DG0799
Demo Guide
PolarFire FPGA 1G Ethernet Loopback Using IOD CDR
## Contents

1 Revision History ................................................................. 1
  1.1 Revision 1.0 ................................................................. 1

2 PolarFire FPGA 1000BASE-T Loopback Using IOD CDR .................... 2
  2.1 Design Requirements ..................................................... 2
  2.2 Prerequisites ............................................................... 3
  2.3 Demo Design ............................................................... 3
    2.3.1 About PF_IOD_CDR .................................................. 4
    2.3.2 Design Implementation ............................................ 5
    2.3.3 IP Configuration ................................................... 6
  2.4 Clocking Structure ....................................................... 20

3 Libero Design Flow ............................................................. 21
  3.1 Synthesize ................................................................. 22
  3.2 Place and Route .......................................................... 22
    3.2.1 PLL, DLL, and Lane Controller Placement .......................... 22
    3.2.2 Resource Utilization ............................................... 23
  3.3 Verify Timing ............................................................. 23
  3.4 Generate FPGA Array Data ............................................. 23
  3.5 Configure Design Initialization Data and Memories ..................... 24
  3.6 Generate Bitstream ....................................................... 26
  3.7 Run PROGRAM Action .................................................... 26

4 Programming the Device Using FlashPro .................................... 28

5 Running the Demo .............................................................. 29

6 Appendix: Installing and Setting Up Cat Karat and Wireshark ............ 35
  6.1 Installing Cat Karat ..................................................... 35
  6.2 Installing and Setting up Wireshark .................................. 40

7 Appendix: References .......................................................... 44
### Figures

<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Figure 1</td>
<td>Hardware Implementation Block Diagram</td>
<td>4</td>
</tr>
<tr>
<td>Figure 2</td>
<td>Top-Level Libero Implementation</td>
<td>5</td>
</tr>
<tr>
<td>Figure 3</td>
<td>PF_IOD_CDR Configurator</td>
<td>6</td>
</tr>
<tr>
<td>Figure 4</td>
<td>CORETSE_0 Configurator</td>
<td>7</td>
</tr>
<tr>
<td>Figure 5</td>
<td>Mi-V Configurator</td>
<td>8</td>
</tr>
<tr>
<td>Figure 6</td>
<td>pf_sram_0 Configurator</td>
<td>9</td>
</tr>
<tr>
<td>Figure 7</td>
<td>PF_CCC_0 Clock Options</td>
<td>10</td>
</tr>
<tr>
<td>Figure 8</td>
<td>PF_CCC_0 Output Clocks</td>
<td>11</td>
</tr>
<tr>
<td>Figure 9</td>
<td>NWC_PLL_0 Input Clock</td>
<td>12</td>
</tr>
<tr>
<td>Figure 10</td>
<td>NWC_PLL_0 Output Clock</td>
<td>13</td>
</tr>
<tr>
<td>Figure 11</td>
<td>NWC_PLL_0 DLL</td>
<td>14</td>
</tr>
<tr>
<td>Figure 12</td>
<td>CoreSPI_0 Configurator</td>
<td>15</td>
</tr>
<tr>
<td>Figure 13</td>
<td>Mi-V Processor Bus Interface Memory Map</td>
<td>16</td>
</tr>
<tr>
<td>Figure 14</td>
<td>CoreAHBLite_0 Configuration</td>
<td>17</td>
</tr>
<tr>
<td>Figure 15</td>
<td>CoreAHBLite_2 Configuration</td>
<td>18</td>
</tr>
<tr>
<td>Figure 16</td>
<td>CoreAPB3 Configuration</td>
<td>19</td>
</tr>
<tr>
<td>Figure 17</td>
<td>Clocking Structure</td>
<td>20</td>
</tr>
<tr>
<td>Figure 18</td>
<td>Libero Design Flow Options</td>
<td>21</td>
</tr>
<tr>
<td>Figure 19</td>
<td>PLL, DLL, and IOD Lane Controller Placement</td>
<td>22</td>
</tr>
<tr>
<td>Figure 20</td>
<td>Configure Design Initialization Data and Memories Option</td>
<td>24</td>
</tr>
<tr>
<td>Figure 21</td>
<td>Fabric RAMs Tab</td>
<td>24</td>
</tr>
<tr>
<td>Figure 22</td>
<td>Fabric RAM Tab Apply Option</td>
<td>25</td>
</tr>
<tr>
<td>Figure 23</td>
<td>Memory Type Selection</td>
<td>25</td>
</tr>
<tr>
<td>Figure 24</td>
<td>Board Setup</td>
<td>27</td>
</tr>
<tr>
<td>Figure 25</td>
<td>Programmer Status After Successful Programming</td>
<td>28</td>
</tr>
<tr>
<td>Figure 26</td>
<td>Cat Karat Packet Builder Window</td>
<td>29</td>
</tr>
<tr>
<td>Figure 27</td>
<td>Host PC Ethernet Network Connection</td>
<td>30</td>
</tr>
<tr>
<td>Figure 28</td>
<td>Interface Selection</td>
<td>30</td>
</tr>
<tr>
<td>Figure 29</td>
<td>Packet Flow and View Settings</td>
<td>31</td>
</tr>
<tr>
<td>Figure 30</td>
<td>Wireshark Main Window</td>
<td>31</td>
</tr>
<tr>
<td>Figure 31</td>
<td>Wireshark Interface Settings</td>
<td>32</td>
</tr>
<tr>
<td>Figure 32</td>
<td>Wireshark - Start a New Live Capture Icon</td>
<td>32</td>
</tr>
<tr>
<td>Figure 33</td>
<td>Wireshark Live Capture</td>
<td>33</td>
</tr>
<tr>
<td>Figure 34</td>
<td>Car Karat - Transmit Packets Icon</td>
<td>33</td>
</tr>
<tr>
<td>Figure 35</td>
<td>Transmitted and Looped Back Packets</td>
<td>34</td>
</tr>
<tr>
<td>Figure 36</td>
<td>Cat Karat Packet Builder Installation Wizard - Welcome Page</td>
<td>35</td>
</tr>
<tr>
<td>Figure 37</td>
<td>Cat Karat License Agreement</td>
<td>36</td>
</tr>
<tr>
<td>Figure 38</td>
<td>Destination Directory Selection</td>
<td>36</td>
</tr>
<tr>
<td>Figure 39</td>
<td>Program Folder Selection</td>
<td>37</td>
</tr>
<tr>
<td>Figure 40</td>
<td>WinPcap Welcome Wizard</td>
<td>37</td>
</tr>
<tr>
<td>Figure 41</td>
<td>WinPcap 4.1.2 Setup Window</td>
<td>38</td>
</tr>
<tr>
<td>Figure 42</td>
<td>WinPcap License Agreement</td>
<td>38</td>
</tr>
<tr>
<td>Figure 43</td>
<td>WinPcap Installation Options</td>
<td>39</td>
</tr>
<tr>
<td>Figure 44</td>
<td>WinPcap 4.1.2 Installation Completed Message</td>
<td>39</td>
</tr>
<tr>
<td>Figure 45</td>
<td>Wireshark Setup Wizard Welcome Page</td>
<td>40</td>
</tr>
<tr>
<td>Figure 46</td>
<td>Wireshark License Agreement</td>
<td>40</td>
</tr>
<tr>
<td>Figure 47</td>
<td>Wireshark Components Selection</td>
<td>41</td>
</tr>
<tr>
<td>Figure 48</td>
<td>Wireshark Additional Tasks Selection</td>
<td>41</td>
</tr>
<tr>
<td>Figure 49</td>
<td>Wireshark Installation Location Selection</td>
<td>42</td>
</tr>
<tr>
<td>Figure 50</td>
<td>Install WinPcap Confirmation Page</td>
<td>42</td>
</tr>
<tr>
<td>Figure 51</td>
<td>Wireshark Installation Completed Message</td>
<td>43</td>
</tr>
<tr>
<td>Figure 52</td>
<td>Wireshark 1.12.4 (64-bit) Setup Finish Page</td>
<td>43</td>
</tr>
</tbody>
</table>
Tables

Table 1 Design Requirements ............................................................. 2
Table 2 I/O Signals ............................................................................ 5
Table 3 Resource Utilization ............................................................... 23
Table 4 Jumper Settings .................................................................... 26
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 1.0

Revision 1.0 is the first publication of this document.
2 PolarFire FPGA 1000BASE-T Loopback Using IOD CDR

Microsemi PolarFire® FPGAs support 1G Ethernet solutions for various networking applications. In PolarFire devices, 10/100/1000 Mbps (1G) Ethernet is implemented using the CoreTSE media access control (MAC) soft IP core. The CoreTSE IP implements a serial gigabit media-independent interface (SGMII) with an Ethernet PHY. This Ethernet interface can be implemented in the FPGA design by using either a transceiver or a GPIO with clock and data recovery (CDR) capability. These features are provided by the PF_XCVR and PF_IOD_CDR IP cores, respectively.

GPjos in PolarFire devices operate at speeds of up to 1.066 Gbps for single-ended standards and 1.25 Gbps for differential standards. Each I/O has an I/O digital (IOD) logic block that supports gearing up of the output data rate and gearing down of the input data rate. The IOD block with CDR circuitry (PF_IOD_CDR IP) deserializes high-speed Ethernet input data and transfers it to the FPGA fabric at lower speeds. It also serializes the lower-speed Ethernet data from the FPGA fabric and transfers to the high-speed Ethernet PHY.

This document describes how to run the 1000BASE-T loopback demo design, which is a reference design created to demonstrate 1000BASE-T Ethernet loopback using GPIO on a PolarFire Evaluation Board. The demo design is built using the CoreTSE, PF_IOD_CDR, and Mi-V soft processor IP cores.

The demo design can be programmed using either of the following options:

- Using the pre-generated .stp file: To program the device using the .stp file provided along with the demo design files, see Programming the Device Using FlashPro, page 28.
- Using Libero® SoC PolarFire: To program the device using Libero SoC PolarFire, see Libero Design Flow, page 21.

To generate and receive Ethernet traffic for this demo, an Ethernet packet generator and a packet analyzer are required on the host PC. For more information about installing these two software applications, see Appendix: Installing and Setting Up Cat Karat and Wireshark, page 35.

A license is required to use the CoreTSE IP core. To request a license, contact soc_marketing@microsemi.com.

2.1 Design Requirements

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

<table>
<thead>
<tr>
<th>Table 1 • Design Requirements</th>
</tr>
</thead>
<tbody>
<tr>
<td>Requirement</td>
</tr>
<tr>
<td><strong>Hardware</strong></td>
</tr>
<tr>
<td>PolarFire Evaluation Kit (MPF300-EVAL-KIT)</td>
</tr>
<tr>
<td>– PolarFire Evaluation Board</td>
</tr>
<tr>
<td>– 12 V/5 A AC power adapter and cord</td>
</tr>
<tr>
<td>– USB 2.0 A to mini-B cable for UART and programming</td>
</tr>
<tr>
<td>RJ45 cable to connect the board with the host PC</td>
</tr>
<tr>
<td>Host PC</td>
</tr>
<tr>
<td><strong>Software</strong></td>
</tr>
<tr>
<td>Cat Karat (Ethernet packet generator)</td>
</tr>
<tr>
<td>Wireshark (network protocol analyzer)</td>
</tr>
<tr>
<td>FlashPro</td>
</tr>
<tr>
<td>Libero® SoC PolarFire Design Suite with a Gold License</td>
</tr>
</tbody>
</table>
2.2 Prerequisites

Before you start:
1. Download the reference design files from http://soc.microsemi.com/download/rsc/?f=mpf_dg0799_liberosocpolarfirev2p0_df.
2. Download and install Libero SoC PolarFire on the host PC from https://www.microsemi.com/products/fpga-soc/design-resources/design-software/libero-soc-polarfire#downloads. The latest versions of ModelSim and Synplify Pro are included in the Libero SoC PolarFire installation package. Make sure you have a Libero Gold license.
3. Download Cat Karat and Wireshark. For more information, see Appendix: Installing and Setting Up Cat Karat and Wireshark, page 35.

2.3 Demo Design

The following is the data flow for the 1000BASE-T loopback demo design:
1. PF_CCC_0 provides the clock to the Mi-V processor and other ABP peripherals.
2. NWC_PLL_0 drives the IOD CDR clocks SGMII_CDR_0:TX_CLK_G and HS_IO_CLK.
3. Mi-V does the following:
   • Executes the application from LSRAM (PF_SRAM IP)
   • Configures the ZL30364 clock generation hardware through the CoreSPI IP to generate reference clocks for the VSC PHY and the IOD CDR fabric module.
   • Configures the CoreTSE registers
   • Configures the VSC PHY through the MDIO interface
   • Enables the auto-negotiation feature in the VSC PHY
   • Sends a request to the CoreTSE IP to negotiate with the on-board VSC8575 PHY.
4. CoreTSE IP allows reads from and writes to the VSC PHY registers through the MDIO interface for control and status.

Table 1 • Design Requirements (continued)

<table>
<thead>
<tr>
<th>Requirement</th>
<th>Version</th>
</tr>
</thead>
<tbody>
<tr>
<td>IP</td>
<td></td>
</tr>
<tr>
<td>PF_INIT_MONITOR</td>
<td>2.0.100</td>
</tr>
<tr>
<td>CORERESET_PF</td>
<td>2.0.112</td>
</tr>
<tr>
<td>PF_CCC</td>
<td>1.0.112</td>
</tr>
<tr>
<td>MIV_RV32IMA_L1_AHB</td>
<td>2.0.100</td>
</tr>
<tr>
<td>CoreJTAGDebug</td>
<td>2.0.100</td>
</tr>
<tr>
<td>COREAHBLite</td>
<td>5.3.101</td>
</tr>
<tr>
<td>COREAHBTOAPB3</td>
<td>3.1.100</td>
</tr>
<tr>
<td>CoreAPB3</td>
<td>4.1.100</td>
</tr>
<tr>
<td>CoreUARTapb</td>
<td>5.6.102</td>
</tr>
<tr>
<td>PF_IOD_CDR</td>
<td>1.0.210</td>
</tr>
<tr>
<td>CoreTSE</td>
<td>3.1.102</td>
</tr>
<tr>
<td>PF_SRAM_AHBL_AXI</td>
<td>1.1.121</td>
</tr>
<tr>
<td>PF_CLK_DIV</td>
<td>1.0.101</td>
</tr>
<tr>
<td>CoreSPI</td>
<td>5.1.104</td>
</tr>
</tbody>
</table>
5. PF_IOD_CDR IP does the following:
   - Interfaces with the on-board VSC8575 PHY.
   - Derives the fabric receive clock (RX_CLK_R) to send data to CoreTSE using the DILL_DELAY_DIFF signal, input data and bank clocks from the NWC_PLL_0.
   - Receives Ethernet data via the RX_P and RX_N input pads, gears down the receive data rate, and deserializes the data.

6. The deserialized data is sent from SGMII_CDR_0:RX_DATA[9:0] to CoreTSE IP: RCG[9:0].
7. The received data is looped back at the CoreTSE IP, and CoreTSE IP:TCG[9:0] is sent to SGMII_CDR_0:TX_DATA[9:0].
8. SGMII_CDR_0 serializes the data, gears up the transmit data rate, and transmits the data to the on-board VSC PHY via the TX_P and TX_N output pads.

The following illustration shows the hardware implementation of the demo design.

**Figure 1 • Hardware Implementation Block Diagram**

2.3.1 About PF_IOD_CDR

The PF_IOD_CDR IP core provides an asynchronous receive and transmit interface that supports up to 1.6 GbE speed for serial data transfers. It supports serial protocols and other similar encoded serial protocols. PF_IOD_CDR uses a 10:1 digital ratio to provide a 10-bit data and clock interface for both transmit and receive modes. The PF_IOD_CDR interface is compatible with the CoreTSE, CoreTSE_AHB, and CoreSGMII IP cores configured in TBI mode. In this demo, the CoreTSE (Non-AMBA) MAC is used in SGMII mode along with the IOD CDR interface to transmit and receive the Ethernet packets.

2.3.1.1 Receive interface

The PF_IOD_CDR receive interface uses four high-speed bank clocks and generates the recovered clock. The lane controller in the IOD includes a clock recovery block. It uses the incoming data and the four bank clocks and generates RX_CLK_R. The CoreTSE logic uses this clock. The serial data is received on an IOA pair and sent to the associated IOD block, which uses a 10:1 digital ratio. The IOD block uses the recovered clock to capture the serial data stream to the core.
The CDR requires four phases of the HS_IO_CLK running at half the frequency of the serial data rate, that is 625 MHz. The lane controller uses the four phases of the HS_IO_CLK and inverts them internally to create eight phases for the CDR. The soft training IP block performs the clock phase selection dynamically on the eight phases to the clock to determine the clock phase that best matches the current data phase. The RX_CLK_R into the fabric includes jitter from the switching of the phase that creates this clock.

2.3.1.2 Transmit Interface
The PF_IOD_CDR transmit interface receives the parallel data (TX_DATA[9:0]), converts it into a serial data stream using the IOD interface, and then transmits it via the I/O ports such as TX_P and TX_N. The transmit interface uses the same PLL as the receive interface. The transmit logic uses the 125-MHz clock from this PLL.

For more information about PF_IOD_CDR, see UG0686: PolarFire FPGA User I/O User Guide.

2.3.2 Design Implementation
The following figure shows the top-level Libero implementation of the demo design.

Figure 2 • Top-Level Libero Implementation

The following table lists the important I/O signals of the design.

Table 2 • I/O Signals

<table>
<thead>
<tr>
<th>Signal</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>RX_P, RX_N</td>
<td>IOD CDR receive signals connected to the VSC PHY receive data signals.</td>
</tr>
<tr>
<td>TX_P, TX_N</td>
<td>IOD CDR transmit signals connected to the VSC PHY transmit data signals.</td>
</tr>
<tr>
<td>REFCLK_N,</td>
<td>125-MHz input clock frequency fed to NWC_PLL_0, received from the on-board ZL30364.</td>
</tr>
<tr>
<td>REFCLK_P</td>
<td>Mi-V reset pin. Asserted by pressing the on-board K22 push-button switch.</td>
</tr>
<tr>
<td>REF_CLK_0</td>
<td>50-MHz input clock frequency received from the on-board 50-MHz oscillator and fed to PF_CCC_0.</td>
</tr>
<tr>
<td>LINK_OK</td>
<td>Link status indicator. Provides the link up or down status received from PHY on CoreTSE_0:ANX_STATE. Mapped to the on-board LED7.</td>
</tr>
</tbody>
</table>
2.3.3 IP Configuration

This section describes the IP blocks and user-defined blocks instantiated in the demo design.

2.3.3.1 SGMII_CDR_0_0

The SGMII_CDR_0_0 block is created when the PF_IOD_CDR IP is instantiated. This IP is configured for 1250 Mbps, as shown in the following figure. The data rate is set to 1250 Mbps because PolarFire GPIOs support data speeds up to 1250 Mbps using differential standards. The Enable BITSLIP port check box is not selected because the CoreTSE IP has a built-in bitslip logic. Selecting the RX enabled and TX enabled check boxes allows IOD blocks to receive and transmit data.

Figure 3 • PF_IOD_CDR Configurator
2.3.3.2 CORETSE_0

The CoreTSE IP is configured in ten-bit interface (TBI) mode to build 8-KB Ethernet packets, as shown in the following figure. The MDIO PHY Address value is used to read and write to the CoreTSE registers.

Figure 4 • CORETSE_0 Configurator

2.3.3.3 PF_INIT_MONITOR_0

The PolarFire Initialization Monitor (PF_INIT_MONITOR component in Libero SoC PolarFire) can be used for issuing a reset signal to the user logic. To ensure a glitch-free reset, the DEVICE_INIT_DONE signal is connected to the CORERESET_PF IP with a lock signal from the PF_CCC macro.

2.3.3.4 CORERESET_PF

The CORERESET_PF block synchronizes the reset of all the blocks to which it is connected when the PolarFire device is powered up.

2.3.3.5 core_jtag_debug_0

The MI-V soft processor debugs the application using the CoreJTAGDebug IP.
2.3.3.6 Mi-V Soft Processor

The Mi-V soft processor executes the application from the LSRAM mapped at 0x80000000. It configures the ZL30364 clock generation hardware through the CoreSPI IP and the VSC PHY through the CoreTSE MDIO interface. It also configures the CoreTSE registers.

The following figure shows the Mi-V soft processor configuration, where the Reset Vector Address is set to 0x8000_000. This is because in the Mi-V processor memory map, the memory range used for the AHB memory interface is 0x8000_0000 to 0x8FFF_FFFC, and the memory range used for the AHB I/O interface is 0x6000_0000 to 0x7FFF_FFFF.

*Figure 5 • Mi-V Configurator*
2.3.3.7 pf_sram_0

The PF_SRAM IP is connected to Mi-V as an AHB slave. The LSRAM blocks are initialized with the user application code from the sNVM.

The processor uses the SRAM memory to execute the application. The following figure shows the LSRAM depth and the interface settings. Under Optimize for, select High Speed if you want the RAMs to be cascaded width wise, and select Low power if you want them to be cascaded depth wise. If the design runs from LSRAM and demands speed of execution, high-speed mode is recommended. In this example, low-power mode is selected. The Fabric Interface type is selected AHBLite because the fabric interfaces with the AHB-based Mi-V processor. The memory depth can be selected based on the application size. This design uses 256 KB RAM (64 KB words).

Figure 6 • pf_sram_0 Configurator
2.3.3.8 **PF_CCC_0**

The PF_CCC_0 (PolarFire Clock Conditioning Circuitry) generates the fabric reference clock that drives the soft processor and the ABP peripherals (CoreTSE and CoreSPI). The PF_CCC_0 IP is configured to produce one output fabric clock using a 50-MHz input.

The following figure shows the PF_CCC_0 input clock configuration.

*Figure 7 • PF_CCC_0 Clock Options*
The following figure shows the PF_CCC_0 output clock configuration. The Mi-V processor supports up to 120 MHz. This design uses an 80-MHz system clock for configuring the APB peripherals.

**Figure 8 • PF_CCC_0 Output Clocks**

2.3.3.9 **NWC_PLL_0**

The NWC_PLL_0 IP block is created when the PF_CCC_0 IP is instantiated. PF_CCC_0 is configured in PLL-DLL cascaded mode to produce five output fabric clocks using a 125-MHz input. The CDR requires four phases of the HS_IO_CLK running at half the frequency of the serial data rate. Therefore, the HSIO clock frequency is selected as 625 MHz with four phases. The lane controller uses the four phases of the HS_IO_CLK and inverts them internally to create eight phases for the CDR. The soft training IP block performs clock phase selection dynamically on the eight phases to select the clock that best matches the current data phase.
The following figure shows the PF_CCC_0 input clock configuration.

**Figure 9 • NWC_PLL_0 Input Clock**
The following figure shows the PF_CCC_0 output clock configuration. A bank clock is generated for 0, 90, 180, and 270 degrees, as shown in Figure 10, page 13. Output clock 2 feeds the DLL. The output clock 0 (dedicated clock of 625 MHz) is sent to the clock divider to generate the 125-MHz frequency required for the IOD TX clock.

*Figure 10* • NWC_PLL_0 Output Clock
The following figure shows the PF_CCC_0 DLL configuration. The settings selected for DLL configuration are:

- **Clock Modes**: Phase Reference Mode.
- **Reference and Phase Shifting**: Output2. This indicates Output clock 2 shown in the preceding figure.

The IOD CDR lane controller uses a process, voltage, and temperature (PVT)-calculated delay code from the DLL while generating the RX clock for the fabric.

**Figure 11** • NWC_PLL_0 DLL

2.3.3.10 **CoreSPI_0**

Mi-V configures the ZL30364 clock generation hardware using the coreSPI_0 IP. The following figure shows the CoreSPI_0 configuration.

- APB Data Width is selected as 32 because the design uses an ABP data width of 32 bit.
- Motorola mode, which is the default selection, is retained because the target SPI slave (VSC PHY) supports this mode.
- Frame size is selected as 16.
- FIFO depth is selected as 32 to store maximum frames (TX and RX) in FIFO.
- Clock rate for the SPI master clock is selected as 7. This clock drives the VSC PHY and becomes system clock/2*(7+1).
- The **Keep SSEL active** check box is enabled to keep the slave peripheral active between back-to-back data transfers.
The following figure shows the CoreSPI configuration.

*Figure 12 • CoreSPI_0 Configurator*
2.3.3.11 Design Memory Map

The following figure shows the Mi-V processor bus interface memory map.

*Figure 13 • Mi-V Processor Bus Interface Memory Map*
CoreAHBLite_0 is configured as shown in the following figure to interface the PF_SRAM for accessing the LSRAM at memory address 0x8000_0000. This configuration is required because the Mi-V processor executes the code from 0x8000_0000.

**Figure 14 • CoreAHBLite_0 Configuration**
CoreAHBLite_2 is configured as shown in the following figure to interface the APB peripherals to the Mi-V processor at 0x6000_0000.

**Figure 15 • CoreAHBLite_2 Configuration**

![CoreAHBLite Configurator](image-url)
CoreAPB3 is configured as shown in the following figure to connect the peripherals CoreTSE, CoreSPI, and CoreUARTapb as slaves.

- APB Master Data bus width: 32 bit
- Number of address bits driven by master: 16. The Mi-V processor addresses slaves using 16 bit, so the final address for these slaves translates to 0x6000_0000, 0x6000_1000, and 0x6000_2000
- Enabled APB Slave Slots: S0, S1, and S2 (for CoreTSE, CoreUARTapb, and CoreSPI, respectively).

**Figure 16 • CoreAPB3 Configuration**

### 2.3.3.12 COREAHBTOAPB3_0

The COREAHBTOAPB3 IP connects to CoreAPB3 using the master interface. This IP retains its default configuration.
2.4 Clocking Structure

In the demo design, there are two clock domains—the on-board 50-MHz oscillator and the on-board ZL30364 clock generation hardware.

- **On-board 50-MHz oscillator**: This oscillator drives the PLL that generates an 80-MHz clock for the Mi-V soft processor and peripherals. The Mi-V soft processor can operate up to 120 MHz. In this design, Mi-V processor runs at 80 MHz.

- **On-board ZL 30364 clock generation hardware**: This hardware generates the reference clocks for the VSC PHY and the IOD CDR fabric module.

The following figure shows the clocking structure of the demo design.

*Figure 17* Clocking Structure
This chapter describes the Libero design flow for running this demo design, which includes:

- Synthesize, page 22
- Place and Route, page 22
- Verify Timing, page 23
- Generate FPGA Array Data, page 23
- Configure Design Initialization Data and Memories, page 24
- Generate Bitstream, page 23
- Configure Design Initialization Data and Memories, page 24
- Generate Bitstream, page 24
- Run PROGRAM Action, page 24

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

**Figure 18 • Libero Design Flow Options**
3.1 Synthesize

To synthesize the design:

   When the synthesis is successful, a green tick mark appears next to Synthesize, as shown in the preceding figure.
2. Right-click Synthesize and select View Report to view the synthesis report and log files in the Reports tab.
   We recommend viewing the iog_cdr_test.srr and iog_cdr_test_compile_netlist.log files for debugging synthesis and compile errors.

3.2 Place and Route

The demo project includes the IO PDC file and the floor planner PDC constraint files. The Place and Route process uses these PDC files to place the I/Os and CCC macros.

To place and route the design:

1. On the Design Flow tab, 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 18, page 21.
2. Right-click Place and Route and select View Report to view the place and route report and the log files in the Reports tab.
   We recommend viewing the iog_cdr_test_place_and_route_constraint_coverage.xml file for place and route constraint coverage.

3.2.1 PLL, DLL, and Lane Controller Placement

PolarFire FPGA I/O pairs are grouped into lanes. Each I/O bank has multiple lanes. Each lane consists of twelve I/Os (six I/O pairs), a lane controller, and a set of high-speed, low-skew clock resources.

All associated I/Os must be placed in one lane. For example, RX_P and RX_N must be placed in the same lane. For more information, see UG0686: PolarFire FPGA User I/O User Guide. The IO Editor shows the placement of the components and I/Os. The PLL and DLL are placed within the reach of the IOD lane controller, as shown in the following figure.

Figure 19 • PLL, DLL, and IOD Lane Controller Placement
3.2.2 **Resource Utilization**

The resource utilization report is written to the `iog_cdr_Top_layout_log.log` file in the **Reports** tab under `iog_cdr_test_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>26149</td>
<td>299544</td>
<td>8.73</td>
</tr>
<tr>
<td>DFF</td>
<td>15984</td>
<td>299544</td>
<td>5.34</td>
</tr>
<tr>
<td>I/O register</td>
<td>0</td>
<td>1536</td>
<td>0.00</td>
</tr>
<tr>
<td>User I/O</td>
<td>21</td>
<td>512</td>
<td>4.10</td>
</tr>
<tr>
<td>– Single-ended I/O</td>
<td>15</td>
<td>512</td>
<td>2.93</td>
</tr>
<tr>
<td>– Differential I/O pairs</td>
<td>3</td>
<td>256</td>
<td>1.17</td>
</tr>
</tbody>
</table>

3.3 **Verify Timing**

To verify timing:

1. On the **Design Flow** tab, double-click **Verify Timing**.
   When the design successfully meets the timing requirements, a green tick mark appears next to **Verify Timing**, as shown in Figure 18, page 21.
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 successfully generated, a green tick mark appears next to **Generate FPGA Array Data**, as shown in Figure 18, page 21.
3.5 Configure Design Initialization Data and Memories

The **Configure Design Initialization Data and Memories** option creates the LSRAM initialization client. When the PolarFire device powers up, the LSRAM memory is initialized with the sNVM contents.

To create the LSRAM initialization client:

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

   ![Configure Design Initialization Data and Memories Option](image)

   *Figure 20* • Configure Design Initialization Data and Memories Option

2. In the **Configure Design Initialization Data and Memories** window, select the **Fabric RAMs** tab, and then select the **pf_sram** file to import the memory information, as shown in the following figure.

   ![Fabric RAMs Tab](image)

   *Figure 21* • Fabric RAMs Tab

3. Import the hex file (`gbe_iod_cdr.hex`) provided with the design files from `...\libero\SoftConsole\gbe_iod_cdr\Release`. The `gbe_iod_cdr.hex` file is a application file generated using SoftConsole v5.2 that configures the ZL clock generation hardware, the CoreTSE registers, and the VSC PHY. The application code is initially stored in sNVM. On device power-up, the system controller copies the code to LSRAM, and the Mi-V processor executes the code from LSRAM. The vector in the first line of the `gbe_iod_cdr.hex` file is deleted to load the file into the pf_lsram block.
4. Click **Apply**, as shown in the following figure.

**Figure 22 • Fabric RAM Tab Apply Option**

![Fabric RAM Tab Apply Option](image)

5. To ensure that the fabric LSRA M contents are stored in sNVM, in the **Design Initialization** tab, under **Memory type for third stage initialization client**, select **sNVM**, as shown in the following figure.

**Figure 23 • Memory Type Selection**

![Memory Type Selection](image)
6. On the Design Flow tab, double-click Generate Design Initialization Data. When the LSRAM initialization client is successfully generated in sNVM, a green tick mark appears next to Generate Design Initialization Data, as shown in Figure 18, page 21. When the device is programmed, the LSRAM block is initialized from the sNVM.

3.6 Generate Bitstream

To generate the bitstream:

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 18, page 21.

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 using the Libero design flow:

Note: If you want to program the board using the .stp file instead, see Programming the Device Using FlashPro, page 28.

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

<table>
<thead>
<tr>
<th>Jumper</th>
<th>Setting</th>
<th>Default Setting</th>
</tr>
</thead>
<tbody>
<tr>
<td>J18, J19, J20, J21, and J22</td>
<td>Close pins 2 and 3 for programming through FTDI.</td>
<td>Closed</td>
</tr>
<tr>
<td>J28</td>
<td>Close pins 1 and 2 for programming through the on-board FlashPro5.</td>
<td>Open</td>
</tr>
<tr>
<td>J4</td>
<td>Close pins 1 and 2 for switching the power manually using SW3.</td>
<td>Closed</td>
</tr>
<tr>
<td>J12</td>
<td>Close pins 3 and 4 for 2.5 V.</td>
<td>Closed</td>
</tr>
</tbody>
</table>

2. Connect the power supply cable to the J9 connector on the board.
3. Connect the USB cable from the host PC to J5 (FTDI port) on the board.
4. Connect the RJ45 cable from the host PC to the J15 connector (RJ45-PORT 0) on the board. The following figure shows the board setup for programming the device.

*Figure 24 • Board Setup*

5. Power up the board using the SW3 slide switch.
6. On the Libero **Design Flow** tab, double-click **Run PROGRAM Action**. When the device is successfully programmed, the LEDs 6, 7, 8, 9, 10, and 11 on the board glow, and a green tick mark appears, as shown in Figure 18, page 21.
7. Right-click **Run Program Action** and select **View Report** to view the corresponding log file in the **Reports** tab. The demo is ready to be run. For information about how to run the demo, see Running the Demo, page 29.
4 Programming the Device Using FlashPro

This chapter describes how to program the PolarFire device with the stp programming file using FlashPro. The stp file is included in the design files folder available at the following location:

```
mpf_dg0799_liberosocpolarfirev2p0_df/programming_file
```

Follow these steps to program the device:

1. Connect the jumpers and set up the board as described in steps 1 to 6 of Run PROGRAM Action, page 26.
2. On the host PC, start the FlashPro software.
3. Click **New Project** to create a new project.
4. In the **New Project** window, do the following, and click OK:
   - Enter a project name.
   - Select **Single device** as the programming mode.
   - Click **Configure Device**.

   When the FlashPro project is successfully created, the message *Created new project 'F:\VOD_C-DR_FP\IOD_CDR_FP.pro'* is displayed in the **Log** pane.

5. In the **Single Device Configuration** pane, click **Browse** to select the ..\programming_file\iod cdr_test.stp file from the design files folder.
6. Click **Program** to program the device.

   The **Programmer List** window in FlashPro, shows details about the programmer including name, type, port, and status.

   When the device is programmed successfully, LEDs 6, 7, 8, 9, 10, and 11 on the board glow, and the **RUN PASSED** status is displayed in FlashPro.

*Figure 25 • Programmer Status After Successful Programming*
Running the Demo

This section describes how to run the 1000BASE-T loopback demo. The procedure involves transmitting packets from the network card of the host PC to the board using Cat Karat and verifying the packets transmitted to and received from the board using Wireshark.

The following procedure assumes that:

- The PolarFire FPGA is programmed with the demo design programming file (.stp). For more information, see Programming the Device Using FlashPro, page 28.
- The Cat Karat and the Wireshark softwares are installed on the host PC. For more information, see Appendix: Installing and Setting Up Cat Karat and Wireshark, page 35.

To run the demo:

1. Ensure that the RJ45 cable is connected from the host PC to the J15 connector on the board.
2. Power up the board using the SW3 slide switch.
3. Confirm that LED 7 is glowing, indicating the Ethernet PHY link is up and running.
4. Open the Cat Karat software from Start menu of the host PC.
   The Cat Karat Packet Builder window opens, as shown in the following figure.

Figure 26 • Cat Karat Packet Builder Window
5. From the control panel of the host PC, note the name of Ethernet Network connection, as shown in the following figure.

**Figure 27 • Host PC Ethernet Network Connection**

6. In the Cat Karat Packet Builder window > Interfaces pane, double-click the Ethernet Network connection noted in the previous step to select that interface, as shown in the following figure.

**Figure 28 • Interface Selection**
7. In **Packet Flow** pane, select the **use RAW** check box, and set **Packets per Burst** to 5 and the **Data Pattern** to 55, as shown in the following figure.

**Figure 29 • Packet Flow and View Settings**

8. Open the Wireshark software from the **Start** menu of the host PC. The following figure shows the Wireshark window.

**Figure 30 • Wireshark Main Window**
9. Double-click **Local Area Connection** and select the interface settings, as shown in the following figure.

**Figure 31 • Wireshark Interface Settings**

![Wireshark Interface Settings](image)

10. Click the **Start a new live capture** icon, as shown in the following figure.

**Figure 32 • Wireshark - Start a New Live Capture Icon**

![Wireshark - Start a New Live Capture Icon](image)
The Wireshark live capture displays the Ethernet packets transferred from the board to the host PC network card, as shown in the following figure.

**Figure 33 • Wireshark Live Capture**

11. In the Cat Karat window, click **Start Transmit** to transmit five packets from the host PC to the board, as shown in the following figure.

**Figure 34 • Car Karat - Transmit Packets Icon**
12. Verify that 10 packets have been captured (transmitted and received), as shown in the following figure.

**Figure 35 • Transmitted and Looped Back Packets**

The preceding figure highlights five packets that were transmitted from the host PC to the board, looped back at the CoreTSE IP, and sent back to the host PC. All packets transmitted from host PC network are looped back in the same way.

13. Select different burst rate and data pattern, and transmit packets to the board.

14. Power down the board, and close the Cat Karat and the Wireshark software.

You have successfully run the demo.
6 Appendix: Installing and Setting Up Cat Karat and Wireshark

This chapter describes how to install the Cat Karat application required for Ethernet packet generation and the Wireshark application required for network protocol analysis.

6.1 Installing Cat Karat

To install Cat Karat:

1. Download the application from the PacketBuilder website.
2. In the download folder, double-click the KaratInstal.exe file.
3. In the User Account Control dialog box, click Yes.

The CatKarat Packet Builder Installation wizard appears, as shown in the following figure.

![Cat Karat Packet Builder Installation Wizard - Welcome Page](image)

4. Click Next to continue.
5. Read the license agreement, select the **I accept the terms of the license agreement** check box, and then click **Next** to continue.

*Figure 37 • Cat Karat License Agreement*

6. In the **Select destination directory** window, retain the default destination directory or click **Browse** to select a different directory, as shown in the following figure.

*Figure 38 • Destination Directory Selection*
7. Click **Next** to continue.
8. In the **Select Program Folder** window, retain the default program folder (Cat Karat Packet Builder), or select a different program folder from the list, as shown in the following figure.

*Figure 39 • Program Folder Selection*

![Program Folder Selection](image)

9. Click **Next** to continue.
10. In the **Summary** window, click **Install**.
11. In the WinPcap 4.1.2 Setup wizard, click **Next**, as shown in the following figure.

*Figure 40 • WinPcap Welcome Wizard*

![WinPcap Welcome Wizard](image)
12. Click **Next** again, as shown in the following figure.

*Figure 41 • WinPcap 4.1.2 Setup Window*

13. Click **I Agree** to accept the WinPcap license agreement.

*Figure 42 • WinPcap License Agreement*
14. On the **Installation options** page, retain the check box selection and click **Install**, as shown in the following figure.

*Figure 43 • WinPcap Installation Options*

15. Click **Finish** to close the **WinPcap 4.1.2 Setup** window.

*Figure 44 • WinPcap 4.1.2 Installation Completed Message*

16. In the **Cat Karat Packet Builder Installation** window, click **Finish** to complete the installation.
6.2 Installing and Setting up Wireshark

To install and set up Wireshark:

1. Download the application from the Wireshark website.
2. In the download folder, double-click the Wireshark-win64.exe file.
3. In the User Account Control dialog box, click Yes.
4. In the Wireshark (64-bit) Setup wizard, click Next to continue, as shown in the following figure.

*Figure 45* • Wireshark Setup Wizard Welcome Page

5. To accept the license agreement, click I Agree, as shown in the following figure.

*Figure 46* • Wireshark License Agreement
6. On the **Choose Components** page, select all the components, and click **Next**.

   *Figure 47 • Wireshark Components Selection*

   ![Wireshark Components Selection](image)

7. On the **Select Additional Tasks** window, select the required shortcuts, and click **Next** to continue.

   *Figure 48 • Wireshark Additional Tasks Selection*

   ![Wireshark Additional Tasks Selection](image)
8. On the **Choose Install Location** page, retain the default installation directory (C drive), or click **Browse** to select the desired directory.

*Figure 49 • Wireshark Installation Location Selection*

![Wireshark Installation Location Selection](image)

9. Click **Next**.

10. When prompted to confirm if you want to install WinPcap, clear the **Install WinPcap 4.1.3** checkbox, and click **Install**. WinPcap was installed during Cat Karat installation, and therefore, does not need to be installed now.

*Figure 50 • Install WinPcap Confirmation Page*

![Install WinPcap Confirmation Page](image)
11. On the **Installation Complete** page, click **Next**.

*Figure 51 • Wireshark Installation Completed Message*

12. Click **Finish** to complete the installation.

*Figure 52 • Wireshark 1.12.4 (64-bit) Setup Finish Page*
This section lists documents that provide more information about and about the IP cores used in the 1000BASE-T loopback demo design and about PolarFire IG Ethernet Solutions in general.

- For more information about PF_IOD_CDR, see *UG0686: PolarFire FPGA User I/O User Guide*.
- For more information about CoreTSE, see *HB0549: CoreTSE Handbook*.
- For more information about PF_CCC, see *UG0684: PolarFire FPGA Clocking Resources User Guide*.
- For more information about PF_SRAM_AXI_AHBL, see *UG0680: PolarFire FPGA Fabric User Guide*.
- For more information about CoreSPI, see *HB0089: CoreSPI Handbook*.
- For more information about PF_INIT_MONITOR, see *UG0725: PolarFire FPGA Device Power-Up and Resets User Guide*.
- For more information about Mi-V soft processor, see *MIV_RV32IMA_L1_AHB_HB.pdf* from the Libero Catalog.
- For general information about PolarFire 1G Ethernet Solutions, see *UG0687: PolarFire FPGA 1G Ethernet Solutions User Guide*.