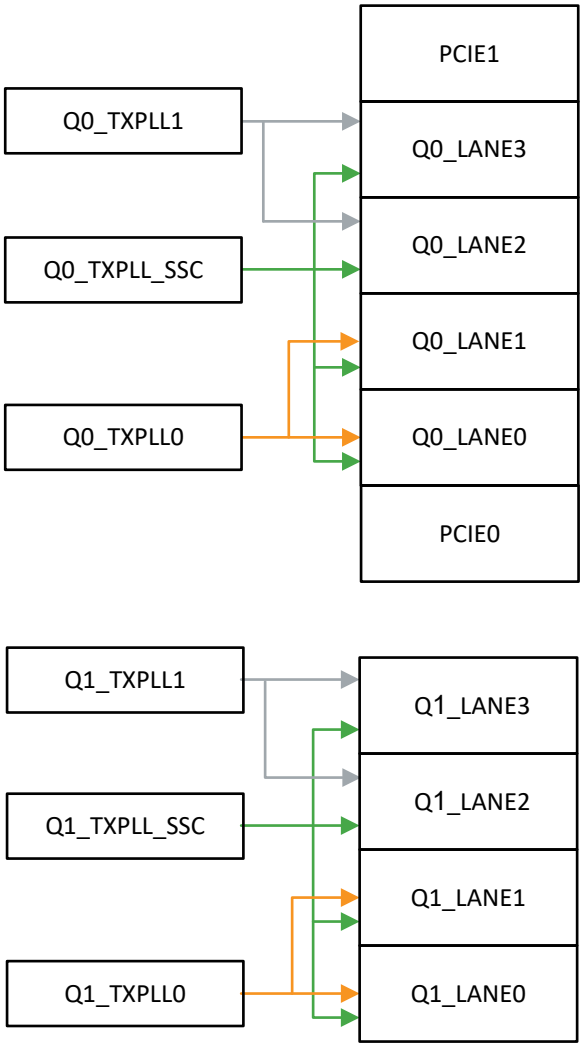
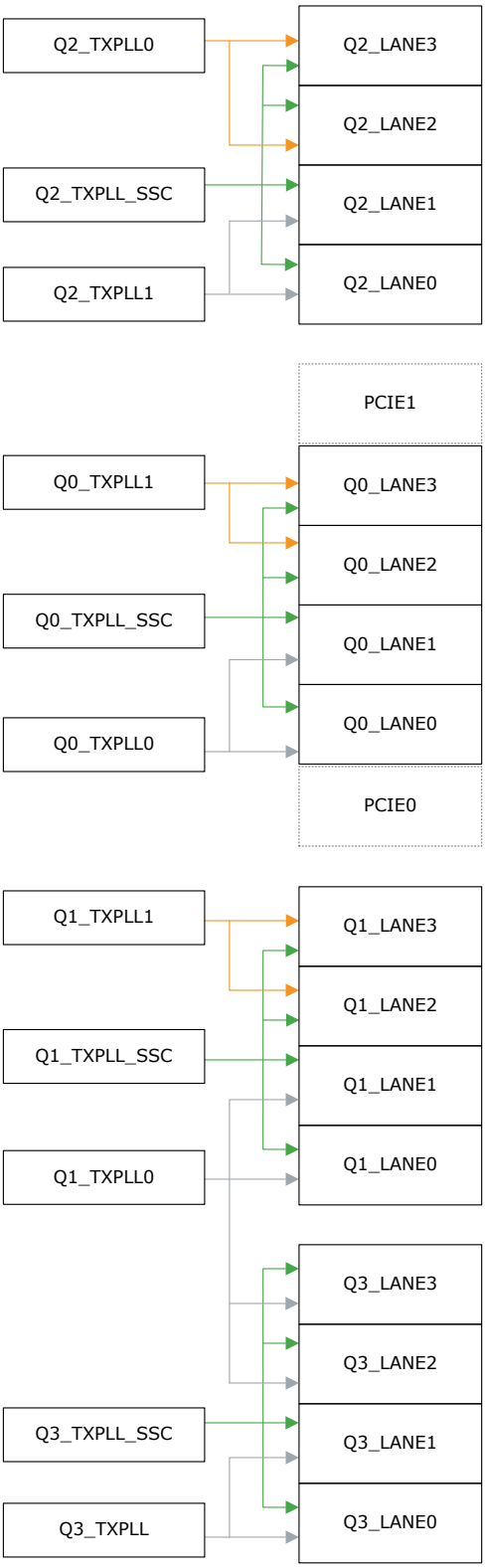


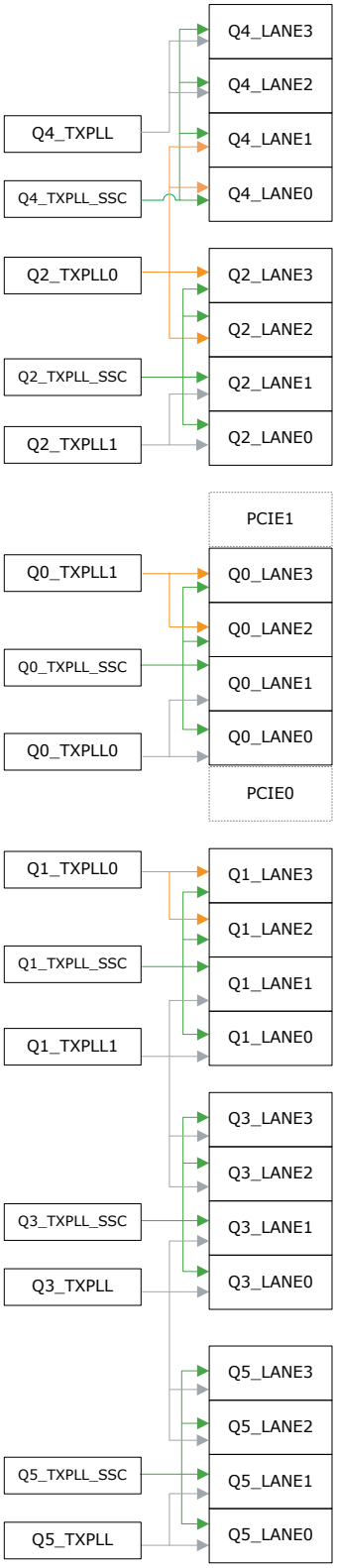
Figure 1-56. MPF200/MPF300/MPFS250 Transceiver and Transmit PLL Layout



Important:

- MPF200–FCG484 and FCVG484 only support up to eight XCVR lanes and six TXPLLs. See Figure 1-55.
- MPFS250-FCVG484 supports four XCVR lanes (Quad 0 only) and three TXPLLS. See Figure 1-54.

Figure 1-57. MPF500/RTPF500/MPFS460 Transceiver and Transmit PLL Layout



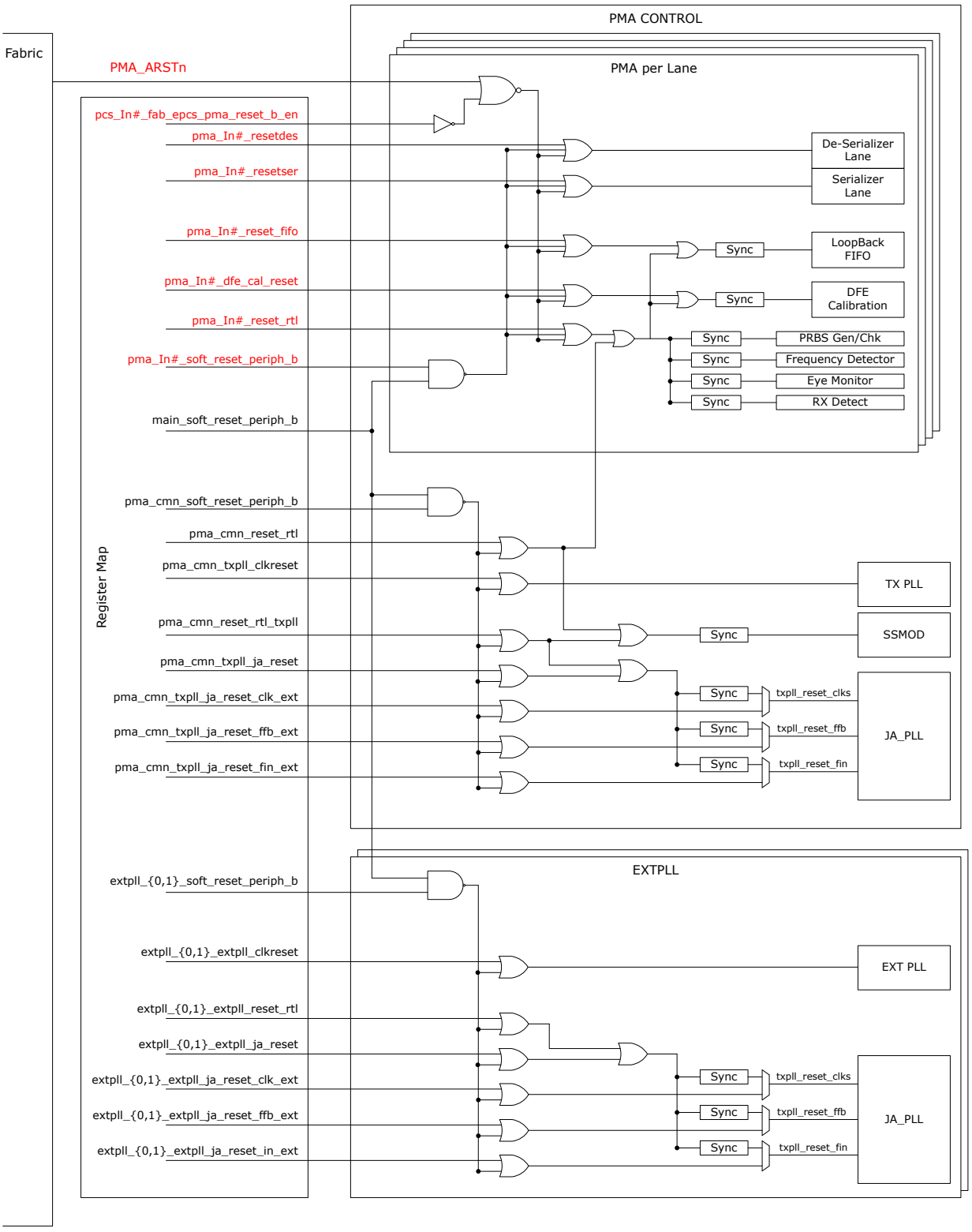
Important: MPFS460 does not include Quad 5.

1.6. PMA and PCS Resets (Ask a Question)

The transceiver uses partitioned resets, one single wire for PMA_ARST_N and one for PCS_ARST_N to the PF_XCVR. Specifically, these two inputs can come from the FPGA fabric to reset the PMA and PCS portions of the transceiver. Both inputs assert the reset asynchronously and de-assertion is internally synchronized.

PMA_ARST_N reset always impacts both the Rx and Tx. This active LOW input resets all internal portions of the transceiver PMA including the serializer/deserializer, DFE, eye monitor, loopback FIFO, and internal analog circuits. The following figure shows the block diagram of PMA_ARST_N.

Figure 1-58. PMA_ARST_N Block Diagram



** Signals in red are reset signals per PMA lane.*

Control registers are available to selectively reset all components within the PMA and TXPLLs. These registers can be accessed through DRI (see Dynamic Reconfiguration Interface). For information about the register map, see respective PolarFire Device Register Map or PolarFire SoC Register Map.

PCS_ARST_N reset impacts either or both the receiver and the transmitter portion of PCS depending on transceiver mode (see Transceiver Modes). Resetting the Tx causes the serial link to be void of data toggling while internally restarting and the PCS transmit data path flushes out. Resetting the Rx causes the PCS active mode circuitry to be restarted. In 8b10b PCS mode, PCS_ARSTN is used to force the symbol alignment to restart when too many data errors are seen in the fabric logic.

PMA_ARSTN causes the RX clocks to stop. Users must not assert/de-assert the PMA_ARSTN reset through logic that is driven from the RX clocks as this would then create a lock-up situation where the PMA Reset is never de-asserted because the RX clock stops to toggle.

The functionality of the resets are register configured by the Libero software. See the respective PolarFire Device Register Map or PolarFire SoC Register Map for information about register maps. After Libero programming, these register controlled bits are written at power up prior to releasing the Tx PLL from reset. By doing this configuration during reset guarantees that the PCS Tx is reset without manual intervention.

In PMA Only XCVR configurations, the PCS_ARST_N can be tied to 1. This configuration assumes that the PCSLANE/LRST_R0/LRST_ULCKD_CDR_RESETS_PCS_RX register is set to 0x1 (default). This configuration resets the fly-wheel FIFOs within the PCS for PMA Only modes when the CDR automatically resets PCS Rx domain logic.

The Libero PCS_ARST_N configuration, Rx Only option, can only reset the Rx PCS when PCS_ARST_N signal falls; and self-reset the Tx PCS when the Tx PLL transitions from unlocked to locked. The Tx PLL lock is typically the last event which impacts the health of the clock going into the PCS from the PMA. However, if there is an application which requires a change to serializer post-divider or a change to the PLL that a Tx lane is using; these events cause a change in PCS clocking. These types of configuration changes must be carried out by first asserting the soft PCS Tx Reset system register, then changing the serializer configuration, and then de-asserting the soft PCS Tx Reset system register. This method assures the PCS plus its FWF sees a consistent clocking at the time when they are out of reset. Rate modifications must be done through DRI using a reset/modify/un-reset sequence and must be implemented by the user design. For such rate changes, the *ARST_N pins do not have to be used since soft resets are available through the DRI.

In 8b10b mode, the word aligner can only find a new pattern following a reset. Therefore, the Libero default is for PCS_ARST_N for Rx Only.

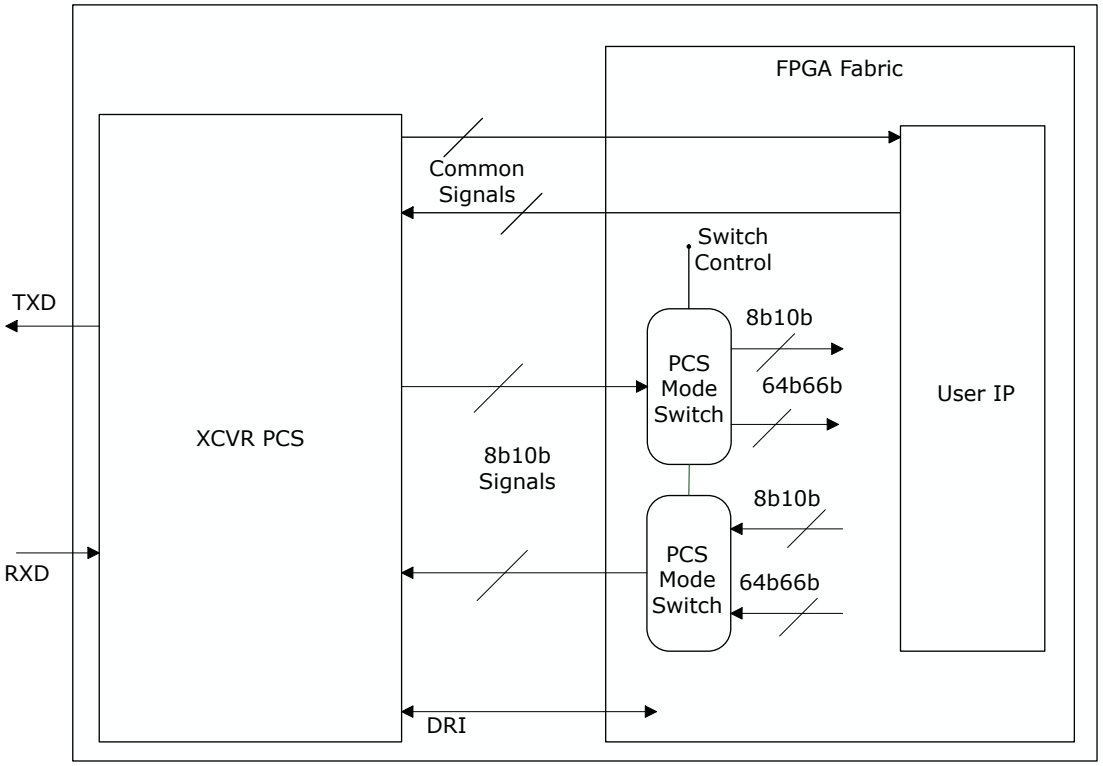
See Transceiver Modes for information about the impact of PMA and PCS resets.

1.7. PCS Rate Switch Between 8b10b and 64b66b Mode for CPRI (Ask a Question)

All CPRI protocol data rates are statically supported with the Libero Transceiver Configurator using either 8b10b or 64b66b modes. 8b10b supports CPRI rates 2, 3, 4, 5, 6, and 7 while 64b66b mode supports rates 7a, 8, and 9. In cases requiring dynamic switching, the user must take steps to accommodate the correct design intentions for switching the PCS mode between the data rates. This case requires a rate switch between the 8b10b and 64b66b modes.

8b10b ports are a superset of 64b66b ports. This implies that the user must plan the FPGA fabric design to interface with 8b10b ports of the XCVR and use the fabric to control the mode switching to the CPRI user IP as shown in the following figure.

Figure 1-59. PCS Rate Switch between 8b10b and 64b66b Mode



The following table lists the port crossover between the 8b10b and 64b66b modes provided by the XCVR PCS to the FPGA fabric.

Table 1-25. Port Crossover between the 8b10b and 64b66b Modes

Direction	8B10B Mode	64B6xB Mode
Input	TX_DISPFNC[15]	TX_ELEC_IDLE
Input	TX_DISPFNC[14]	TX_BYPASS_DATA
Input	TX_DISPFNC[13:12]	RESERVED_IN[1:0]
Input	TX_DISPFNC[11:8]	RESERVED_IN[5:2]
Input	TX_DISPFNC[7:6]	RESERVED_IN[7:6]
Input	TX_DISPFNC[5:2]	RESERVED_IN[11:8]
Input	TX_DISPFNC[1]	RESERVED_IN[12]
Input	TX_DISPFNC[0]	RESERVED_IN[13]
Input	TX_K[7:6]	RESERVED_IN[15:14]
Input	TX_K[5]	RESERVED_IN[16]
Input	TX_K[4]	TX_SOS
Input	TX_K[3:0]	TX_HDR[3:0]
Input	TX_DATA[63:33]	TX_DATA[63:33]
Input	TX_DATA[32:19]	TX_DATA[32:19]
Input	TX_DATA[18:17]	TX_DATA[18:17]
Input	TX_DATA[16]	TX_DATA[16]
Input	TX_DATA[15]	TX_DATA[15]
Input	TX_DATA[14]	TX_DATA[14]
Input	TX_DATA[13:12]	TX_DATA[13:12]

Table 1-25. Port Crossover between the 8b10b and 64b66b Modes (continued)

Direction	8B10B Mode	64B6xB Mode
Input	TX_DATA[11:10]	TX_DATA[11:10]
Input	TX_DATA[9:8]	TX_DATA[9:8]
Input	TX_DATA[7:5]	TX_DATA[7:5]
Input	TX_DATA[4]	TX_DATA[4]
Input	TX_DATA[3]	TX_DATA[3]
Input	TX_DATA[2]	TX_DATA[2]
Input	TX_DATA[1]	TX_DATA[1]
Input	TX_DATA[0]	TX_DATA[0]
Output	RX_K[7]	RX_BYPASS_DATA
Output	RX_K[6:0]	RESERVED_OUT[6:0]
Output	RX_CODE_VIOLATION_7	RESERVED_OUT[13:7]
	RX_DISPARITY_ERROR_7	
	RX_CODE_VIOLATION_6	
	RX_DISPARITY_ERROR_6	
	RX_CODE_VIOLATION_5	
	RX_DISPARITY_ERROR_5	
	RX_CODE_VIOLATION_4	
Output	RX_DISPARITY_ERROR_4	RX_HDR_VAL
Output	RX_CODE_VIOLATION_3	STATUS_HI_BER
Output	RX_DISPARITY_ERROR_3	STATUS_LOCK
Output	RX_CODE_VIOLATION_2	RX_SOS
Output	RX_DISPARITY_ERROR_2	RX_DATA_VAL
Output	RX_CODE_VIOLATION_1	RX_HDR[3:0]
	RX_DISPARITY_ERROR_1	
	RX_CODE_VIOLATION_0	
	RX_DISPARITY_ERROR_0	
Output	RX_DATA[63:51]	RX_DATA[63:51]
Output	RX_DATA[50:49]	RX_DATA[50:49]
Output	RX_DATA[48:9]	RX_DATA[48:9]
Output	RX_DATA[8:5]	RX_DATA[8:5]
Output	RX_DATA[4]	RX_DATA[4]
Output	RX_DATA[3:1]	RX_DATA[3:1]
Output	RX_DATA[0]	RX_DATA[0]
Input	TX_BIT_CLK[4]	
Input	TX_PLL_LOCK[4]	
Input	TX_PLL_REF_CLK[4]	
Input	CTRL_CLK[5]	
Input	CTRL_ARST_N[5]	
Input	CALIB_REQ[5]	
Input	LOS[5]	
Input	CDR_REF_CLK	
Output	CALIBRATING	
Input	TX_WCLK[6]	
Input	PCS_ARST_N	
Input	PMA_ARST_N	

Table 1-25. Port Crossover between the 8b10b and 64b66b Modes (continued)

Direction	8B10B Mode	64B6xB Mode
Output	RX_VAL	
Output	RX_READY	
Output	RX_IDLE	
Output	TX_CLK_STABLE	
Output	RX_CLK_[R:G]	
Output	TX_CLK_[R:G]	
Input	RXD_P	
Input	RXD_N	
Output	TXD_P	
Output	TXD_N	

Notes:

1. Port names in Libero has prefix lane#_ appended to name.
2. Refer to specific PCS port list tables for port description.
3. Ports prefixed by RESERVED for a specific mode is not used for the MODE.
4. Port is included in CLKS_FROM_TXPLL BIF.
5. Port is included with Enhanced Receiver Management (ERM).
6. TX_WCLK is included when Global (Shared) TX Clock is used.

The user control logic must accommodate switching transceiver control registers listed in the following table. This is typically by modifying the registers using the DRI to the transceiver from an APB within the design.

The following table outlines the affected registers that must be modified during rate switching.

Table 1-26. System Registers Affecting 8B10B and 64B6xB Data Paths

Register Page xls	Register Name	Field Name	Description	Required Value for 8B10B	Required Value for 64B6xB
pcslane	L8_R0	L8_TXENCSWAPSEL	Selects between 1000BASE-X/T and Fibre Channel octet-swapping modes.	Optional: 0=1000BASE-X/T, 1=Fibre Channel	Don't-care
		L8_GEARMODE[1:0]	Sets data path width of FWF interfaces.	Must be consistent with clock selections for txfwf_rclk and rxfwf_wclk.	Don't-care
	LOVR_R0	FAB_IFC_MODE[3:0]	Selects path through fabric and FWF overlay blocks.	Register changes the meanings of the epcs_tx_data and epcs_rx_data pins based on the setting of the pcslane LOVR_R0:PCSPMA_IFC_MODE[3:0] control register field. The PCSPMA_IFC_MODE is one-hot encoded as follows: 0b0100 == 8B10B mode	PCSPMA_IFC_MODE is one-hot encoded as follows 0b0010 == 64B6xB mode
		PCSPMA_IFC_MODE[3:0]	Selects lane mode for driving data into the SerDes serializer.		
	LCLK_R0	LCLK_EPCS_RX_CLK_SEL [1:0]	Chooses which clock is sent to fabric on epcs_rx_clk port.	Usually this must be set to 2'd1 so that the frequency of the fabric is the same as the internal side of the FWF. However variations are possible if the use of the Rx FWF synchronous enable will be employed. See FWF description for further information.	2'd1
		LCLK_EPCS_TX_CLK_SEL [1:0]	Chooses which clock is sent to fabric on epcs_tx_clk port.		2'd1
pcslane	LCLK_R0	LCLK_PCS_RX_CLK_SEL [1:0]	Defines clock module's source for pcs_rx_clk.	Must be set to 2'd3 for all applications using 8B10B function.	2'd3
		LCLK_PCS_TX_CLK_SEL [1:0]	Defines clock module's source for pcs_rx_clk.		
		LCLK_RXFWF_WCLK_SEL [1:0]	Defines clock module's source for rxfwf_wclk.	Must be consistent with L8_GEARMODE setting.	2'd2
		LCLK_TXFWF_RCLK_SEL [1:0]	Defines clock module's source for txfwf_rclk.		2'd2
		LCLK_RXFWF_WCLK_PIPE	Defines whether Rx FWF is clocked by Tx side clocks or Rx side clocks.	Must be set to 1'd0 for 8B10B functionality.	1'd0

Table 1-26. System Registers Affecting 8B10B and 64B6xB Data Paths (continued)

Register Page xls	Register Name	Field Name	Description	Required Value for 8B10B	Required Value for 64B6xB
pcslane (continued)	LCLK_R1	LCLK_ENA_8B10B_RX_CLK	Instructs clock module to drive 8B10B pcs_rx_clk.	Must be set to 1'd1 for 8B10B operation.	1'd0
		LCLK_ENA_8B10B_RXFWF_WCLK	Instructs clock module to drive 8B10B rxfwf_wclk.		
		LCLK_ENA_8B10B_TX_CLK	Instructs clock module to drive 8B10B pcs_tx_clk.		
		LCLK_ENA_8B10B_TXFWF_WCLK	Instructs clock module to drive 8B10B txfwf_rclk.		
		LCLK_ENA_64B6XB_RX_CLK	Instructs clock module to drive 64B6xB pcs_rx_clk.	1'd0	1'd1
		LCLK_ENA_64B6XB_RX_CLK_DIV2	Instructs clock module to drive 64B6xB pcs_rx_clk_div2.	1'd0	1'd1 for 64-bit fabric 1'd0 for 32-bit fabric
		LCLK_ENA_64B6XB_TX_CLK	Instructs clock module to drive 64B6xB pcs_tx_clk.	1'd0	1'd1
		LCLK_ENA_64B6XB_TX_CLK_DIV2	Instructs clock module to drive 64B6xB pcs_tx_clk_div2.	1'b0	1'b1 for 64-bit fabric 1'b0 for 32-bit fabric
pma_lane	DES_CLK_CTRL	DESMODE[2:0]	Selects parallel bus width of deserializer interface.	Must select the 40-bit wide bus mode for 8b10b functionality (3'd7).	Must select the 32-bit wide bus mode for 64B6xB functionality (3'd6).
	SER_CLK_CTRL	SERMODE[2:0]	Selects parallel bus width of serializer interface.		
	DES_CDR_CTRL3	SLIP_DES_CDR_SEL	Selects source of CDR slip control.	Below setting is dependent on RX_SLIP_BIT enabled with the XCVR Configurator GUI. This has dependencies on the users fabric IP	
				Must be 1'd0 so that the fabric can control the symbol alignment.	Should be 1'd1 so that fabric cannot control the symbol alignment.
		SLIP_DES_CDR_EN	Optionally turns slip control OFF.	Must be set to 1'd1.	Must be set to 1'd0

Note: See respective PolarFire Device Register Map or PolarFire SoC Register Map for more information.

2. Implementation (Ask a Question)

Transceiver blocks support many high-speed serial protocols. These protocols are supported using multiple transceiver building blocks that the user constructs using the transceiver configurators in the Libero design software. The Libero configurator allows the user to set the reference clock and data rates for particular protocols. This information is then used to properly generate the configuration settings for the PMA, and the associated interface logic. The configurators build components that are used to instantiate/configure the transceiver-specific hardware macros including the PMA and PCS blocks using the Libero SmartDesign software.

2.1. Libero Configurators (Ask a Question)

Three explicit PolarFire configurators are the preferred tool for wrapper generation needed to instantiate transceiver primitive macros called PF_XCVR_REF_CLK, PF_TX_PLL, and PF_XCVR. The configurator is part of the Libero SoC design tools and is available when the PolarFire macros are downloaded from the Libero catalog.

The following table provides details on three Libero transceiver configurators in the Libero Software: transmit PLL, transceiver reference clock, and transceiver interface modules. The transmit PLL (PF_TX_PLL) and transceiver interface (PF_XCVR) modules are used when the transceivers are implemented in the Libero FPGA design. The transceiver reference clock (PF_XCVR_REF_CLK) is used when the dedicated input clock from the top-level pins are used. Optionally, this is not used if the transceiver reference clock comes from the PLL or from the FPGA fabric. The user must instantiate and configure these three blocks in their transceiver design.

Table 2-1. Transceiver Configurator Component List

Configurator	Macro	Details
Transmit PLL	PF_TX_PLL	Generates the TxPLL/TxPLL_SSC based on the provided input to the GUI. The PF_TX_PLL generates the BIT_CLK for the transceiver.
Transceiver Reference Clock	PF_XCVR_REF_CLK	Generates the reference clock based on the provided input to the GUI—selection of differential or single-end input buffer and selection of single or dual clock inputs to the transmit PLL clock interface.
Transceiver Interface	PF_XCVR_ERM	Configures the requested number of lanes (4 lane maximum) with the same PMA and PCS settings—the lanes required by the design and CDRPLL settings.

As the FPGA designer makes selections in the each transceiver module configurators, it automatically guides and narrows down the subsequent choices and defaults. Each configurator maintains a module diagram while the designer selects the module properties. Once all the choices are made, the configurator generates an RTL netlist that instantiates the required macros specific to the requirements of the design. Only the relevant ports appear in the generated macro. This section describes how to enter these configuration parameters in the transceiver configurator GUIs.

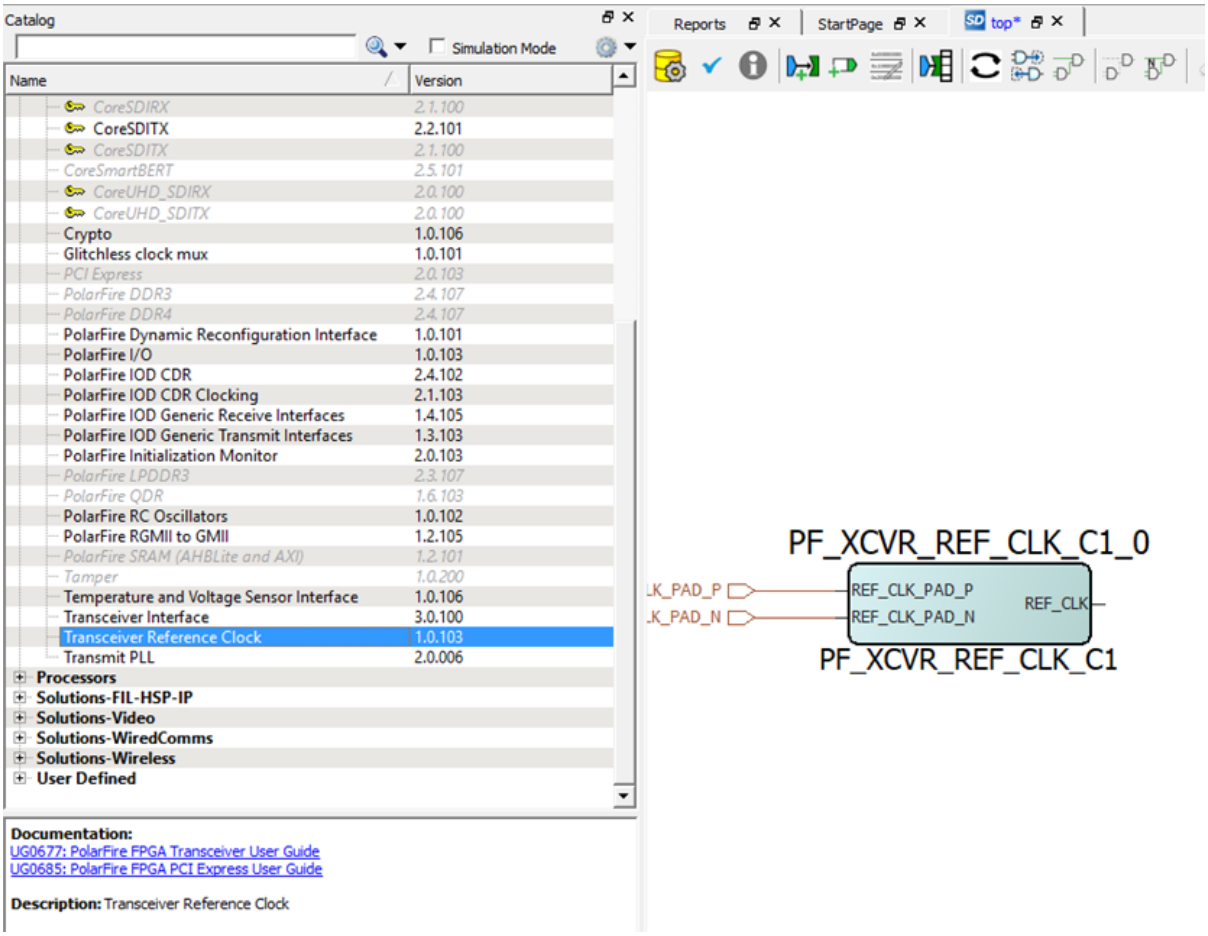
2.1.1. Transceiver Reference Clock Configurator (Ask a Question)

The Transceiver Reference Clock Configurator is used to build the correct reference clock input to the transceiver and to the Tx PLL. The user can pick the input type and various input options.

To initiate the Reference Clock Configurator, perform the following steps:

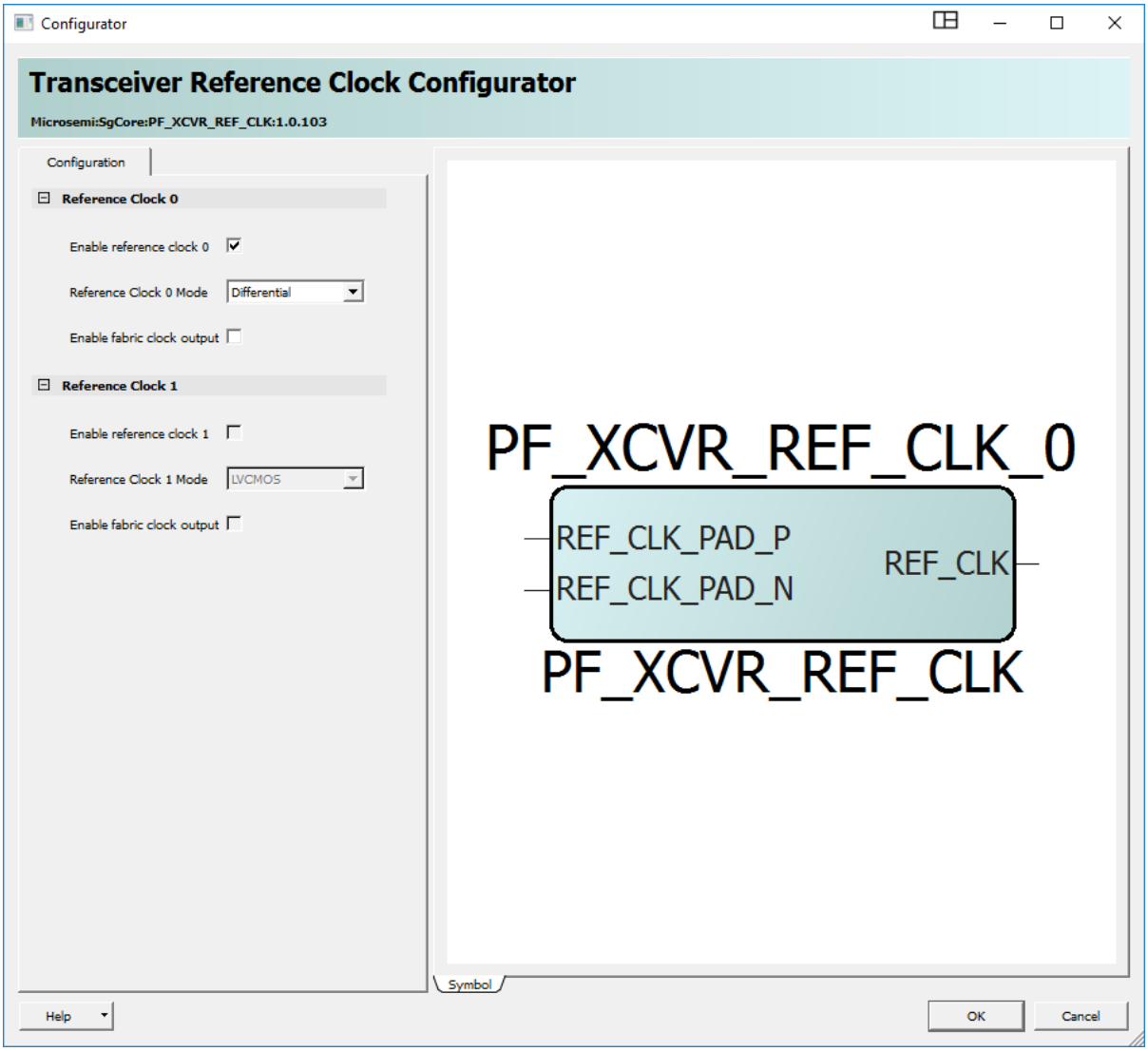
1. Access the **Transceiver Reference Clock** cores under **Features** from the **Catalog** window, as shown in the following figure.

Figure 2-1. Transceiver Reference Clock Selection from Catalog



2. Double-click each PF_XCVR_REF_CLK block from the catalog to launch the configurator. A GUI allows the selection of the related reference clock properties.

Figure 2-2. Transceiver Reference Clock Configurator GUI



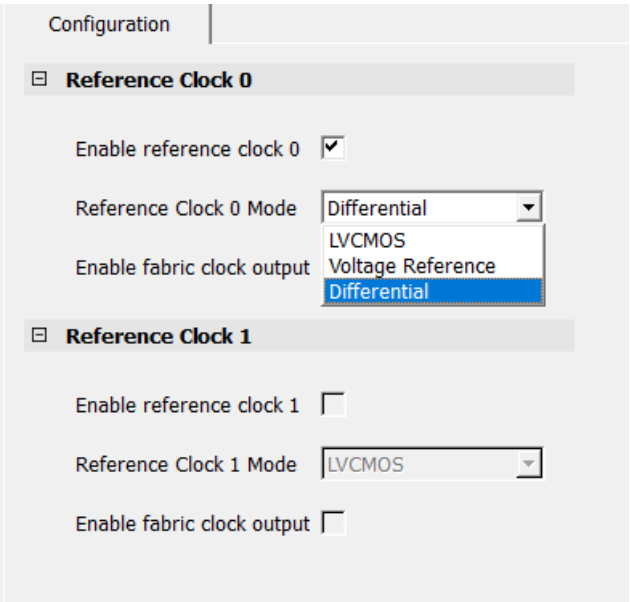
The following table lists transceiver reference clock configurator GUI options.

Table 2-2. Transceiver Reference Clock Configurator GUI Options

	Options	Default	Details
Reference Clock 0 Configuration			
Enable reference clock 0	Enable and disable	Enabled	Checked = enabled
Reference Clock 0 Mode	LVCMOS, voltage reference, and differential	Differential	
Enable fabric clock output	Enable and disable	Disabled	Checked = enabled When enabled, a port is exposed for fabric routing.
Reference Clock 1 Configuration			
Enable reference clock 1	Enable and disable	Disabled	Checked = enabled
Reference Clock 1 Mode	LVCMOS and voltage reference	LVCMOS	
Enable fabric clock output	Enable and disable	Disabled	Checked = enabled When enabled, a port is exposed for fabric routing.

3. Select the reference clock mode type based on the input buffer type in the application. Single-ended **Differential** is the default mode.

Figure 2-3. Transceiver Reference Clock Mode Type



In the case of **LVCMOS** or **Voltage Reference** inputs, the design can have up to two individual reference clock inputs in one instance of the PF_XCVR_REF_CLK. This configuration can access either or both Reference Clock 0 or/and 1.

Figure 2-4. PF_XCVR_REF_CLK With One Single-Ended Input and Single Output Clock

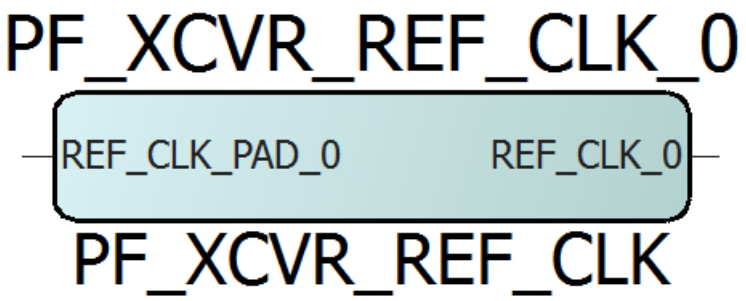
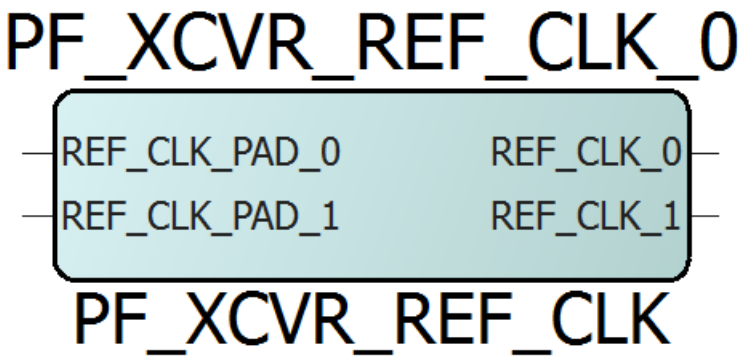
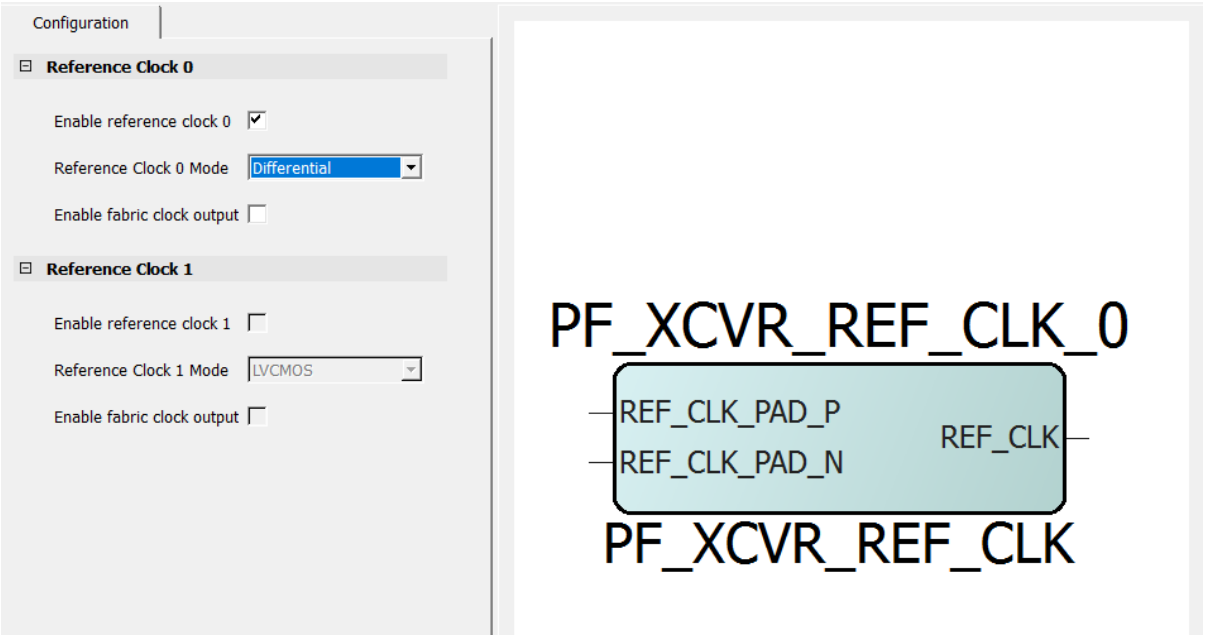


Figure 2-5. PF_XCVR_REF_CLK With Two Single-Ended Input and Two Output Clock



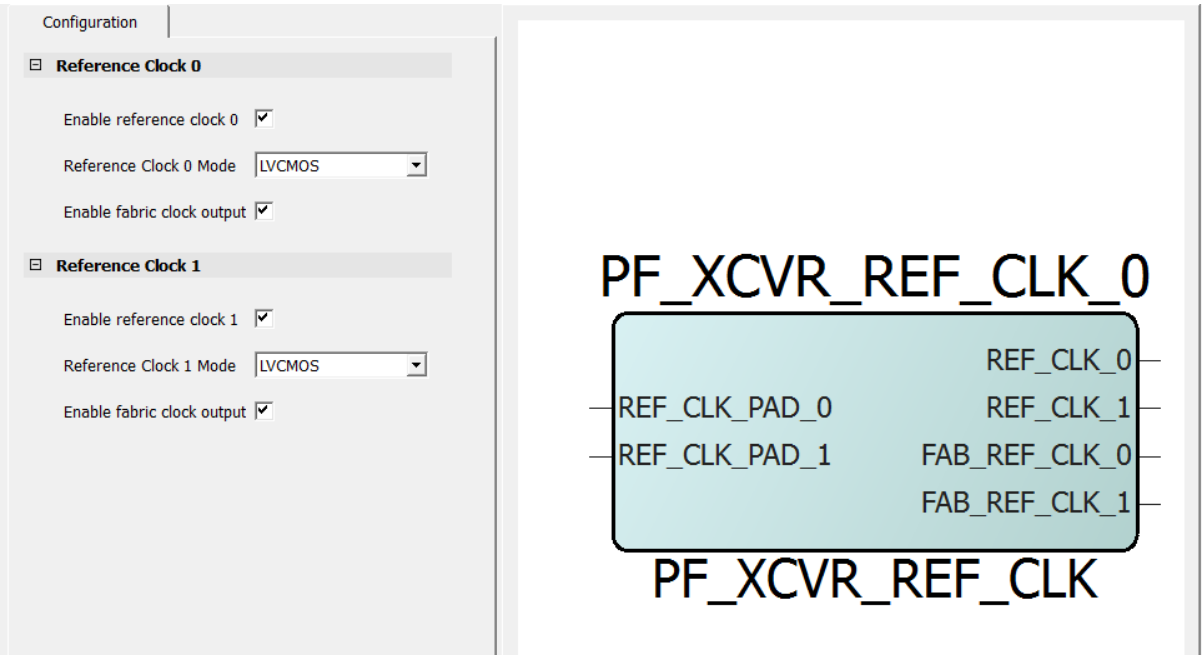
However, only one reference clock input is available when the designer selects **Differential** (Figure 2-3). In this case, only **Reference Clock 0** can be accessed and use a differential clock source signal.

Figure 2-6. PF_XCVR_REF_CLK With Differential Input and Single Output Clock



4. Optionally enable a connection to the FPGA fabric for either/both reference clock 0 or reference clock 1. When enabled, a related port of the associated reference clock is exposed for fabric routing.

Figure 2-7. PF_XCVR_REF_CLK With Fabric Output Clock



5. Click **OK** after making desired selections.

When the Reference Clock configurator generates the reference clock block, specify the desired IP Standard. This is completed by adding the desired IO to the PDC file. For more information about adding the IO to the PDC file, see Physical Constraints.

For more information about XCVR REFCLK input configuration, see XCVR REFCLK Usage.

The PF_XCVR_REF_CLK allows a global routing connection to the FAB_REF_CLK output through a CLKINT global buffer. The FAB_REF_CLK output uses regular fabric routing resources and the connection traverse half the device before connecting onto a CLKINT fabric Global Buffer at the center of the chip. FAB_REF_CLK clock is potentially more susceptible to fabric switching noise, depending on the design, which could lead to an unpredictable amount of clock jitter on that clock being broadcasted to the fabric.

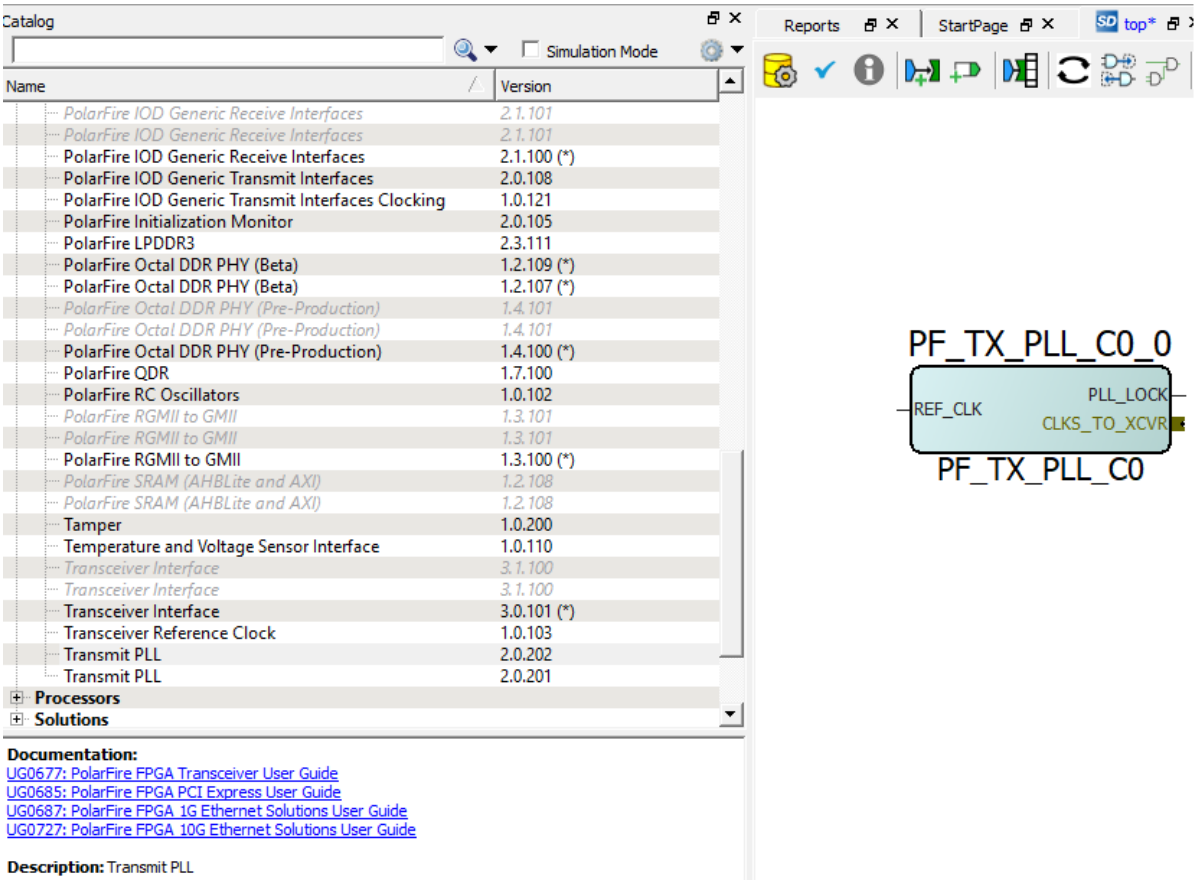
2.1.2. Transmit PLL Configurator (Ask a Question)

The Transceiver Transmit PLL Configurator is used to build the correct transmit PLL to the transceiver. The user can pick from many of the PLL options used for the transceiver based on the application.

To initiate the Transceiver Transmit PLL Configurator, perform the following steps:

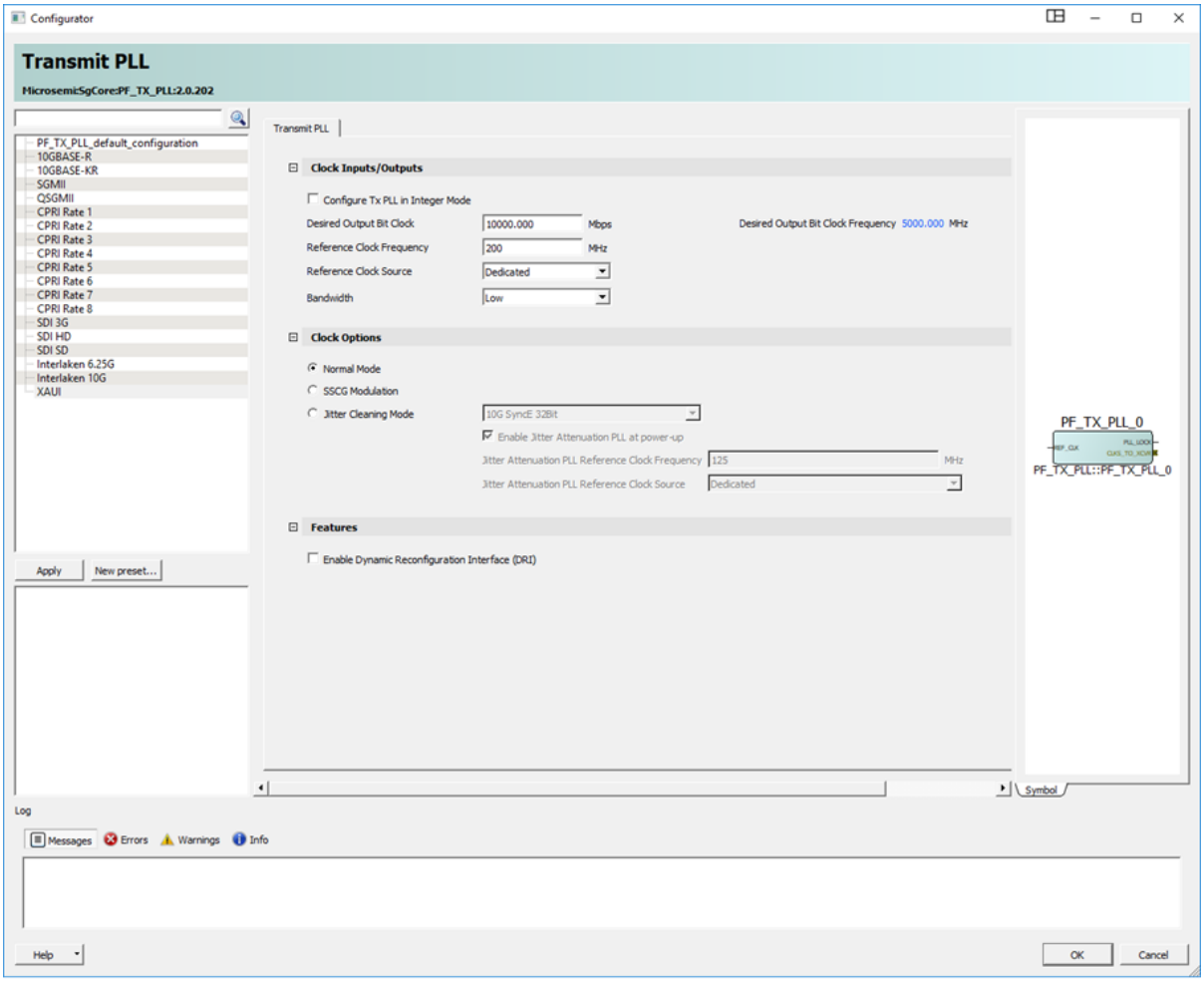
1. Access the **Transmit PLL** module under **Features** from the **Catalog** window, as shown in the following figure.

Figure 2-8. Transceiver Transmit PLL Selection from Catalog



2. Double-click each PF_TX_PLL block from the catalog to launch the configurator. A GUI allows the option of selecting the related transmit PLL properties.

Figure 2-9. Transmit PLL Configurator GUI



The following table lists the transmit PLL configurator GUI options.

Table 2-3. Transmit PLL Configurator GUI Options

Clock Inputs	Options	Default	Details
Configure Tx PLL in Integer Mode	Enable and disable	Disabled	When disabled, it is in TXPLL Fractional N Mode. When enabled, it allows fixed value of available reference clock frequencies.
Reference Clock Source	Dedicated and fabric	Dedicated	Dialogue box requires input clock rate (200 MHz default)
Desired Output Bit Clock	VCO rate, and output clock (MHz)	10000 Mbps, 5000 MHz	Only valid combinations can be entered based on reference clock and PLL capability. This value represents the BIT_CLK output speed.
Bandwidth	Low/High	Low	By default, the Tx PLL selects the low bandwidth option in order to filter more of the reference clock jitter. If the system has a low-jitter reference clock, the user can then set it to high bandwidth mode to decrease the overall jitter. Low bandwidth has longer lock times than high bandwidth.
Clock options (Only one option can be selected)			
Normal Mode	Enable and disable	Enabled	Radio-button on = enabled
SSCG Modulation	Enable and disable	Enabled	Radio-button on = disabled

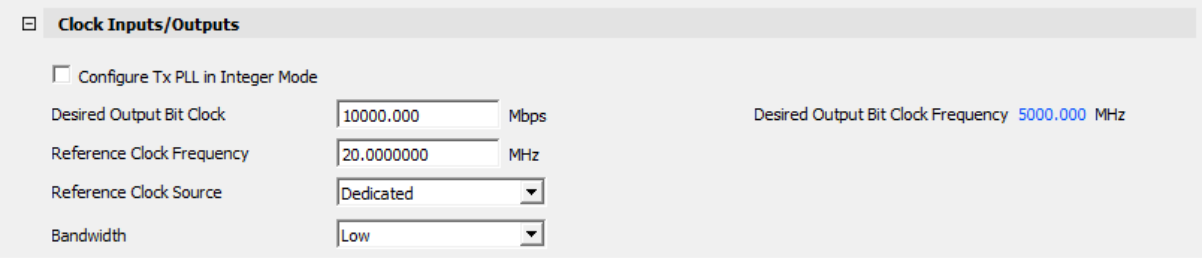
Table 2-3. Transmit PLL Configurator GUI Options (continued)

Clock Inputs	Options	Default	Details
Jitter Cleaning Mode	Enable and disable	Disabled	Radio-button on = disabled The following options are available when enabled: 10G SyncE 32Bit 10G SyncE 64Bit 1G SyncE 10Bit CPRI Rate 1 CPRI Rate 2 CPRI Rate 3 CPRI Rate 4 CPRI Rate 5 CPRI Rate 6 CPRI Rate 8 – 64-bit SDI 3G SDI HD SDI SD Custom Protocol Settings
Features			
Enable Dynamic Reconfiguration Interface (DRI)	Enable and disable	Disabled	Adds Pins to TXPLL component for using Dynamic Reconfiguration Interface. See PolarFire Family Device Power-Up and Resets User Guide for DRI information.

Note: For information about pin functions, see Table 1-18.

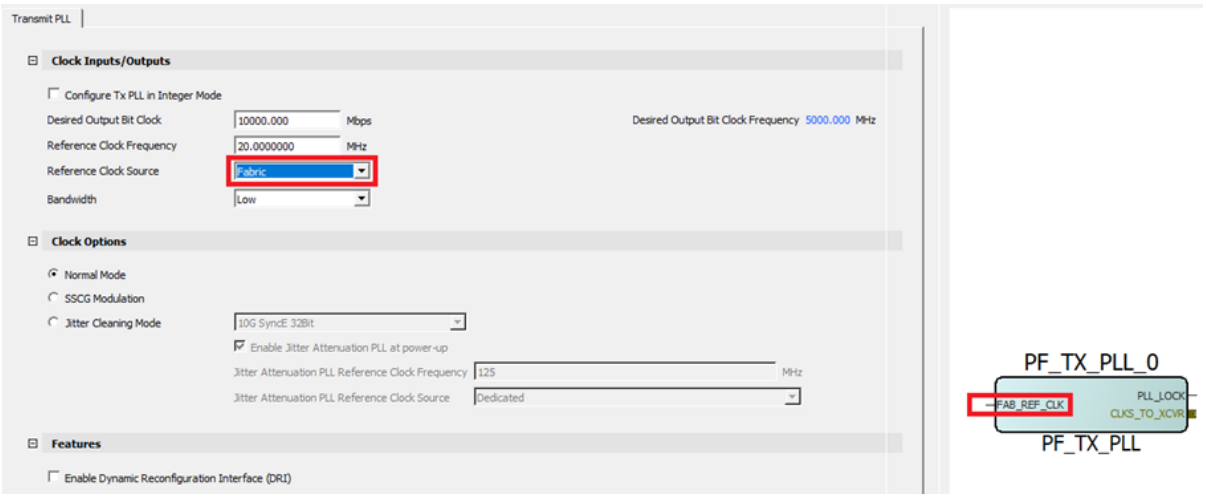
3. Select one of the **Reference Clock** sources (dedicated source is the default) to define **Clock Inputs** and enter the reference clock value.

Figure 2-10. Clock Inputs



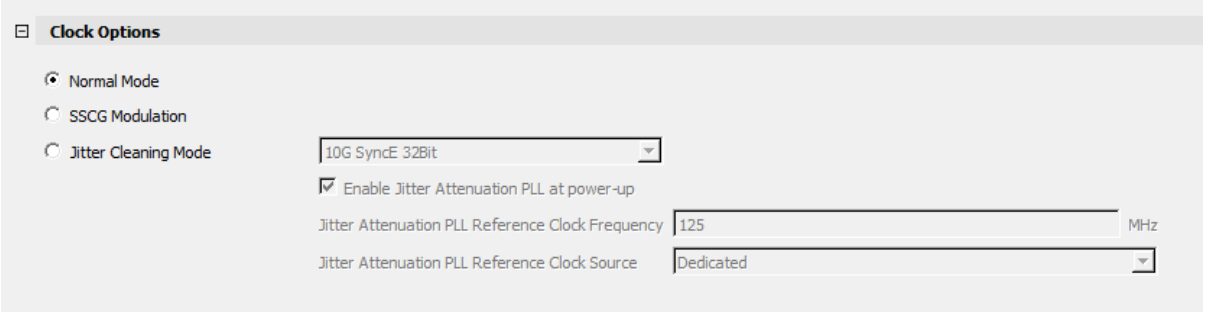
4. Enter the values in **Desired Output Bit Clock**. Frequency cannot be entered—it is calculated automatically based on the speed (data rate in Mbps) and the reference clock.

Figure 2-11. Fabric Clock Input



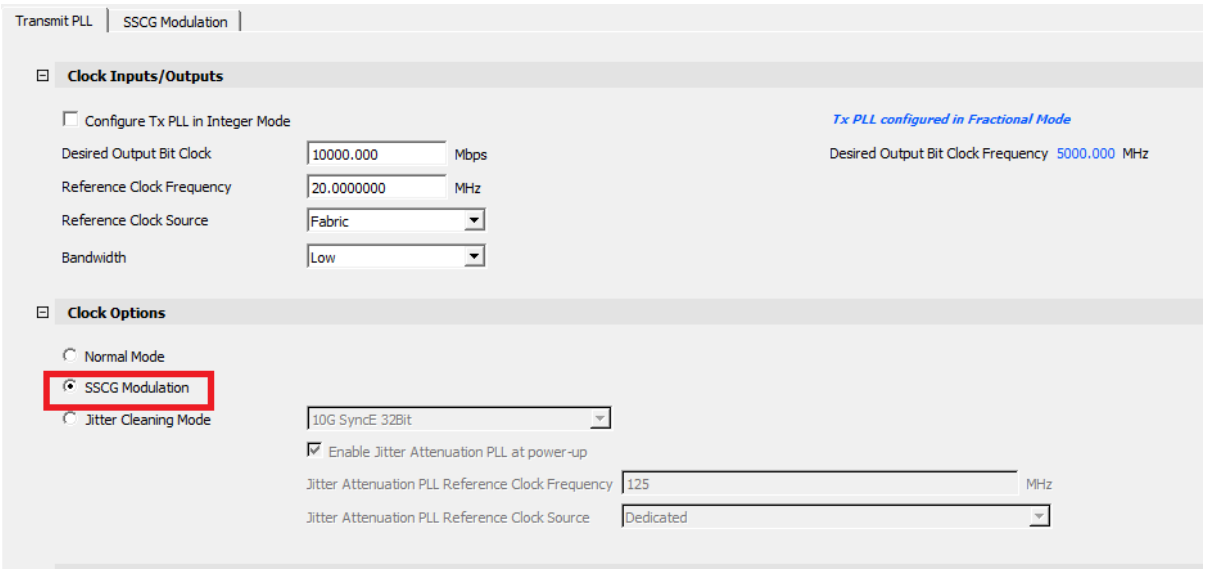
Normal Mode configures the TX PLL to default operation based on the clock Inputs/Outputs setting. Whereas, Jitter Cleaning mode customizes the attenuation coefficients for the pre-defined standards to be used where the recovered clock can be used as a TX reference clock to meet protocol jitter specifications.

Figure 2-12. Clock Options



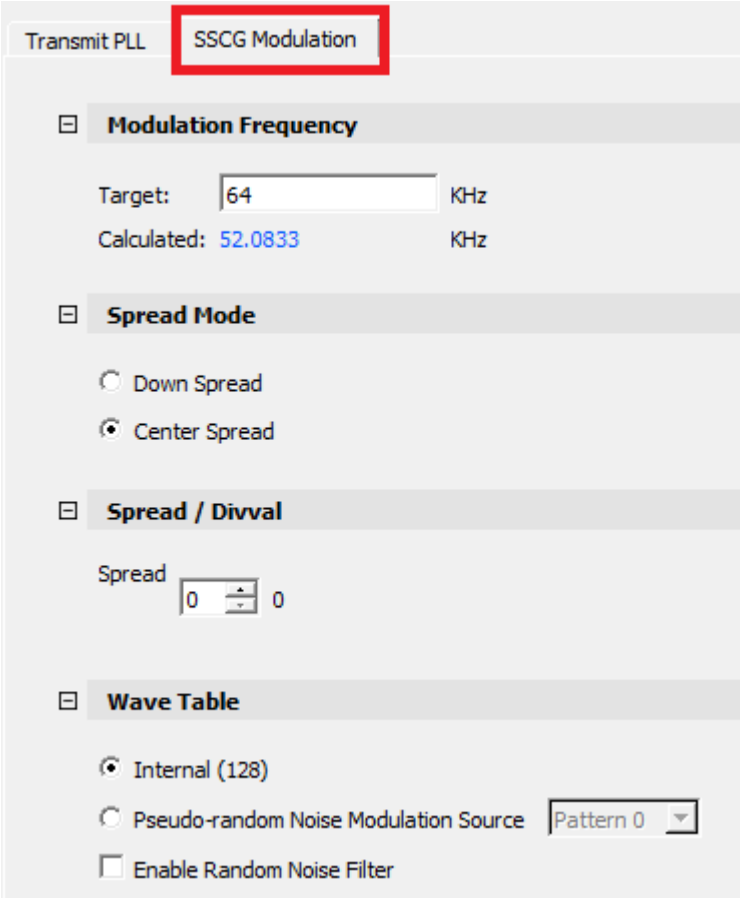
5. Click **Spread Spectrum** clock generation mode (Not available prior to Libero SoC v12.3). This feature allows spread-spectrum generation to be enable from the transmit phase-locked loop. See respective PolarFire FPGA Datasheet, RT PolarFire FPGA Datasheet or PolarFire SoC Datasheet for more information on spread spectrum specifications.

Figure 2-13. Spread Spectrum Clock Generation Enable



6. Select SSCG mode to enable the **SSCG Modulation Feature** tab.

Figure 2-14. Spread Spectrum Modulation Options



7. Click **OK** after making desired selections.
   For more information about Spread Spectrum, see Spread Spectrum Clocking.

Table 2-4. Spread Spectrum

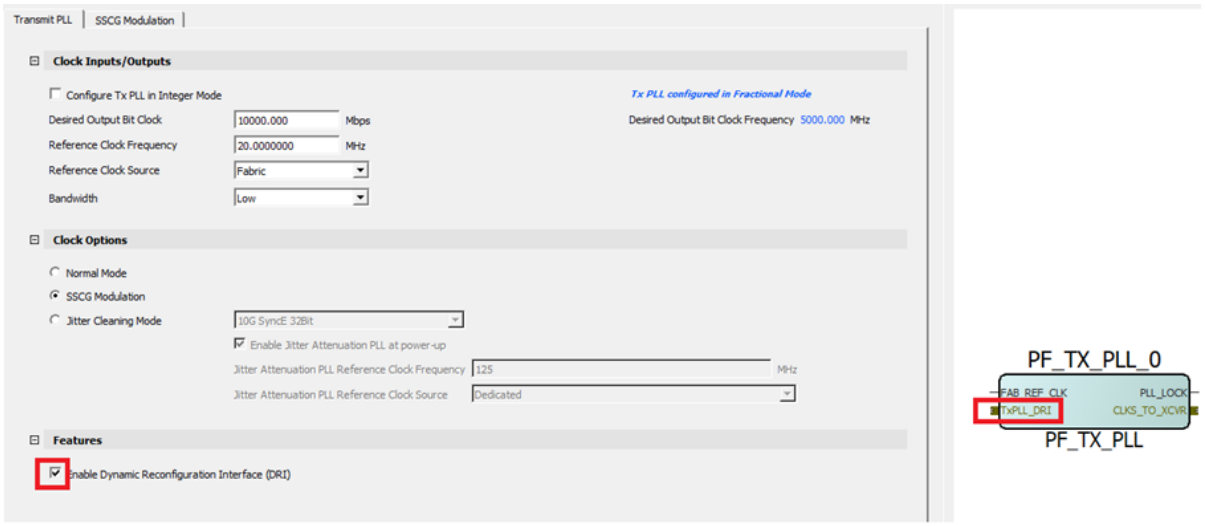
	Options	Default	Details
Modulation Frequency			
Target	User entry	64 KHz	Target Spread Spectrum modulation.
Calculated		60.0962 KHz	Calculation is based on TXPLL reference clock settings.
Spread Mode	Down spread/Center Spread	Down	Configures the modulation style
Spread/Divval			
Spread	Pull down	0	Sets up to 32 divider settings (0, 01.,....3.1). This value sets the percent of modulation amplitude down or center spread.
Wave Table			
Internal (128)		Enabled	Selects internal 128 point triangle wave
Pseudo-random Noise Modulation Source		Disabled	Selection between 3 predefined modulation patterns.
Enable Random Noise Filter	Enable/Disable[1]	Disabled	Enable to turn on filtering to reduce jitter if needed.

(1) When pseudo-random noise modulation source is enabled after the user selects one of the predefined modulation patterns. The optional high pass filtered PRBS is modulated to the PLL reducing the harmonic frequency amplitude peaks.

For more information about jitter cleaning operation, see Jitter Attenuator.

8. Click **Enable Dynamic Reconfiguration Interface (DRI)** checkbox to add the ports needed to connect the Transmit PLL to the DRI. See PolarFire Family Device Power-Up and Resets User Guide for more information about DRI usage and specifications.

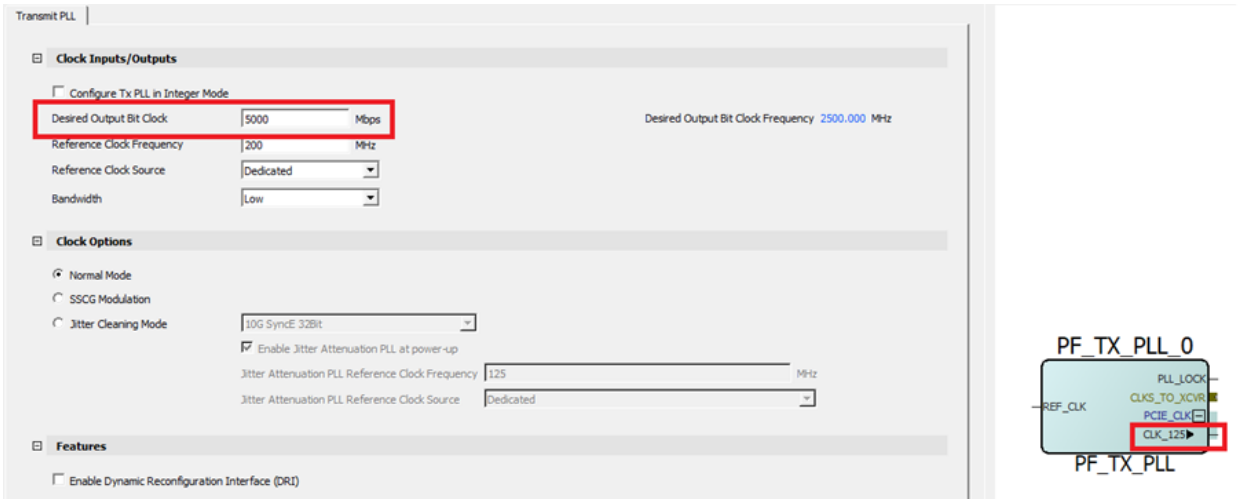
Figure 2-15. Enable Dynamic Reconfiguration Interface



9. Click **OK** after making desired selections.

The TxPLL supports an additional clock output. This CLK_125 output port is automatically exposed when TxPLL BIT_CLK is 5 Gbps. This clock is used with the PCIE macro as input to the transaction layer or AXI clock. For more information about CLK_125, see PolarFire Family PCI Express User Guide.

Figure 2-16. CLK_125 GUI



Note: PLL_LOCK port is automatically exposed as an output port on the TXPLL block. It is the PLL lock indicator signal that can be used to drive logic in the fabric. It is a fabric routed signal.

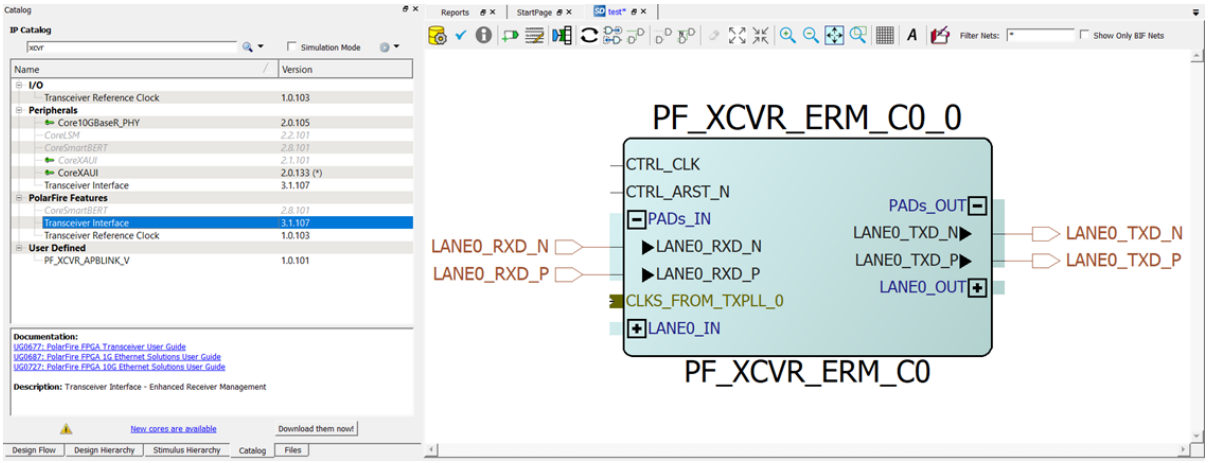
2.1.3. Transceiver Interface Configurator (Ask a Question)

The Transceiver Interface Configurator is used to build the transceiver based on protocol requirements. The user selects the number of lanes, data rate, and protocol-specific settings.

To initiate the Transceiver Interface Configurator, perform the following steps:

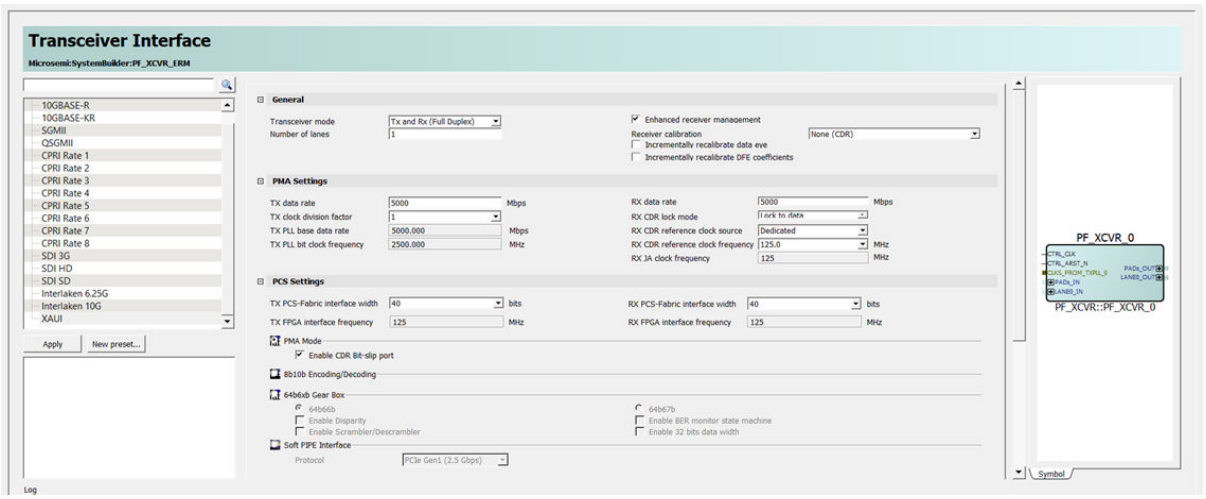
1. Access the **Transceiver Interface** module under **Features** from the **Catalog** window, as shown in the following figure.

Figure 2-17. Transceiver Interface Selection From Catalog



2. Double-click each PF_XCVR block in the catalog to launch the configurator. A GUI allows the option to select the related XCVR properties.

Figure 2-18. Transceiver Interface Configuration GUI



The following tables list Transceiver Interface options.

Table 2-5. Transceiver Interface General Settings

General	Options	Default	Details
Number of lanes	1 to 4	1	
Transceiver mode	Tx and Rx (Full Duplex)	Tx and Rx (Full Duplex)	See Transceiver Modes for more information.
	Tx Only		
	Rx Only		
	Tx and Rx (Independent)		
Enhanced Receiver Management	Enable/Disabled	Enabled	Includes the enhanced receiver management solution when checkbox is checked (See Enhanced Receiver Management).
Receiver Calibration	None (CDR), On-Demand, On-Demand & First Lock, and None (DFE), Incrementally Recalibrate Data Eye and Incrementally Recalibrate DFE Coefficients	On-Demand & First Lock	See Receiver.

Table 2-6. Transceiver Interface PMA Settings

PMA Settings	Options	Default	Details
Tx data rate	250 – 12700 Mbps	5000 Mbps	10312.5 Mbps (STD maximum)
TX clock division factor	1, 2, 4, 8, and 11	1	—
TXPLL base data rate	Computed[1]	—	—
TX PLL bit clock frequency	Computed	—	—
RX Data rate	250 Mbps – 12700 Mbps	5000 Mbps	10312.5 Mbps (STD maximum)
RX CDR lock mode	Lock to reference, Lock to data, Burst Mode Receiver[2], lock to data with 2x gain[3]	Lock to data	—
RX CDR reference clock source	Dedicated and fabric	Dedicated	—
RX CDR reference clock frequency[4]	Based on transceiver data rate	—	—
RX JA Clock Frequency	Calculated based on configuration	—	—

Notes:

a. Enter the transceiver data rate (lane rate), and the TX clock division factor in the XCVR UI. Based on these settings, the TX_PLL base data rate is calculated. The TX_BIT_CLK frequency is half of the TX_PLL base data rate. The TX_PLL base data rate must be entered under the desired output clock option of the PF_TX_PLL block. The PF_TX_PLL generates the BIT_CLK output (connected to the TX_BIT_CLK_0/1 input of PF_XCVR).

b. When Burst Mode Receiver (BMR) is selected, LANE_X_CDR_LOCKMODE[1:0] port is exposed.

c. Default Gain, slower CDR lock time, lower jitter tolerance 2x Gain, faster CDR lock time, higher jitter.

d. This input frequency is given by the user to support the integer feedback divider of the receiver PLL. From the drop-down, enter a CDR reference clock frequency (MHz) value equal to the reference clock used to the Receiver PLL. The computation derives the feedback divider used to clock the receiver data path.

Table 2-7. Transceiver Interface PCS Settings

PCS Settings	Options	Default	Details
PCS-fabric interface width	8, 10, 16, 20, 32, 40, 64, and 80[1]	40	—
FPGA interface frequency[2]	Computed	—	—
PMA Mode	Enable CDR Bit-slip port	Enable	—
8b10b encoding/decoding	None	—	—
64b6xb gear box	64b66b 64b67b	64b66b	If 64b6xb mode is enabled, then PCS-Fabric interface width must be 32- or 64-bit.
64b66b gear box[3]	Enable disparity	Disabled	Enabled for 64b67b
	Enable scrambler/de-scrambler	Disabled	—
	Enable BER monitor state machine	Disabled	—
	Enable 32 bits data width	Disabled	—
64b67b gear box[3]	Enable BER monitor state machine	Disabled	Enable 32 bits data width
	Enable Disparity	Disabled	Cannot be enabled for 64b66b
	Enable Scrambler/de-scrambler	Disabled	—
	Enable 32 bits data width	Disabled	—
Soft PIPE interface	PCIe Gen1 (2.5 Gbps) PCIe Gen2 (5.0 Gbps)	PCIe Gen1 (2.5 Gbps)	—

Notes:

a. Dependent on PCS settings.

b. TX_CLK_G/R frequency = RX_CLK_G/R frequency = FPGA Interface frequency = data rate/(PMA-PCS width × PCS Gearing).

c. When **64b6xb Gear Box** is enabled, the **Enable Disparity**, **Enable BER monitor state machine**, **Enable Scrambler/Descrambler**, and **Enable 32 bits data width** options can be enabled or disabled independently for both **64b66b** and **64b67b**.

Table 2-8. Clocks and Resets

Interface Options	Options	Default	Details
Interface clock	Use as PLL reference clock	Disabled	When Use as PLL reference clock is selected, this exposes additional ports that permit connection to the PLL REFCLKs. LANEn_TX_CLK_TO_PLL_REFCLK LANEn_RX_CLK_TO_PLL_REFCLK This allow designs that require gearing other than 2:1 with wider fabric interfaces and use dedicated routing to the PLL. This is used in place of CLKDIV, which does not have the dedicated routing to PLL.
TX clock [2]	Global, Regional, Regional (Deterministic), Global Shared	Regional	See Table 1-16
RX clock [2]	Global, Regional, Regional (Deterministic), Global Shared, and NA	Regional	NA option must be selected when the PCS is configured in Soft PIPE mode (PCIe). See Table 1-16
Interface Resets	PMA Reset[1]	TX and RX	—
	PCS Reset[1]	Tx Only, Rx Only, Tx and Rx	RX Only – only RX side can be reset from the fabric. TX Only – only TX side can be reset from the fabric. TX and RX – both RX and TX sides can be reset from the fabric.
Optional Ports	Enable/Disable	—	TX_BYPASS port/TX_ELEC_IDLE port. See Table 1-10 or Table 1-12.
			RX_READY_CDR and RX_VAL_CDR ports. See Enhanced Receiver Management.
			JA_CLK port. See Jitter Attenuator.
Dynamic Reconfiguration	Enable Dynamic Reconfiguration Interface (DRI)	Disabled	—

Notes:

a. The minimum pulse width required is 16 clock cycles.

b. To toggle TX clock and RX clock, an active pulse is provided on PCS Reset and PMA Reset.

Preset configurations are available within the **Transceiver Interface** Configurator to speed up the transceiver configuration. Factory provided presets are available with the Libero release. Additionally, customized presets can be saved. See Transceiver Modes for more information.

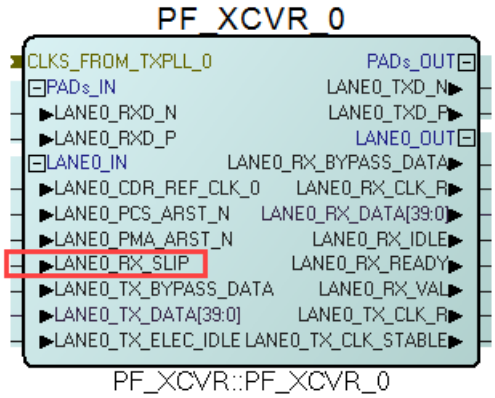
3. Select **Number of lanes** from 1 to 4 in the general settings configuration.
4. Enter the **Transceiver data rate** and select one of the **TX clock division factors**. The **TX PLL base data rate** is calculated in the GUI. The calculated TX PLL base data rate must be entered under the desired output clock option inside the PF_TX_PLL configurator. LANE#_TX_PLL_REF_CLK_#, LANE#_TX_BIT_CLK_0, and LANE#_TX_PLL_LOCK_# are included in CLKS_FROM_TXPLL_# BIF (bus interface). This connection is required between the TXPLL and Transceiver Interface.
5. Select the desired **CDR reference clock mode** and **CDR reference clock frequency** from the drop-down list based on the application.

Important: CDR reference clock frequency drop-down list is populated with valid frequencies based on the data rate.

6. Select the **CDR reference clock** source based on the design requirements. The dedicated clock adds a dedicated CDR_REF_CLK port whereas the fabric port only includes a port that can be connected to the fabric resources. The dedicated CDR_REF_CLK_0/1 port must be connected to the REF_CLK or REF_CLK_0/1 output of the PF_XCVR_REF_CLK block.
7. Select **PCS-Fabric interface width** from the GUI. This selection computes the FPGA interface frequency. The FPGA interface frequency is calculated based on the transceiver data rate, PCS-Fabric width, and the PCS settings/mode.
8. Click the GUI radio button to select the desired PCS mode. See Transceiver PCS Interface Modes for more information on PCS mode.
9. Select the desired interface clock options in the **Interface Options** GUI. See PCS/FPGA Fabric Interface.
10. For PMA Only modes - CDR Bit-slip, select the **Enable CDR Bit-slip port** to add the LANE#_RX_SLIP pin.

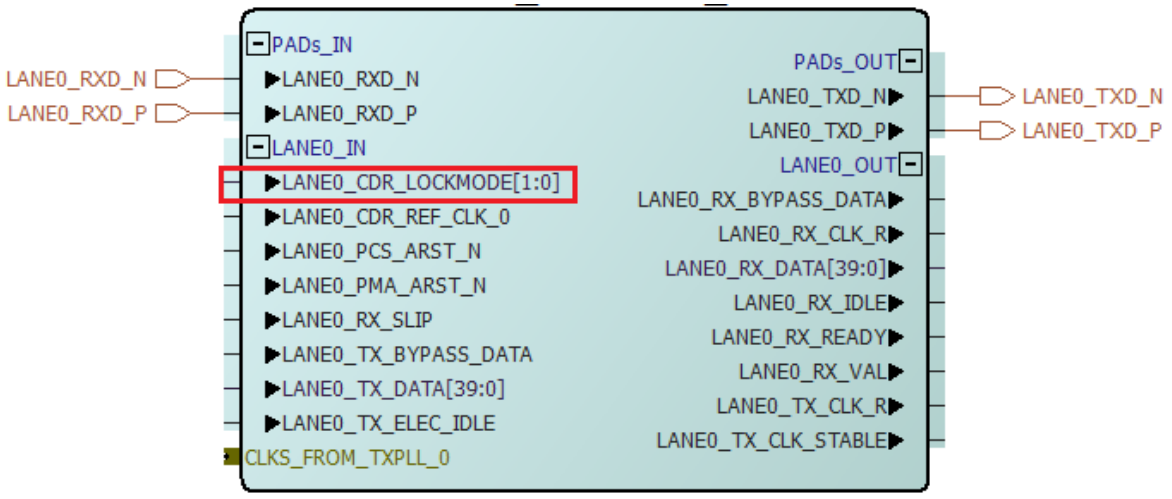
Figure 2-19. PMA Mode—Enable CDR Bit-Slip Port

Figure 2-20. XCVR Component With CDR Bit-Slip Port Enabled



Important: For information about RX_SLIP, see Bit Slip.

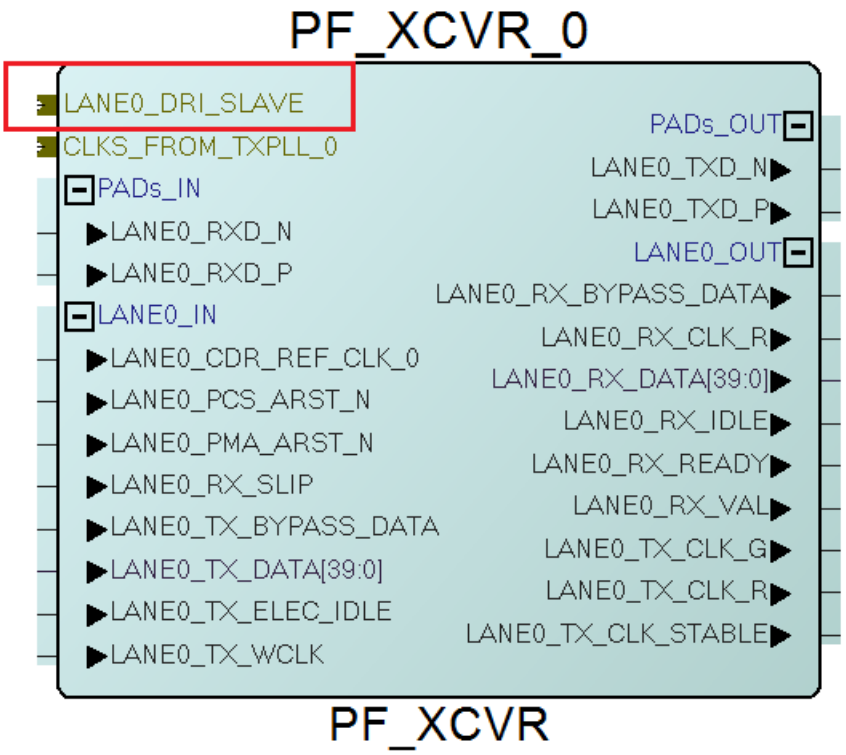
Figure 2-21. XCVR Component With BMR Port Enabled



Important: For information about CDR_LOCKMODE pins, see burst mode receiver in CDR Options.

11. Select the **Enable Dynamic Reconfiguration** to add the LANE#_DRI_SLAVE pins. See Dynamic Reconfiguration Interface for usage details.

Figure 2-22. XCVR Component With DRI Port Enabled



12. After making all of the selections in the Transceiver Interface Configurator, click **OK**.

When the transceiver interface configuration is complete, a PF_XCVR macro is generated by the Libero Software. The macro includes the ports based on the configuration. Figure 2-23 to Figure 2-27 shows sample PCS macros. The PF_XCVR macro is instantiated into the user design to customize the connectivity of the application.

Figure 2-23. Transceiver with ERM Example SmartDesign Component

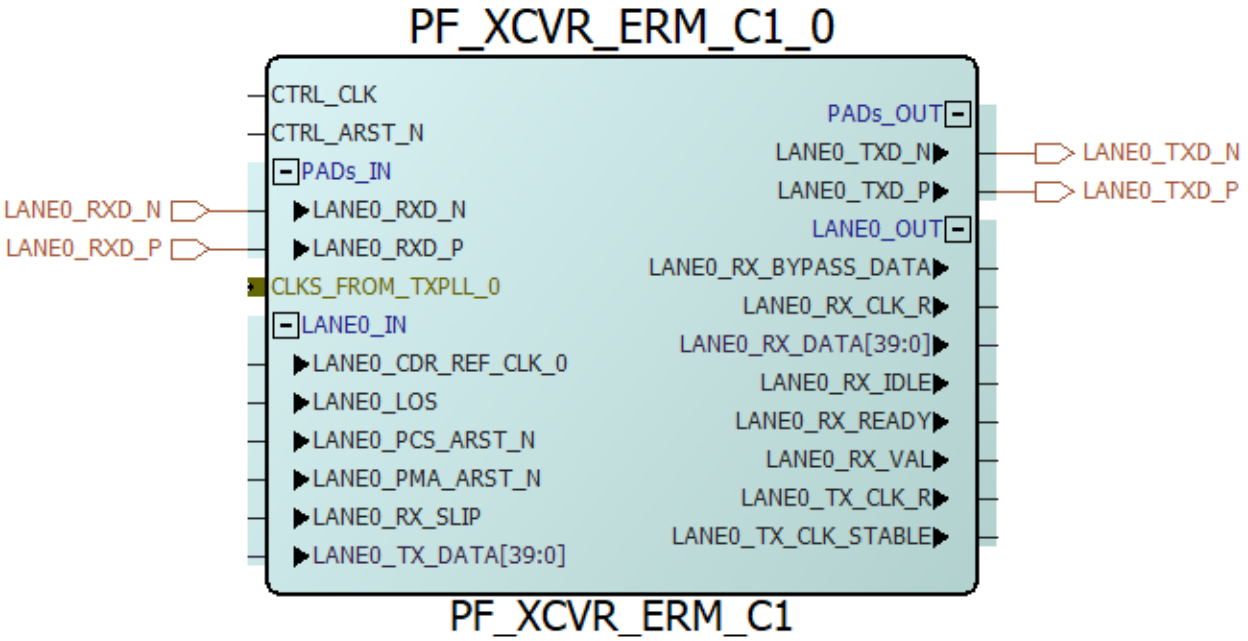


Figure 2-24. PMA Only PCS Example SmartDesign Component

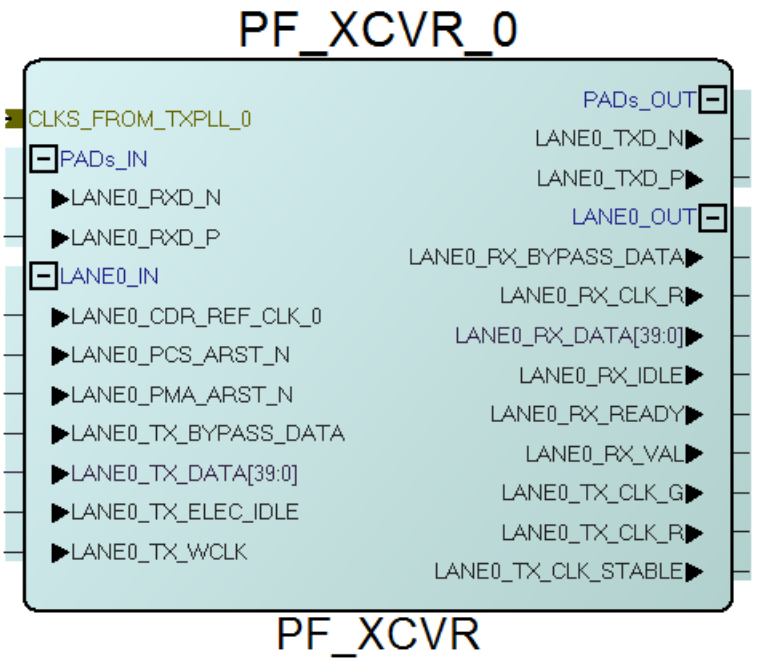


Figure 2-25. 8b10b PCS Example SmartDesign Component

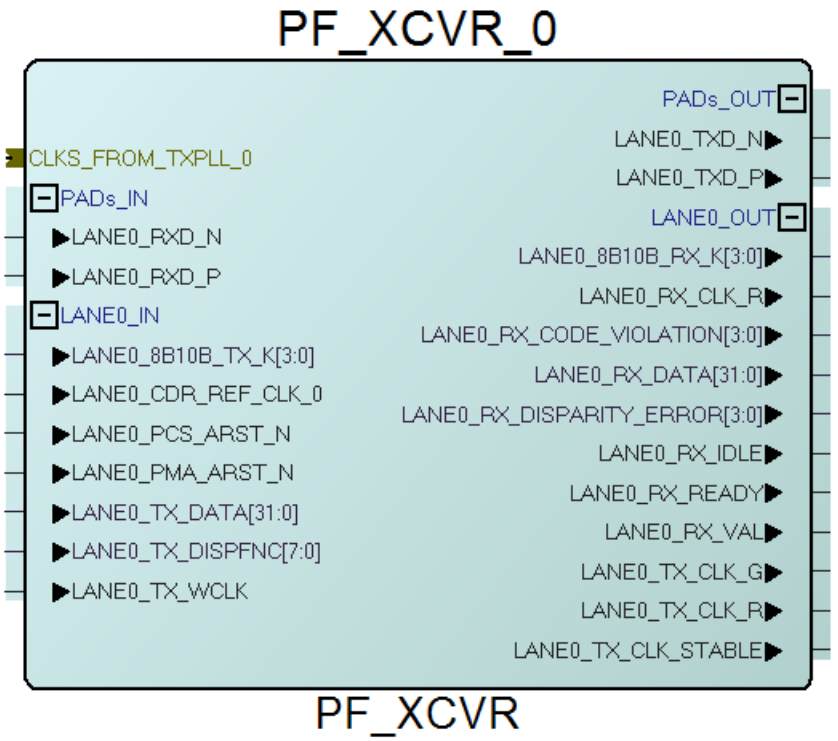


Figure 2-26. 64b66b PCS Example SmartDesign Component

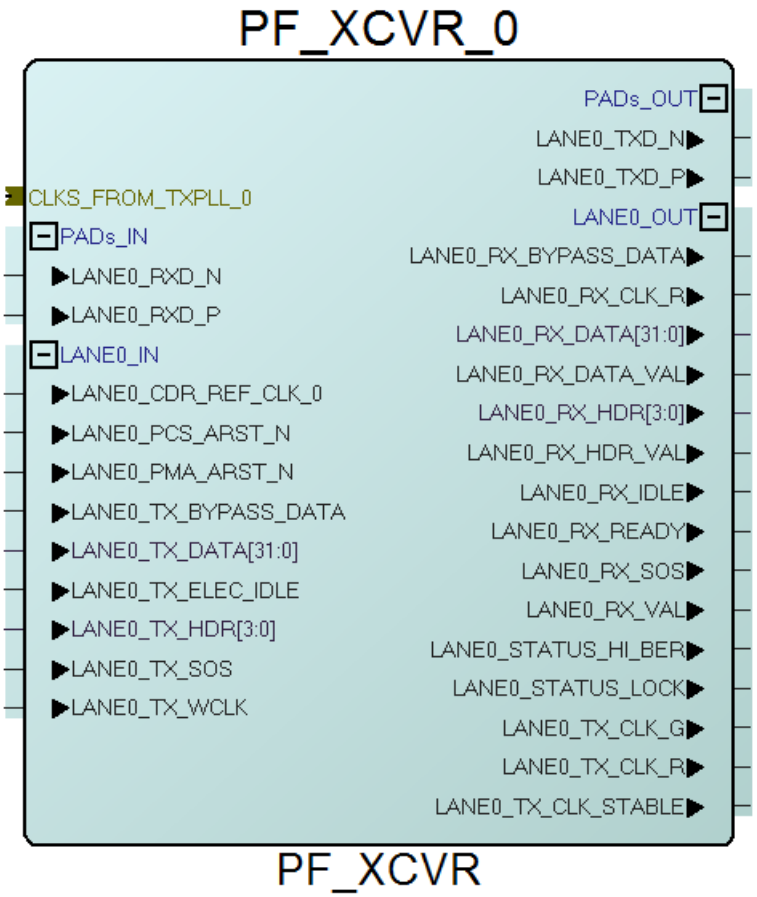
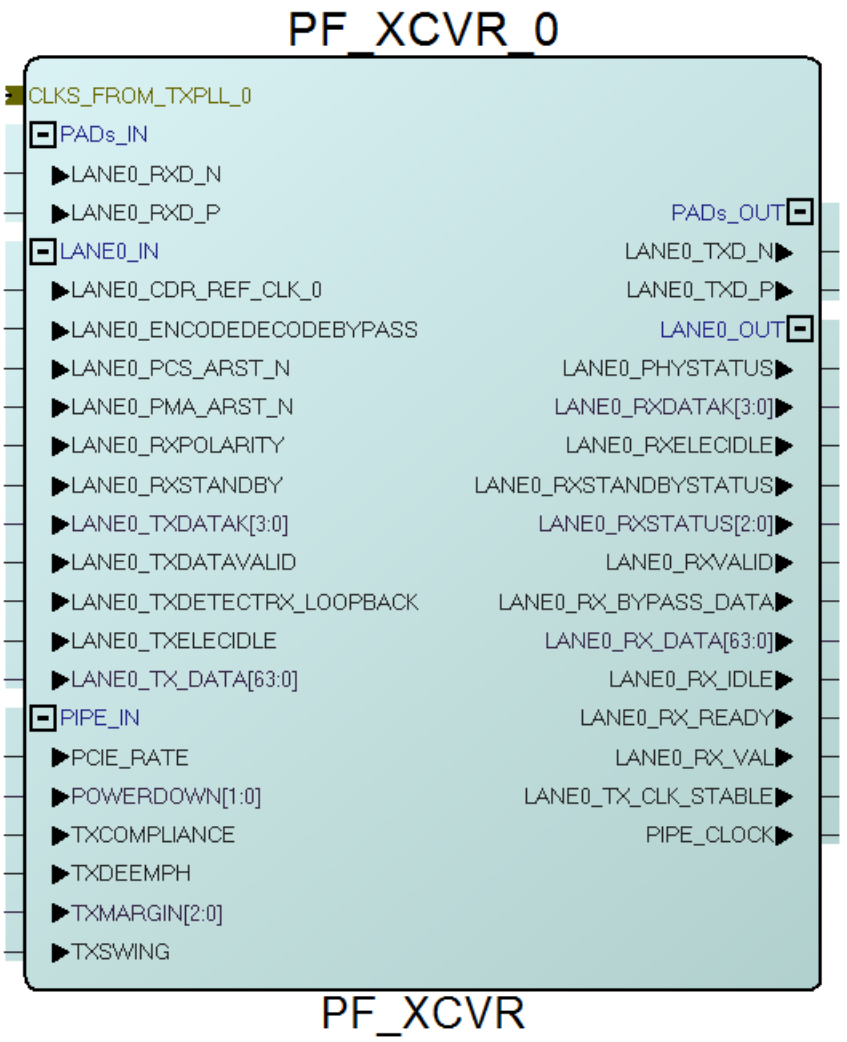


Figure 2-27. Soft PIPE PCS Example SmartDesign Component



After building the PF_XCVR, PF_TX_PLL, and PF_XCVR_REF_CLK cores, the transceiver subsystem must be connected together in the SmartDesign canvas. Typically, the REF_CLK and/or FAB_REF_CLK outputs of the PF_XCVR_REF_CLK are connected to the respective inputs of the PF_XCVR and the input REF_CLK of the PF_TX_PLL. LANE#_TX_PLL_REF_CLK_#, LANE#_TX_BIT_CLK_0, and LANE#_TX_PLL_LOCK_# are included in CLKS_FROM_TXPLL_# BIF (bus interface). This connection is required between the TXPLL and transceiver interface. The SmartDesign component must then be generated.

Figure 2-28. Completed Transceiver Subsystem

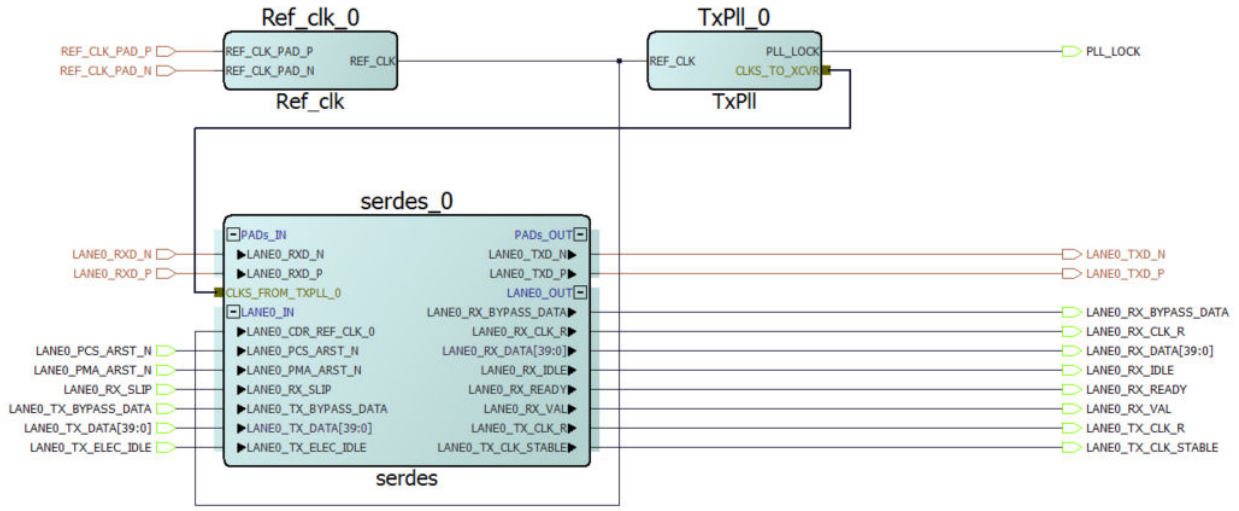
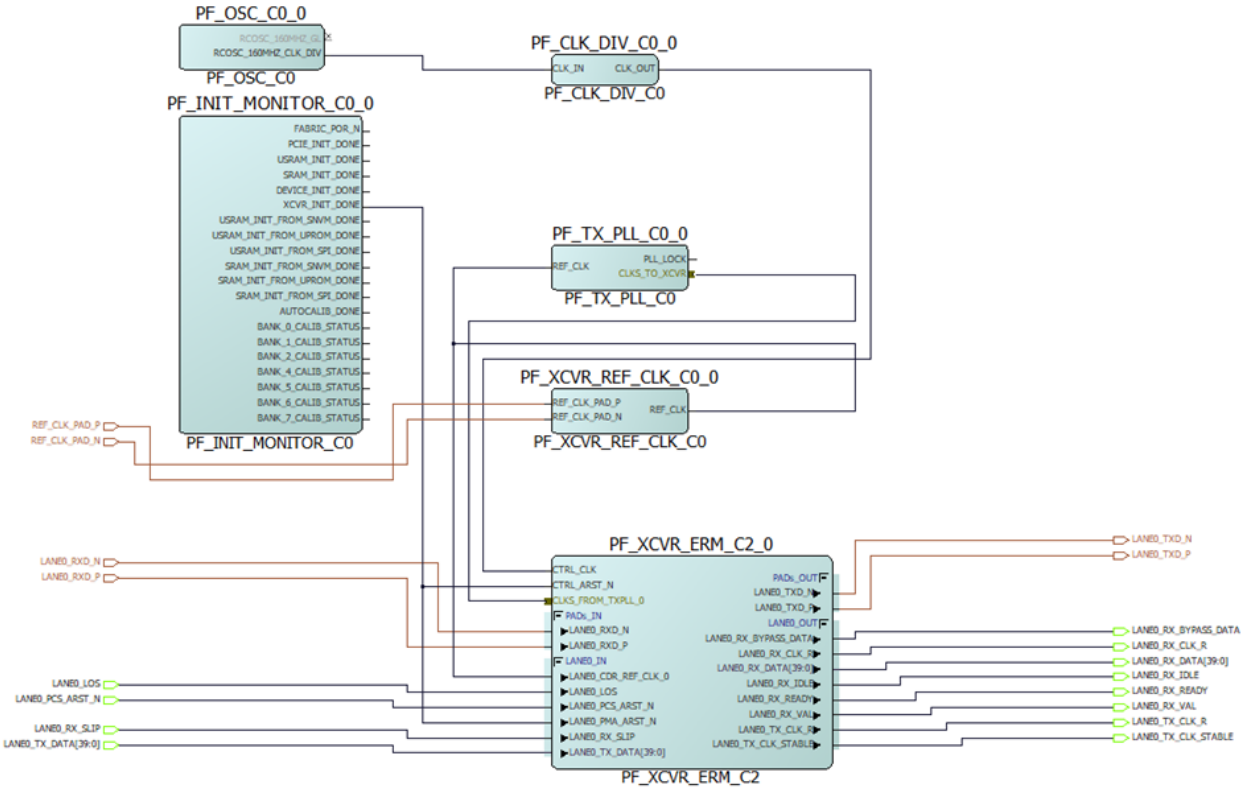


Figure 2-29. Completed Transceiver Subsystem with ERM



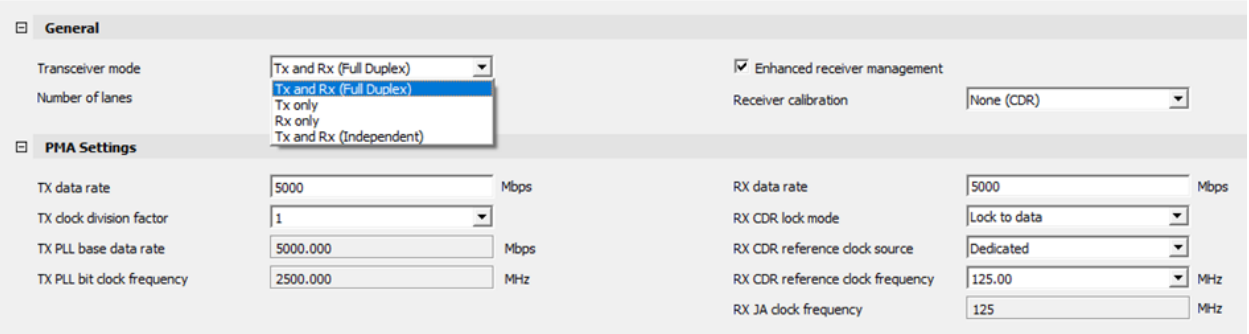
See the port list tables in Transceiver PCS Interface Modes for complete pin descriptions generated with the Transceiver Configurator.

2.2. Transceiver Modes (Ask a Question)

The transceiver architecture permits several ways to use the transmit and receive portions of the PMA and PCS. A TXPLL is not instantiated by the Transceiver configurator. The user must instantiate the TXPLL. The Transceiver configurator gives the user options to use the Tx and Rx together or independently or alone. The transceiver modes require the user to input the specific

design requirements to generate the transceiver block. The modes also define the operation of the PMA_ARSTN and PCS_ARSTN pins and how these pins reset the transceiver functionality.

Figure 2-30. Transceiver Modes shown in Transceiver GUI



2.2.1. Full-Duplex Mode (Ask a Question)

Tx and Rx Duplex support is the most common use of the transceiver and often referred to as full-duplex. It is defined as the mode where the Rx and Tx portions of the PMA share resources such as clocking and reset functions. Full duplex uses a common clock for both the Tx and Rx, that is, the base rates and PCS widths are the same. Both PMA_ARSTN and PCS_ARSTN resets both the Tx and Rx of the PMA and PCS, respectively.

Full-duplex configures the XCVR with common Tx and Rx data rates. Although, users can see independent control for data rate and PCS-Fabric widths, a Libero SoC DRC prevent users from selecting different rates/widths when TX and RX is in full-duplex mode. In full-duplex mode, the user must match the Rx and Tx requirements.

2.2.2. Half-Duplex Mode (Ask a Question)

Another mode of the transceiver is half-duplex mode. It is defined by allowing parts of the Rx and Tx to operate independently. Half-duplex modes optimize the use of lanes within XCVR quads by allowing the separation of Rx and Tx within a lane. To address the half-duplex modes, the following options are available to the transceiver.

- Tx only
- Rx only
- Tx and Rx (Independent)

When users select Tx only or Rx only option, then one physical XCVR lane is completely used, and the unused Rx or Tx cannot be used. When users select Rx only in the GUI, the Tx GUI options are marked as N/A for display purposes. In the same way, when users select Tx only in the GUI, the Rx GUI options are marked as N/A for display purposes. However, in both cases, the unused Tx or Rx parameters are still set to some valid values in the generated core (based on the default parameter values from the core). PMA_ARSTN resets both the Tx and Rx of the PMA, however, the PCS_ARSTN will reset the Tx only or Rx only accordingly.

If users intend to implement two independent half-duplex channels in the same physical XCVR lane then select the 'Tx and Rx (Independent)' option in Transceiver mode. By using the 'Tx and Rx (Independent)' option, users can specify a different data rate for the Rx channel and Tx channel. When configured Tx and Rx (Independent), an additional input pin (LANE#_REF_CLK) is exposed on the component. See the related port list for pin definition. Additionally, the PCS-FABRIC width can also be different between Rx and Tx. However, if half-duplex applications are to be merged into one XCVR lane, then both applications must use the same basic mode of the PCS. The reason for this is, system register control for steering the data path among the four basic modes (PIPE, 64B6xB, 8B10B, NATIVE/PMA) is shared for Tx and Rx.

Libero SoC programs the appropriate registers within the transceiver to power down the serializer or de-serializer in Tx only /Rx only Transceiver modes.

Table 2-9. PCS Modes Supported

Transceiver Mode	PCS Mode supported
Tx and Rx (Full Duplex)	PMA, 8b10b, 64b6xb, PIPE
Tx only	PMA and 8b10b
Rx only	PMA and 8b10b
Tx and Rx (Independent)	PMA and 8b10b

Tx only, Rx only, and Tx and Rx (Independent) modes are supported only in PMA mode. In 8b10b mode, both Tx and Rx must use the same fabric interface width. Tx/Rx Independent option PCS mode applies for both Tx and Rx and cannot be different within the lane. For example, the Tx cannot be 8b10b when the Rx is 64b66b.

The TXPLL supports Fractional-N frequency synthesizer, and the CDR PLL only supports integer-based synthesis. Typically, the same REFCLK is used for both the TXPLL and CDR PLL for full-duplex lane. The user has the option to use integer-based for the TXPLL if the frequency is achieved without issue.

If the TXPLL uses Frac-N rather than integer-based to synthesize the bit clock, the CDR PLL will not have the Frac-N capability. Consequently, the user needs to determine if the PPM tolerance of the CDR is enough to compensate for the TXPLL rate.

2.3. Libero Generated Files (Ask a Question)

Libero SoC software automatically generates the required files after stepping through the design entry steps of the transceiver. The following files are created:

- Netlist file—the RTL netlist instantiates the transceiver macros and related RTL wrappers based on protocol specific functions.
- <design name>.sdc file—Timing constraints file in the case of TXPLL and XCVR configurators.
- <module name>.v, _syn_comps.v, _pre_comps.v
  - HDL source files for all Synthesis and Simulation tools.
  - HDL source files for Synopsys SynplifyPro Synthesis tool.
  - HDL source files for Mentor Precision Synthesis tool.

Note: The entire file list and their file paths are maintained in mainfest.txt at <Project Directory>/component/work/<Component name>/*_manifest.txt files.

Note: XCVR and PCIESS initialization data is available in the generated netlists of the TXPLL, XCVR, and PCIE blocks. This is used to configure the blocks specifically for users design customization. See Transceiver Initialization.

2.4. Design Constraints (Ask a Question)

Design constraints are either requirements or properties in the design. Constraining ensures transceiver designs meets their performance goals, embedded block locations, and pin assignment requirements.

The software supports timing, physical, and netlist optimization constraints for the transceivers. Design constraints can be set by either using Microsemi's interactive tools or by importing constraint files directly into the design session.

2.4.1. Timing Constraints (Ask a Question)

Timing constraints for the designs are required to meet the performance goals of the transceiver interface. Specify timing constraints directly in the SDC or by using the timing constraints editor.

The following timing constraints must be methodically introduced into the design. Libero assists by automatically generating timing constraints related to transceiver clock usage.

A component-level SDC file is created for every XCVR instance, which is pulled into the SDC file created for the entire Libero design project as shown in the following example.

CDR reference clock source

Dedicated

```
create_clock -period <T> [get_pins {LANE<n>/REF_CLK_P}]
```

Fabric

```
create_clock -period <T> [get_pins {LANE0<n>RX_REF_CLK}]
```

Interface clock

Global

```
create_clock -period <T> [get_pins {LANE<n>/TX_CLK_G}]
create_clock -period <T> [get_pins {LANE<n>/RX_CLK_G}]
```

Regional or Regional (Deterministic)

```
create_clock -period <T> [get_pins {LANE<n>/TX_CLK_R}]
create_clock -period <T> [get_pins {LANE<n>/RX_CLK_R}]
```

Global-shared Mode

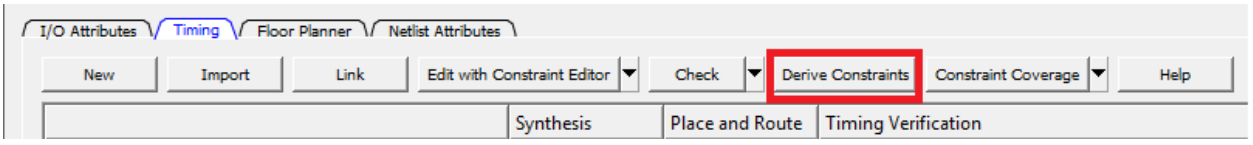
```
create_clock -period <T> [get_pins {LANE<n>/TX_CLK_G}]
create_clock -period <T> [get_pins {LANE<n>/RX_CLK_G}]
create_clock -period <T> [get_pins {LANE<n>/TX_CLK_R}]
create_clock -period <T> [get_pins {LANE<n>/RX_CLK_R}]
```

Users need to use the **Derived SDC** file generated by clicking the **Derive Constraints** in the **Timing** tab of the **Constraint Manager** window of Libero SoC software.

The following example shows a Libero project with transceiver.

```
create_clock -name {REF_CLK_PAD_P} -period 6.4 [ get_ports { REF_CLK_PAD_P } ]
create_clock -name {My_Project_0/my_xcvr_0/PF_XCVR_0/LANE0/TX_CLK_R} -period 16 [ get_pins
{ My_Project_0/my_xcvr_0/PF_XCVR_0/LANE0/TX_CLK_R } ]
create_clock -name {My_Project_0/my_xcvr_0/PF_XCVR_0/LANE0/RX_CLK_R} -period 16 [ get_pins
{ My_Project_0/my_xcvr_0/PF_XCVR_0/LANE0/RX_CLK_R } ]
create_clock -name {My_Project_0/my_pll_0_0/pll_xvier_0_0/txpll_isnt_0/DIV_CLK} -period 8
[ get_pins { My_Project_0/my_pll_0_0/my_pll_0_0/txpll
```

Figure 2-31. Derive Constraints using Constraints Editor



The transceiver can source three different clocks into the fabric—TX_CLK, RX_CLK, and the REFCLK (FAB_REF_CLK). These clock resources have several optional networks such as global or regional routing structures. These global clock resources are susceptible to clock jitter induced by design-specific simultaneous switching activity and the response of the internal device and board-level power distribution network (PDN).

Libero SoC (v2022.1 and later) design software flow is enhanced to automatically apply clock uncertainty constraints to account for the jitter on these clock nets during static timing analysis

(STA). The Libero SoC routine analyzes the loaded clock network and uses data driven calculations to apply clock uncertainty that can be anticipated on a typical switching clock net. In addition to global clocks sourced from the transceiver, other FPGA global clock resources are susceptible to clock jitter, including PLLs, DLLs, internal RC Oscillators, and the global nets themselves. See PolarFire Family Clocking Resources User Guide for more information about FPGA clocking resources. Additionally, the device specific Datasheet includes the clock jitter specifications that are used by the Libero SoC automatic clock uncertainty flow.

2.4.2. Physical Constraints (Ask a Question)

Transceiver designs require physical constraints. These constraints provide placement guidance for the embedded transceiver blocks such as XCVR_REF_CLK, XCVR_TXPLL, and XCVR. Select the placement of the RXD and TXD transceiver pins and reference clock input pins by adding location constraints to the design PDC file. Placement fails if the XCVR lane TX and RX I/Os, XCVR_REF_CLK I/Os and TX_PLL locations are not explicitly placed. The Libero I/O Editor assists with the creation of physical constraints using a GUI, see Adding Physical Constraints Using Libero.

For physical constraining by the user, the instance name is mapped to top-level physical pins based on the device and package pin-outs found in the physical pin assignment tables (PPAT). The user provides the constraint to the PDC file as listed in the following table.

For XCVR Lanes:

```
set_io –port_name <port name> \-pin_Name <package pin name> \-DIRECTION INPUT
```

For XCVR REFCLK:

```
set_io -port_name <refclk port name> \
-pin_name <package pin>\
-DIRECTION INPUT \
```

`-io_std`: See PolarFire FPGA and PolarFire SoC FPGA User I/O User Guide for information about available IO Standards.

`-ODT VALUE`: See PolarFire FPGA and PolarFire SoC FPGA User I/O User Guide for information about available ODT values.

Table 2-10. Physical Constraint Instances For XCVR

XCVR Instance[1]	Corresponding Top-level Pins[2]
Q#_LANE0	XCVR_Q#_RX0_P/N XCVR_Q#_TX0_P/N
Q#_LANE1	XCVR_Q#_RX1_P/N XCVR_Q#_TX1_P/N
Q#_LANE2	XCVR_Q#_RX2_P/N XCVR_Q#_TX2_P/N
Q#_LANE3	XCVR_Q#_RX3_P/N XCVR_Q#_TX3_P/N
Q#_TXPLL_SSC	Dependent on design clocking requirements (See Figure 1-56 and Figure 1-57.
Q#_TXPLL0	Dependent on design clocking requirements (See Figure 1-56 and Figure 1-57.
Q#_TXPLL1[3]	Dependent on design clocking requirements (See Figure 1-56 and Figure 1-57.
Q#_REFCLK_A	XCVR_Q#A_REFCLK_P/N
Q#_REFCLK_B	XCVR_Q#B_REFCLK_P/N
Q#_REFCLK_C	XCVR_Q#C_REFCLK_P/N

(1) Q# = Transceiver Quad identifier (Q0, Q1, and so on.)
(2) Pin-mapping to top-level pins can be found in the pin-assignment tables (PPAT) for each device and package type. For more information about PPATs, see or PolarFire SoC FPGA webpage.
(3) For PolarFire FPGA, only TXPLL_SSC and TXPLL0 is available for Quad4 and Quad5. Refer Figure 1-56. The default differential REFCLK value is set to 100 (`-ODT_VALUE 100`). To disable the ODT value, set the value of the differential REFCLK to 0 (`-ODT_VALUE 0` in the PDC file.

2.4.2.1. DFE Coefficients (Ask a Question)

DFE Calibration of the DFE Coefficients is not be performed in Static DFE. See DFE Calibration.

To set the required registers with static values, users must enhance the “set_io” PDC command to add new attributes. The new attributes that need to be added are highlighted in the below PDC command.

PDC command example:

```
set_io -port_name LANE0_RXD_N \
-RX_DFE_COEFFICIENT_H1 20 \
-RX_DFE_COEFFICIENT_H2 20 \
-RX_DFE_COEFFICIENT_H3 20 \
-RX_DFE_COEFFICIENT_H4 20 \
-RX_DFE_COEFFICIENT_H5 20 \
-DIRECTION INPUT
```

The RX_DFE_COEFFICIENT attributes are optional (applied only when Static calibration is selected). These attributes take integer values between 0 and 15. The corresponding register fields are 5 bits wide in all cases with the MSB bit reserved for sign bit.

2.5. Adding Physical Constraints Using Libero (Ask a Question)

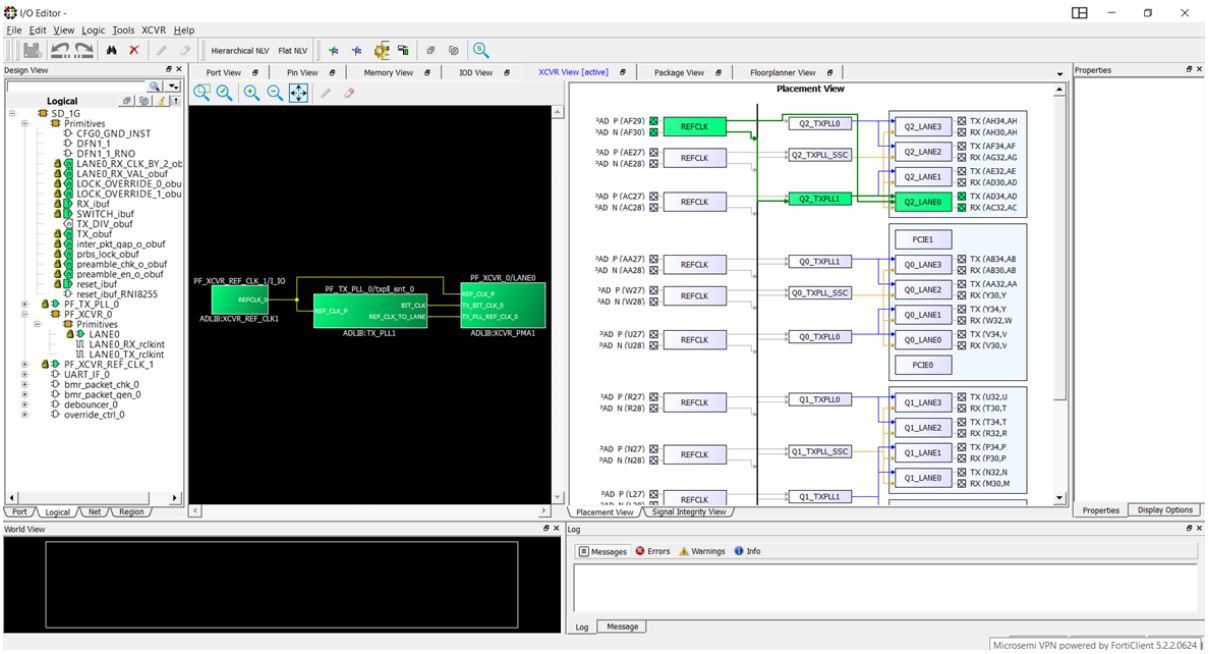
The transceiver I/O pin assignments can be made with PDC commands and passed as physical design constraints to the physical layout tool. The I/O PDC constraints (for XCVR LANE RX pins, TX pins, and XCVR_REF_CLK pins) must be specified in an I/O PDC file. The chip plan location constraints for TXPLL must be specified in a floor plan PDC file. The I/O Editor provides a GUI tool designed to make I/O pin assignments graphically and user-friendly, as an easy alternative to writing PDC commands. When the pin assignment is committed and saved in the Pin Planner, a PDC file is created. This PDC file can then be passed to the Place and Route tool as a physical design constraint.

2.5.1. Invoking the Pin Planner (Ask a Question)

To invoke the Pin Planner, the design must be in the post-synthesis state.

1. Invoke the Constraint Manager from the Design Flow window (**Design Flow** > **Manage Constraints** > **Open Manage Constraints View**).
2. In the Constraints Manager, select the **I/O Attributes** tab and then select Edit > Edit with I/O Editor (**I/O Attributes** > **Edit** > **Edit with I/O Editor**). The I/O Editor opens.
   Placement View tab—Presents a physical view of the transceiver connectivity, including transceiver lanes (XCVR), Reference Clock (REFCLK), and Transmit PLLs (TxPLL).

Figure 2-32. IO Editor GUI

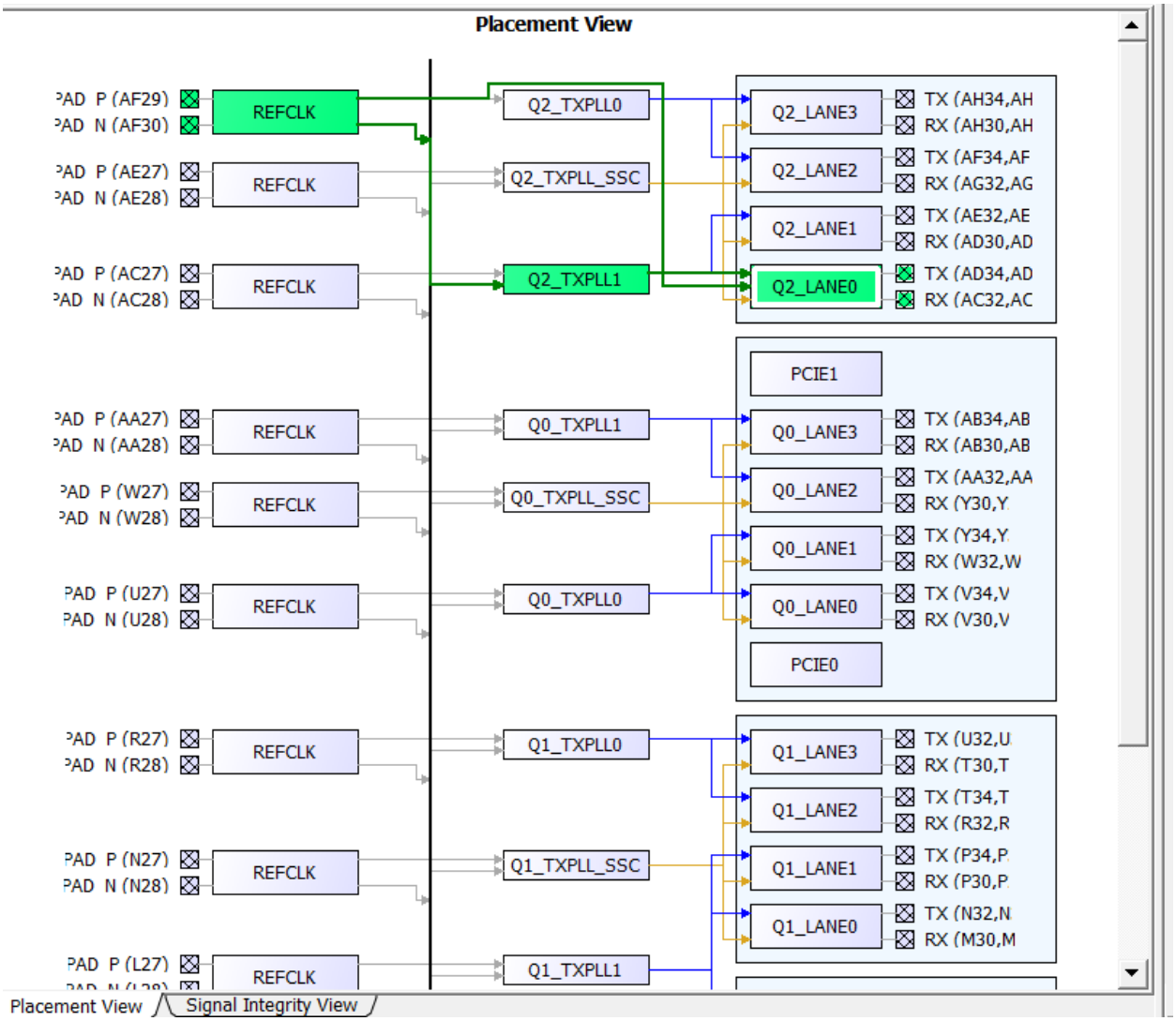


The transceiver view allows you to make assignments—XCVR, REFCLK, and TxPLLs. It has two views:

- A schematic view of the REFCLK, the TXPLL, and the XCVRs they drive.
- A graphical physical view of the REFCLK, its connection from the PADS to the TXPLL and the XCVR lanes.

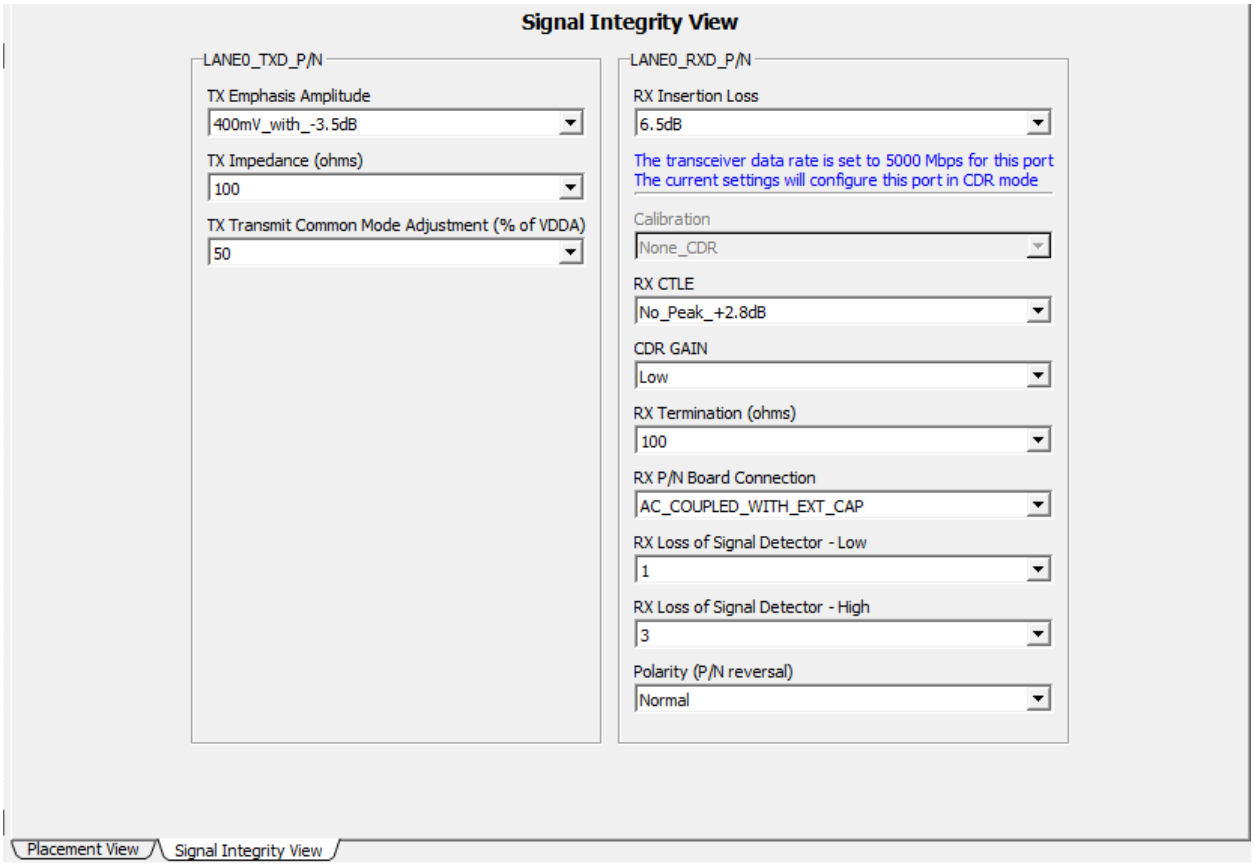
The schematic and physical views provide design rule guidance for selecting legal connectivity combinations between the XCVR I/O, TxPLLs, and REFCLK inputs. It provides logical mapping to the device physical resources using connection rules of the device.

Figure 2-33. XCVR Placement Tab



The following figure shows the XCVR **Signal Integrity View** tab. When user selects lane in Left Panel, the user can view and change the signal integrity parameters for the Rx and Tx transceiver ports.

Figure 2-34. XCVR Signal Integrity Tab



For more information about tuning the transceiver signal integrity, see Signal Integrity Conditioning.

2.6. Transceiver Initialization (Ask a Question)

The transceiver is initialized after the user customizes the transceiver or PCIe features with the associated configurators. The transceiver initialization data refers to the memory files and the initialization clients required for the three stages of initialization that is executed by the device at start-up. For proper functioning, the user design must generate initialization data before programming (running Generate Bitstream or the Export Programming File, Export Programming Job, Export Programming Job Data, Export SmartDebug Job Data, Generate SPI Flash Image, Generate SPI Flash Image, and Export SPI Flash Image).

1. The operations of Configuration and Generation are separated into two individual design flow steps called **Configure Design Initialization Data and Memories** and **Generate Design Initialization Data**.
2. Invoking the **Configure Design Initialization Data and Memories** opens up the initialization UI.
3. Clicking **Apply** in the **Design Initialization** tab only saves the configuration for initialization data, it does not generate the initialization data.

For design with transceivers, the Libero software automatically incorporates the required initialization data into the design project without any manual steps by the user. With transceiver designs, there are instances where a custom configuration file is used for cases not directly supported by Libero SoC. In these rare instances, the custom configuration file is used to compile the design to initialize the transceiver components. XCVR configuration data present in the design is initialized during power-up using initialization clients placed in the non-volatile sNVM memory. Initialization data is generated and stored in binary (*.mem) file of the second stage client, which initializes the TXPLL, PCIe and XCVR blocks present in the design.

The programming steps (Generate Bitstream and the Export steps Export Programming File, Export Programming Job, Export Programming Job Data, Export SmartDebug Job Data and Export SPI Flash Image) depend on the Generate Design Initialization Data. Libero automatically runs the Generate Design Initialization Data, if it is not already in pass state, when any of these programming steps are run. This ensures that initialization data is always present and up-to-date before programming.

2.6.1. Transceiver Initialization Data (Ask a Question)

After completion of the Libero SoC customization, the XCVR, PCIe, TxPLL, and CCCs are autonomously configured and initialized during the design generation. The register customization can be reviewed in a report generated during the **Generate Design Initialization Data** step within Libero SoC. A “configuration Report for SERDES XCVR, PCIe, CC, Transmit PLL” is generated that records the value in the related registers. The report is located in the *project\designer\impl\Design_Initialization_Data_report.xml*.

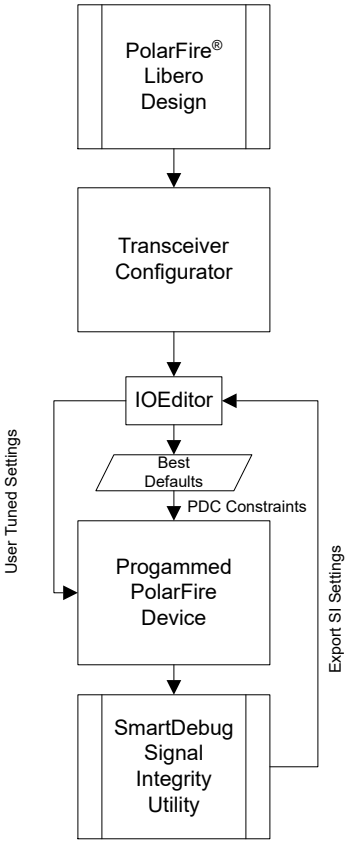
3. Signal Integrity Conditioning (Ask a Question)

The transceivers have many tuning adjustments for the analog portions of the PMA allowing for signal integrity optimizations to the system. These features include Rx Continuous Time Line Equalization (CTLE), Rx termination, polarity inversion, Pre- and Post-cursor output emphasis, output impedance settings, and Tx amplitude adjustment. Transceivers are programmed at startup from default configuration information known to the Libero SoC software based on protocol and data rate specifics supplied by the user during design customization and generation. The Libero configurators generate the Rx and Tx default settings that initialize the transceivers at power on or device reset by toggling the DEVRSTN pin.

Other embedded transceiver features include Decision Feedback Equalization (DFE) within the receiver PMA that further corrects signal imperfections caused by PCB losses and quality of the signal being driven to the Rx. The Rx path also includes an input signal eye monitoring feature.

The user can also control the transceiver settings after device programming using the FlashPro programming cable and the SmartDebug Transceiver software utility. Transceivers have a memory mapped Dynamic Reconfiguration Interface (DRI) allowing SmartDebug to communicate with the transceiver blocks to change and monitor the transceivers in real-time. This feature provides debugging capabilities and altering of the transceivers for optimized performance in the system. After the final SmartDebug signal integrity optimization, the user can export the tuned information back into the Libero SoC software for future design regeneration.

Figure 3-1. Signal Integrity Conditioning Flow



Building a transceiver based design using Libero SoC software flow allows flexibility to improve the transceiver performance within the system. The Libero SoC software sets initial good defaults for

the user's custom design based on input information to the transceiver configurators. The user can change the associated transceiver input and output settings using the IO Editor after initial design generation. The user can further enhance the signal integrity qualities of the transceiver after design place, route, and bitstream generation by using the SmartDebug capabilities. This allows user to continue the signal integrity tuning after programming the device in the system and provides a feedback path to include the customized signal integrity settings in subsequent design regeneration.

Both the Signal Integrity views of the IO Editor and SmartDebug Transceiver present the same signal integrity settings for both the transmitter and receiver. The available settings are predefined by the factory providing a flexible means to adjust the XCVR.

3.1. Transmitter (Ask a Question)

The transmitter takes parallel data from the FPGA fabric through the PCS-fabric interface block and gearing logic. The data passes through the PMA-PCS interface to the serializer to create a high-speed serial data stream using the serial clock provided from the transmit PLL. For more information about transmitter, see Transmitter.

3.1.1. Transmit Emphasis and DC Amplitude (Ask a Question)

Transmit emphasis and DC amplitude settings adjust the transmitter output drivers. The settings are provided from a drop-down menu of pre-defined combinations of Tx emphasis and DC amplitude.

The DC amplitude is adjusted by changing the driver segments to deliver a differential swing to the receiving device. The trade-off between large output-swing is power consumption and output return loss. The transmit emphasis adjusts the magnitude of the output based on the prior bit values reducing the successive bits. This transition emphasis compensates for the channel losses and opens the signal eye at the far-end receiver. This is accomplished by deemphasizing or weighting driver taps with negative dB adjustment.

Adjustment of the Tx amplitude and Tx emphasis are used to match the PCB interconnect losses. For information about Predefined Transmit Emphasis and DC Amplitude settings, see AC483: PolarFire FPGA Transceiver Signal Integrity Application Note.

3.1.2. Impedance (Differential) (Ask a Question)

This feature is used to add—85Ω or 100Ω or 150Ω—calibrated internal impedances onto the differential outputs. This setting adjusts the impedance selected to match the system to optimize performance.

3.1.3. Tx Insertion Loss (Ask a Question)

Table 3-1 lists the recommended transmit amplitude and emphasis values for short, medium, and long applications. User must manually adjust the amplitude/emphasis to match the Tx channel of the PCB. For Rx, these values can be set in Libero.

Table 3-1. Amplitude and Emphasis

Channel Type	Amplitude (mV)	Emphasis (dB)
Short	400	–3.5
Medium	800	–6
Long	1000	–6

3.1.4. Transmit Common Mode Level (Ask a Question)

Transmit common mode level is used as a percentage—50% or 60% or 70% or 80%—of full common mode level or VDDA. It is only adjusted when DC coupled (for example with SDI or high CID protocols). This is to match the common-mode restoration to the DC coupled receiver. For AC coupled systems, the level must remain as default.

3.2. Receiver (Ask a Question)

The receiver deserializes high-speed serial data received through the input buffer by creating a parallel data stream for the FPGA fabric and recovering the clock information from the received data. For more information about receiver, see Receiver.

3.2.1. Rx Insertion Loss (Ask a Question)

Receiver insertion loss is used to match the PCB qualities of the system. The predefined settings are used to statically adjust the receiver CDR and DFE. The CDR and DFE parameters are assigned based on a targeted data rate and backplane model. CDR uses only the CTLE capabilities of the PMA receiver. The DFE uses combination of CTLE with DFE optimizations. To start the transceiver design, select the Rx Insertion loss values in Libero. Libero adjusts to the best known setting based on the data rate. If the pre-defined settings do not achieve the best performance, users can fine-tune the Rx CTLE settings to precisely match their system requirements after initial system bring-up.

Backplane or PCB Length, that is, Reach definitions are based upon number of connectors and overall insertion loss as per the CEI-11G SR, MR and LR specifications defined by Tyco Electronics Z-Pack Tinman testing platform. See Transceivers Insertion Loss.

See AC483: PolarFire FPGA Transceiver Signal Integrity Application Note for information about the data rate range and models of backplane length.

3.2.2. Rx CTLE (Ask a Question)

CTLE at the receiver end is a typical equalization technique for equalizing the incoming signal to a flat response. CTLE is used to reduce the low-frequency component of the signal while boosting the high-frequency component. The receiver equalization settings are a function of the cut-off frequency and the amplitude gain across data rates. The equalizer circuitry can be tuned to compensate for the signal distortion due to the high frequency reduction of the physical channel of the PCB and interconnect.

For example, an under-equalized channel cannot adequately open the eye, whereas over-equalization can produce a channel with high jitter.

Correct equalization has optimal eye opening with low noise and low jitter. Rx insertion loss default settings sets the CTLE. AC483: PolarFire FPGA Transceiver Signal Integrity Application Note lists the combination of settings that help the user to find the best response to the incoming signal based on data rate “buckets”. Within each bucket, there are optimized settings based on gain and peaking. When using DFE designs, users must not alter the predefined values of CTLE. These values are optimized to work in conjunction with the DFE configuration based on Rx insertion defaults as listed in following table.

3.2.3. Rx Termination (Ask a Question)

Within the Rx buffer, a calibrated input termination can be set having three—85Ω or 100Ω or 150Ω —available differential impedances. The input impedance can be configured to match the system requirements.

3.2.4. AC/DC Coupled Connection (Ask a Question)

DC coupled connection option is set for using AC or DC coupled channels to the receiver. The device does include an internal coupling capacitor option, however, it also includes internal biasing circuitry required for AC coupled applications.

- AC coupled with external cap. VICM is internally generated.
- DC coupled – Link partner is in control of the Vcim through the incoming signal.

Note: For AC coupled with external cap, user must still place ac-coupling capacitor on PCB for AC coupled connection.

AC-coupling or DC-coupling is configured through the XCVR Signal Integrity options, which is available in Libero SoC v12.2 and later. Coupling options can be set in the Signal Integrity View tab (open the I/O Editor > XCVR View).

3.2.5. Loss-of-Signal Detector (Ask a Question)

Loss-of-signal (LOS) detector (LOW and HIGH value) sets the requirement of the voltage detector. The following table lists the upper and lower set points for a LOS to ensure that a good signal is applied into the receiver. Referring to data sheet Loss-of-Signal detect, Peak Detect Range (VDETLOW), below the Peak Detect Range minimum value setting is interpreted as “no signal” and above the maximum number will interpreted as “signal”. In between the minimum and maximum values are considered indeterminate.

Table 3-2. LOS Range

Setting	Range
PCIE	Preset PCIe low threshold setting = 1 For Min and MAX values of VDETLOW, see respective PolarFire FPGA Datasheet or PolarFire SoC Datasheet.
	Preset PCIe high threshold setting = 2 For Min and MAX values of VDETLOW, see respective PolarFire FPGA Datasheet or PolarFire SoC Datasheet.
SATA	Preset SATA low threshold setting = 2 For Min and MAX values of VDETLOW, see respective PolarFire FPGA Datasheet or PolarFire SoC Datasheet.
	Preset SATA high threshold setting = 3 For Min and MAX values of VDETLOW, see respective PolarFire FPGA Datasheet or PolarFire SoC Datasheet.
0	For Min and MAX values of VDETLOW, see respective PolarFire FPGA Datasheet or PolarFire SoC Datasheet.
1	For Min and MAX values of VDETLOW, see respective PolarFire FPGA Datasheet or PolarFire SoC Datasheet.
2	For Min and MAX values of VDETLOW, see respective PolarFire FPGA Datasheet or PolarFire SoC Datasheet.
3	For Min and MAX values of VDETLOW, see respective PolarFire FPGA Datasheet or PolarFire SoC Datasheet.
4	For Min and MAX values of VDETLOW, see respective PolarFire FPGA Datasheet or PolarFire SoC Datasheet.
5	For Min and MAX values of VDETLOW, see respective PolarFire FPGA Datasheet or PolarFire SoC Datasheet.
6	For Min and MAX values of VDETLOW, see respective PolarFire FPGA Datasheet or PolarFire SoC Datasheet.
7	For Min and MAX values of VDETLOW, see respective PolarFire FPGA Datasheet or PolarFire SoC Datasheet.

Software defaults to LOS-Low = 1 and LOS-High = 3.

3.2.6. Polarity Invert (Ask a Question)

Polarity invert is used to swap the P and N receiver pins, which provide flexible PCB routing by interchanging the devices physical pin to the logical signal. If the polarity of differential traces is swapped on the PCB, the differential data can be swapped to correct it. In PIPE mode, there is an input port for rxpolarity inversion, but that is not the case for the other modes. Polarity inversion is controlled by a system register (INV_RX_POLARITY_LN#), not an input port for PMA/8b10b/64b6xb modes.

3.3. IO Editor for Signal Integrity (Ask a Question)

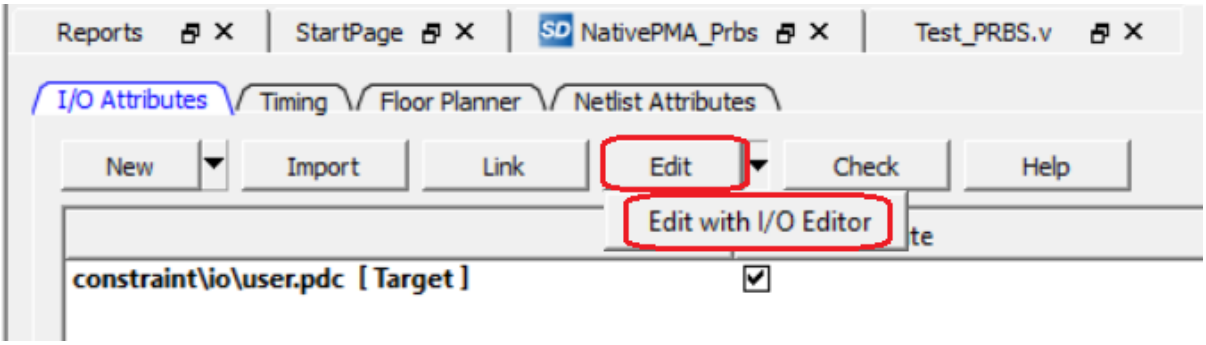
Using the Libero SoC IO Editor, the default signal Integrity parameter settings can be adjusted. These settings are exported from the IO Editor in the PDC format along with port name, pin, direction and so on. The `io.pdc` file created contains physical placement and signal integrity options of all XCVR lane instances used in the SmartDesign.

3.3.1. IO Editor—Signal Integrity (Ask a Question)

To access the Signal Integrity tab from IO Editor:

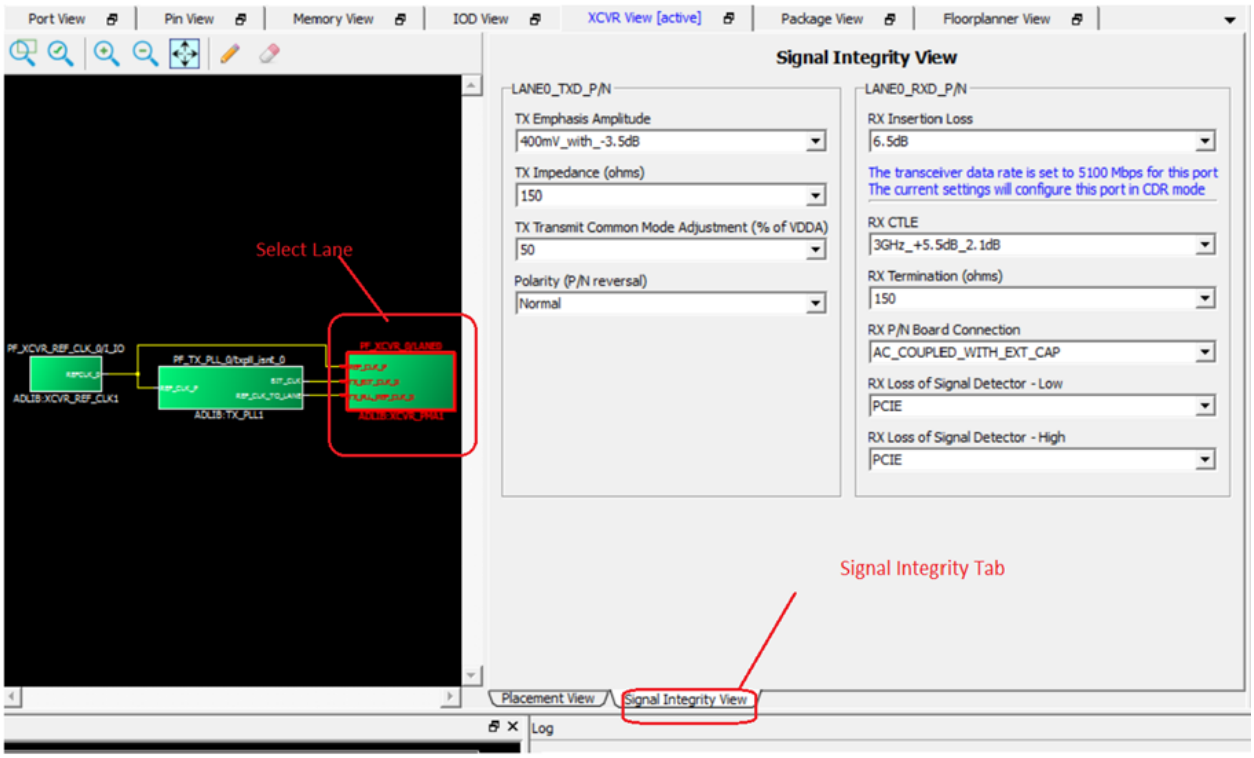
1. Open **Constraint Manager** > **Edit** > **Edit with I/O Editor**.

Figure 3-2. IO Editor—XCVR View



2. In the **XCVR View [active]** tab, select **Signal Integrity View** tab and the XCVR lane to edit signal integrity settings as shown in the following figure.
   Signal Integrity panel displays the default values from the initial PDC when XCVR is created by the Libero SoC software. If user modifies the default values in the **Signal Integrity** tab of the IO Editor, it writes in the PDC.

Figure 3-3. IO Editor—Signal Integrity View



3.3.2. PDC Constraint File Commands for XCVR Signal Integrity (Ask a Question)

PDC constraint support uses the set_io pdc commands to set the related parameters.The settings are applied only to the dedicated XCVR ports—TX_P, TX_N (outputs), RX_P, and RX_N (inputs)

Example: `set_io -port_name TX_P -TX_EMPHASIS_AMPLITUDE 100mV_with_0dB`

Note: User can set the parameter on the P or N side and it applies to both.

3.3.2.1. PDC Supported Attributes and Values (Ask a Question)

The following table lists the PDC supported attributes and values. For more information about he description of the attributes, see Transmitter.

Table 3-3. TX Attributes and Values

Name	Direction	Values	Default
TX_EMPHASIS_AMPLITUDE	Output	400mV_with_-3.5dB	400mV_with_-1.0dB
		100mV_with_0dB	
		200mV_with_0dB	
		200mV_with_-1.0dB	
		200mV_with_-2.5dB	
		200mV_with_-3.5dB	
		200mV_with_-4.4dB	
		200mV_with_-6.0dB	
		300mV_with_0dB	
		400mV_with_0dB	
		400mV_with_-1.0dB	
		400mV_with_-2.5dB	
		400mV_with_-4.4dB	
		400mV_with_-6.0dB	
		500mV_with_0dB	
		600mV_with_-3.5dB	
		600mV_with_-6.0dB	
		800mV_with_0dB	
		800mV_with_-1.0dB	
		800mV_with_-2.5dB	
		800mV_with_-3.5dB	
		800mV_with_-4.4dB	
		800mV_with_-6.0dB	
		1000mV_with_0dB	
		1000mV_with_-1.0dB	
		1000mV_with_-2.5dB	
		1000mV_with_-3.5dB	
		1000mV_with_-4.4dB	
		1000mV_with_-6.0dB	
TX_IMPEDANCE	Output	150	100
		100	
		85	
TX_TRANSMIT_COMMON_MODE_ADJUSTMENT	Output	50	50
		60	
		70	
		80	

The following table lists the PDC supported attributes and values. For more information about the description of the attributes, see Receiver.

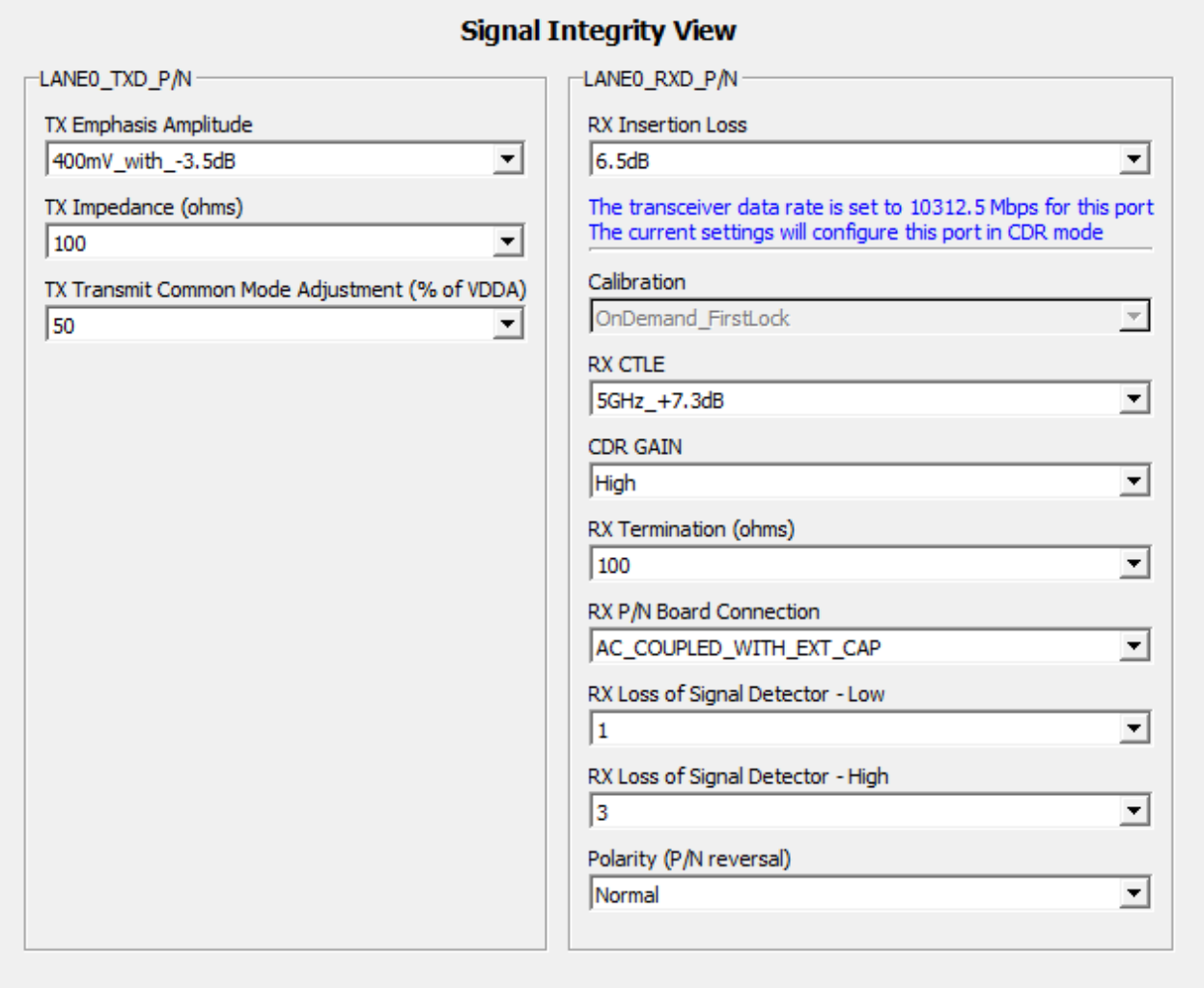
Table 3-4. RX Attributes and Values

Name	Direction	Values	Default
RX_INSERTION_LOSS	Input	6.5 dB	6.5 dB
		17.0 dB	
		25.0 dB	
RX_CTLE	Input	See Rx CTLE Settings table in AC483: PolarFire FPGA Transceiver Signal Integrity Application Note.	Set based on data rate and Rx insertion loss model.
RX_TERMINATION	Input	100	100
		150	
		85	
RX_PN_BOARD_CONNECTION	Input	AC_COUPLED_WITH_EXT_CAP	AC_COUPLED_WITH_EXT_CAP
		DC_COUPLED	
RX_LOSS_OF_SIGNAL_ DETECTOR_LOW	Input	OFF	OFF
		PCIE	
		SATA	
		1	
		2	
		3	
		4	
		5	
		6	
		7	
RX_LOSS_OF_SIGNAL_DETECTOR_HIGH	Input	OFF	OFF
		PCIE	
		SATA	
		1	
		2	
		3	
		4	
		5	
		6	
		7	
POLARITY	Input	Normal	Normal
		Inverted	

3.4. SmartDebug Signal Integrity (Ask a Question)

SmartDebug signal integrity is invoked from debug transceiver. SmartDebug Signal Integrity helps to evaluate the reliability of a high-speed serial link using transceivers. The signal integrity GUI follows the same format as the IO Editor. For more information about software operation, see PolarFire FPGA SmartDebug User Guide. For more information about transceiver signal tuning, see AC483: PolarFire FPGA Transceiver Signal Integrity Application Note.

Figure 3-4. SmartDebug Signal Integrity GUI Panel



The Signal Integrity UI panel can be viewed in SmartBERT, Loopback Modes, Static Pattern Transmit, and Eye Monitor debug tabs. This allows the user to move between test operations and signal integrity tuning, hence, allowing the user to interactively test while making adjustments to the signal integrity settings.

Consistent with debugging techniques of high-speed serial channels, users utilize the SmartDebug GUI to select integrated test data streams and enable device embedded loop backs. This allows the active channel to be adjusted to match the users PCB losses and discontinuities. For more information about test pattern capabilities, see PRBS Generator/Checker.

3.4.1. Loopback Modes (Ask a Question)

Loopback operations are embedded within the XCVR and are commonly used in debug practice. These loop backs can be tested solely within the device by sending and receiving on-chip or can test real-data to and from the system side. For information, see Loopback.

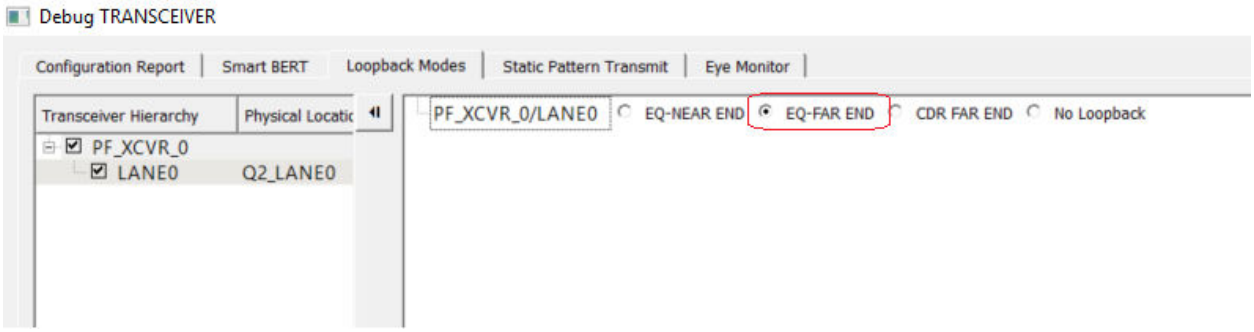
For example, by placing the transceiver into EQ Far-End loopback, a system can drive data into the Rx and send it back directly onto the Tx. In this case, the system is passing the serial data stream within the input and output buffers. Change the Signal Integrity panel settings to find the optimal setting to match the system.

In the Loopback Modes tab, select the targeted lane to test under Transceiver Hierarchy, select EQ-FAR END and then click Apply to monitor the system performance from the far-end. Adjust the

signal integrity setting to optimize the Rx and Tx performance of the transceiver. The following figure shows the loopback modes of the Debug TRANSCEIVER.

Note: The Apply button is enabled when you make a selection for any parameter.

Figure 3-5. Loopback Modes—Far-end Example



Similarly, the user can also continue to test a deeper path into the XCVR path using CDR Far-end loopback. In this manner, the same system data stream would pass from the input buffer through the Rx PMA and the parallel data stream would be routed back through a pathway to the Tx PMA to the output buffer.

3.4.2. SmartBERT (Ask a Question)

The CoreSmartBERT IP core provides a broad-based evaluation and demonstration platform for transceivers (PF_XCVR). See HB0788: CoreSmartBERT Handbook (download the handbook from Libero Catalog). For any transceiver design, PRBS tests from XCVR PMA are available by default. SmartBERT enables you to run diagnostic tests on the transceiver lanes. The self-testing capability can be used for isolating faults either during development debug or for in-field diagnostics using the transceiver built-in PRBS generator and checker.

SmartBERT uses the PRBS generator and checker functionality available in each transceiver lane to determine the bit error rate (BER) of a lane. The various PRBS patterns supported are PRBS7, PRBS9, PRBS15, PRBS23, and PRBS31. Near-end loopback can be performed using one of these PRBS patterns. Bit Error Rate (BER) displays the BER for the PRBS test in progress.

To run a PRBS pattern:

1. Select one of the **Patterns** from the drop-down list.
2. Select the **EQ-NearEnd** check box. When checked, the selected lane gets added to the right hand side where PRBS test can be performed. When unchecked, the selected lane gets removed from the added list (see Figure 3-7).
3. Click **Start** in the bottom-left corner of the window. The loopback cycle is initiated and the result is displayed. It enables both transmitter and the receiver for a particular lane and for a particular PRBS pattern.The GUI shows the status of the TXPLL, RXPLL, Lock to Data, Data rate, and the BER
   (see Figure 3-7).
4. Click **Stop** in the bottom-right corner of the window to stop the loopback.

Monitor the Lock indicators and error counters to check the quality of the link. This test ensures proper power supplies, clocks and resets to the XCVR and traffic is not going off-chip to the system. The following figures show the Smart BERT options of the debug transceiver.

Figure 3-6. PRBS Self Test Near-End Loopback Example

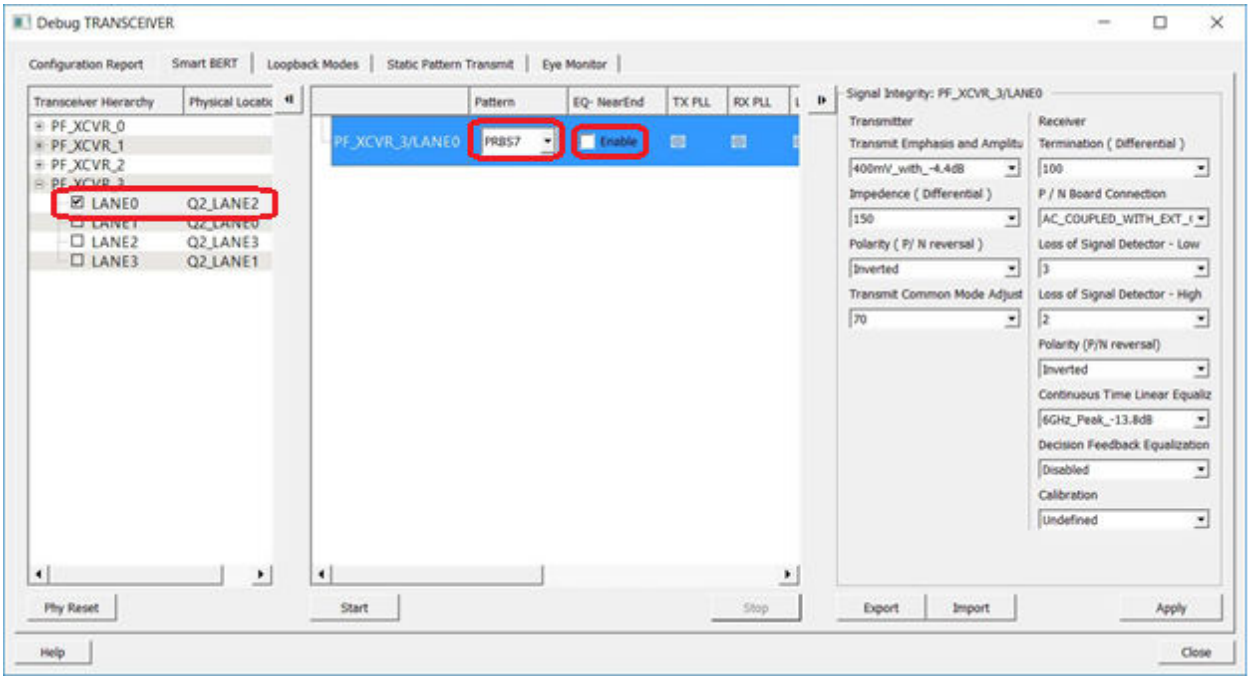
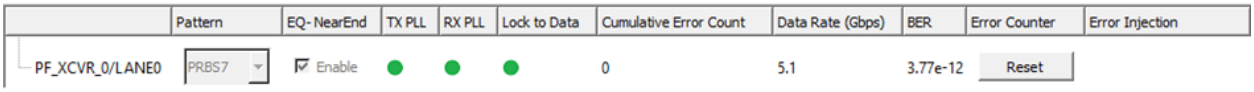


Figure 3-7. Test Status Indicators



3.4.3. Eye Monitoring (Ask a Question)

An eye diagram is a visual representation of a digital signal from an oscilloscope, showing the signal's eye opening as a function of voltage over time. It helps in identifying issues such as timing jitter, amplitude noise, and other signal integrity problems. Similarly, the Eye Monitor within the PolarFire XCVR can do the same without the needed instrumentation of an oscilloscope. Both an oscilloscope and the Eye Monitor represent a "typical" data stream in Bit Error Rate (BER) measurements, including noisy channel impediments and bandwidth limiting effects from stressful data stream patterns when it is transmitted through a real system.

The PolarFire Eye Monitor path within the receiver uses an exact copy of the Continuous-Time Linear Equalizer (CTLE) and Decision Feedback Equalizer (DFE) blocks as the Rx data path to provide accurate monitoring and analysis of the signal quality. This path ensures that the measurement activity does not introduce any disruption to the actual data being received. By incorporating an independent phase rotator and offset setting, the Eye Monitor varies the phase and sampling offset relative to the DFE path. This parallel measurement pathway provides a precise capture of the eye diagram dimensions, which is crucial for assessing signal integrity and performance. This approach helps in identifying issues such as jitter and noise more accurately, as it provides a clearer picture of the signal's on-chip behavior.

When comparing a waveform from an oscilloscope to the resulting waveform of the Eye Monitor, the waveforms are likely to be different. This is because the Eye Monitor is post-equalization thus the Rx CTLE settings impose a negative DC gain, resulting in a change of the overall input signal characteristics as detected inside the receiver. This gives a true measure of signal quality after the PCB variations, package pin including the device equalization corrections rather than working with the distortions that occur when performing off-chip oscilloscope probing.

The Eye Monitor requires a reasonably good signal to provide meaningful measurements. If the incoming signal is so poor in quality that the CDR cannot reliably lock onto it, the Eye Monitor cannot operate either. This is because the Eye Monitor depends on the CDR to provide a stable clock reference for accurate signal analysis. Without a stable clock, the eye diagram cannot be accurately constructed, rendering the Eye Monitor ineffective. Both the DFE and Eye Monitor require a certain level of signal quality to function properly. If the signal quality is too poor, neither the CDR nor the Eye Monitor can perform their respective tasks effectively. Also, both the eye monitor and DFE operation are not permitted below 3 Gbps.

The embedded Eye Monitor logic measures the eye of the incoming data by using the XCVR register interface to control and monitor the features. The eye monitor circuitry works with all bit widths of the deserializer data path. It samples the differences between Eye Monitor and DFE data by sweeping the serial data across 32 phase steps across one complete UI while monitoring the BER. These results provide a statistical eye scan representing the margin analysis of the channel over which the RX receives data showing input voltage compared to BER. The input voltage is expressed as a value after DC-offset cancellation where the differential input is near zero.

The Eye Monitor/DFE comparison block is embedded in the XCVR. The block samples the differences between Eye Monitor and DFE data, maintaining a count of up to 10 differences per DFE clock. It is operated by programming user-defined samples to EYEMONITOR SAMPLE COUNT and transitioning RUN EYEMONITOR COMPARISON from 1'b0 to 1'b1. It counts for EYEMONITOR SAMPLE COUNT clock samples, asserts EYEMONITOR COMPARISON DONE, and presents the difference count on EYEMONITOR COMPARISON OUT. The count is self-reset between comparisons by resetting the accumulating errors between measurement steps.

The BER for an Eye Monitor scan is dependent on the number of samples taken at each point. As the maximum number of differences per sample is 10, the error rate (BER) is calculated using the following equation:

```
EYEMONITOR COMPARISON OUT / (10 × EYEMONITOR SAMPLE COUNT)
```

To restart the block, de-assert and then re-assert RUN EYEMONITOR COMPARISON which restarts the entire scan.

Figure 3-8. Eye Monitor/DFE Comparison Block

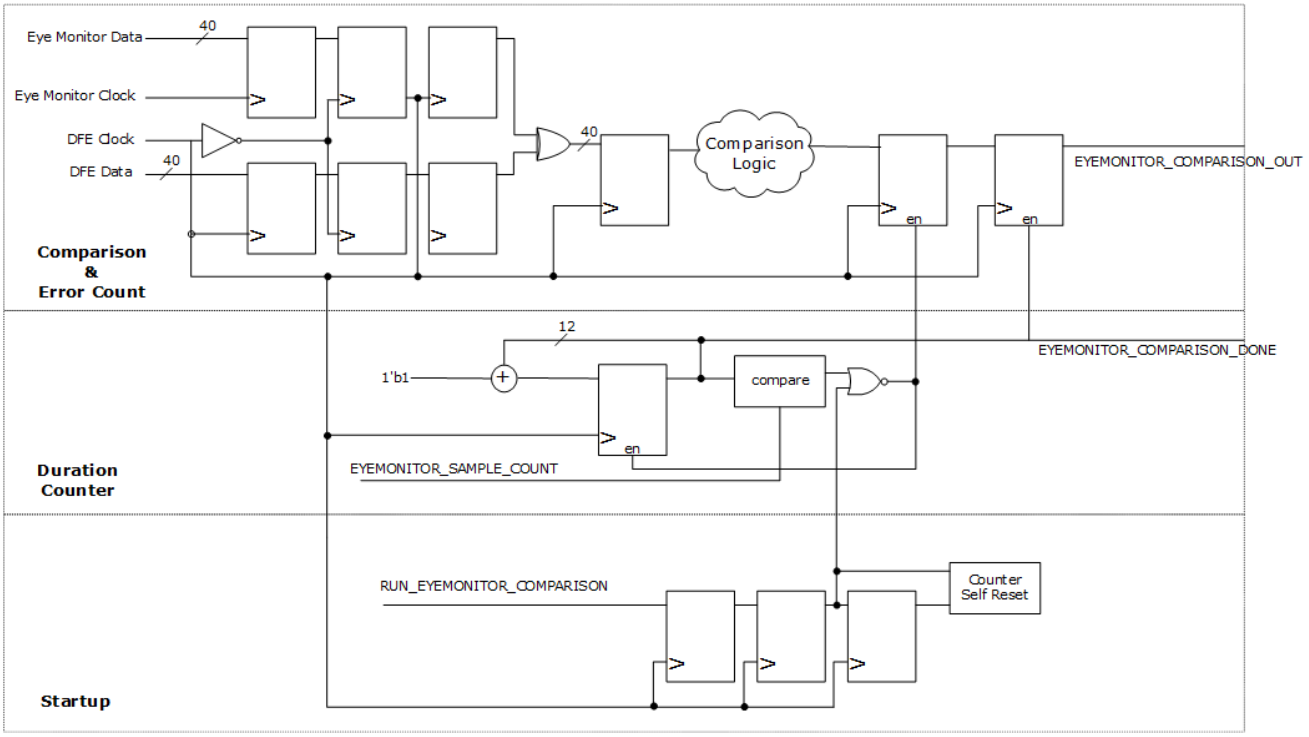


Table 3-5. Eye Monitor PMA_LANE Registers

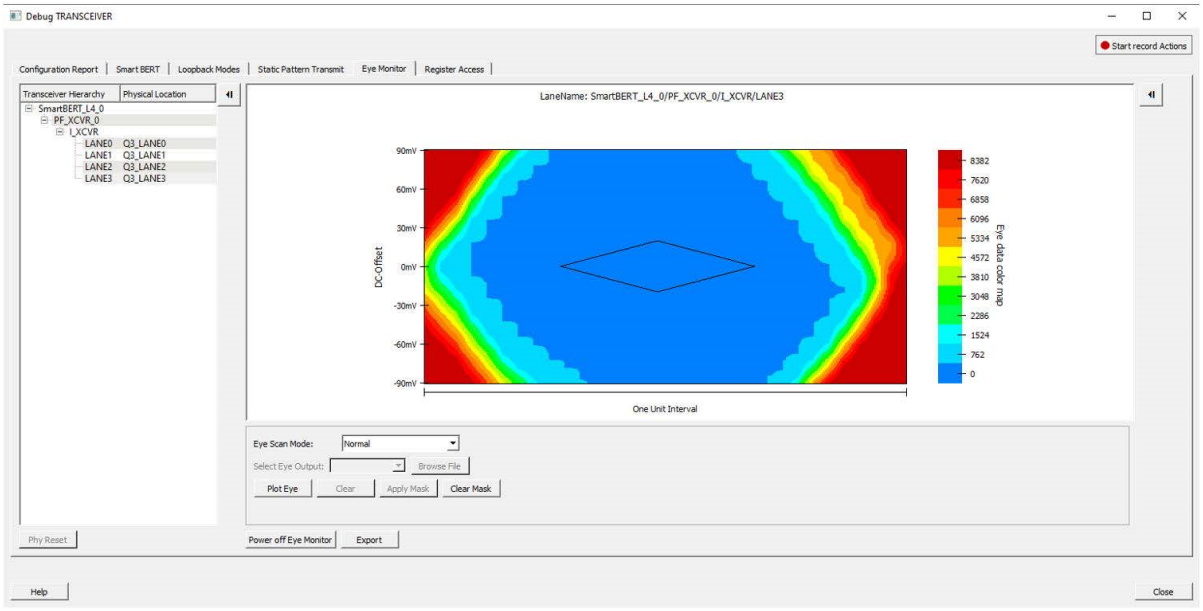
Register Name	Field Name	Width	Field Offset	Type	Default	Field Description
DES_RTL_EM	RUN_EYEMONITOR_COMPARISON	1	0	RW	0x0	• Eye Monitor Comparison circuitry starts on a 0 -> 1 transition of RUN_EYEMONITOR_COMPARISON • Eye monitor accumulates EYEMONITOR_COMPARISON_SAMPLES, and then asserts EYEMONITOR_COMPARISON_DONE, and presents output on EYEMONITOR_SAMPLE_COUNT
	EYEMONITOR_SAMPLE_COUNT	12	1	RW	0x64	• Eye Monitor comparison sample count • Error rate is EYEMONITOR_COMPARISON_OUT / (10*EYEMONITOR_SAMPLE_COUNT) • There are 10 samples taken for every clock period.
	EYEMONITOR_COMPARISON_DONE	1	13	RO	0x0	Eye Monitor Done Signal
	EYEMONITOR_COMPARISON_OUT	16	16	RO	0x0	Eye Monitor comparison error count

3.4.3.1. SmartDebug Eye Monitor Utility (Ask a Question)

As described in the preceding section, the SmartDebug Debug Transceiver utility implements an algorithm using the embedded Eye Monitor capabilities to show an eye diagram during debugging. The SmartDebug Eye Monitor feature creates an eye diagram waveform to actively measure the incoming signal quality. This SmartDebug feature assembles a 2-dimensional graphical depiction of

the incoming eye area as represented by the horizontal or vertical opening. It uses the real-time data collected in the system from XCVRs and creates a plot showing the margins of the incoming signal. The eye diagram is shown as a function of errors detected while the Eye Monitor sweeps the incoming serial input signal. The eye waveform plots the input voltage (Y-axis) versus bit-error of one UI (x-axis) and is displayed to the SmartDebug GUI. The eye waveform does this by executing the described PMA_LANE / DES_RTL_EM register accesses through the SmartDebug JTAG port to the embedded DRI bus therefore allowing run time software GUI to interpret the eye monitor results. Click the **Eye Monitor** tab in the **Debug TRANSCEIVER** window to see the eye monitor representation within the receiver.

Figure 3-9. SmartDebug Eye Monitor Diagram



The plot produces an eye diagram by overlaying many bits whereas a color gradient shows the density hits of the signal. Thus, the opening is represented as the area with least hits.

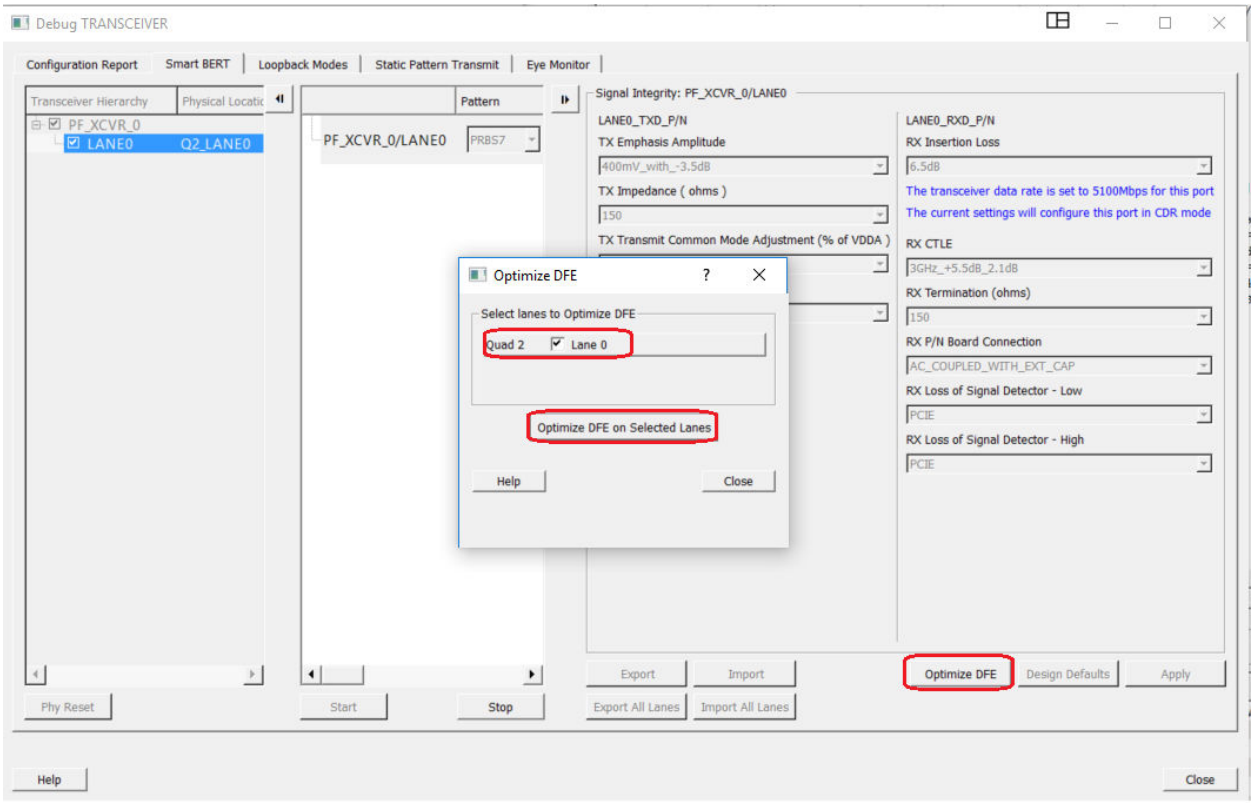
For more information, see the following documents:

- For information on SmartDebug, see SmartDebug User Guide.
- For more information on Transceiver Signal Integrity, see AN5667: PolarFire FPGA Transceiver Signal Integrity Application Note.

3.4.3.2. SmartDebug Decision Feedback Equalization Support (Ask a Question)

SmartDebug is used to adapt the Decision Feedback Equalization (DFE) coefficients to optimize the settings for the overall signal integrity at the receiver for the system under test environment. On-demand, the SmartDebug utility runs the adaptive algorithm to the DFE filter to resolve the TAP values of the DFE coefficients. This can be applied on any XCVR design greater than 5G. The routine is invoked from the Optimize DFE in the GUI. The TXPLL needs to be locked to begin the DFE procedure. This runs the algorithm to set the optimal settings. The optimized settings remain intact until the next DEVRSTn or power cycle. The following figure shows the Optimize DFE window.

Figure 3-10. Example of Optimize DFE

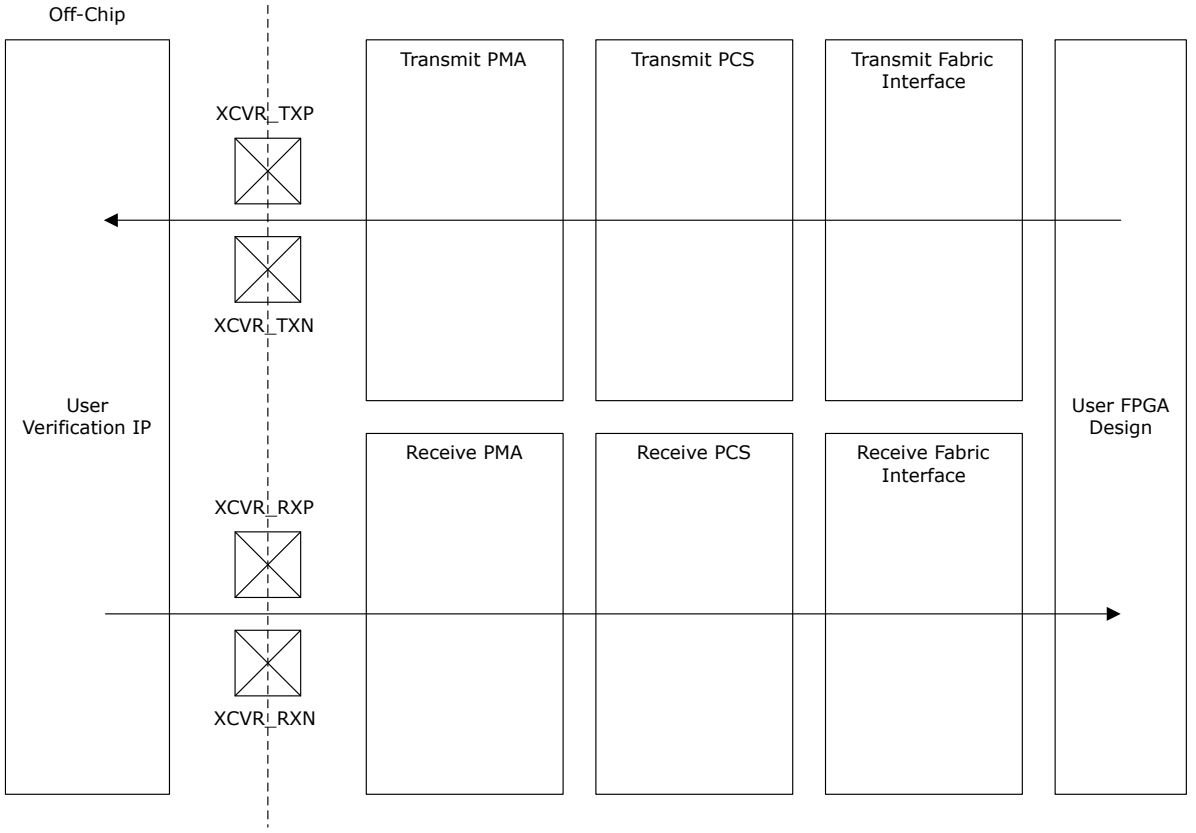


For more information about the Optimized DFE feature, see PolarFire FPGA SmartDebug User Guide.

4. Simulation (Ask a Question)

RTL simulation mode is available for all of the transceiver modes. This simulation mode enables the simulation of all the protocol communication layers (including the PMA, PCS, and fabric interfaces) and provides cycle-accurate simulation for the design. In RTL simulation mode, the recovered clock frequency is derived from the reference clock and not from the incoming data. Also, using RTL simulation incurs some run-time penalties. Microchip provides a specific PCIe BFM model for enhanced simulation of PCIe designs using the embedded PCIe controllers, see PolarFire Family PCI Express User Guide.

Figure 4-1. RTL Simulation Block Diagram



4.1. RTL Simulation Mode (Ask a Question)

RTL simulation mode simulates the XCVR block from the fabric interface to the serial I/O interface. RTL simulation mode is available for all of the XCVR modes. This mode supports all of the protocol communication layers, including the physical layer, and provides accurate cycle simulation for the design. Using RTL simulation, however, experiences some run-time penalties. As the IP user block is off-chip, it must be connected to the user design in the top-level test bench. It is the user's responsibility to provide the model for the off-chip IP that can communicate with the XCVR block in the same protocol used by the XCVR block when using this mode. Post-synthesis simulation is available with PF_XCVR_ERM designs when configured with both ERM OFF and ON.

To minimize simulation time, certain peripherals in the transceiver do not have full behavioral models. These models are replaced with memory models that output a message indicating when the memory locations inside the peripheral are accessed. The memory models are created by using register information that is generated by Libero. The XCVR register data is found at `<Libero`

Project>\component\work\<top_level>\<xcvr_component_name>. The peripheral signals do not toggle based on any writes to registers, or react to any signal inputs on the protocol pins.

Using RTL simulation mode, the FPGA designer can have an off-chip verification IP model that communicates with the transceiver. For example, if the design uses a 8b10b XCVR block, the FPGA designer must have a 8b10b verification IP off-chip block to communicate with the XCVR block using the required protocol. When the IP user block is off-chip, it must be connected to the design in the top-level testbench.

5. Debug and Testing (Ask a Question)

The PolarFire family of devices include debug and testing features for multi-gigabit transceivers. It provides capabilities for diagnostic test setups and inserting test patterns during FPGA testing and debugging. This chapter describes the embedded transceiver capabilities that allow high-speed link diagnostics.

5.1. PRBS Generator/Checker (Ask a Question)

Each transceiver has embedded blocks with a built-in PRBS generator and checker that can be used to perform link testing and diagnostics. These test capabilities are available to the user through the SmartDebug toolset. For more information on PRBS generator and checker, see SmartBERT.

The implementation of the PRBS generator uses a linear feedback shift register (LFSR). The generator produces a pre-defined sequence of 1s and 0s, occurring with the same probability. A sequence of consecutive n × (2^{n-1}) bits comprise one data pattern, and this pattern repeats itself over time. This sequence is compared within the checker to ensure no errors in the sequence are detected.

The PRBS generator/checker supports the following test patterns for 32- and 40-bit wide PMA parallel buses.

- PRBS31: $x^{31} + x^{28} + 1$
- PRBS23: $x^{23} + x^{18} + 1$
- PRBS15: $x^{15} + x^{14} + 1$
- PRBS9: $x^{9} + x^{5} + 1$
- PRBS7: $x^{7} + x^{6} + 1$

PRBS7 is also supported in widths of 8, 10, 16, and 20 bits.

Note: Some PRBS pattern polynomials are used as part of several standards such as ITU-T recommendations. The PRBS7 polynomial is not necessarily a telecommunications standard but is typically used by test equipment because its similarity with 8b10b-encoded patterns.

The PRBS checker is on the parallel side of the de-serializer. If LANE#_RX_SLIP is toggled during PRBS testing, then the next word will not be in the same PRBS pattern as the one before it which causes a failure to match.

5.2. Loopback (Ask a Question)

There are three loopbacks supported within the transceiver blocks to assist designers in debugging the system by segmenting the link (Figure 5-1). The loopbacks are accessed through the SmartDebug tools. For information about data rate performance of loopback paths, see respective PolarFire FPGA Datasheet or PolarFire SoC Datasheet.

5.2.1. EQ Far-End Loopback (Ask a Question)

Far-end serial loopback (receiver to transmitter) or EQ FELB bypasses all input equalization (CTLE and DFE) as detailed in Receiver and therefore, only supports lower speed operation since these features cannot be utilized.

Because it does not use the CDR (the receive and transmit lanes are essentially shorted together), there is no PPM relationship between the receive and transmit data.

5.2.2. EQ Near-End Loopback (Ask a Question)

Near-end serial loopback (transmitter to receiver) or EQ NELB uses the digital serialized transmit data that bypasses the transmit output buffer and loops the data back to the third-stage CTLE input, bypassing the receiver input and connecting to the CDR only.

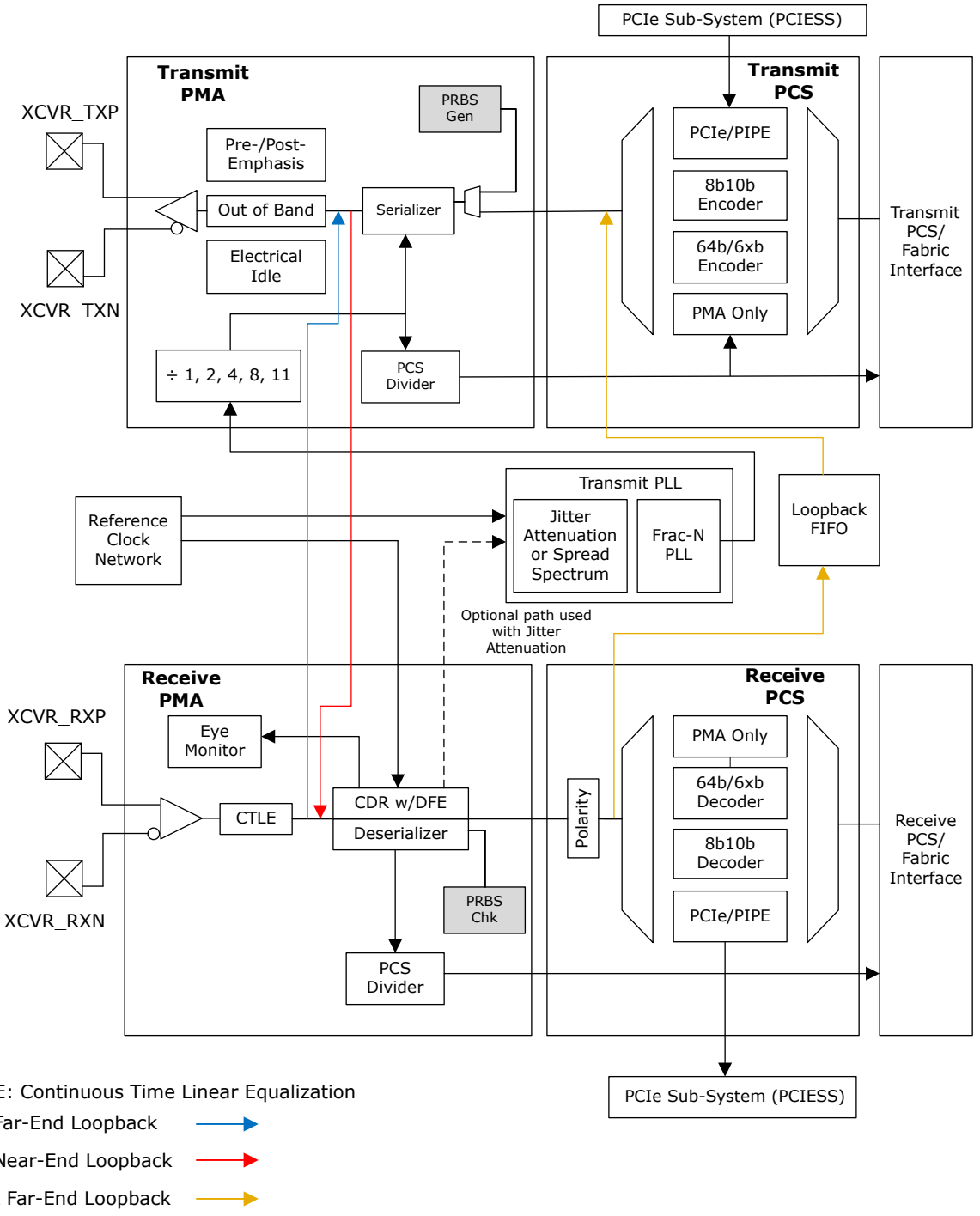
The EQ NELB mode does not test the transmit buffer, high-speed receive buffer, low-speed receive buffer, CTLE stages 1 and 2 for the CDR, CTLE stages 1, 2, and 3 for the DFE and eye monitor circuits.

5.2.3. CDR Far-End Loopback (Ask a Question)

CDR FELB (CDR far-end loop back), occurs after the parallel word creation in the PCS through a loop-back FIFO. The transmit word is then serialized and sent out of the transmitter.

This loopback is after the CDR which requires the receive and transmit paths to have exactly matched clock rates or 0 ppm differences. In the Far-end loopback case, the LANE#_RX_READY must be used instead of LANE#_RX_VAL to indicate valid data path.

Figure 5-1. Transceiver Loopbacks



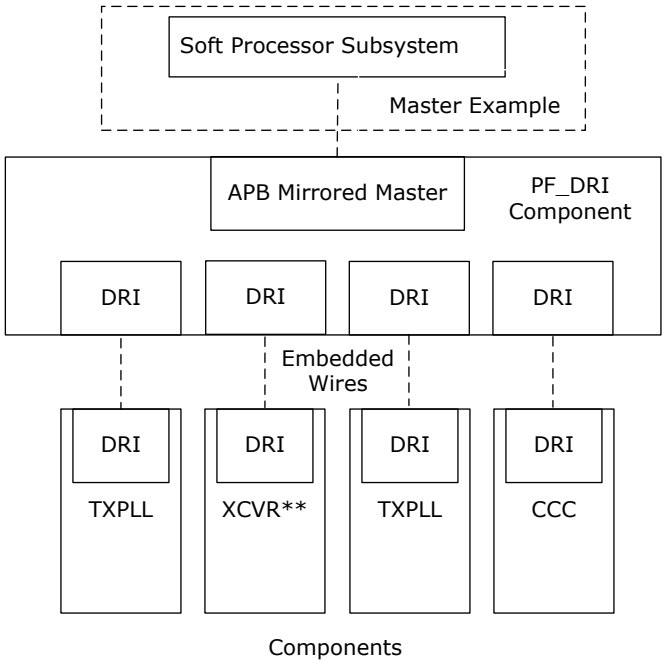
5.3. Dynamic Reconfiguration Interface (Ask a Question)

The dynamic reconfiguration interface (DRI) is used with transceiver to access the memory map of the transceiver blocks. DRI is an APB slave that allows global access to all transceiver lanes, PCIe blocks, transmit PLLs, and FPGA PLLs. The DRI allows changing key features of the transceiver before and during operation. The DRI connectivity is dedicated within the device requiring no FPGA fabric routing. For more information about DRI, see PolarFire Family Device Power-Up and Resets User

Guide and PolarFire Family DRI User Guide. For information about design targeted configuration, see Transceiver Initialization Data.

Care must be exercised when using the DRI to alter transceiver settings as changes from the factory settings can cause undesired results.

Figure 5-2. PF_DRI Example



Note: For implementation example, see AN4592: PolarFire FPGA Dynamic Reconfiguration Interface Application Note (Earlier AC475).

Note:
** The PCIESS blocks do not connect directly to a DRI port, however, the associated XCVR LANE for Quad0 must be connected to the DRI for dynamic control of the XCVR features used with PCIESS. The PCIE[0:1] blocks have dedicated APB port for access to the register control within the PCIE subsystem.

6. Board Design Recommendations (Ask a Question)

User must have knowledge of the following PCB design topics before designing a PCB that uses transceivers.

- Device interfacing
- Transmission line impedance and routing
- Power supply design filtering and distribution
- Component selection
- PCB layout and stack-up design

See respective PolarFire FPGA Board Design User Guide or PolarFire SoC FPGA Board Design Guidelines User Guide for implementing transceiver designs on printed circuit boards.

6.1. Transceiver Top-Level Pin Out (Ask a Question)

The transceiver quad includes four differential receive and transmit pairs. The reference clock to the transmit PLLs can be provided either by the provided primary differential reference clock pins or by the FPGA clock resources.

The transceiver pins, power pins, and associated clock are listed in the following table.

Table 6-1. Transceiver Device Level Pin List (continued)[1]

Pin Name[2]	Direction	Description
XCVR_#_TX3_P	Output	Transmit data. Transceiver differential positive output. Each transceiver quad consists of four transmit+ signals.
XCVR_#_TX2_P	Output	Transmit data. Transceiver differential positive output. Each transceiver quad consists of four transmit+ signals.
XCVR_#_TX1_P	Output	Transmit data. Transceiver differential positive output. Each transceiver quad consists of four transmit+ signals.
XCVR_#_TX0_P	Output	Transmit data. Transceiver differential positive output. Each transceiver quad consists of four transmit+ signals.
XCVR_#_TX3_N	Output	Transmit data. Transceiver differential negative output. Each transceiver quad consists of four transmit− signals.
XCVR_#_TX2_N	Output	Transmit data. Transceiver differential negative output. Each transceiver quad consists of four transmit− signals.
XCVR_#_TX1_N	Output	Transmit data. Transceiver differential negative output. Each transceiver quad consists of four transmit− signals.
XCVR_#_TX0_N	Output	Transmit data. Transceiver differential negative output. Each transceiver quad consists of four transmit− signals.
XCVR_#_RX3_P	Input	Receive data. Transceiver differential positive input. Each transceiver quad consists of four receive+ signals.
XCVR_#_RX2_P	Input	Receive data. Transceiver differential positive input. Each transceiver quad consists of four receive+ signals.
XCVR_#_RX1_P	Input	Receive data. Transceiver differential positive input. Each transceiver quad consists of four receive+ signals.
XCVR_#_RX0_P	Input	Receive data. Transceiver differential positive input. Each transceiver quad consists of four receive+ signals.
XCVR_#_RX3_N	Input	Receive data. Transceiver differential negative input. Each transceiver quad consists of four receive− signals.
XCVR_#_RX2_N	Input	Receive data. Transceiver differential negative input. Each transceiver quad consists of four receive− signals.
XCVR_#_RX1_N	Input	Receive data. Transceiver differential negative input. Each transceiver quad consists of four receive− signals.

Table 6-1. Transceiver Device Level Pin List (continued)[1] (continued)

Pin Name[2]	Direction	Description
XCVR_#_RX0_N	Input	Receive data. Transceiver differential negative input. Each transceiver quad consists of four receive– signals.
XCVR_#[A,B,C]_REFCLK_P[3]	Input	This pin is used as the positive terminal when used with a differential clock source.
XCVR_#[A,B,C]_REFCLK_N[3]	Input	This pin is used as the negative terminal when used with a differential clock.
XCVR_VREF	Power	This pin is used as a reference voltage for the REFCLK input buffers. It is used for single-ended clock signals. This signal is common for all transceiver on device.
VDDA25	Power	2.5 V analog supply. All transmit PLLs and associated high-speed clock routes in each transceiver PMA are connected on-chip but isolated from the other transmit PLLs on the device.
VDDA	Power	Supply for receive, transmit, and common circuits. Common for all lanes within the PMA block.
VDD_XCVR_CLK	Power	Provides common power to all transceiver reference clock buffers. VDD_XCVR_CLK power supply operates using a voltage of 2.5 V to 3.3 V.

(1) See the related PolarFire Package Pin Assignment Table or PolarFire SoC Package Pin Assignment Table for recommended USED or UNUSED conditions.
(2) # Indicates the associated transceiver quad (that is, Q0=0, Q1=1, ...Q5=5).
(3) There is one pin per transmit PLL per transceiver quad. There is a minimum of two differential reference clock input pairs per quad with an additional pair for specific quads. It is limited to driving only one clock source for the transceiver block when used differentially or two when used in the single-ended mode.

6.2. Design for Protocols (Ask a Question)

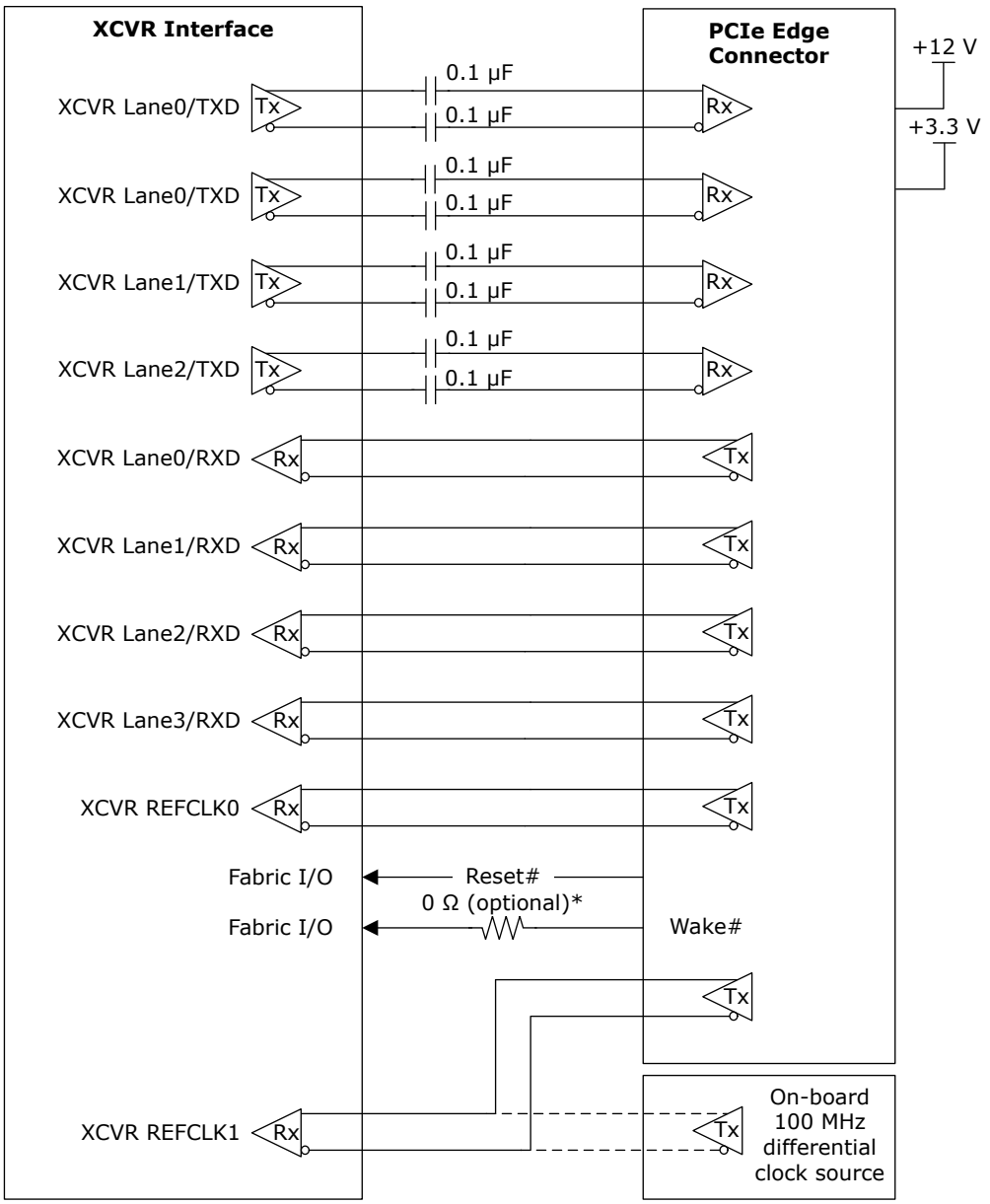
Transceiver designs are used in many high-speed protocols. Each protocol specifies the system requirements to meet the specific standards of the protocol. The electrical performance requirements for these protocols must be addressed by proper design of the PCB. The following sections describes the PCB requirements of transceivers for specific protocols. For more information, see respective PolarFire SoC FPGA Board Design Guidelines User Guide or PolarFire FPGA Board Design User Guide.

6.2.1. PCI Express (Ask a Question)

PCIe is a point-to-point serial differential low-voltage interconnect supporting up to four channels. Each lane consists of two pairs of differential signals: a transmit pair, XCVR_x_TXy_P/N, and a receive pair, XCVR_x_RXy_P/N. Each signal has a 2.5 GHz embedded clock.

The following figure shows the connectivity between the transceiver interface and the PCIe edge connector.

Figure 6-1. Connectivity Between XCVR Interface and PCIe Edge Connector



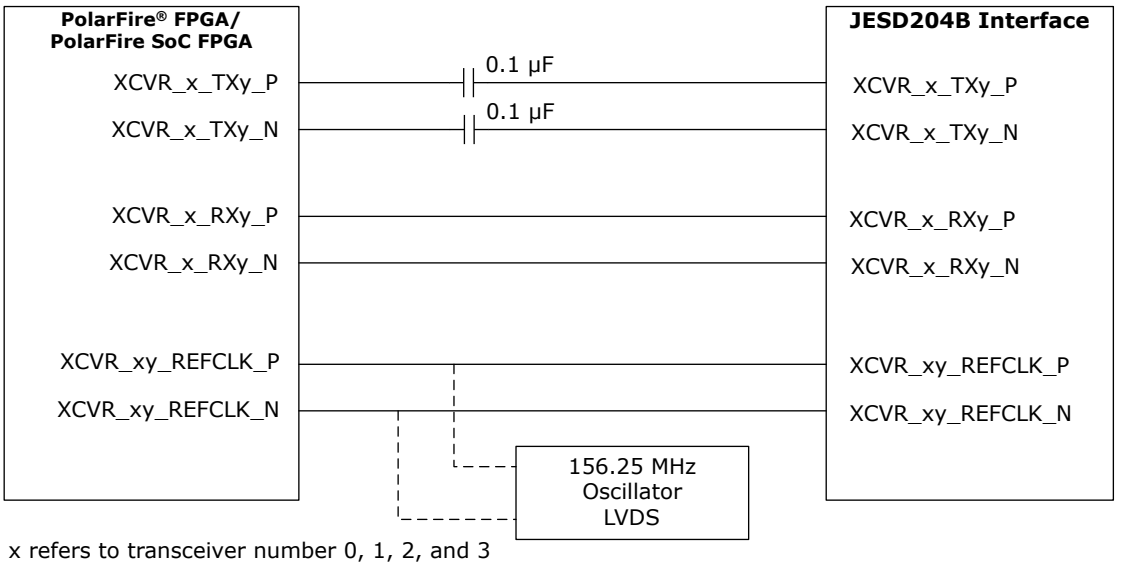
Note: The ceramic 0201 and 0402 AC coupled capacitors are preferred for transceivers. The transmitter must have AC coupling capacitors.

6.2.2. JESD204B (Ask a Question)

JESD204B version increases the supported lane data rates to 12.5 Gbps and divides devices into three different speed grades. The source and load impedance is the same for all three speed grades being defined as 100Ω. The first speed grade aligns with the lane data rates from the JESD204 and JESD204A versions of the standard, and defines the electrical interface for lane data rates up to 3.125 Gbps. The second speed grade in JESD204B defines the electrical interface for lane data rates up to 6.375 Gbps. This speed grade lowers the minimum differential voltage level to 400 mV peak-to-peak, down from 500 mV peak-to-peak for the first speed grade. The third speed grade in JESD204B defines the electrical interface for lane data rates up to 12.5 Gbps.

The following figure shows the connectivity between the device and the JESD204B interface.

Figure 6-2. Connectivity Between Device and JESD204B Interface

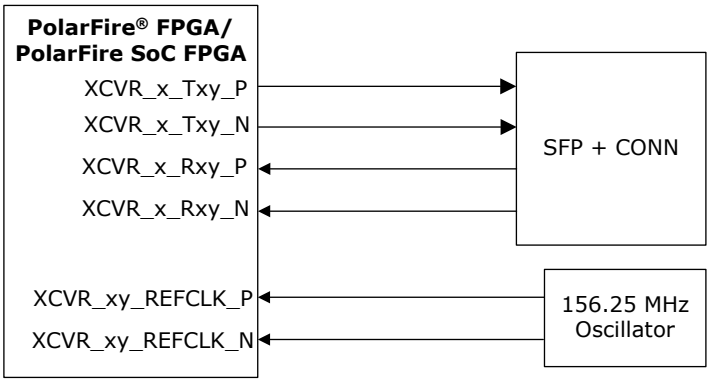


x refers to transceiver number 0, 1, 2, and 3
y refers to lane number 0, 1, 2, and 3
Xy refers to location

6.3. 10G Interface (Ask a Question)

The following figure shows the connection between device and SFP+ interface.

Figure 6-3. Connectivity Between Device and SFP+ Interface



6.3.1. Unused Transceiver Pins (Ask a Question)

If the transceiver interface is not used in the design, the transceiver pins must be connected as defined in the related PolarFire Package Pin Assignment Table (PPAT) or PolarFire SoC Package Pin Assignment Table (PPAT).

6.4. Transceivers Insertion Loss (Ask a Question)

The following table lists the type of insertion loss.

Table 6-2. Transceivers Insertion Loss

Type	Trace Distance	Insertion Loss	PCB length	Di-Electric material	Trace width and Spacing
Short	Connections within a board or through one connector to a daughter card	6.5 dB at 5 GHz	8 inches strip-line	Nelco FR-4	trace width = 7mil, space between the P to N signal = 10mil material
Medium	Backplane application with 2 connectors	17.0 dB at 5 GHz	24 inches strip-line	Nelco FR-4	trace width = 7mil, space between the P to N signal = 10mil material
Long	Backplane application with 2 connectors	25.0 dB at 5 GHz	40 inches strip-line	Nelco FR-4	trace width = 7mil, space between the P to N signal = 10mil material

Notes:

1. For FR-4 and good quality connectors, the insertion Loss is ~0.5 dB per inch at 5 GHz and each connector is ~2.5 dB per connector at 5 GHz.
2. Medium reach backplane is meant for an 18” backplane with two 3” length line-card traces.
3. Long reach backplane is meant for a 34” backplane with two 3” line card traces.
4. DFE Equalization mode is not supported at < 3 Gbps (rates from 3 Gbps to 5 Gbps, including PCI-Express, could be supported with DFE but are not set up to do so in this RX_PRESETS table).

6.5. JTAG Support (Ask a Question)

The Boundary-Scan Description Language (BSDL) files are available on the PolarFire FPGA, PolarFire SoC FPGA, and RT PolarFire FPGA Microchip website that includes a full description of the Boundary-scan functionality within the PolarFire transceivers.

The transceiver I/O implementation is fully compliant with AC-JTAG standard (IEEE1149.6), meaning it works in traditional boundary scan (IEEE1149.1) with the addition of the AC function. RX boundary scan cells (BC-4) are in the scan chain, but they are for capturing DC or AC signals, from the neighboring device, which are annotated in the BSDL as "Observe_only".

Important: Clamp and HighZ instructions are not supported for the SerDes I/Os or the reference clock inputs.

7. Revision History (Ask a Question)

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

Table 7-1. Revision History

Revision	Date	Description
H	05/2025	The following is a summary of the changes made in this revision: • Added a note related to the CDR lock mode option **Lock to Data with 2X Gain** in CDR Options. • Updated the default setting of the **Enable Disparity** option to "Disabled" for 64b67b in Table 2-7. • Added a footnote in Table 2-7 of Transceiver Interface Configurator section to describe that all the options, under **64b6xb Gear Box** in the Transceiver Interface PCS Settings can be independently enabled or disabled for 64b66b and 64b67b.
G	04/2025	The following is a summary of the changes made in this revision: • Updated the supported PCS modes for Tx and Rx (Independent) in Table 2-9 of Half-Duplex Mode. • Added information related to eye monitor, see Eye Monitoring. • Updated the information related to SmartDebug Eye Monitor feature in SmartDebug Eye Monitor Utility.
F	07/2024	The following is a summary of the changes made in this revision: • Added a sentence that the recovered clock is derived from the reference clock and not from the incoming data in the Simulation section. • Updated PCIE_RATE[1:0] to 1-bit signal from 2-bit and also updated the description, see PCIE_RATE.
E	05/2024	The following is a summary of the changes made in this revision: • Updated the port name from "LANE#_CLK_REF" to "LANE#_REF_CLK" in Table 1-6, Table 1-10, Table 1-11, Table 1-12, and Half-Duplex Mode. • Updated the LANE#_RX_VAL clock type to Synchronous in Table 1-6. • Added a note on how to provide the clock source to LANE#_REF_CLK and the accuracy requirement of the source clock with respect to LANE#_RX_CLK_R. See Table 1-6, Table 1-10, Table 1-11, and Table 1-12. • Added fourth note under Table 1-24. • Updated the title of Figure 1-54 to include “MPF200T/MPF300T/MPFS095/MPFS160T/MPFS250T-FCSG536”. • Added a paragraph in JTAG Support to state that the transceiver I/O implementation is fully compliant with AC-JTAG standard (IEEE1149.6). • Added a footnote recommending 32-bit PMA latency to clarify 64b66b latency requirements in Table 1-15. • Removed the “Total Latency (UI)” column from Table 1-15. • Added 64b66b Latency Considerations.

Table 7-1. Revision History (continued)

Revision	Date	Description
D	10/2023	The following is a summary of the changes made in this revision: • Added information about reference clock interface for each die. See Reference Clock Input Pins. • Changed the PCS_ARSTN signal name to PMA_ARSTN. See PMA and PCS Resets.
C	04/2023	The following is a summary of the changes made in this revision: • Updated the document title to "PolarFire Family Transceiver User Guide". Added support for RT PolarFire throughout the document. • Added a note about ERM logic during lock-to-reference mode. See Enhanced Receiver Management. • Added Interface Latency. • Updated Transmit Lane Alignment. • Added Broadcasting REFCLK to the FPGA Fabric. • Added JTAG Support.
B	04/2022	The following is a summary of the changes made in this revision: • Updated information about SATA support. See Table 2. • Updated information about usage of lower data rates in ERM. See Enhanced Receiver Management. • Updated Tx lane alignment. See Transmit Lane Alignment. • Updated information about clock uncertainty. See Timing Constraints.
A	08/2021	The first publication of the document. This user guide was created by merging the following documents: • UG0677: PolarFire FPGA Transceiver User Guide • UG0915: PolarFire SoC FPGA Transceiver User Guide The revision history tables of both the user guides are retained here for the future reference. For information, see Table 7-2 and Table 7-3.

The following revision history table describes the changes that were implemented in the UG0677: PolarFire FPGA Transceiver User Guide document. The changes are listed by revision.

Important: UG0677: PolarFire FPGA Transceiver User Guide document is now obsolete and the information in the document has been migrated to PolarFire Family Transceiver User Guide.

Table 7-2. Revision History of UG0677: PolarFire FPGA Transceiver User Guide

Revision	Date	Description
Revision 9.0	5/21	The following is a summary of the changes in this revision. • Information about incremental DFE calibration was added. See DFE Calibration. • Information about Enhanced Receiver Management was updated. • Information about Transceiver Reference Clock Interface was updated. • Information about spread spectrum was updated. See Table 28. • Information about PIPE Port List and PMA Port List was updated. • Information about Loss of Signal Detect (LOS) was added. • Information about PMA and PCS Resets was updated. • Information about Transceiver Reference Clock Configurator was updated. • Information about Dynamic Reconfiguration Interface was updated. • Information about Jitter Attenuator was updated. • Information about Transceiver Initialization was updated. • Information about Libero Generated Files was updated.
Revision 8.0	9/20	The following is a summary of the changes in this revision. • Information about 8b10b Data Path Interface, 8b10b Bit and Octet Sequencing, and 8b10b System Registers was added. • Information about 64b6xb Data Path Interface, 64b6xb System Registers, 64b66b Receiver, 64b66b Transmit, 64b67b Transmit, and 64b67b Receive was added. • Information about None_DFE (Static DFE) was updated. See Enhanced Receiver Management. • Information about LANE#_CLK_REF port name was added. See Table 7, Table 11, Table 12, and Table 13. • Information about Table 21 was updated. • Information about DFE Coefficients was added. • Information about AC/DC Coupled Connection was updated. • Information about Table 39 was updated. • Information about Design for Protocols and Unused Transceiver Pins was updated. • Information about Half-Duplex Mode was updated. • Information about Receive Input Buffer was updated.
Revision 7.0	4/20	The following is a summary of the changes in this revision. • Information about physical constraint instances was updated. See Table 34. • Information about TX and RX interface clock was updated. See Table 16. • Information about Enhanced Receiver Management was updated.

Table 7-2. Revision History of UG0677: PolarFire FPGA Transceiver User Guide (continued)

Revision	Date	Description
Revision 6.0	1/20	The following is a summary of the changes in this revision. • Information about capability to support switching between two TXPLLs or two CDR REFCLKs through the DRI interface was added. See Transceiver Interface Configurator. • Information about LiteFast was updated. See Table 1. • Information about PCS/FPGA Fabric Interface was added. • Information about new latency was updated. See Table 15. • Information about Transceiver Data Path Latency was added. • Information about footnote was updated. See Table 21. • Information about PCS Rate Switch Between 8b10b and 64b66b Mode for CPRI was added. • Information about MPF500 Transceiver and Transmit PLL Layout was updated. See Figure 53. • Information about JA PLL settings for each preset was added. See Jitter Attenuator. • Information about Transceiver Modes was added. • Information about PMA and PCS Resets was updated. • Information about Custom Protocol Settings was added. • Information about Table 7, Table 11, Table 12, and Table 13 was updated.
Revision 5.0	3/19	The following is a summary of the changes in this revision. • Information about SATA sub-mode was removed from PIPE. • Information about RX_IDLE was updated. See Table 7, Table 11, and Table 13. • Information about LANE#_TX_WCLK input pin was added. See PCS/FPGA Fabric Interface. • Information about Transceiver Clock Regions was added. • Information about Reference Clock Disruptions was added. • Information about Jitter Attenuator was added. • Information about REFCLK input pins were updated. See Reference Clock Input Pins. • Information about PMA and PCS Resets was added. • Information about the interface clocks for the Transceiver PLL was updated. See Table 32. • Information about Enhanced Receiver Management was added. • Information about PIPE Interface Compliance Exceptions was added.
Revision 4.0	10/18	The following is a summary of the changes in this revision. • Updated the document for Libero® SoC PolarFire v2.3 release. • Added information about burst mode receiver, see CDR Options. • Added information about the alignment of transmit lanes, see Transmit Lane Alignment. • Added information about Tx loss insertion, see Tx Insertion Loss. • Added a footnote under Table 34 that describes how to disable the ODT value for the differential REFCLK. • Added a footnote under Table 32 that describes the minimum pulse width for the PMA reset signal.
Revision 3.0	1/18	The following is a summary of the changes in this revision. • Information about 8b10b, 64b66b, and PMA only features was added. See 8b10b, 64b66b/64b67b, and PMA Only. • Information about signal integrity was added. See Signal Integrity Conditioning. • Information about Bit-slip was updated. See Bit Slip. 8b10b does not support bit slip mechanism.

Table 7-2. Revision History of UG0677: PolarFire FPGA Transceiver User Guide (continued)

Revision	Date	Description
Revision 2.0	6/17	The following is a summary of the changes in this revision. • Information about 8b10b, 64b66b, and PMA only features was added. See 8b10b, 64b66b/64b67b, and PMA Only. • Information about Word Alignment was added as sub-section to 8b10b. See Word Alignment (Byte Boundary or Comma Detect). • Updated Figure 52 and Figure 53. • Information about LANE#_RX_VAL port name description was updated. See Table 4, Table 7, and Table 11.
Revision 1.0	2/17	The first publication of UG0677: PolarFire FPGA Transceiver User Guide.

The following revision history table describes the changes that were implemented in the UG0915: PolarFire SoC FPGA Transceiver User Guide document. The changes are listed by revision.

Note: UG0915: PolarFire SoC FPGA Transceiver User Guide document is now obsolete and the information in the document has been migrated to PolarFire Family Transceiver User Guide.

Table 7-3. Revision History of UG0915: PolarFire SoC FPGA Transceiver User Guide

Revision	Date	Description
Revision 3.0	5/21	The following is a summary of the changes in this revision. • Information about incremental DFE calibration was added. See DFE Calibration. • Information about Enhanced Receiver Management was updated. • Information about Transceiver Reference Clock Interface was updated. • Information about spread spectrum was updated. See Table 28. • Information about PIPE Port List and PMA Port List was updated. • Information about Loss of Signal Detect (LOS) was added. • Information about PMA and PCS Resets was updated. • Information about Transceiver Reference Clock Configurator was updated. • Information about Dynamic Reconfiguration Interface was updated. • Information about Jitter Attenuator was updated. • Information about Transceiver Initialization was updated. • Information about Libero Generated Files was updated.
Revision 2.0	9/20	The following is a summary of the changes in this revision. • Information about 8b10b Data Path Interface, 8b10b Bit and Octet Sequencing, and 8b10b System Registers was added. • Information about 64b6xb Data Path Interface, 64b6xb System Registers, 64b66b Receiver, 64b66b Transmit, 64b67b Transmit, and 64b67b Receive was added. • Information about None_DFE (Static DFE) was updated. See Enhanced Receiver Management. • Information about LANE#_CLK_REF port name was added. See Table 7, Table 11, Table 12, and Table 13. • Information about Table 21 was updated. • Information about DFE Coefficients was added. • Information about AC/DC Coupled Connection was updated. • Information about Table 39 was updated. • Information about Design for Protocols and Unused Transceiver Pins was updated. • Information about Half-Duplex Mode was updated. • Information about Receive Input Buffer was updated.

Table 7-3. Revision History of UG0915: PolarFire SoC FPGA Transceiver User Guide (continued)

Revision	Date	Description
Revision 1.0	4/20	The first publication of UG0915: PolarFire SoC FPGA Transceiver User Guide.

Microchip FPGA Support

Microchip FPGA products group backs its products with various support services, including Customer Service, Customer Technical Support Center, a website, and worldwide sales offices. Customers are suggested to visit Microchip online resources prior to contacting support as it is very likely that their queries have been already answered.

Contact Technical Support Center through the website at www.microchip.com/support. Mention the FPGA Device Part number, select appropriate case category, and upload design files while creating a technical support case.

Contact Customer Service for non-technical product support, such as product pricing, product upgrades, update information, order status, and authorization.

- From North America, call **800.262.1060**
- From the rest of the world, call **650.318.4460**
- Fax, from anywhere in the world, **650.318.8044**

Microchip Information

Trademarks

The “Microchip” name and logo, the “M” logo, and other names, logos, and brands are registered and unregistered trademarks of Microchip Technology Incorporated or its affiliates and/or subsidiaries in the United States and/or other countries (“Microchip Trademarks”). Information regarding Microchip Trademarks can be found at https://www.microchip.com/en-us/about/legal-information/microchip-trademarks.

ISBN: 979-8-3371-1228-2

Legal Notice

This publication and the information herein may be used only with Microchip products, including to design, test, and integrate Microchip products with your application. Use of this information in any other manner violates these terms. Information regarding device applications is provided only for your convenience and may be superseded by updates. It is your responsibility to ensure that your application meets with your specifications. Contact your local Microchip sales office for additional support or, obtain additional support at www.microchip.com/en-us/support/design-help/client-support-services.

THIS INFORMATION IS PROVIDED BY MICROCHIP “AS IS”. MICROCHIP MAKES NO REPRESENTATIONS OR WARRANTIES OF ANY KIND WHETHER EXPRESS OR IMPLIED, WRITTEN OR ORAL, STATUTORY OR OTHERWISE, RELATED TO THE INFORMATION INCLUDING BUT NOT LIMITED TO ANY IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY, AND FITNESS FOR A PARTICULAR PURPOSE, OR WARRANTIES RELATED TO ITS CONDITION, QUALITY, OR PERFORMANCE.

IN NO EVENT WILL MICROCHIP BE LIABLE FOR ANY INDIRECT, SPECIAL, PUNITIVE, INCIDENTAL, OR CONSEQUENTIAL LOSS, DAMAGE, COST, OR EXPENSE OF ANY KIND WHATSOEVER RELATED TO THE INFORMATION OR ITS USE, HOWEVER CAUSED, EVEN IF MICROCHIP HAS BEEN ADVISED OF THE POSSIBILITY OR THE DAMAGES ARE FORESEEABLE. TO THE FULLEST EXTENT ALLOWED BY LAW, MICROCHIP’S TOTAL LIABILITY ON ALL CLAIMS IN ANY WAY RELATED TO THE INFORMATION OR ITS USE WILL NOT EXCEED THE AMOUNT OF FEES, IF ANY, THAT YOU HAVE PAID DIRECTLY TO MICROCHIP FOR THE INFORMATION.

Use of Microchip devices in life support and/or safety applications is entirely at the buyer’s risk, and the buyer agrees to defend, indemnify and hold harmless Microchip from any and all damages, claims, suits, or expenses resulting from such use. No licenses are conveyed, implicitly or otherwise, under any Microchip intellectual property rights unless otherwise stated.

Microchip Devices Code Protection Feature

Note the following details of the code protection feature on Microchip products:

- Microchip products meet the specifications contained in their particular Microchip Data Sheet.
- Microchip believes that its family of products is secure when used in the intended manner, within operating specifications, and under normal conditions.
- Microchip values and aggressively protects its intellectual property rights. Attempts to breach the code protection features of Microchip products are strictly prohibited and may violate the Digital Millennium Copyright Act.
- Neither Microchip nor any other semiconductor manufacturer can guarantee the security of its code. Code protection does not mean that we are guaranteeing the product is “unbreakable”. Code protection is constantly evolving. Microchip is committed to continuously improving the code protection features of our products.