Microsemi makes no warranty, representation, or guarantee regarding the information contained herein or the suitability of its products and services for any particular purpose, nor does Microsemi assume any liability whatsoever arising out of the application or use of any product or circuit. The products sold hereunder and any other products sold by Microsemi have been subject to limited testing and should not be used in conjunction with mission-critical equipment or applications. Any performance specifications are believed to be reliable but are not verified, and Buyer must conduct and complete all performance and other testing of the products, alone and together with, or installed in, any end-products. Buyer shall not rely on any data and performance specifications or parameters provided by Microsemi. It is the Buyer’s responsibility to independently determine suitability of any products and to test and verify the same. The information provided by Microsemi hereunder is provided “as is, where is” and with all faults, and the entire risk associated with such information is entirely with the Buyer. Microsemi does not grant, explicitly or implicitly, to any party any patent rights, licenses, or any other IP rights, whether with regard to such information itself or anything described by such information. Information provided in this document is proprietary to Microsemi, and Microsemi reserves the right to make any changes to the information in this document or to any products and services at any time without notice.

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

1 Revision History ................................................. 1
   1.1 Revision 6.0 ................................................. 1
   1.2 Revision 5.0 ................................................. 1
   1.3 Revision 4.0 ................................................. 1
   1.4 Revision 3.0 ................................................. 1
   1.5 Revision 2.0 ................................................. 1
   1.6 Revision 1.0 ................................................. 2

2 PolarFire FPGA 10GBASE-R Ethernet Loopback ......................... 3
   2.1 Design Requirements ........................................ 3
   2.2 Prerequisites ............................................... 4
   2.3 Demo Design ............................................... 4
      2.3.1 Design Implementation .................................. 6
      2.3.2 IP Configuration ........................................ 6
      2.3.3 Clocking Structure ...................................... 11
      2.3.4 Reset Structure ......................................... 12
   2.4 Simulating the 10GBASE-R Ethernet Loopback Design ................ 13
      2.4.1 Prerequisites ........................................... 13
      2.4.2 Design Description ....................................... 14
      2.4.3 Design Implementation ................................... 15
      2.4.4 Simulation Flow ......................................... 16

3 Libero Design Flow .............................................. 20
   3.1 Synthesize .................................................. 20
   3.2 Place and Route ............................................ 20
      3.2.1 Resource Utilization .................................... 22
   3.3 Verify Timing ............................................... 22
   3.4 Generate FPGA Array Data .................................... 22
   3.5 Configure Design Initialization Data and Memories .................. 23
   3.6 Generate Bitstream .......................................... 23
   3.7 Run Program Action ......................................... 23

4 Programming the Device Using FlashPro .............................. 25

5 Running the Demo .................................................. 27

6 Appendix: Enabling SyncE in 10G BaseR Design ......................... 30
   6.1 Prerequisite .................................................. 30
   6.2 Design Implementation ....................................... 31
      6.2.1 Transceiver .............................................. 32
   6.3 Libero Design Flow .......................................... 33
   6.4 Programming the Device Using Flash Pro ........................ 33
   6.5 Running the Demo ............................................ 34

7 Appendix: References ............................................... 37
Figures

Figure 1 Hardware Implementation Block Diagram .................................................. 5
Figure 2 10GBASE-R Loopback Hardware Design Libero Implementation .......................... 6
Figure 3 CORE10GMAC Configuration ................................................................. 7
Figure 4 Transceiver Interface Configuration ......................................................... 8
Figure 5 Transmit PLL Configuration ..................................................................... 9
Figure 6 Transceiver Reference Clock Configuration ............................................ 10
Figure 7 Clocking Structure ................................................................................. 11
Figure 8 Reset Structure ...................................................................................... 12
Figure 9 Testbench and 10GBASE-R Ethernet Loopback Design Interaction ............... 13
Figure 10 Libero Implementation of the top SmartDesign Module ............................... 15
Figure 11 Libero Implementation of the top-tb SmartDesign Module ......................... 15
Figure 12 Project Directory Selection .................................................................. 16
Figure 13 User Defined DO File Location ............................................................. 17
Figure 14 Simulate Option in Libero Design Flow .................................................. 17
Figure 15 Use Automatic DO File Option Selected ................................................ 17
Figure 16 O_CORE_RX_SRESET and O_CORE_RX_SRESET at 0 ......................... 18
Figure 17 Ethernet Packet Sent ......................................................................... 18
Figure 18 Good Packets Count Incremented by 1 .................................................. 19
Figure 19 Libero Design Flow Options .................................................................. 20
Figure 20 Edit with I/O Editor Option ................................................................. 21
Figure 21 I/O Editor Transceiver View .................................................................. 21
Figure 22 Component Locations Updated in user.pdc File ..................................... 22
Figure 23 Design and Memory Initialization Window ............................................ 23
Figure 24 PolarFire Evaluation Board Setup ......................................................... 24
Figure 25 FlashPro New Project Window ............................................................. 25
Figure 26 Programming File Details in FlashPro .................................................. 26
Figure 27 Programming Passed .......................................................................... 26
Figure 28 10G Ethernet Settings ....................................................................... 27
Figure 29 Link Status ......................................................................................... 27
Figure 30 Stream Block Addition ....................................................................... 28
Figure 31 Stream Block Added ........................................................................... 28
Figure 32 Port Load Settings ............................................................................. 29
Figure 33 Ethernet Traffic Data ......................................................................... 29
Figure 34 Hardware Implementation .................................................................... 30
Figure 35 10GBASE-R Loopback Hardware Design Libero Implementation —SyncE .......................... 31
Figure 36 Transceiver—SyncE ........................................................................... 32
Figure 37 Transmit PLL—SyncE .......................................................................... 33
Figure 38 Frequency Offset—SyncE Disabled ...................................................... 34
Figure 39 Frequency Offset Between TX and RX ................................................... 35
Figure 40 Frequency Offset—SyncE Enabled ........................................................ 35
Figure 41 Waveform—Tx and Rx Output ............................................................. 36
## Tables

<table>
<thead>
<tr>
<th>Table</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Table 1</td>
<td>Design Requirements</td>
<td>3</td>
</tr>
<tr>
<td>Table 2</td>
<td>Hardware Design Clock Frequencies</td>
<td>5</td>
</tr>
<tr>
<td>Table 3</td>
<td>CORE10GMAC Configuration</td>
<td>7</td>
</tr>
<tr>
<td>Table 4</td>
<td>Simulation Signals</td>
<td>14</td>
</tr>
<tr>
<td>Table 5</td>
<td>Resource Utilization</td>
<td>22</td>
</tr>
<tr>
<td>Table 6</td>
<td>PolarFire Evaluation Board Jumper Settings</td>
<td>24</td>
</tr>
</tbody>
</table>
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 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: Enabling SyncE in 10G BaseR Design, page 30.

1.2 Revision 5.0

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

1.3 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 6.
- IP configuration details were updated. For more information, see 10GBASE-R Loopback Hardware Design Libero Implementation, page 6.
- 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 12.
- 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 13.

1.4 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 27.

1.5 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 22.
1.6 Revision 1.0

The first publication of this document.
2 PolarFire FPGA 10GBASE-R Ethernet Loopback

The PolarFire® FPGA 10G Ethernet solution is compliant with the IEEE 802.3ae standard, which supports data transfer rates of up to 10.3125 Gbps. Advantages offered by using PolarFire FPGAs for building 10G Ethernet solutions include the use of low-power transceivers, low-power FPGA fabric, and 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

2.1 Design Requirements

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

<table>
<thead>
<tr>
<th>Requirement</th>
<th>Version</th>
</tr>
</thead>
<tbody>
<tr>
<td>Operating system</td>
<td>Windows 7, Windows 8.1, or Windows 10</td>
</tr>
<tr>
<td>Hardware**</td>
<td>Rev D or later</td>
</tr>
<tr>
<td>PolarFire Evaluation Kit (MPF300-EVAL-KIT):</td>
<td></td>
</tr>
<tr>
<td>– PolarFire Evaluation Board</td>
<td></td>
</tr>
<tr>
<td>– 12 V/5 A power adapter</td>
<td></td>
</tr>
<tr>
<td>– USB 2.0 A-male to mini-B cable for</td>
<td></td>
</tr>
<tr>
<td>programming</td>
<td></td>
</tr>
<tr>
<td>Spirent test module for 10G Ethernet traffic generation</td>
<td></td>
</tr>
<tr>
<td>Optical fiber cable</td>
<td></td>
</tr>
<tr>
<td>Small form-factor pluggable (SFP+) module</td>
<td></td>
</tr>
<tr>
<td>Software</td>
<td></td>
</tr>
<tr>
<td>Libero SoC PolarFire</td>
<td>v2.2</td>
</tr>
<tr>
<td>Synplify Pro</td>
<td>N-2017.09M-SP1-1</td>
</tr>
<tr>
<td>Spirent TestCenter**</td>
<td>4.69.9486.0000</td>
</tr>
<tr>
<td>FlashPro**</td>
<td>v2.2</td>
</tr>
<tr>
<td>ModelSim**</td>
<td>SE 10.5C or ME 10.5 Pro</td>
</tr>
<tr>
<td>IP</td>
<td></td>
</tr>
<tr>
<td>CORE10GMAC</td>
<td>2.1.126</td>
</tr>
<tr>
<td>Transceiver Interface</td>
<td>1.0.232</td>
</tr>
<tr>
<td>CoreABC</td>
<td>3.7.101</td>
</tr>
<tr>
<td>PF_XCVR_REF_CLK</td>
<td>1.0.103</td>
</tr>
</tbody>
</table>
2.2 Prerequisites

Before you start:

1. Download the design files from the following location:

2. Download and install Libero SoC PolarFire on the host PC. The latest version of Synplify Pro is included in the Libero SoC PolarFire installation package.
   https://www.microsemi.com/product-directory/design-resources/3863-libero-soc-polarfire#downloads

3. From the Project menu, click Open Project.

4. Navigate to the mpf_dg0757_liberosocpolarfirev2p2_df/Middleware/PF_10GBASER/ folder, select the PF_10GBASER.prjx file, and click Open.

5. Download the following IP cores from Libero SoC PolarFire > Catalog:
   - CORE10GMAC
   - PF_XCVR
   - PF_TX_PLL
   - PF_XCVR_REF_CLK

Note: To simulate this demo design, see Simulating the 10GBASE-R Ethernet Loopback Design, page 13.

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 is received by the Spirent test center. 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**

<table>
<thead>
<tr>
<th>Clock</th>
<th>Frequency (MHz)</th>
</tr>
</thead>
<tbody>
<tr>
<td>CDR reference clock</td>
<td>156.25</td>
</tr>
<tr>
<td>Transceiver bit clock</td>
<td>156.25</td>
</tr>
<tr>
<td>I_SYS_CLOCK</td>
<td>156.25</td>
</tr>
<tr>
<td>I_CORE_TX_CLK</td>
<td>322.26</td>
</tr>
<tr>
<td>I_CORE_RX_CLK</td>
<td>322.26</td>
</tr>
<tr>
<td>PCLK</td>
<td>50</td>
</tr>
</tbody>
</table>

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

**Figure 1 • Hardware Implementation Block Diagram**

---

Microsemi Proprietary and Confidential DG0757 Demo Guide Revision 6.0
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 IP Configuration

The following sections describe the IP blocks and user-defined blocks used in this demo.

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**

<table>
<thead>
<tr>
<th>Register</th>
<th>Address</th>
<th>Offset</th>
<th>Bit</th>
<th>Binary Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>MAC Tx Config Register</td>
<td>(0xA)</td>
<td>0x3</td>
<td>cfg_sys_mac_tx_en</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0x4</td>
<td>sys_mac_tx_fcs_ins</td>
<td>1</td>
</tr>
<tr>
<td>MAC Rx Config Register</td>
<td>(0xB)</td>
<td>0x0</td>
<td>mac_rx_fcs_remove</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0x3</td>
<td>cfg_sys_mac_rx_en</td>
<td>1</td>
</tr>
</tbody>
</table>
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 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.

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 in a systematic way. 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 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 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.

### 2.3.2.10 Retime_1_bit

The Retime_1_bit block is a four-stage synchronizer that synchronizes the signals from the transceiver module to the Fabric Clock generated from the PF_CCC_0 that is used to reset the transmit and receive logic.

### 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 following figure shows the clocking structure of the design.

---

**Figure 7** Clocking Structure

![Clocking Structure Diagram](image_url)
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 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.
2.4 Simulating the 10GBASE-R Ethernet Loopback Design

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

2.4.1 Prerequisites

Before you start:

1. Start Libero SoC PolarFire, 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_liberosocpolarfirev2p2/mpf_10BaseR_df/Libero_Project/Simulation/BaseR folder, select the BaseR.prjx file, and click Open.
   - The PolarFire 10G Ethernet Simulation design project opens in Libero.
5. Download the following IP cores from Libero SoC PolarFire Catalog:
   - PF_XCVR
   - PF_TX_PLL
   - PF_XCVR_REF_CLK
   - CORE10GMAC

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 RTL.

The following table lists the simulation signals transmitted between the Testbench\_10g, top, and top\_tb blocks.

<table>
<thead>
<tr>
<th>Output From</th>
<th>Input To</th>
<th>Signal</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>testbench_10g</td>
<td>top, top_tb</td>
<td>SYSCLK</td>
<td>156.25-MHz clock</td>
</tr>
<tr>
<td></td>
<td></td>
<td>OUT0_FABCLK_0_net_0</td>
<td>50-MHz clock</td>
</tr>
<tr>
<td>top</td>
<td>testbench_10g</td>
<td>LANE0_RXD_P</td>
<td>Rx port signals</td>
</tr>
<tr>
<td></td>
<td></td>
<td>LANE0_RXD_N</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>LANE0_TXD_P</td>
<td>Tx port signals</td>
</tr>
<tr>
<td></td>
<td></td>
<td>LANE0_TXD_N</td>
<td></td>
</tr>
<tr>
<td>top_tb</td>
<td>testbench_10g</td>
<td>LANE0_RXD_P_0</td>
<td>Rx port signals</td>
</tr>
<tr>
<td></td>
<td></td>
<td>LANE0_RXD_N_0</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>LANE0_TXD_P_0</td>
<td>Tx port signals</td>
</tr>
<tr>
<td></td>
<td></td>
<td>LANE0_TXD_N_0</td>
<td></td>
</tr>
</tbody>
</table>
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 either ModelSim SE 10.5C installed separately or ModelSim ME 10.5C Pro provided with the Libero SoC PolarFire installation. The following sections describe the simulation flow for each of these ModelSim versions.

2.4.4.1 Initiating Simulation with ModelSim ME 10.5C Pro

Follow these steps to initiate the simulation using ModelSim ME 10.5C Pro.

1. From the design files folder, copy the precompiled library (the `core10gmac_library` folder) to the host PC.

2. From the `mpf_dg0757_liberosocpolarfirerev2p2_df/mpf_10GBaseR_df/Libero_Project/Simulation/BaseR/simulation` folder, open the `run_10gprecompiled_lib.do` file.

3. Set the project directory to the location where the project files are copied on the host PC, for example, `F:/mpf_dg0757_liberosocpolarfirerev2p2_df/mpf_10GBaseR_df/Libero_Project/Simulation/BaseR`

   ![Figure 12 • Project Directory Selection](image)

   The pre-compiled library mapping is added to the DO file, as shown in the preceding figure. Ensure that the library mapping points to the folder where the pre-compiled libraries are copied on the host PC.

4. Verify that the PolarFire Library is mapped to the location where Libero Soc PolarFire software is installed in the host PC.

5. Save the file.

6. In Libero SoC PolarFire, go to Project > Project Settings > Simulation options > DO file, and clear the Use automatic DO file check box.

7. Under User defined DO file, enter the location of the saved DO file, as shown in the following figure.
8. Click **Save**, and then click **Close**.

9. Double-click **Simulate** under **Verify Pre-Synthesized Design** in the **Design Flow** tab of Libero SoC PolarFire.

10. The **Simulate** option is highlighted in the following figure.

---

### 2.4.4.2 Initiating Simulation with ModelSim SE 10.5C

To simulate the design using ModelSim SE 10.5C:

1. In Libero SoC PolarFire, 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.

---

2. Double-click **Simulate** under **Verify Pre-Synthesized Design** in the **Design Flow** tab of Libero SoC PolarFire, as shown in **Figure 14**, page 17.

---

### 2.4.4.3 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. At the 20631.147 ns, 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 16 • O_CORE_RX_SRESET and O_CORE_RX_SRESET at 0

3. At 30100 ns, 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 17 • Ethernet Packet Sent

4. At 31094.4 ns, 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. At 31151.945 ns, it increments the good packets (good_pckts) count by 1, as shown in the following figure.

Figure 18 • 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.
This chapter describes the Libero design flow, which involves the following steps:

- **Synthesize**, page 20
- **Place and Route**, page 20
- **Verify Timing**, page 22
- **Generate FPGA Array Data**, page 22
- **Configure Design Initialization Data and Memories**, page 23
- **Generate Bitstream**, page 23

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

### Figure 19 • Libero Design Flow Options

![Libero Design Flow Options](image_url)

#### 3.1 Synthesize

Go to the **Design Flow** tab, and double-click **Synthesize**.

When the synthesis is successful, a green tick mark appears next to **Synthesize**, as shown in **Figure 19**, page 20.

Right-click **Synthesize** and select **View Report** to view the synthesis report and log files in the **Reports** tab. View the `top.srr` and `top_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** tab, double-click **Manage Constraints**.
2. On the I/O Attributes tab, click **Edit with I/O Editor**, as shown in the following figure.

*Figure 20 • 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 21 • I/O Editor Transceiver View*

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

The resource utilization report is written to the top_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>
<thead>
<tr>
<th>Type</th>
<th>Used</th>
<th>Total</th>
<th>Percentage</th>
</tr>
</thead>
<tbody>
<tr>
<td>4LUT</td>
<td>4644</td>
<td>299544</td>
<td>1.55</td>
</tr>
<tr>
<td>DFF</td>
<td>4764</td>
<td>299544</td>
<td>1.59</td>
</tr>
<tr>
<td>I/O register</td>
<td>0</td>
<td>510</td>
<td>0.00</td>
</tr>
<tr>
<td>Logic element</td>
<td>6033</td>
<td>299544</td>
<td>2.01</td>
</tr>
</tbody>
</table>

3.3 Verify Timing


When the design meets the timing requirements, a green tick mark appears next to Verify Timing, as shown in Figure 19, page 20.

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 19, page 20.
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.

Follow these steps to create the transceiver initialization client.

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

![Design and Memory Initialization Window](image)

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** tab, 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 19, page 20.

3.6 Generate Bitstream

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 19, page 20.

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>
<thead>
<tr>
<th>Jumper</th>
<th>Pin From</th>
<th>Pin To</th>
</tr>
</thead>
<tbody>
<tr>
<td>J27</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>J28</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>J39</td>
<td>1</td>
<td>2</td>
</tr>
</tbody>
</table>

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 24 • 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 19, page 20.

For information about running the demo, see Running the Demo, page 27.
This chapter describes how to program the PolarFire device with the .stp programming file using a FlashPro programmer. The default location of the .stp file is:

```
mpf_dg0757_liberosocpolarfirerrev2p2_df\mpf_10GBaseR_df\Programming_file
```

Follow these steps to program the device.

1. Connect the jumpers and set up the PolarFire Evaluation Board as described in steps 1 to 5 of Run Program Action, page 23.
2. On the host PC, start the FlashPro software.
3. Click New Project, and in the New Project window, enter the project name.

**Figure 25 • FlashPro New Project Window**

4. Click Browse, and navigate to the location where the project is required to be saved.
5. Select Single device as the programming mode.
6. Click OK to save the project.
   The Configure Device option (disabled in the preceding figure) is enabled.
7. Click Configure Device.
8. In the **Programming File** section, click **Browse**, navigate to the location where the **PF_10GMAC_Demo.stp** file is located, and select the file. Details of the selected file are displayed, as shown in the following figure.

**Figure 26 • Programming File Details in FlashPro**

9. In the **Action** list, select **PROGRAM**, as shown in the preceding figure.

10. Click the main **PROGRAM** option (also highlighted in the preceding figure) to program the device. Wait until the programmer status changes to **RUN PASSED** and the **PROGRAM PASSED** message appears in the log window.

**Figure 27 • Programming Passed**
5 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 10G 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 28 • 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 29 • Link Status
7. Click **Traffic Generator**, and in the **Add** list, click **Add Raw Stream Block**, as shown in the following figure.

**Figure 30 • 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.

**Figure 31 • Stream Block Added**
8. Click **Port Load** (highlighted in blue in the preceding figure), and make sure that under **Fixed load settings**, the speed is set to 10000 Mbps, as shown in the following figure.

**Figure 32 • 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 33 • 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 the **Stop Generating Traffic** icon to stop generating traffic from the Spirent test module.
6 Appendix: 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.

**Note:** SyncE option through Libero is supported from version 2.2 onwards. In Libero v2.2 the support is only for 10G rate and 32 bit transceiver interface.

### 6.1 Prerequisite

Before you start:

1. Start Libero SoC PolarFire, 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_liberoscpolarfirerev2p2_df\mpf_SYNC_df\Libero` Project folder, select the `PF_10GBASER.prjx` file, and click **Open**. The PolarFire 10G BaseR Ethernet SyncE reference design opens in the Libero.
5. Download the following IP cores from Libero SoC PolarFire Catalog:
   - PF_XCVR
   - PF_TX_PLL
   - PF_XCVR_REF_CLK
   - CORE10GMAC

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

**Figure 34 • 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 35*  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 36 • 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 37 • Transmit PLL—SyncE

![Transmit PLL](image)

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 33.

6.4 **Programming the Device Using Flash Pro**

The PolarFire device with the .stp programming file using a FlashPro programmer is located at: `mpf_dg0757_liberosocpolarfirerev2p2_df\mpf_SyncE_df\Programming Files`

For more information, see Programming the Device Using Flash Pro, page 33.
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 following figure.

*Figure 38 • 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 39 • 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 40 • Frequency Offset—SyncE Enabled*
4. The observed Tx and Rx clock outputs of the transceiver are Phase Locked in the Oscilloscope plot shown in the figure 39. The Lane0_TX_CLK_G (pin D4) and Lane0_Rx_CLK_G (Pin A2) are extracted on J7 header of the Evaluation board for monitoring purpose.

**Figure 41 • Waveform—Tx and Rx Output**
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 the *Microsemi Libero SoC PolarFire webpage*.