DG0757
Demo Guide
PolarFire FPGA 10GBASE-R Ethernet Loopback

Microsemi
a Microchip company

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

About Microsemi

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

Contents

Figures

Tables

1 Revision History

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

1.1 Revision 8.0

The following is a summary of the changes made in this revision.

- Updated the document for Libero SoC v12.2.
- Removed the references to Libero version numbers.

1.2 Revision 7.0

Updated the document to include features and enhancements introduced in the Libero SoC v12.0.

1.3 Revision 6.0

The following is a summary of the changes made in this revision.

- Updated the document to include features and enhancements introduced in the Libero SoC PolarFire v2.2 release.
- Added new Appendix for Enabling SyncE in 10G BaseR Design. For more information, see Appendix 2: Enabling SyncE in 10G BaseR Design, page 30.

1.4 Revision 5.0

Updated the document for Libero SoC PolarFire v2.1 release.

1.5 Revision 4.0

The following is a summary of the changes made in this revision.

- The document was updated to include features and enhancements introduced in the Libero SoC PolarFire v2.0 release.
- The design requirements were updated. For more information, see Design Requirements, page 3.
- Details about the demo design, including the hardware implementation block diagram, were updated. For more information, see Demo Design, page 4.
- Libero design implementation details were updated. For more information, see Figure 2, page 5.
- IP configuration details were updated. For more information, see 10GBASE-R Loopback Hardware Design Libero Implementation, page 5.
- Clocking Structure diagram was added. For more information, see Figure 7, page 11.
- A new section which details the reset structure of the design is added. For more information, see Reset Structure, page 11.
- Reset Structure diagram is added. For more information, see Figure 8, page 12.
- Information about simulating the design was updated. For more information, see Simulating the 10GBASE-R Ethernet Loopback Design, page 12.

1.6 Revision 3.0

The following is a summary of the changes made in this revision.

- The document was updated to include features and enhancements introduced in the Libero SoC PolarFire v1.1 SP1 release.
- Hardware requirements were added, and software requirements were updated to include Spirent TestCenter and FlashPro. For more information, see Design Requirements, page 3.
- Information about how to program the device was added. For more information, see Programming the Device Using FlashPro, page 25.
- Information about how to run the hardware reference design was added. For more information, see Running the Demo, page 24.

1.7 Revision 2.0

The following is a summary of the changes made in revision 2.0 of this document.

- The document was updated for Libero SoC PolarFire v1.1 release.
- Information about resource utilization was added. For more information, see Resource Utilization, page 21.

1.8 Revision 1.0

The first publication of this document.

2 PolarFire FPGA 10GBASE-R Ethernet Loopback

The PolarFire® FPGA 10G Ethernet solution is compliant to IEEE 802.3ae standard, which supports data transfer rates up to 10.3125 Gbps. The advantages offered using PolarFire FPGAs for building 10G Ethernet solutions include: the use of low-power transceivers, low-power FPGA fabric, and an in-built SyncE-compliant jitter attenuation.

The 10G Ethernet solution is implemented using the CORE10GMAC soft IP Media Access Control (MAC) core, which can be configured either in 10GBASE-KR mode or 10GBASE-R mode.

This demo design includes the following two designs, which can be used as reference designs for building a 10GBASE-R Ethernet loopback application:

- a 10GBASE-R Ethernet 64-bit loopback design that can be used for simulation
- a 10GBASE-R Ethernet 32-bit loopback design that can be run on the PolarFire Evaluation Board using Spirent TestCenter

These demo designs can be programmed using either of the following options:

- **Using the `.job` file**: To program the device using the `.job` file provided with the design files, see Appendix 1: Programming the Device Using FlashPro Express, page 27
- **Using Libero SoC**: To program the device using Libero SoC, see Libero Design Flow, page 19. Use this option when the demo design is modified.

2.1 Design Requirements

The following table lists the hardware and software requirements for running the demo.

Table 1 • **Design Requirements**

Requirement	Version
Operating system	Windows 7 or Windows 10
Hardware	
PolarFire Evaluation Kit (MPF300-EVAL-KIT)	Rev D or later
Spirent test module for 10G Ethernet traffic generation	
Optical fiber cable	
Small form-factor pluggable (SFP+) module	
Software	
Libero SoC	**Note:** Refer to the readme.txt file provided in the design files for the software versions used with this reference design.
Synplify Pro	
Spirent TestCenter	
FlashPro Express	
ModelSim	

Note: Libero SmartDesign and configuration screen shots shown in this guide are for illustration purpose only. Open the Libero design to see the latest updates.

2.2 Prerequisites

Before you begin:

1. For demo design files download link:
   http://soc.microsemi.com/download/rsc/?f=mpf_dg0757_df
2. Download and install Libero SoC (as indicated in the website for this design) on the host PC from the following location:
   https://www.microsemi.com/product-directory/design-resources/1750-libero-soc#downloads
3. From the Project menu, click **Open** Project.
4. Navigate to the `mpf_dg0757_df/mpf_10GBaseR_df/Libero_Project` folder, select the `Libero_Project.prjx` file, and click **Open**.
5. Download all the IP cores from **Libero SoC > Catalog**:

Note: To simulate this demo design, see Simulating the 10GBASE-R Ethernet Loopback Design, page 12.

2.3 Demo Design

The 10GBASE-R Ethernet loopback hardware design loops back the Ethernet traffic generated by the Spirent test module through the CORE10GMAC IP. A FIFO logic is implemented in the RTL to loop the Rx signals of the Core10G MAC back to the Tx signals of the MAC.

This looped back data is sent through the TX interface of the transceiver that is received by the Spirent TestCenter. Using the Spirent TestCenter software, the received data is analyzed for throughput rate and errors in the incoming packets.

The 10GBASE-R Ethernet loopback design includes the following components:

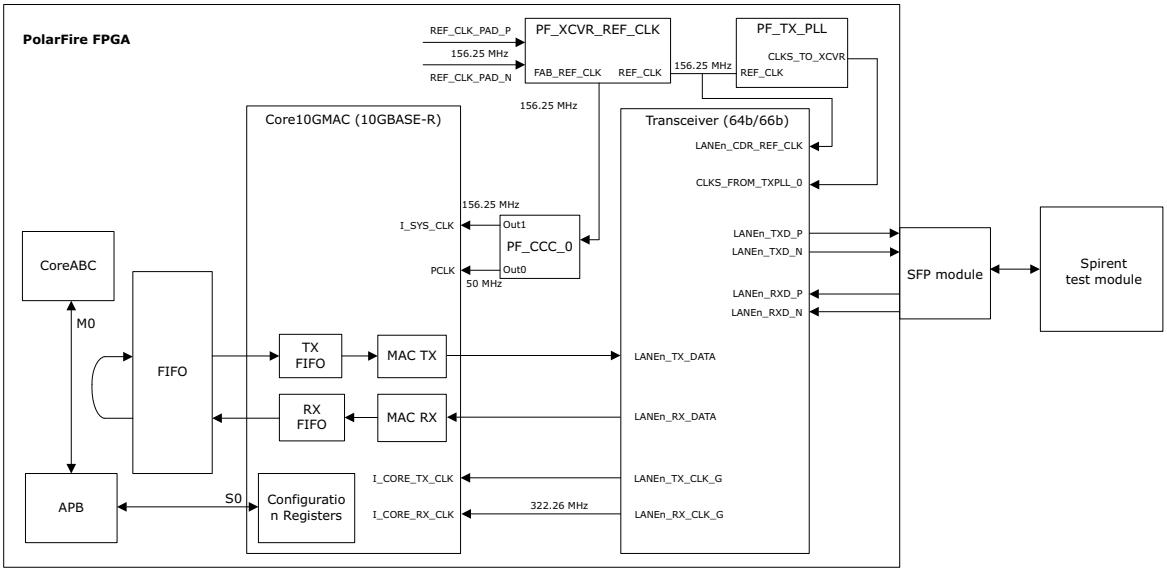
- CORE10GMAC: Serves as a 10-Gbps Ethernet MAC that transmits and receives the Ethernet packets.
- Transceiver: Acts as a 10GBASE-R physical interface for data transfers; configured for 64b/66b encoding/decoding with scrambler/descrambler enabled with a PCS interface width of 32 bits to the CORE10GMAC.
- CoreABC: Configures the CORE10GMAC registers.
- FIFO interface logic: Loops back the CORE10GMAC Rx data to Tx data.
- PF_TX_PLL: Generates the bit clock required for the transceiver.
- PF_XCVR_REF_CLK: Generates the fabric clock and the reference clock for the transceiver and the TX_PLL.
  The following table lists the clock frequencies used in the design.

Table 2 • **Hardware Design Clock Frequencies**

Clock	Frequency (MHz)
CDR reference clock	156.25
Transceiver bit clock	5156.25
I_SYS_CLOCK	156.25
I_CORE_TX_CLK	322.26
I_CORE_RX_CLK	322.26
PCLK	50

The following figure shows the top-level block diagram of the PolarFire 10GBASE-R Ethernet loopback hardware implementation.

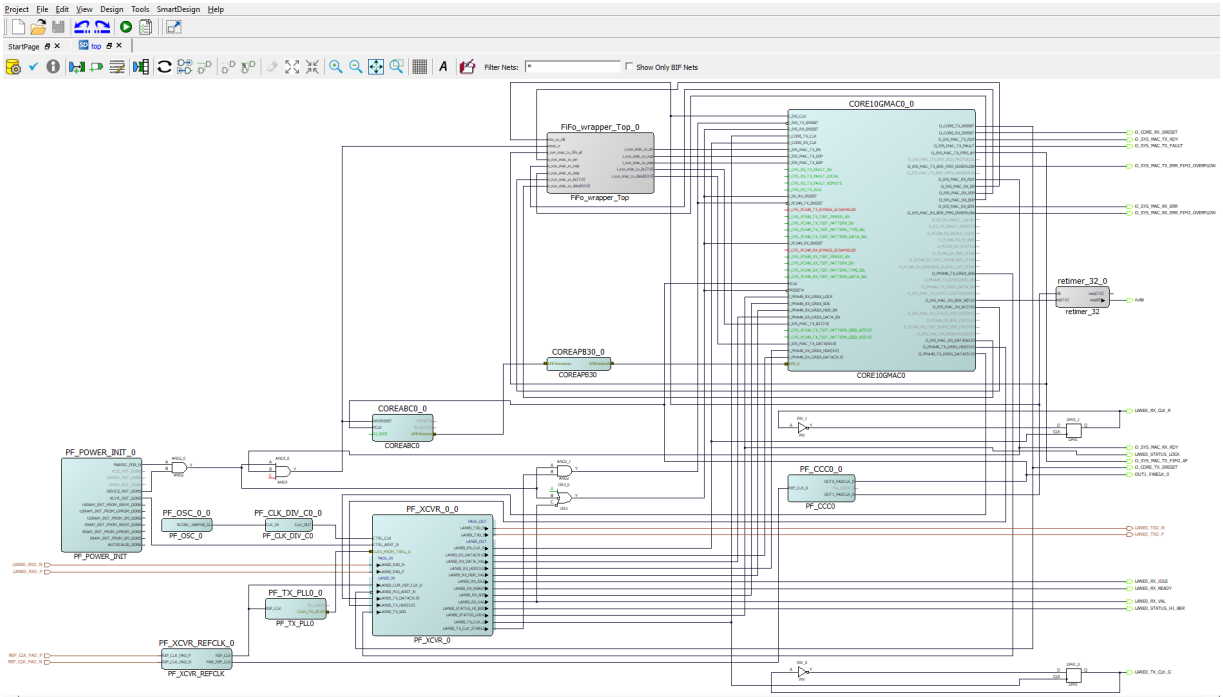
Figure 1 • **Hardware Implementation Block Diagram**



2.3.1 Design Implementation

The following figure shows the Libero implementation of the 10GBASE-R Ethernet loopback hardware design.

Figure 2 • **10GBASE-R Loopback Hardware Design Libero Implementation**



2.3.2 Design Blocks and IP Configuration

The following IPs need to be configured before simulating and implementing the demo design.

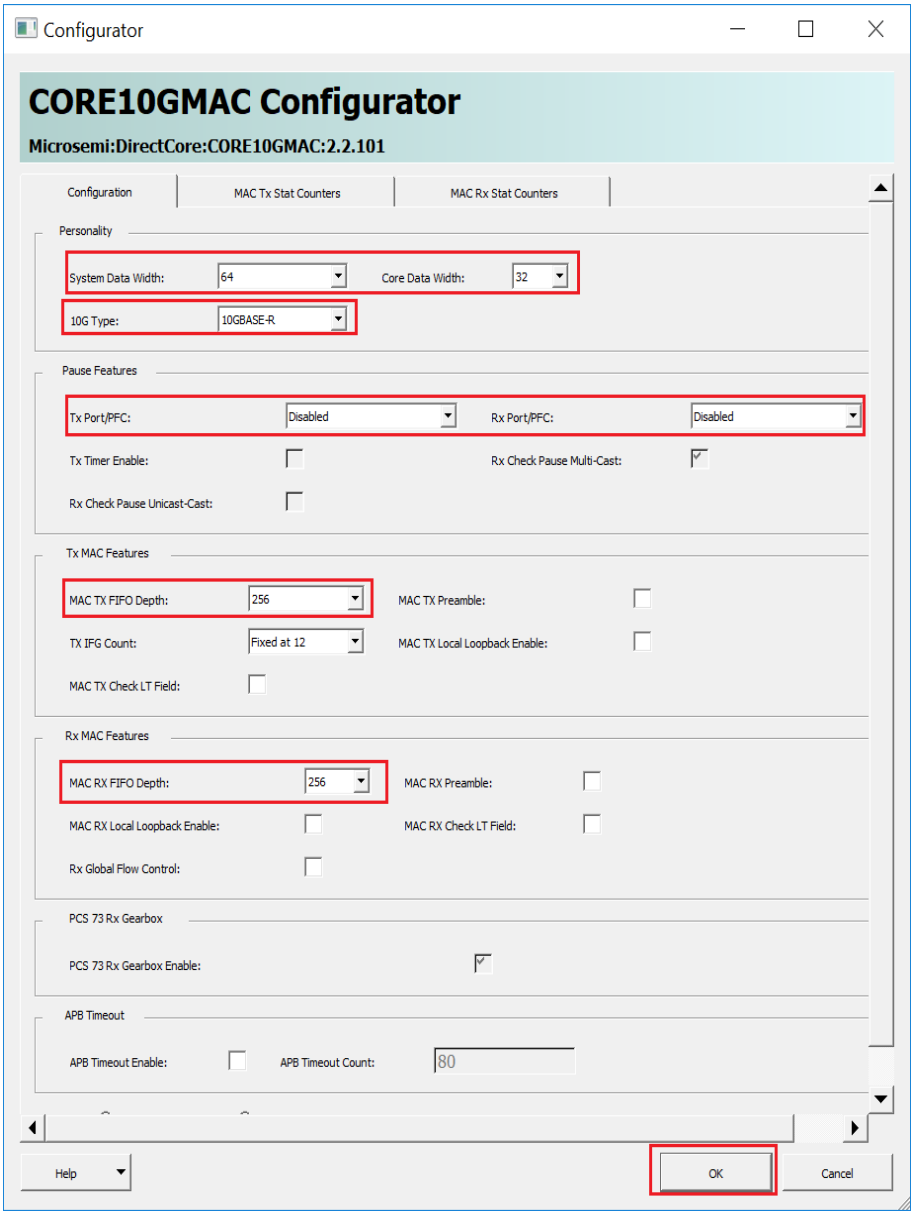
2.3.2.1 Core10GMAC

Core10GMAC is configured for 10GBASE-R mode with a core data width of 32 bits. Core data width is the width of the data path connected to the transceiver interface. The system data width, that is, the width of the interface to the user logic, is configured as 64 bits. (In this demo, the FiFo_wrapper_top module provides this interface.)

The Tx and Rx Pause features are disabled, and both the MAC TX FIFO depth and MAC RX FIFO depth are set to 256.

The following figure shows the settings selected in the CORE10GMAC Configurator.

Figure 3 • **CORE10GMAC Configuration**



The Core10GMAC IP is configured using the CoreABC soft processor. The Core10GMAC configuration for the demo design is as follows:

Table 3 • **Core10GMAC Configuration**

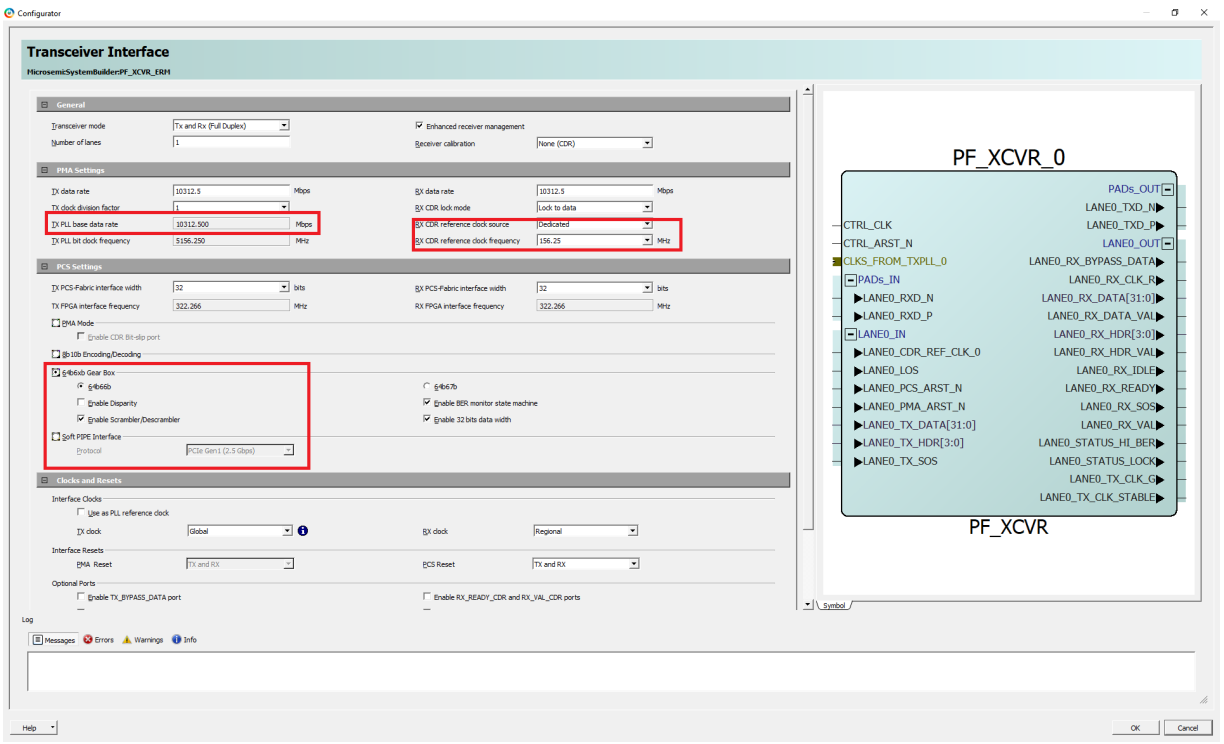
Register	Address	Offset	Bit	Binary Value
MAC Tx Config Register	(0xA)	0x3	cfg_sys_mac_tx_en	1
		0x4	sys_mac_tx_fcs_ins	1
MAC Rx Config Register	(0xB)	0x0	mac_rx_fcs_remove	1
		0x3	cfg_sys_mac_rx_en	1

For information about the features and registers of Core10GMAC, see **Libero SoC > Catalog> Core10GMAC Handbook**.

2.3.2.2 Transceiver Interface

The PolarFire high-speed transceiver (PF_XCVR) is a hard IP block and supports data rates from 250 Mbps to 12.5 Gbps. In this demo, PF_XCVR is configured for the data rate of 10312.5 Mbps. It is configured with a CDR reference clock of 156.25 MHz with lock to data selected as the CDR lock mode. The PCS of the transceiver is interfaced with CORE10GMAC. It is configured for the 64/66b mode with scrambler/descrambler enabled. The scrambler, which is self-synchronizing, generates sufficient transitions to aid data and clock recovery at the CDR. The following figure shows the transceiver interface configuration.

Figure 4 • **Transceiver Interface Configuration**

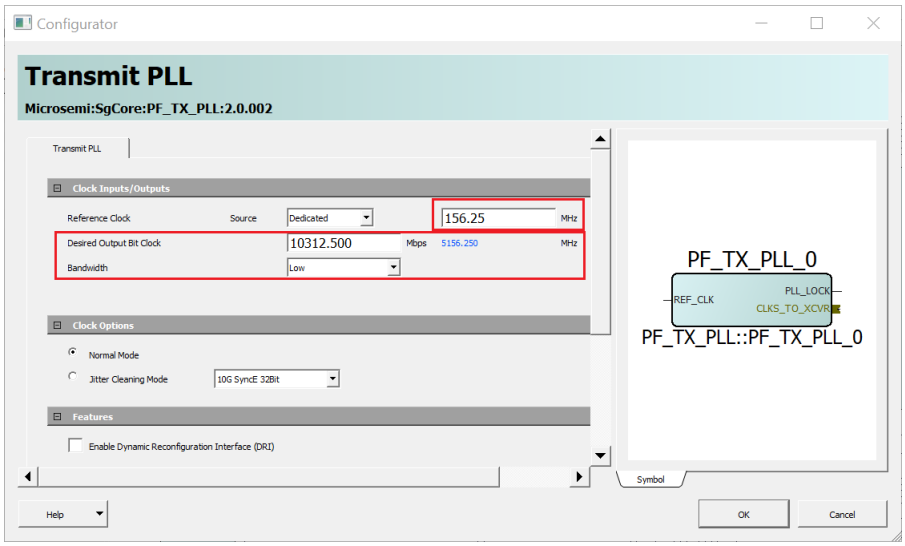


2.3.2.3 Transmit PLL

The PolarFire transmit PLL (PF_TX_PLL) is a hard IP block that provides a bit clock and a reference clock to the transceiver block. The transmit PLL is configured with a reference clock of 156.25 MHz and generates an output clock of 10312.5 Mbps.

The following figure shows the transmit PLL configuration.

Figure 5 • **Transmit PLL Configuration**

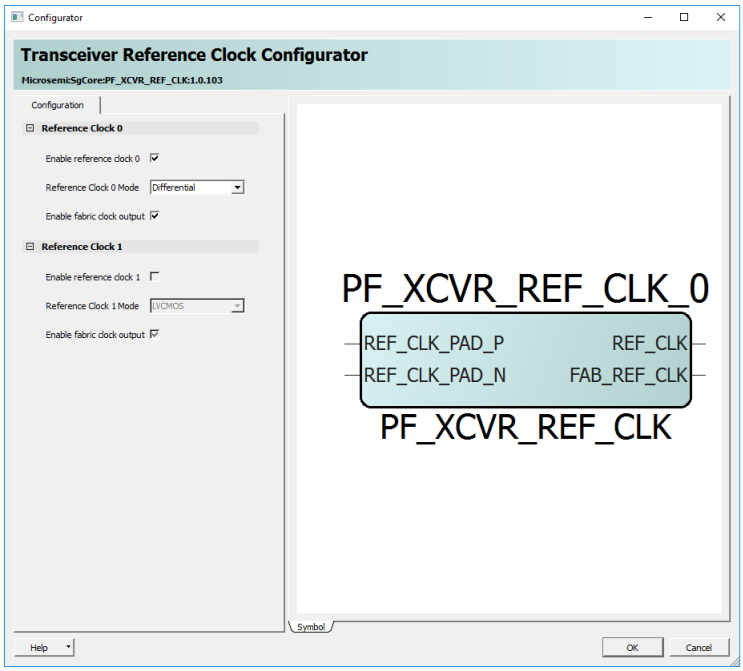


2.3.2.4 Transceiver Reference Clock

The transceiver reference clock (PF_XCVR_REF_CLK) is a hard IP block that provides a reference clock (REF_CLK) of 156.25 MHz to the transmit PLL and a fabric reference clock (FAB_REF_CLK) which is provided as an input to the Clock Conditioning circuit (CCC) to generate the pclk (for configuration) and I_SYS_CLK of the CORE10GMAC.

The following figure shows the transceiver reference clock configuration.

Figure 6 • **Transceiver Reference Clock Configuration**



2.3.2.5 CoreABC

CoreABC is a configurable, low-gate count controller intended for Advanced Microcontroller Bus Architecture Advanced Peripheral Bus (AMBA APB)-based designs. Because this demo design requires only a few registers to be configured, and no dynamic changes are required in the configuration, the CoreABC processor is used in this design. Depending on the application requirements, RISC-V, Cortex-M1, or any other soft processor may be used for configuring the registers.

2.3.2.6 CoreAPB3

CoreAPB3 is a bus component that provides an AMBA AHB fabric for interconnection between an APB master and up to 16 APB slaves. CoreAPB3 supports a single APB3 master. In this design, CoreAPB3 is used to connect the CoreABC APB master interface to the CORE10GMAC APB slave interface.

2.3.2.7 PF_POWER_INIT

The PF_POWER_INIT block ensures the device is powered up systematically. The process of powering up the device includes three steps:

1. Power-on reset
2. Programmed device boot
3. Design initialization

During design initialization, the transceiver configuration is initialized using the data stored in the non-volatile memory. The output of the PF_POWER_INIT block is ANDed with the resets used in the design to reset the entire logic.

2.3.2.8 PF_CCC_0

The PolarFire Clock Conditioning Circuitry (CCC) block takes an input clock of 156.25 MHz from the FAB_REF_CLK signal (output of PF_XCVR_REF_CLK) and generates a 50 MHz clock at OUT0 and 156.25 MHz clock at OUT1. The OUT0 port of the CCC is used for the configuration and OUT1 is used for the user logic in the design.

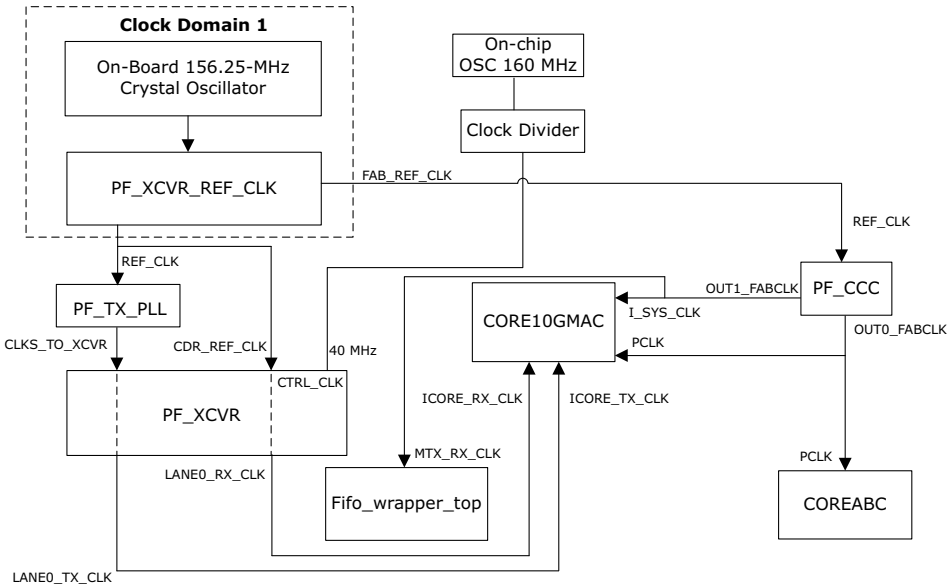
2.3.2.9 FiFo_wrapper_Top

FiFo_wrapper_Top is a user-defined RTL module, which uses the CoreFIFO IP to loop the MAC RX PACKET INTERFACE to the MAC TX PACKET INTERFACE.

2.3.3 Clocking Structure

This design has one clock domain. The on-board 156.25 MHz crystal oscillator generates the clocks used in the demo design. The clock divider generates 40 MHz clock for the XCVR_ERM. The following figure shows the clocking structure of the design.

Figure 7 • **Clocking Structure**



2.3.4 Reset Structure

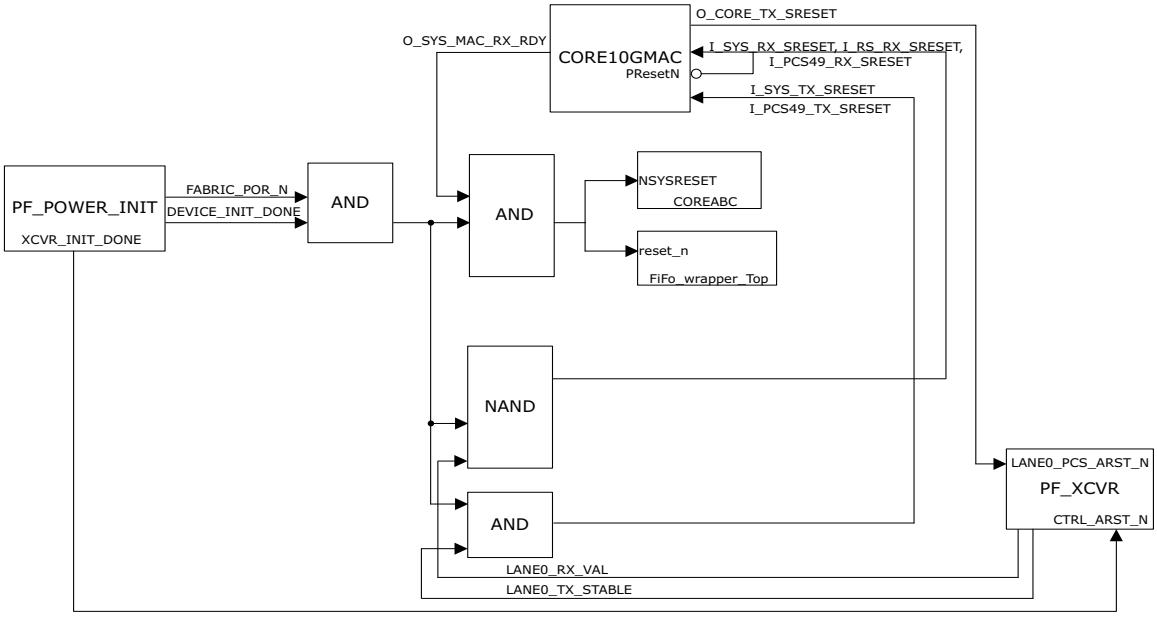
The reset structure of the design is shown in Figure 8, page 12 The output of the **PF_POWER_INIT** monitor ANDed with **PLL_Lock_0** is used to reset the logic in the design. This output is combined with following signals to reset the modules in the design:

- The output is ANDed with **O_SYS_MAC_RX_RDY** to reset the **COREABC** and **FiFo_wrapper_top** module. **CoreABC** configures the **CORE10GMAC** when **MAC RX** is ready.
- The output is NAND with the transceiver control signal, **LANE0_RX_VAL**, and is used to reset the **RX** path of the **CORE10GMAC**. The **MAC RX** path is held in reset until the transceiver, **LANE0_RX_VAL** is driven to '1'.

The **PF_POWER_INIT** monitor output, ANDed with the transceiver control signal, **LANE0_TX_STABLE**, issued to reset the **TX** path of the **CORE10GMAC**.

The **MAC TX** path is held in reset until the transceiver, **LANE0_TX_STABLE** is driven to '1'. The **LANE0_PCS_ARST_N** signal of **PF_XCVR** is reset using **O_CORE_TX_SRESET** of the **CORE10GMAC**.

Figure 8 • **Reset Structure**



2.4 Simulating the 10GBASE-R Ethernet Loopback Design

The following sections list the prerequisites for simulation of the 10GBASE-R Ethernet loopback design, provide details about the design implementation, and describe the simulation flow.

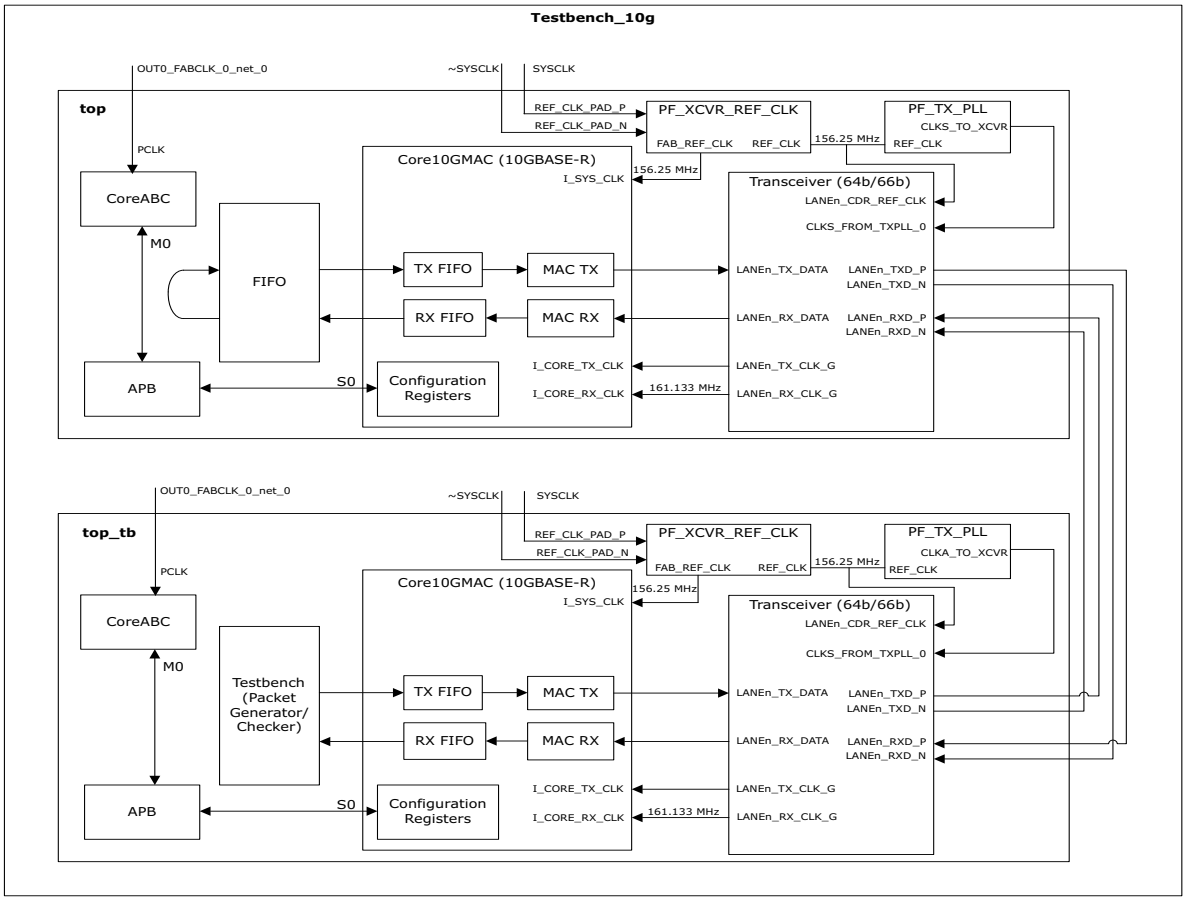
2.4.1 Prerequisites

Before you begin:

1. Start **Libero SoC**, and select **Project** > **Tool Profiles**.
2. In the Tool Profiles window, select **Synthesis** and **Simulation** on the **Tools** panes, and select the latest active installation directory paths for these two tools.
3. From the Project menu, click **Open Project**.
   The Open Project dialog box opens.
4. Navigate to the `mpf_dg0757_df/mpf_10GBaseR_df/Simulation_Project/BaseR_Sim` folder, select the `BaseR_Sim.prjx` file, and click **Open**.
   The PolarFire 10G Ethernet Simulation design project opens in Libero.
5. Download the following IP cores from Libero SoC Catalog:
   - PF_XCVR
   - PF_TX_PLL
   - PF_XCVR_REF_CLK
   - CORE10GMAC
   - CoreABC

The following figure shows the interaction between testbench and the design.

Figure 9 • **Testbench and 10GBASE-R Ethernet Loopback Design Interaction**



2.4.2 Design Description

The 10GBASE-R Ethernet loopback simulation design includes the following components:

- **Testbench_10g:** Top testbench module that generates the clocks required for the device under test (DUT) and the testbench submodule, and interconnects the ports from DUT to the testbench submodule.
- **top_tb:** Testbench submodule, which consists of the following major blocks:
  - CoreABC: Configures 10G MAC registers.
  - packet_generator_checker: Performs the packet generator function of defining the Ethernet frame to be transmitted to CORE10GMAC (packet generator). Performs the packet checker function of receiving the looped back Ethernet frame from the CORE10GMAC and comparing it with the transmitted frame.
  - CORE10GMAC: 10-Gbps Ethernet MAC configured in Base-R mode which transmits and receives the Ethernet packets.
- **top:** DUT block of the design, which consists of the following major blocks:
  - CORE10GMAC: 10-Gbps Ethernet MAC configured in Base-R mode which transmits and receives the Ethernet packets.
  - Transceiver: 10GBASE-R physical interface for data transfers; configured in 64b/66b mode with a PCS fabric width of 64 bits.
  - CoreABC: Configures the CORE10GMAC registers.

- FIFO interface logic: Loops back the Ethernet packets; implemented in Verilog RTLThe following table lists the simulation signals transmitted between the Testbench_10g, top, and top_tb blocks.

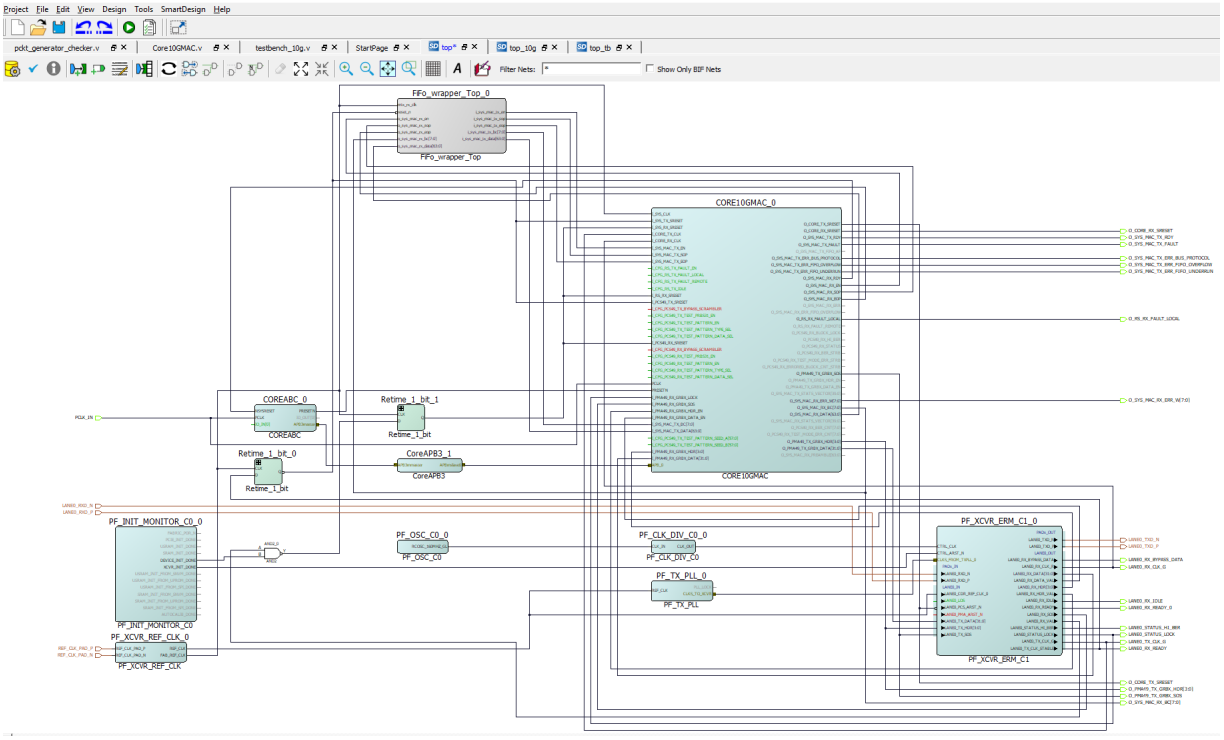
Table 4 • **Simulation Signals**

Output From	Input To	Signal	Description
testbench_10g	top, top_tb	SYSCLK	156.25-MHz clock
		OUT0_FABCLK_0_net_0	50-MHz clock
top	testbench_10g	LANE0_RXD_P	Rx port signals
		LANE0_RXD_N	
		LANE0_TXD_P	Tx port signals
		LANE0_TXD_N	
top_tb	testbench_10g	LANE0_RXD_P_0	Rx port signals
		LANE0_RXD_N_0	
		LANE0_TXD_P_0	Tx port signals
		LANE0_TXD_N_0	

2.4.3 Design Implementation

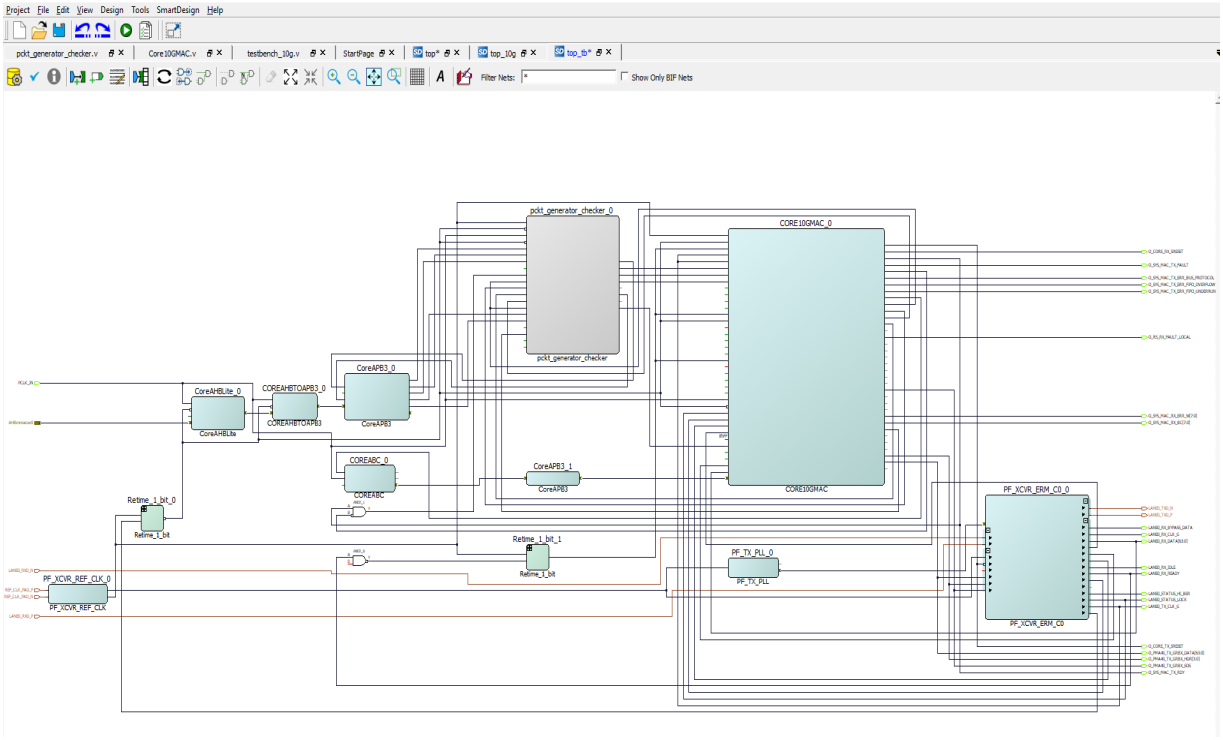
The following figure shows the Libero implementation of the *top* SmartDesign module.

Figure 10 • **Libero Implementation of the top SmartDesign Module**



The following figure shows the Libero implementation of the top_tb SmartDesign module.

Figure 11 • **Libero Implementation of the top-tb SmartDesign Module**



2.4.4 Simulation Flow

The design can be simulated using ModelSim ME 10.7C Pro provided with the Libero SoC installation. The following sections describe the simulation flow for each of these ModelSim versions.

2.4.4.1 Initiating Simulation with ModelSim ME 10.7C Pro

1. In Libero SoC, go to **Project** > **Project Settings** > **Simulation options** > **DO file**, and ensure the **Use automatic DO file** check box is selected, as shown in the following figure.

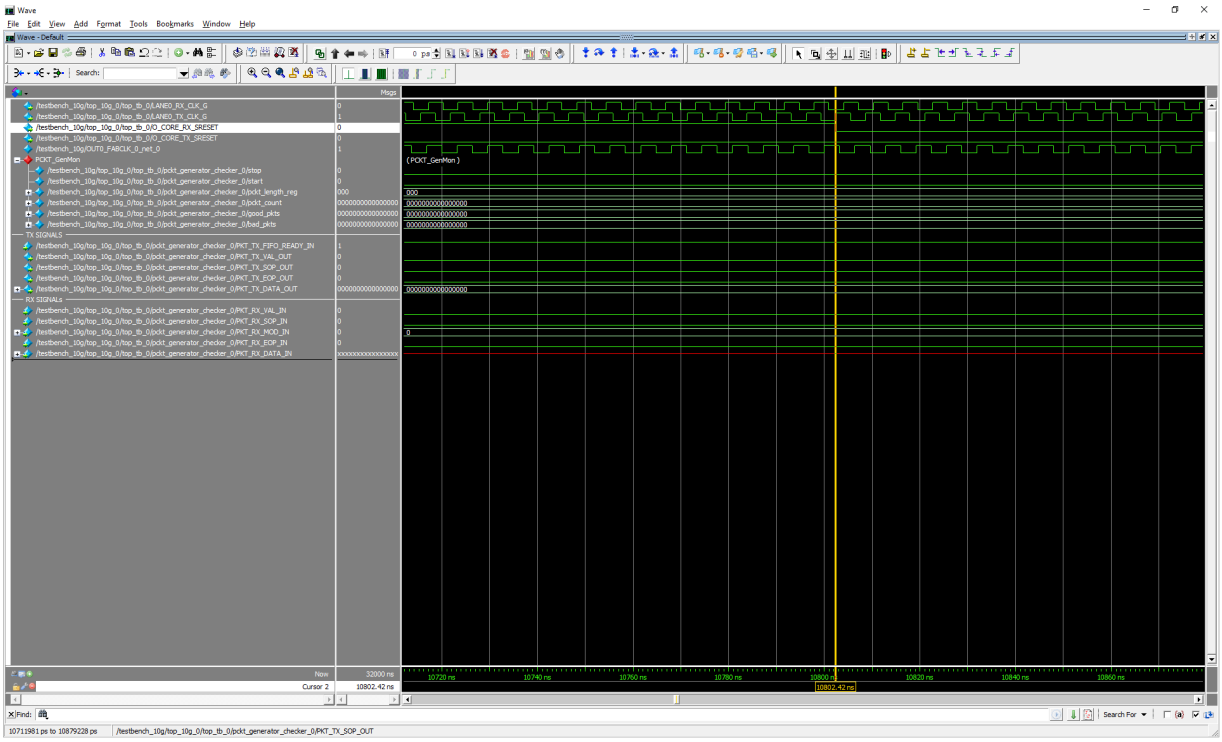
Figure 12 • **Use Automatic DO File Option Selected**

2.4.4.2 Simulation Results

When the simulation is initiated, ModelSim compiles all the design source files, runs the simulation, and launches the waveform viewer to show the simulation signals. The simulation results of the 10GBASE-R loopback design are as follows:

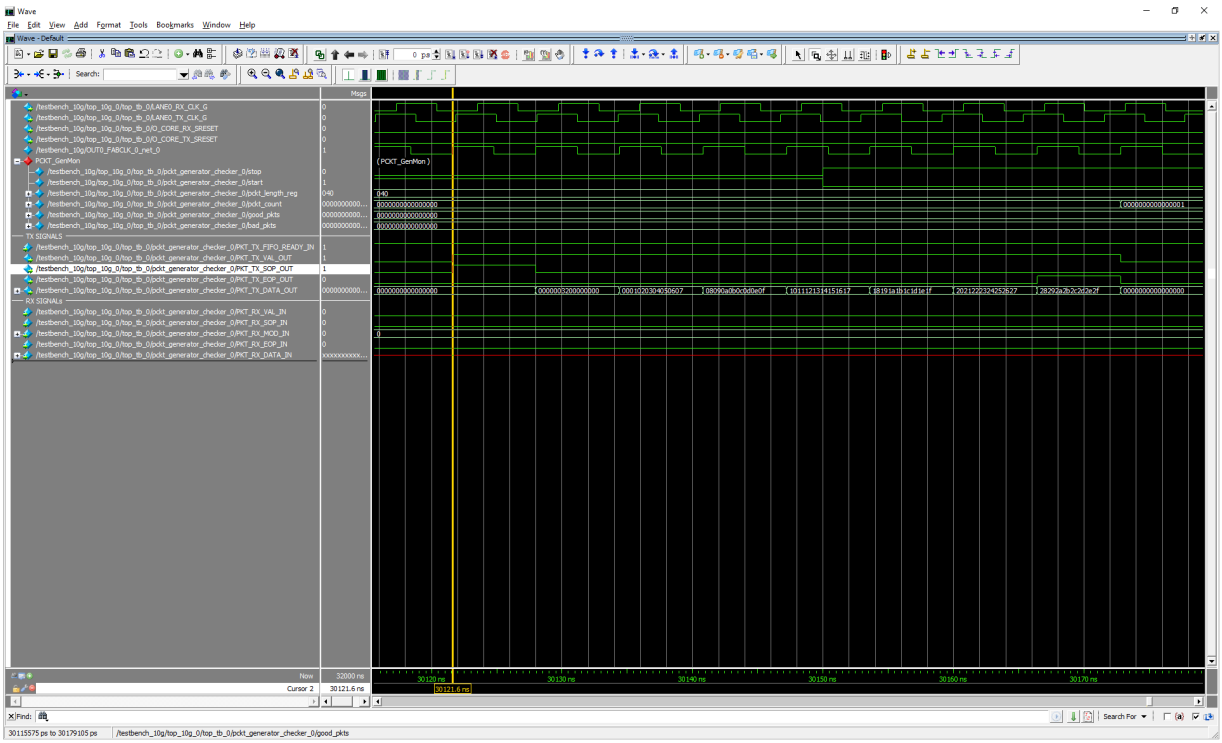
1. At 0 ns, the testbench drives the 156.25 MHz system clock to the DUT.
2. The MAC is released out of the reset. Signal O_CORE_RX_SRESET and O_CORE_RX_SRESET are at 0, as shown in the following figure.

Figure 13 • **O_CORE_RX_SRESET and O_CORE_RX_SRESET at 0**



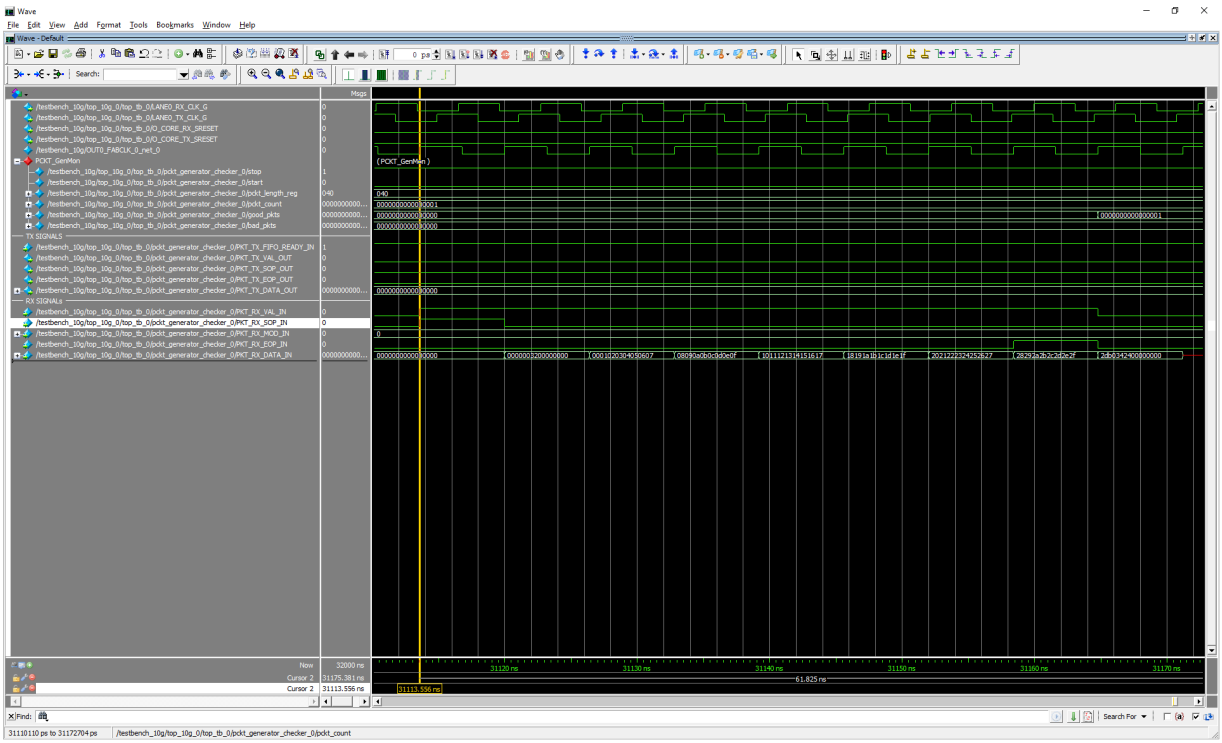
3. The packet generator starts to send the packet. The start signal triggers the packet generation. The size of the packet is set to 0x32 (80 bytes). The sent packet can be viewed on the signals under the TX SIGNALS divider in the wave window.

Figure 14 • **Ethernet Packet Sent**



4. The packet checker receives the sent packet. The signals can be viewed under the RX SIGNALS divider in the wave window. The packet sent matches with the packet received.
5. The packet checker compares the incoming packet with the sent packet and increments the good packets (good_pckts) count by 1, as shown in the following figure.

Figure 15 • **Good Packets Count Incremented by 1**



The sent packet is looped back, and no errors are observed in the received packet, showing successful completion of 10GBASE-R Ethernet loopback.

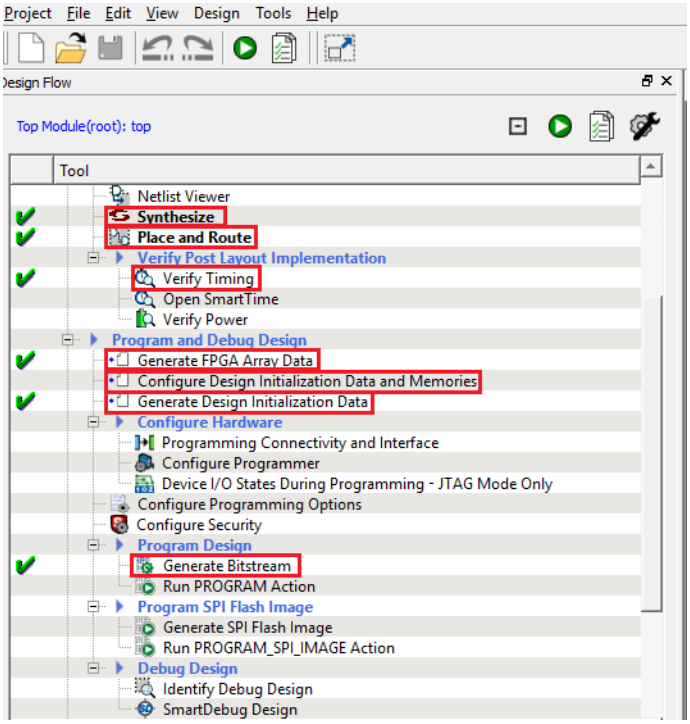
3 Libero Design Flow

This chapter describes the Libero design flow, which involves the following processes:

- Synthesize, page 19
- Place and Route, page 20
- Verify Timing, page 21
- Generate FPGA Array Data, page 21
- Configure Design Initialization Data and Memories, page 22
- Generate Bitstream, page 22
- Run Program Action, page 23

The following figure shows these options in the Design Flow tab.

Figure 16 • **Libero Design Flow Options**



3.1 Synthesize

To synthesize the design, perform the following steps:

1. On the **Design Flow** window, double-click **Synthesize**.

   When the synthesis is successful, a green tick mark appears next to **Synthesize**, as shown in Figure 16, page 19.

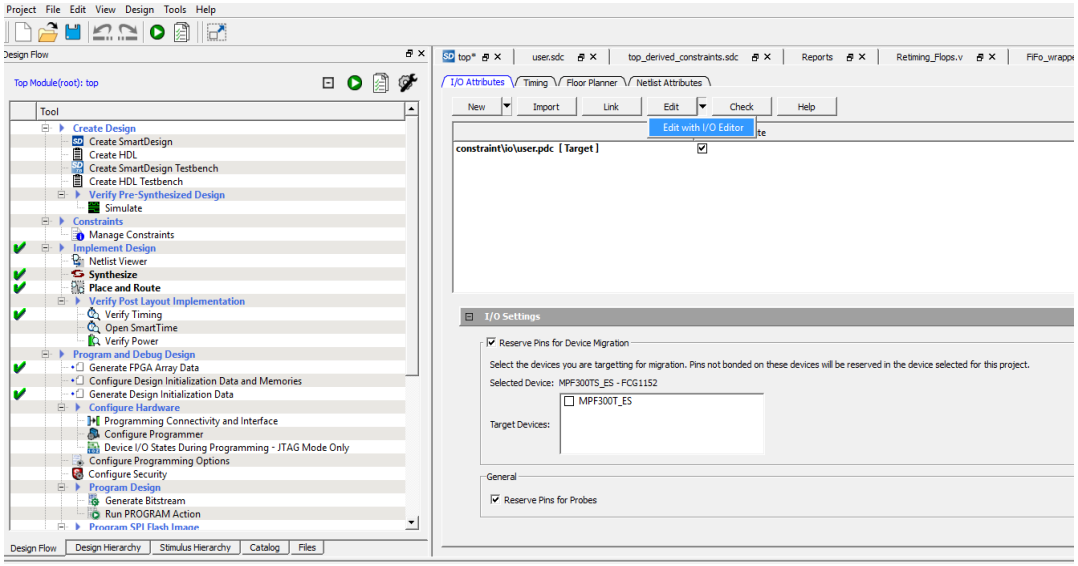
2. Right-click **Synthesize** and select **View Report** to view the synthesis report and log files in the **Reports** tab. View the `top_SD.srr` and `top_SD_compile_netlist.log` files to debug synthesis and compile errors.

3.2 Place and Route

To place and route the design, TX_PLL, XCVR_REF_CLK, and PF_XCVR must be configured using the I/O Editor. Follow these steps to configure the components and place and route the design.

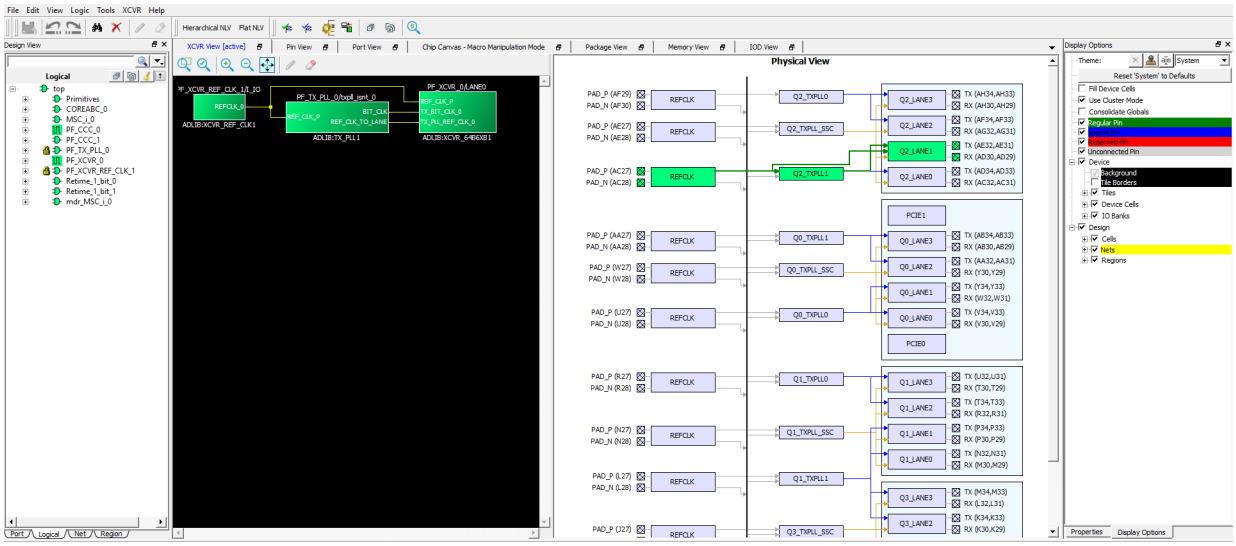
1. On the **Design Flow** window, double-click **Manage Constraints**.
2. On the **I/O Attributes** tab, click **Edit with I/O Editor**, as shown in the following figure.

Figure 17 • **Edit with I/O Editor Option**



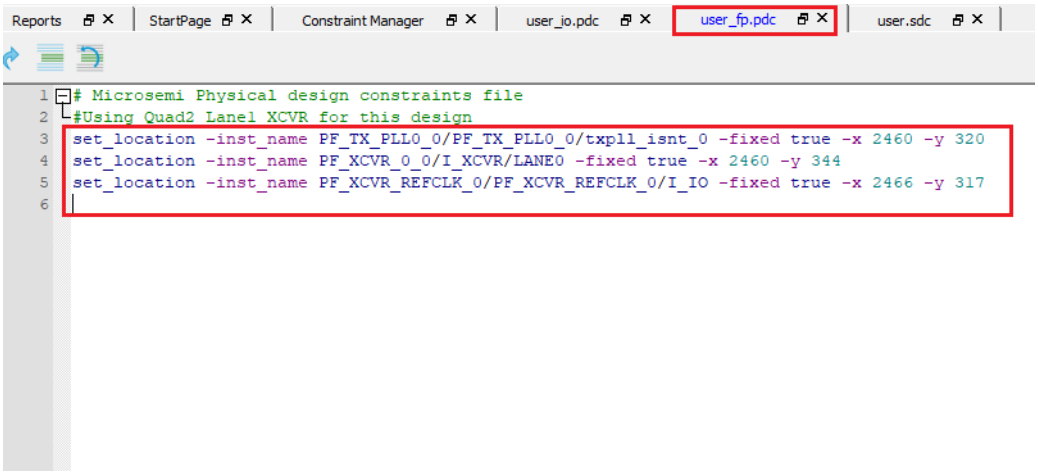
3. Using the **XCVR View** in I/O Editor, place TX_PLL, XCVR_REF_CLK, and PF_XCVR TX as shown in the following figure.

Figure 18 • **I/O Editor Transceiver View**



When all the components are placed, the location of the components is updated in the `user_fp.pdc` file (located in Constraint Manager > Floor planner tab), as shown in the following figure.

Figure 19 • **Component Locations Updated in user_fp.pdc File**



4. On the **Design Flow** window, double-click **Place and Route**.
   When place and route is successful, a green tick mark appears next to **Place and Route**, as shown in Figure 16, page 19.
5. Right-click **Place and Route** and select **View Report** to view the place and route report and log files in the **Reports** tab. View the `top_place_and_route_constraint_coverage.xml` file for place and route constraint coverage.

3.2.1 Resource Utilization

The resource utilization report is written to the `TOP_SD_layout_log.log` file. To view this file, go to the **Reports** tab > **top reports** > **Place and Route**. The following table lists the resource utilization of the design after place and route. These values may vary slightly for different Libero runs, settings, and seed values.

Table 5 • **Resource Utilization**

Type	Used	Total	Percentage
4LUT	4695	299544	1.57
DFF	4787	299544	1.60
I/O register	510	1536	0.00
Logic element	6134	299544	2.05

3.3 Verify Timing

To verify timing, perform the following steps:

1. On the **Design Flow** window, double-click **Verify Timing**.
   When the design meets the timing requirements, a green tick mark appears next to **Verify Timing**, as shown in Figure 16, page 19.
2. Right-click **Verify Timing** and select **View Report** to view the verify timing report and log files in the **Reports** tab.

3.4 Generate FPGA Array Data

On the **Design Flow** tab, double-click **Generate FPGA Array Data**.

When the FPGA array data is generated, a green tick mark appears next to **Generate FPGA Array Data**, as shown in Figure 16, page 19.

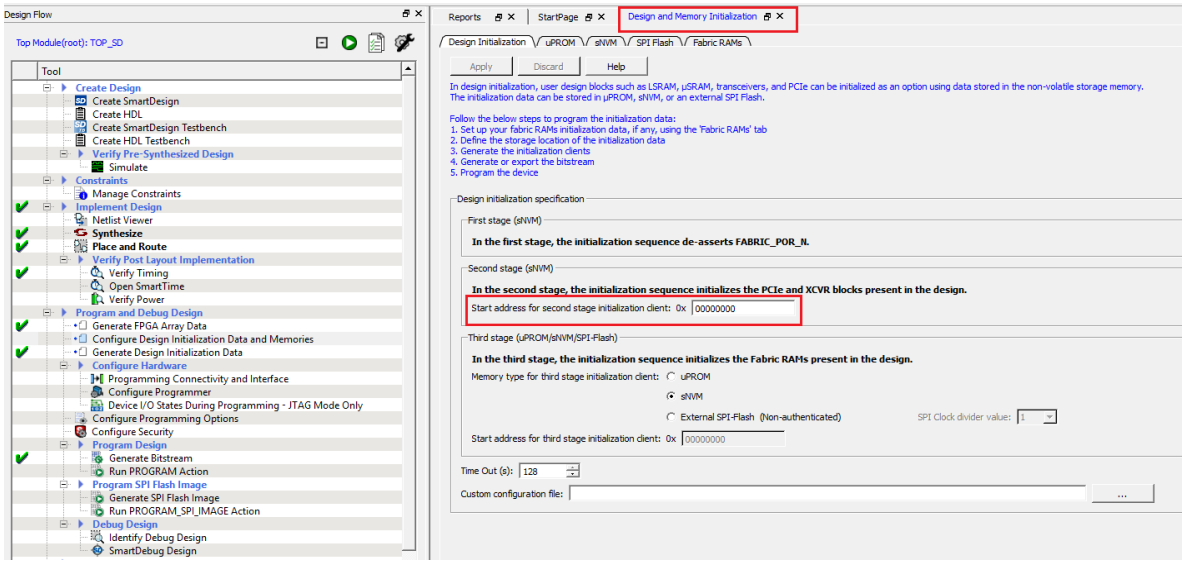
3.5 Configure Design Initialization Data and Memories

The **Configure Design Initialization Data and Memories** option creates the non-PCIe transceiver initialization client, which initializes the transceiver block when the PolarFire device powers up.

To create the transceiver initialization client, perform the following steps:.

1. On the Design Flow window, double-click **Configure Design Initialization Data and Memories**. The Design and Memory Initialization window opens, as shown in the following figure.

Figure 20 • **Design and Memory Initialization Window**



2. Under **Second stage pane (sNVM)** enter the start address where the transceiver initialization client should be created in the sNVM, as shown in the preceding figure.
3. On the **Design Flow** window, double-click the **Generate Design Initialization Data** to generate the initialization client.
   When the initialization client is generated, a green tick mark appears next to **Generate Design Initialization Data**, as shown in Figure 16, page 19.

3.6 Generate Bitstream

To generate the bitstream, perform the following steps:

1. On the **Design Flow** tab, double-click **Generate Bitstream**.

   When the bitstream is successfully generated, a green tick mark appears next to **Generate Bitstream**, as shown in Figure 16, page 19.

2. Right-click **Generate Bitstream** and select **View Report** to view the corresponding log file in the **Reports** tab.

3.7 Run Program Action

After generating the bitstream, the PolarFire device must be programmed. Follow these steps to program the PolarFire device.

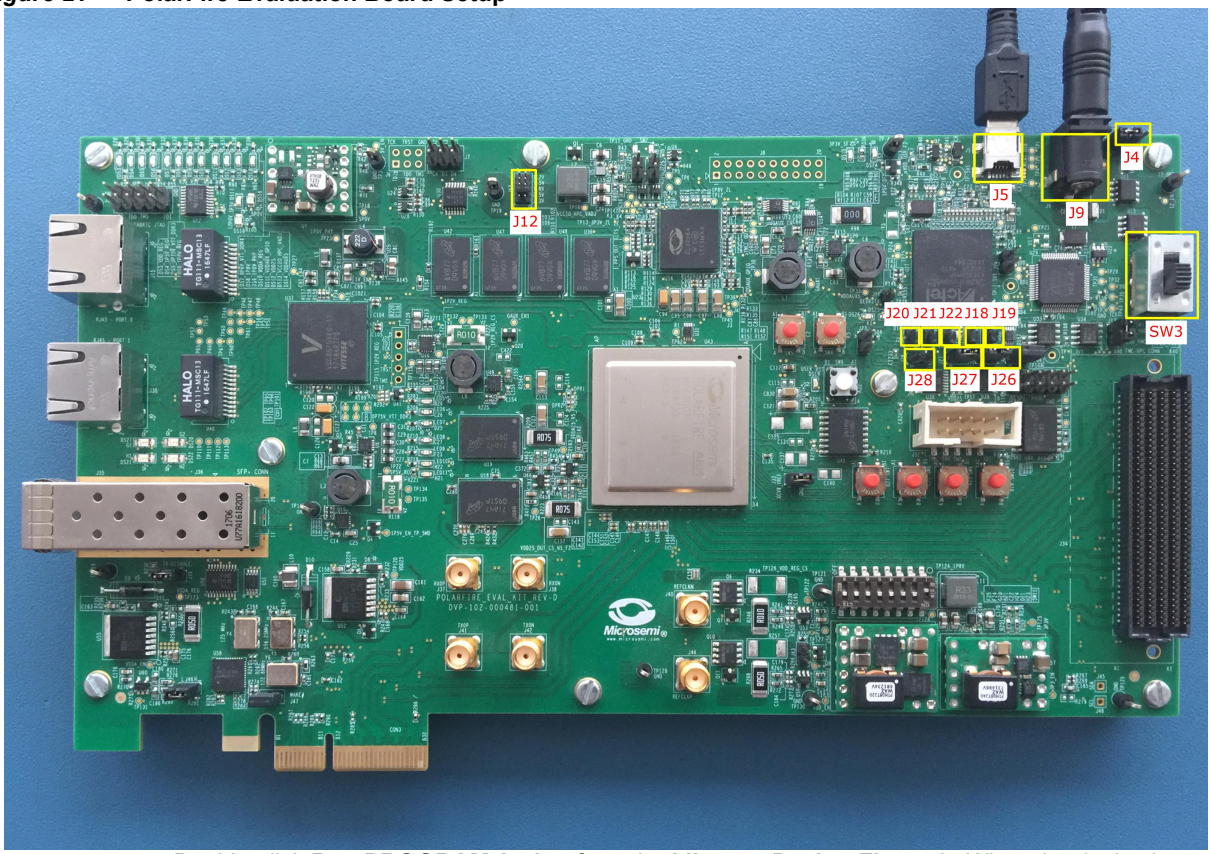
1. Ensure that the jumper settings on the board are as listed in the following table

Table 6 • **Jumper Settings for PolarFire Device Programming**

Jumper	Description
J18, J19, J20, J21, and J22	Short pin 2 and 3 for programming the PolarFire FPGA through FTDI
J28	Short pin 1 and 2 for programming through the on-board FlashPro Express
J26	Short pin 1 and 2 for programming through the FTDI SPI
J27	Short pin 1 and 2 for programming through the FTDI SPI
J39	Short pin 1 and 2 for enabling the TX
J4	Short pin 1 and 2 for manual power switching using SW3

2. Connect the power supply cable to the **J9** connector on the board.
3. Connect the host PC to the **J5** connector (FTDI port) on the board using a USB cable.
4. Power on the board using the **SW3** slide switch.
   The following figure shows the PolarFire Evaluation Board setup for programming the device and running the reference design.

Figure 21 • **PolarFire Evaluation Board Setup**



5. Double-click **Run PROGRAM Action** from the **Libero > Design Flow** tab. When the device is successfully programmed, a green tick mark appears next to **Run PROGRAM Action**, as shown in Figure 16, page 19.

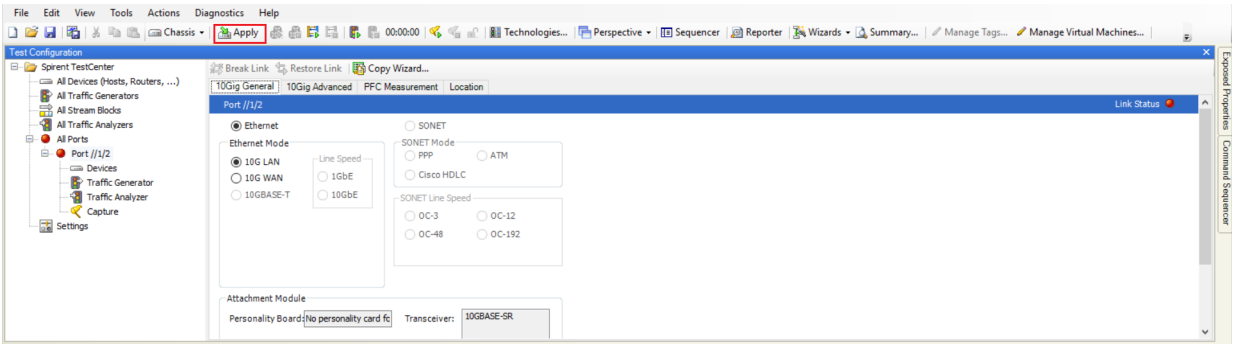
For information about running the demo, see Running the Demo, page 24.

4 Running the Demo

Follow these steps to run the PolarFire 10GBASE-R loopback hardware demo design on the PolarFire Evaluation Board.

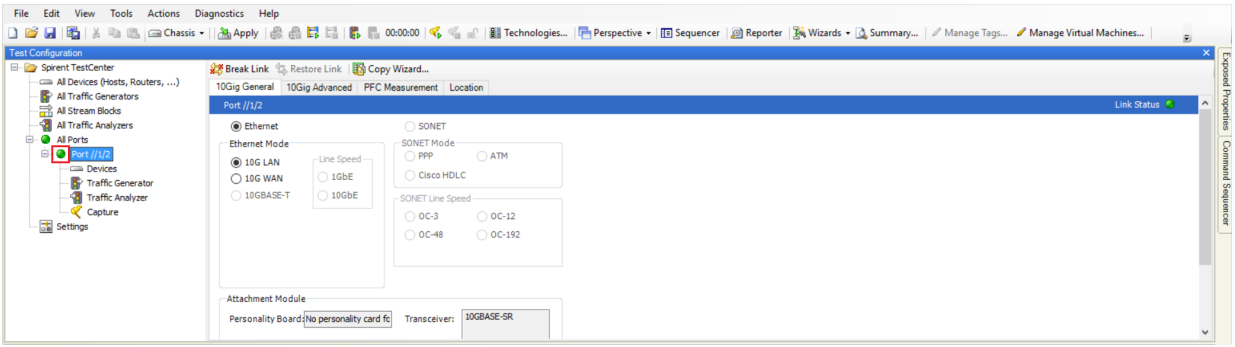
1. Insert a 10G SFP+ module into the J36 connector (SFP+ module cage) on the board.
2. Connect the J36 connector to the Spirent test module using an optical fiber cable.
3. On the host PC, start the Spirent TestCenter software.
4. In the Spirent TestCenter tree, under **All Ports,** click the port to which the optical fiber cable is connected. (In this demo, it is Port //1/2.)
5. On the **10Gig General** tab, select **Ethernet**.
6. Under **Ethernet Mode**, select **10G LAN**, and click **Apply**, as shown in the following figure. The attachment module details are automatically detected and populated.

Figure 22 • **10G Ethernet Settings**



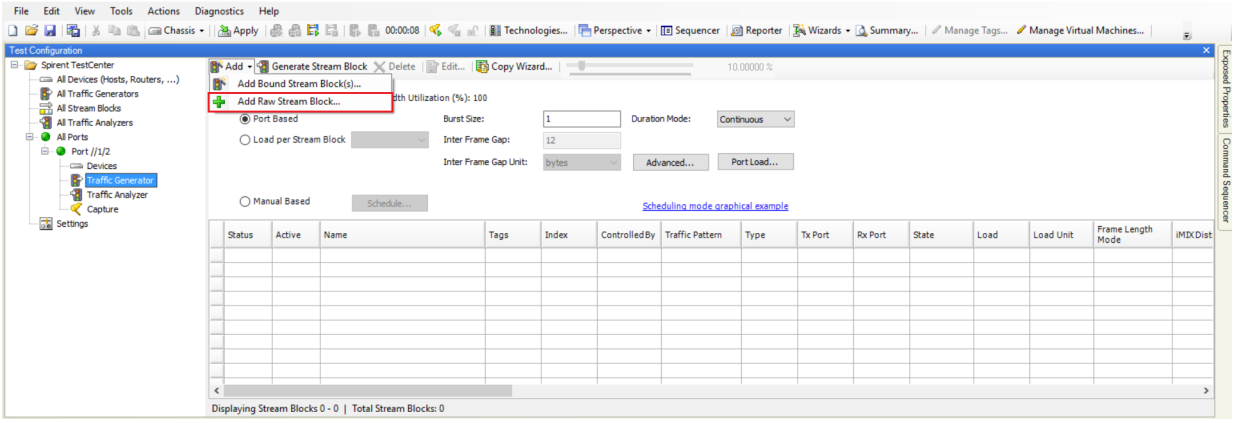
After the changes are applied, the link status icon for the port turns green, indicating that communication has been established between the Spirent test module and the PolarFire Evaluation Board.

Figure 23 • **Link Status**



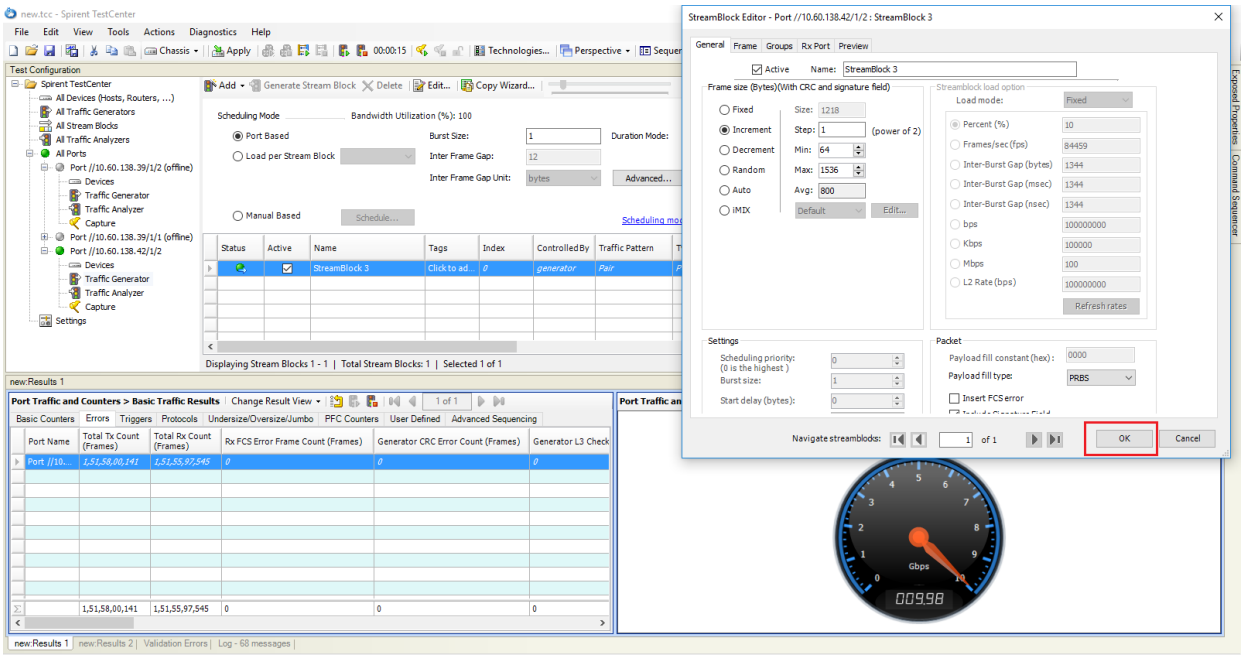
7. Click **Traffic Generator**, and in the **Add** list, click **Add Raw Stream Block**, as shown in the following figure.

Figure 24 • **Stream Block Addition**



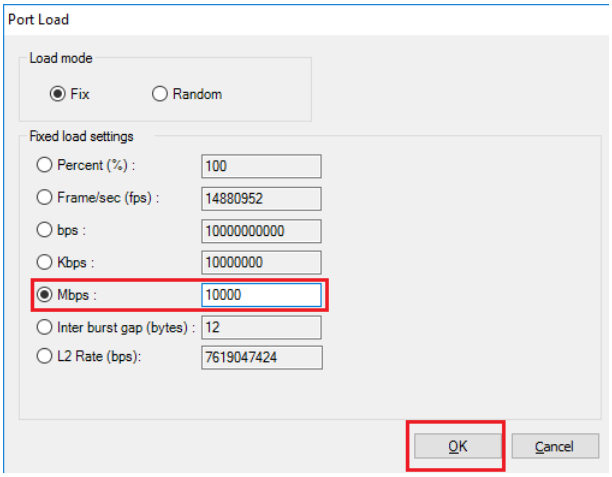
Select any one of the frame type from the options. Configure the frame size and frame format as shown in the following figure. Click **OK**.

Figure 25 • **Stream Block Added**



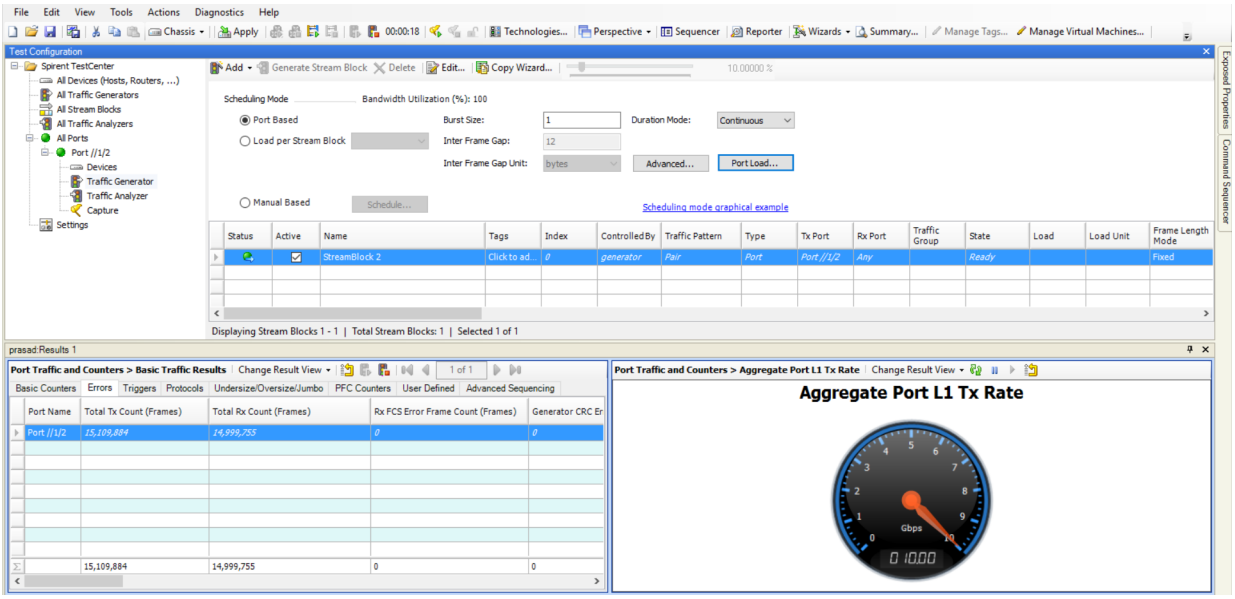
8. Click **Port Load** (highlighted in blue in the Figure 25, page 25), and make sure that under **Fixed load settings**, the speed is set to 10000 Mbps, as shown in the following figure.Click **OK**.

Figure 26 • **Port Load Settings**



9. To begin Ethernet traffic generation, click the **Start Traffic on all ports** icon.
The **Port Traffic and Counters** section starts displaying details of the Ethernet traffic:
   - The **Basic Traffic Results** section displays a real-time count of the Ethernet frames being transmitted and received. The number of Ethernet frames transmitted from the Spirent test module to the PolarFire Evaluation Board is displayed in the **Total Tx Count** field, and the number of Ethernet frames looped back by the PolarFire Evaluation Board to the Spirent test module is displayed in the **Total Rx Count** field.
   - The **Aggregate Port Tx Rate** section shows the rate at which the data is being transmitted.

Figure 27 • **Ethernet Traffic Data**



As shown in the preceding figure, 10G Ethernet packets transmitted from the Spirent test module are successfully looped back by the PolarFire Evaluation Board.

10. Click **Stop Generating Traffic** icon to stop generating traffic from the Spirent test module.

5 Appendix 1: Programming the Device Using FlashPro Express

This chapter describes how to program the PolarFire device with the Job programming file using a FlashPro programmer. The default location of the Job file are located at the following location:
`mpf_dg0757_df\Programming_Files\DG0757_PF_10GBaseR.job`

and

`mpf_dg0757_df\Programming_Files\DG0757_PF_10GBaseR_SYNCE.job`

To program the PolarFire device using FlashPro Express, complete the following steps:

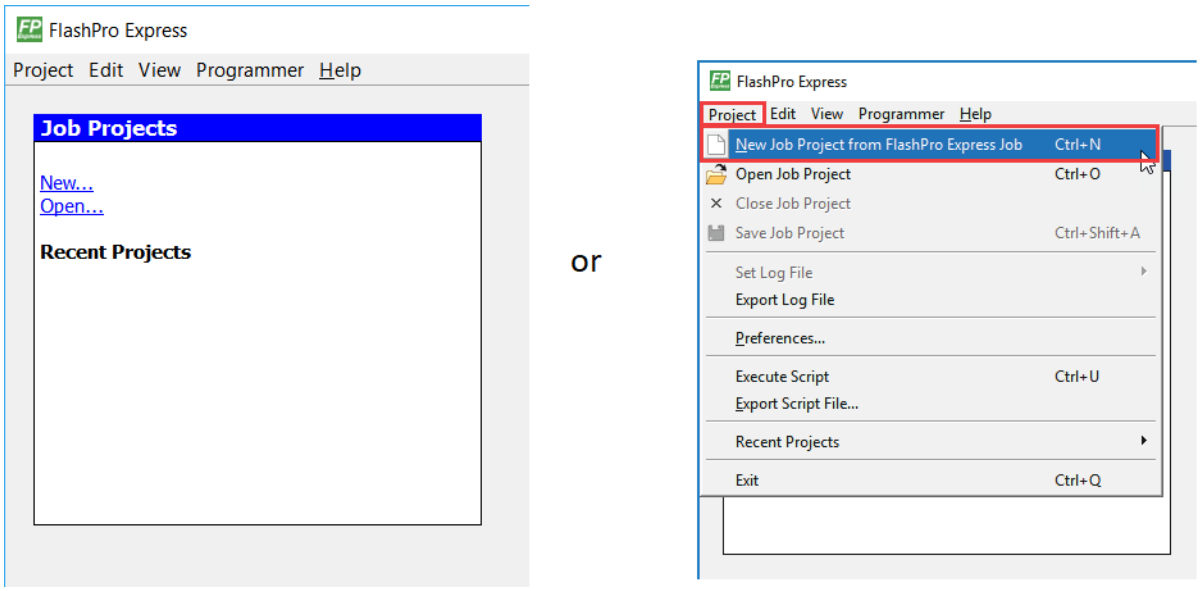
1. Ensure that the jumper settings on the board are the same as listed in Table 6, page 23.

Note: The power supply switch must be switched off while making the jumper connections.

2. Connect the power supply cable to the **J9** connector on the board.
3. Connect the USB cable from the Host PC to the **J5** (FTDI port) on the board.
4. Power on the board using the **SW3** slide switch.
5. On the host PC, launch the FlashPro Express software.
6. To create a new job project, click **New** or

   In the **Project** menu, select **New Job Project from FlashPro Express Job,** as shown in the following figure.

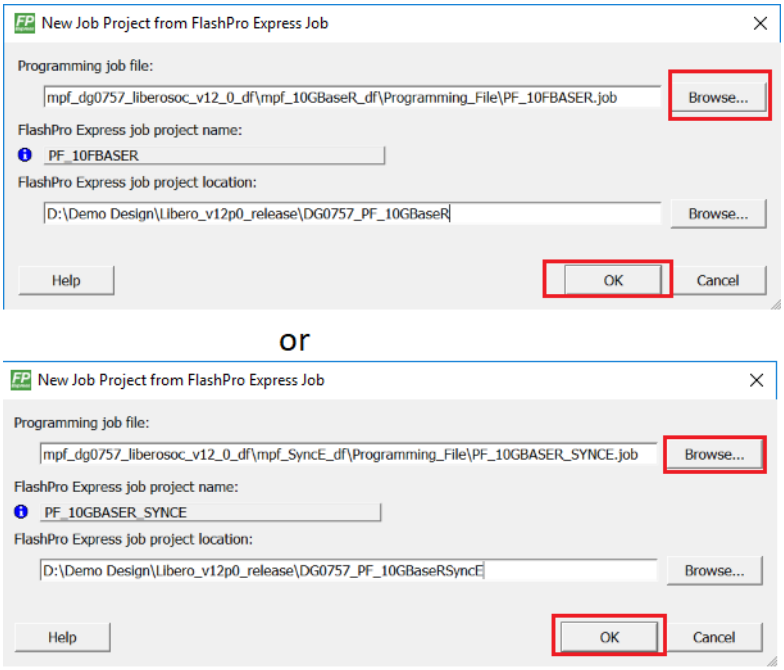
Figure 28 • **FlashPro Express Job Project**



7. Enter the following in the New Job Project from FlashPro Express Job dialog box:

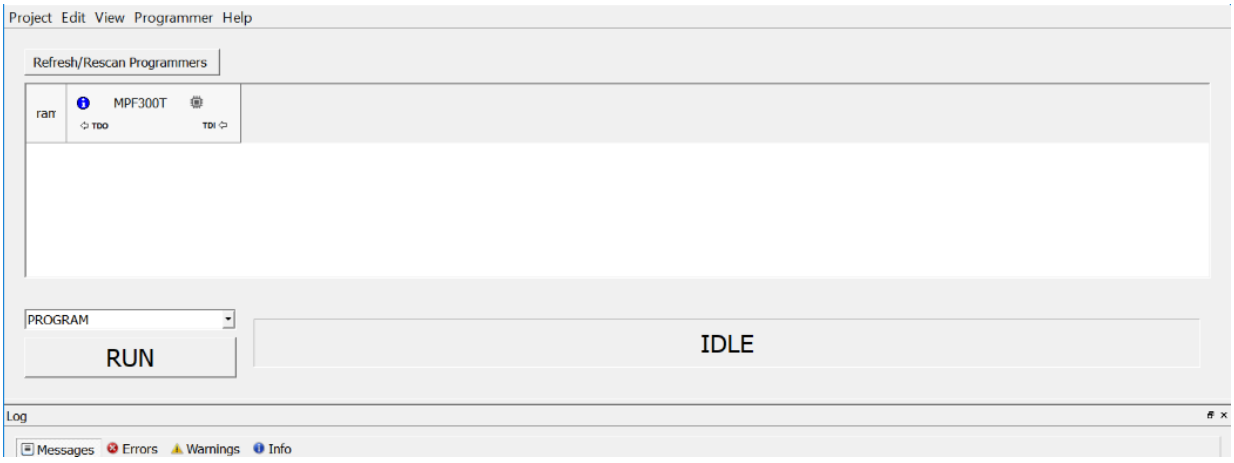
- Programming job file: Click **Browse**, and navigate to the location where the .job file is located and select the file. The default location is: *<download_folder>*`\mpf_dg0757_df\Programming_Files` and *<download_folder>*`mpf_dg0757_df\Programming_Files`
- FlashPro Express job project location: Click **Browse** and navigate to the location where you want to save the project.

Figure 29 • **New Job Project from FlashPro Express Job**



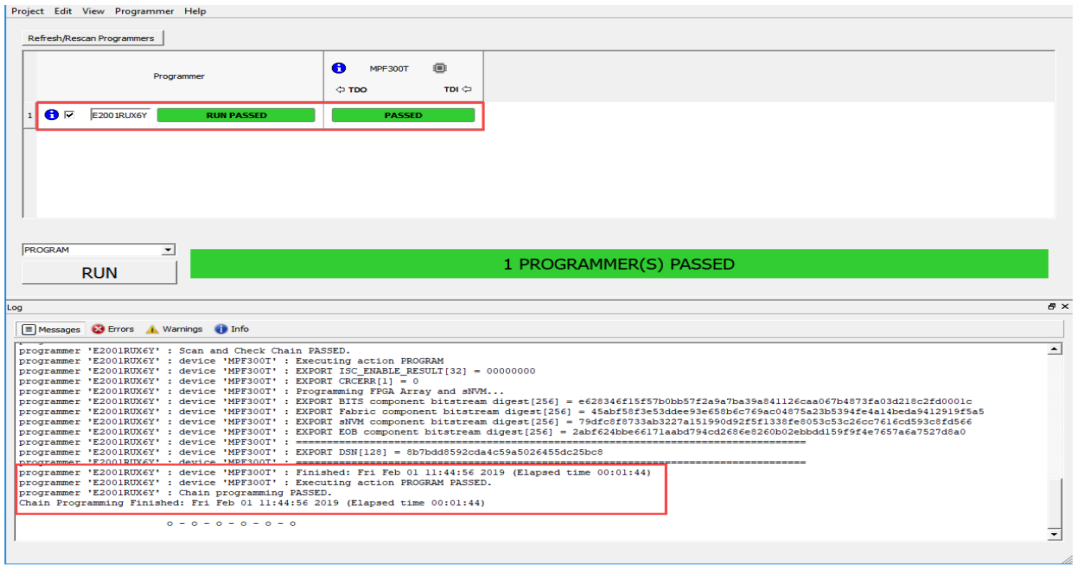
8. Click **OK**. The required programming file is selected and ready to be programmed in the device.
9. The FlashPro Express window appears as shown in the following figure. Confirm that a programmer number appears in the Programmer field. If it does not, confirm the board connections and click **Refresh/Rescan** Programmers.

Figure 30 • **Programming the Device**



10. Click **RUN.** When the device is programmed successfully, a RUN PASSED status is displayed as shown in the following figure.

Figure 31 • **FlashPro Express—RUN PASSED**



11. Close **FlashPro Express** or in the **Project tab, click Exit**.

6 Appendix 2: Enabling SyncE in 10G BaseR Design

This section describes how to enable the SyncE Complaint Jitter Attenuation Phase Locked Loop (JA PLL) in the 10G BaseR Design. The design is validated using Optical Network Tester (ONT). The reference design describes how to build the 10G BaseR design with the JA PLL enabled.

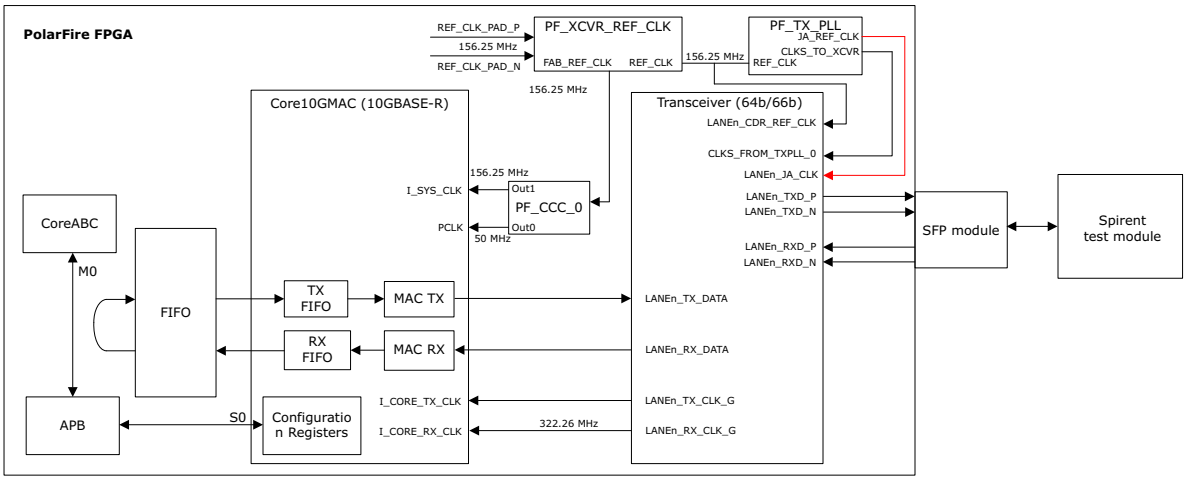
6.1 Prerequisite

Before you begin:

1. Start Libero SoC, and select **Project** > **Tool Profiles**.
2. In the Tool Profiles window, select **Synthesis** and **Simulation** on the **Tools** panes, and select the latest active installation directory paths for these two tools.
3. From the Project menu, click **Open Project**. The Open Project dialog box appears.
4. Navigate to the *mpf_dg0757_df\mpf_SyncE_df\Libero_Project* folder, select the Libero_Project.prjx file, and click **Open**.
   The PolarFire 10G BaseR Ethernet SyncE reference design opens in the Libero.
5. Download the IP cores from Libero SoC Catalog.

The following figure shows the Hardware implementation of the 10G Base-R Ethernet SyncE loopback design.

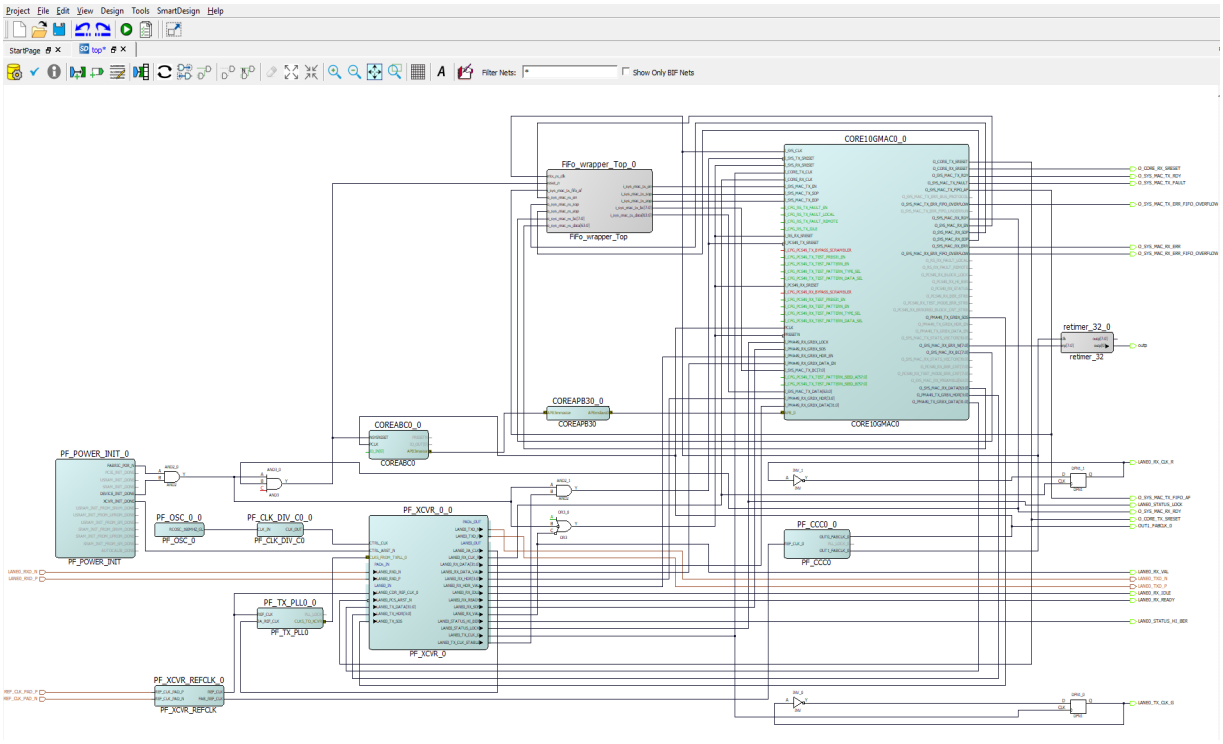
Figure 32 • **Hardware Implementation**



6.2 Design Implementation

The following figure shows the Libero implementation of the 10G Base-R Ethernet loopback design with Jitter Attenuator PLL enabled.

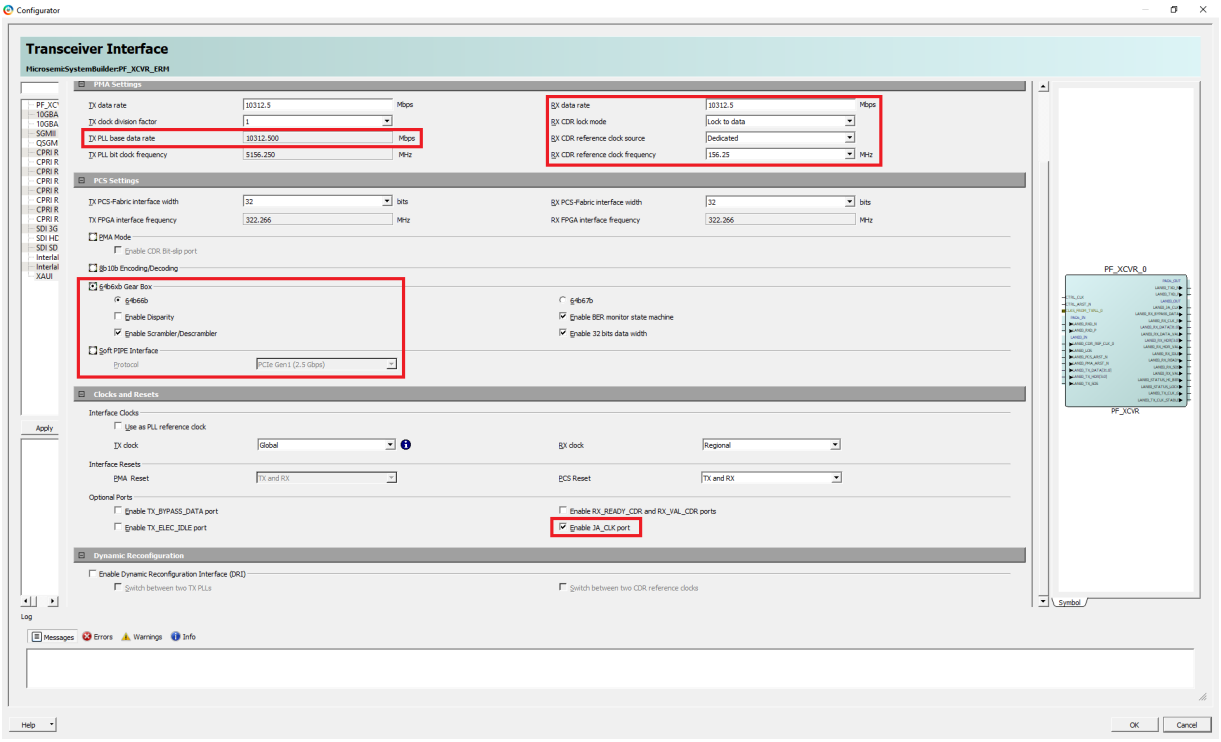
Figure 33 • **10GBASE-R Loopback Hardware Design Libero Implementation —SyncE**



6.2.1 Transceiver

To enable the Jitter attenuator DPLL, select the Enable JA_CLK in addition to the settings specified in Transceiver as shown in the following figure.

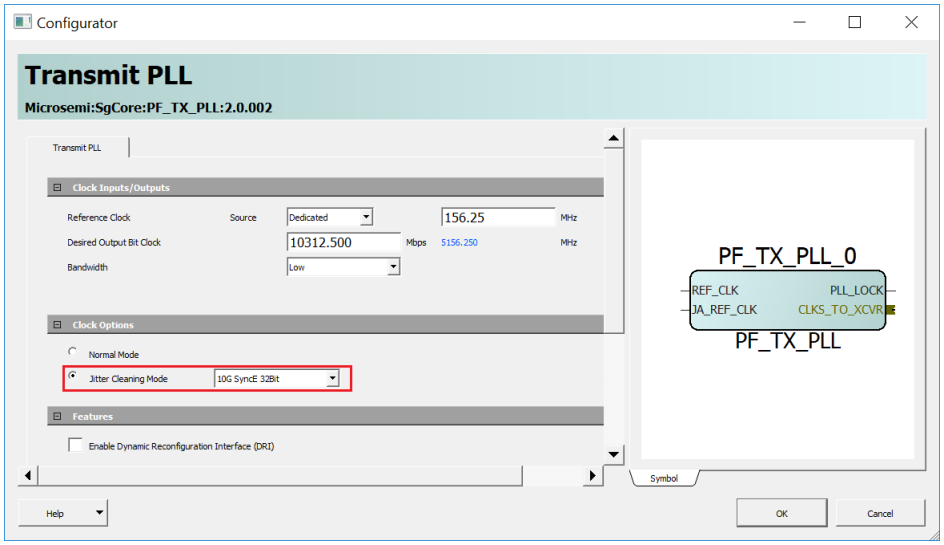
Figure 34 • **Transceiver—SyncE**



6.2.1.1 Transmit PLL

The clock option is set to Jitter cleaning mode to enable the Jitter attenuator PLL as shown in the following figure.

Figure 35 • **Transmit PLL—SyncE**



The remaining IP configurations, clocking structure, and the reset structure of this design is same as the demo design explained earlier. For more information, see Demo Design, page 4.

6.3 Libero Design Flow

For more information, see Libero Design Flow, page 19.

6.4 Programming the Device Using FlashPro Express

The .job programming file is programmed using a FlashPro Express programmer. The file is located at:

```
mpf_dg0757_df\Programming_Files
```

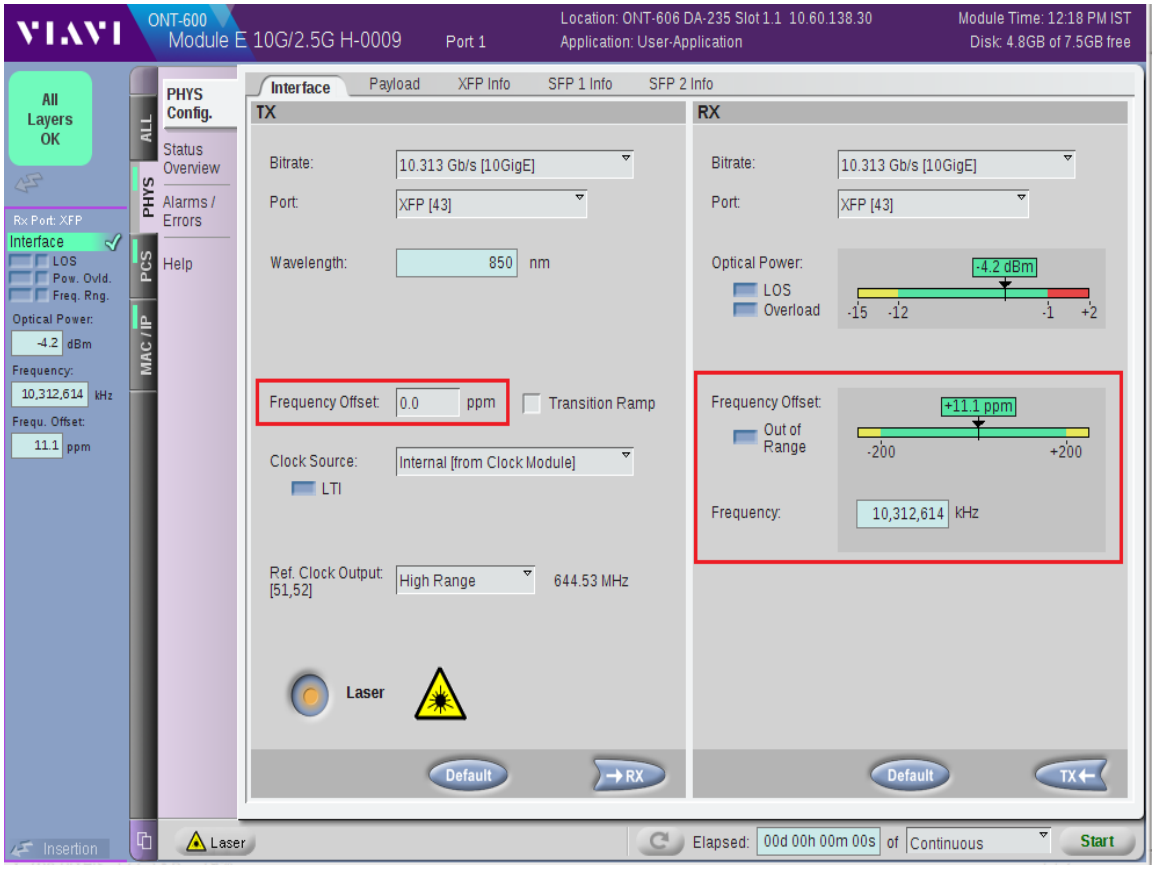
For more information, see Appendix 1: Programming the Device Using FlashPro Express, page 27.

6.5 Running the Demo

Follow these steps to run the PolarFire 10GBASE-R loopback hardware demo design on the PolarFire Evaluation Board.

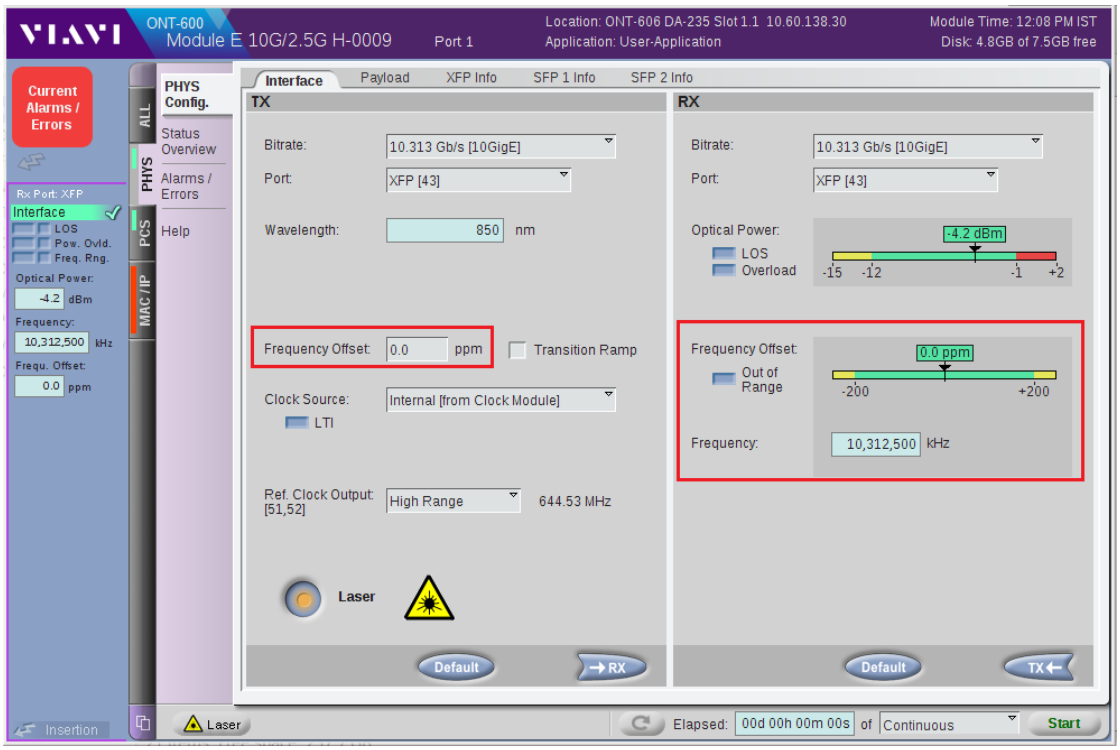
1. The reference design is validated using the ONT. There exists the ppm offset in the TX and RX frequencies when the Jitter cleaning Mode is not enabled in the TX PLL as shown in the following figure.

Figure 36 • **Frequency Offset—SyncE Disabled**



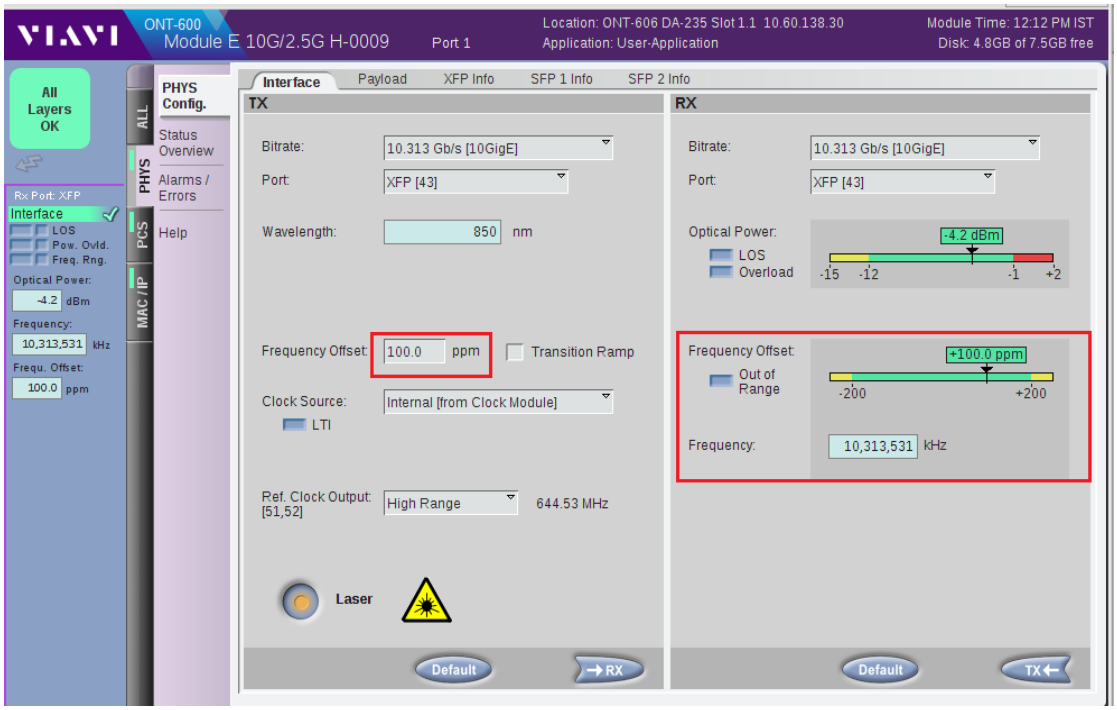
2. When the design is built with Jitter Cleaning Mode of TX PLL enabled the Frequency offset between Tx and Rx is 0 ppm.

Figure 37 • **Frequency Offset Between TX and RX**



3. When the Tx clock frequency is offset by 100 ppm, the Rx Clock frequency also gets adjusted by 100 ppm, which shows the JA PLL is tracking and adjusting the clock as per the offset in the received Clock.

Figure 38 • **Frequency Offset—SyncE Enabled**



7 Appendix 3: References

This section lists documents that provide more information about the concepts and features covered in this demo guide.

- For more information about 10G Ethernet, refer to the IEEE 802.3 standard in the *IEEE website*.
- For more information about PolarFire transceiver blocks, see *UG0677: PolarFire FPGA Transceiver User Guide*.
- For more information about Libero, ModelSim, and Synplify, visit *Microsemi Libero SoC webpage*.