<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Figure 1</td>
<td>Block Diagram</td>
<td>3</td>
</tr>
<tr>
<td>Figure 2</td>
<td>PCIe Root Port demo design</td>
<td>5</td>
</tr>
<tr>
<td>Figure 3</td>
<td>MIV_SS_0 SmartDesign</td>
<td>6</td>
</tr>
<tr>
<td>Figure 4</td>
<td>PCIe_RP SmartDesign</td>
<td>7</td>
</tr>
<tr>
<td>Figure 5</td>
<td>PCIe_TL_CLK_0 SmartDesign</td>
<td>8</td>
</tr>
<tr>
<td>Figure 6</td>
<td>Clocking Structure</td>
<td>9</td>
</tr>
<tr>
<td>Figure 7</td>
<td>Reset Structure</td>
<td>9</td>
</tr>
<tr>
<td>Figure 8</td>
<td>Libero Design Flow Options</td>
<td>10</td>
</tr>
<tr>
<td>Figure 9</td>
<td>I/O Editor—XCVR View</td>
<td>11</td>
</tr>
<tr>
<td>Figure 10</td>
<td>PF_DDR3_SubSystem_0 Placement</td>
<td>11</td>
</tr>
<tr>
<td>Figure 11</td>
<td>Design and Memory Initialization Window</td>
<td>13</td>
</tr>
<tr>
<td>Figure 12</td>
<td>Fabric RAMs Tab</td>
<td>13</td>
</tr>
<tr>
<td>Figure 13</td>
<td>Edit Fabric RAM Initialization Client Dialog Box</td>
<td>14</td>
</tr>
<tr>
<td>Figure 14</td>
<td>Applying Fabric RAM Content</td>
<td>14</td>
</tr>
<tr>
<td>Figure 15</td>
<td>Third Stage INIT Client</td>
<td>15</td>
</tr>
<tr>
<td>Figure 16</td>
<td>Board Setup—Evaluation Kit</td>
<td>16</td>
</tr>
<tr>
<td>Figure 17</td>
<td>Demo Setup</td>
<td>17</td>
</tr>
<tr>
<td>Figure 18</td>
<td>PCIe Root Port GUI</td>
<td>18</td>
</tr>
<tr>
<td>Figure 19</td>
<td>Endpoint Device Information</td>
<td>19</td>
</tr>
<tr>
<td>Figure 20</td>
<td>Endpoint Config Space-Basic</td>
<td>19</td>
</tr>
<tr>
<td>Figure 21</td>
<td>Endpoint Config Space-Advanced</td>
<td>20</td>
</tr>
<tr>
<td>Figure 22</td>
<td>Single LED Control</td>
<td>20</td>
</tr>
<tr>
<td>Figure 23</td>
<td>LED ON/OFF Walk</td>
<td>21</td>
</tr>
<tr>
<td>Figure 24</td>
<td>DIP SW Status Option</td>
<td>21</td>
</tr>
<tr>
<td>Figure 25</td>
<td>Endpoint DIP SW Status</td>
<td>22</td>
</tr>
<tr>
<td>Figure 26</td>
<td>Enable Interrupt Session Option</td>
<td>22</td>
</tr>
<tr>
<td>Figure 27</td>
<td>Interrupt Counter4</td>
<td>23</td>
</tr>
<tr>
<td>Figure 28</td>
<td>BAR2-LSRAM Read Option</td>
<td>23</td>
</tr>
<tr>
<td>Figure 29</td>
<td>BAR2-LSRAM Write</td>
<td>24</td>
</tr>
<tr>
<td>Figure 30</td>
<td>BAR2-LSRAM Write Successful</td>
<td>24</td>
</tr>
<tr>
<td>Figure 31</td>
<td>Initiating RP LSRAM to EP LSRAM DMA</td>
<td>25</td>
</tr>
<tr>
<td>Figure 32</td>
<td>RP LSRAM to EP LSRAM Throughput</td>
<td>26</td>
</tr>
<tr>
<td>Figure 33</td>
<td>Memory Test Feature</td>
<td>27</td>
</tr>
<tr>
<td>Figure 34</td>
<td>The View Memory Option</td>
<td>27</td>
</tr>
<tr>
<td>Figure 35</td>
<td>Root port DDR4 Memory Content</td>
<td>28</td>
</tr>
<tr>
<td>Figure 36</td>
<td>Endpoint LSRAM Memory Content</td>
<td>28</td>
</tr>
<tr>
<td>Figure 37</td>
<td>FlashPro Express Job Project</td>
<td>30</td>
</tr>
<tr>
<td>Figure 38</td>
<td>New Job Project from FlashPro Express Job</td>
<td>31</td>
</tr>
<tr>
<td>Figure 39</td>
<td>Programming the Device</td>
<td>31</td>
</tr>
<tr>
<td>Figure 40</td>
<td>FlashPro Express—RUN PASSED</td>
<td>32</td>
</tr>
<tr>
<td>Figure 41</td>
<td>PF_DDR4 Configurator—General</td>
<td>33</td>
</tr>
<tr>
<td>Figure 42</td>
<td>PF_DDR4 Configurator—Memory Initialization</td>
<td>34</td>
</tr>
<tr>
<td>Figure 43</td>
<td>PF_DDR4 Configurator—Controller</td>
<td>35</td>
</tr>
</tbody>
</table>
Tables

<table>
<thead>
<tr>
<th>Table</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Table 1</td>
<td>Design Requirements</td>
<td>2</td>
</tr>
<tr>
<td>Table 2</td>
<td>Mi-V and PF_PCIE Address Maps</td>
<td>4</td>
</tr>
<tr>
<td>Table 3</td>
<td>Resource Utilization</td>
<td>12</td>
</tr>
<tr>
<td>Table 4</td>
<td>Jumper Settings</td>
<td>15</td>
</tr>
<tr>
<td>Table 5</td>
<td>Allocated MSIs</td>
<td>23</td>
</tr>
<tr>
<td>Table 6</td>
<td>Throughput Summary</td>
<td>29</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 most current publication.

1.1 Revision 7.0
The following is a summary of the changes made in this revision.
- Updated the document for Libero SoC v2021.2.
- Added support for PERSTn signal generation.
- Replaced Figure 1, page 3 through Figure 7, page 9.
- Updated Table 2, page 4 and Table 3, page 12.
- Updated section Design Implementation, page 5.
- AHBtoAXIAPB subsystem is removed from the MiV subsystem.
- AXI_Interconnect_0 and Core_APB_0 were added in the MiV subsystem.
- Enabled AXI master interface and APB master interface on MiV core.
- Updated the reset structure.

1.2 Revision 6.0
The following is a summary of the changes made in this revision.
- Updated the document for Libero SoC v12.6.
- Removed the references to Libero version numbers.

1.3 Revision 5.0
The document was updated for Libero SoC v12.0.

1.4 Revision 4.0
The document was updated for Libero SoC PolarFire v2.3.

1.5 Revision 3.0
The document was updated for Libero SoC PolarFire v2.2.

1.6 Revision 2.0
The document was updated for Libero SoC PolarFire v2.1.

1.7 Revision 1.0
The first publication of this document.
2 PolarFire FPGA PCIe Root Port

Microsemi PolarFire® FPGAs support fully integrated PCIe Endpoint and Root Port subsystems with optimized embedded controller blocks that use the physical layer interface (PHY) of the transceiver. Each PolarFire device includes two embedded PCIe subsystem (PCIESS) blocks that can be configured either separately, or as a pair, using the PF_PCIE IP configurator in the Libero® SoC software.

The PF_PCIE IP core is compliant with the PCI Express Base Specification, Revision 3.0 with Gen1/2. It implements memory-mapped Advanced Microcontroller Bus Architecture (AMBA) advanced extensible interface 4 (AXI4) access to the PCIe space and the PCIe access to the memory-mapped AXI4 space. For more information, see UG0685: PolarFire FPGA PCI Express User Guide.

This document describes the Root Port capabilities of the PolarFire FPGA PCIe controller using Mi-V soft processor. The PCIe Root Port capabilities like the enumeration of an Endpoint device, low-speed and high-speed data transfers are demonstrated using the PCIe Root Port Demo GUI application.

The demo design includes a Mi-V soft processor, which initiates PCIe control and data plane functions. For more information about the PCIe Root Port design implementation, and the necessary blocks and IP cores instantiated in Libero SoC, see Demo Design, page 3.

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

- Using the job file: To program the device using the job file provided along with the design files, see Setting Up the Demo, page 17.
- Using Libero SoC: To program the device using Libero SoC, see Libero Design Flow, page 10. Use this option when the demo design is modified.

To run the demo, perform the following steps:

- The Root Port demo design must be programmed on a PolarFire Evaluation board.
- The Endpoint demo design must be programmed on another PolarFire Evaluation board.
- Both the boards must be connected using a PCIe Adapter card.

For more information about setting up the PCIe Root Port demo, see Setting Up the Demo, page 17.

2.1 Design Requirements

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

<table>
<thead>
<tr>
<th>Requirement</th>
<th>Version</th>
</tr>
</thead>
<tbody>
<tr>
<td>Operating system</td>
<td>64-bit Windows 7 or 10</td>
</tr>
<tr>
<td>Hardware</td>
<td></td>
</tr>
<tr>
<td>Two PolarFire Evaluation Kits (MPF300TS-FCG1152I)</td>
<td>Rev D or later</td>
</tr>
<tr>
<td>Microsemi PCIe Adapter Card</td>
<td>PCIE-ROOTPORT-AD</td>
</tr>
<tr>
<td>Software</td>
<td></td>
</tr>
<tr>
<td>Libero SoC</td>
<td></td>
</tr>
<tr>
<td>SoftConsole</td>
<td></td>
</tr>
<tr>
<td>ModelSim</td>
<td></td>
</tr>
<tr>
<td>Synplify Pro</td>
<td></td>
</tr>
</tbody>
</table>

**Note:** Refer to the readme.txt file provided in the design files for the software versions used with this reference design.

**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_dg0802_df
3. 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

The latest versions of ModelSim, Synplify Pro, and FTDI drivers are included in the Libero SoC installation package.

2.3 Demo Design

The top-level block diagram of the PCIe Root Port design is shown in the following figure. The PolarFire PCIe Root Port can establish a PCIe link with any PCIe Endpoint or Bridge. The user application enumerates the Endpoint device using the ECAM (enhanced configuration access mechanism) feature. The user application also initiates the AXI transactions from the Root Port. These AXI transactions are converted to PCIe Configuration space or memory transactions (CfgWr/CfgRd/MWr/MRd) to Endpoint.

Figure 1 • Block Diagram
As shown in Figure 1, page 3, the following points describe the data flow in the PCIe Root Port design:

1. The CoreUART block interfaces with the GUI.
2. The Mi-V soft processor reads/writes data to the Core_UART_0 block through the core Core_APB_0.
3. The Mi-V soft processor forwards the corresponding PCIe command to the PF_PCIE_0 block through the PCIe_APB/PCIe_AXI slave interface.
4. The PCIe request and completion TLPs are transmitted and received between the Root Port and the Endpoint through the serial link.
5. Inbound TLPs are reflected as AXI transactions on the AXI_1_Master port of PF_PCIE.
6. The Mi-V soft processor uses the PCIe_AXI bus interface to read the data from AXI_1_SLAVE.
7. The Mi-V soft processor uses the core Core_APB_0 and writes the data to the UART_APB slave interface to forward the data to the GUI.

As shown in Figure 1, page 3, the following points describe the DMA flow from Root Port to Endpoint:

1. The Mi-V soft processor enumerates the Endpoint by accessing the Root Port and the Endpoint configuration space through the SLAVE ATR0 path.

**Note:** ATRs (address translation registers) perform address translation from PCIe address space to the AXI master. For more information about ATRs, see UG0685: PolarFire FPGA PCI Express User Guide.

2. The Mi-V soft processor accesses the Endpoint BAR 0/2 through the SLAVE ATR 0/1 path.
3. The Mi-V soft processor accesses the Root Port LSRAM memory through the SLAVE ATR 3 path.
4. The DMA is performed according to the user selection on the GUI application.

### 2.3.1 Memory and Peripheral Address Map

This section lists the memory and peripheral address map of the Root Port demo design.

The address map of the Mi-V peripherals and main memory are:

- APB/AXI MMIO Interface: 0x60000000 to 0x7FFFFFFF
- AHB MEM Interface: 0x80000000 to 0x8FFFFFFF

The address map of the bus interfaces connecting Mi-V to PF_PCIE is listed in the following table.

<table>
<thead>
<tr>
<th>Bus Interface/Component</th>
<th>Description</th>
<th>Memory Map</th>
</tr>
</thead>
<tbody>
<tr>
<td>PCIe_APB</td>
<td>This bus interface is used to access the PCIe register</td>
<td>0x63000000 to 0x63FFFFFF</td>
</tr>
<tr>
<td>CoreUARTapb</td>
<td>This block establishes a UART interface to connect the Mi-V processor to the external world</td>
<td>0x64000000 to 0x64FFFFFF</td>
</tr>
<tr>
<td>CoreGPIO_C0</td>
<td>This component is used to generate the PERSTn signal for the link partner that is connected to the Root port</td>
<td>0x65000000 to 0x65FFFFFF</td>
</tr>
<tr>
<td>PCIe_AXI</td>
<td>This bus interface is the PCIe AXI slave for EP configuration or BAR space access</td>
<td>0x70000000 to 0x7000FFFF – Configuration space (Mi-V configures through PCIe APB) 0x71000000 to 0x7100FFFF – EP BAR0 space 0x72000000 to 0x7200FFFF – EP BAR2 space 0x73000000 to 0x73FFFFFF – RP AXI Master - LSRAM/DDR4 (Mi-V configures through PCIe APB)</td>
</tr>
<tr>
<td>TCM</td>
<td>This block is the main memory of the Mi-V processor</td>
<td>0x80000000 to 0x80FFFFFF</td>
</tr>
</tbody>
</table>
The PF_PCIE block connects to the DDR4 and LSRAM blocks through the AXI_1_MASTER bus interface. The address maps of DDR4 and LSRAM are 0x10000000 to 0xFFFFFFFF and 0x00000000 to 0x00000FFF respectively.

### 2.3.2 Design Implementation

Figure 2, page 5 shows the Libero SoC software design implementation of the PCIe Root Port demo design.

---

The top-level design includes the following SmartDesign components and memory controller subsystems:

- MIV_SS_0
- PCIe_RP_0
- DDR4
- CoreGPIO_C0_0

**Note:** PERSTn is a fundamental reset signal defined in both PCI Express Base Specification and PCI Express Card Electromechanical Specification. It is a reset signal issued by the Root port via PCIe slots to reset the entire PCIe fabric hierarchy. CoreGPIO_C0_0 is used to generate the PERSTn signal for the link partner that is connected to the root port. The Root port firmware running on the MI-V processor can assert the PERSTn signal through CoreGPIO_C0_0. When the host is power cycled, the PERSTn signal is asserted by the PCIe_INIT_MONITOR_0 until the PCIe controller in the Root port is initialized.
2.3.2.1 Mi-V Subsystem

The sub-blocks of MIV_SS_0 are shown in the following figure.

**Figure 3 • MIV_SS_0 SmartDesign**

![SmartDesign Diagram](image)

2.3.2.1.1 MIV_RV32_C0

The MIV_RV32_C0 (MIV_RV32_C0) is configured with a Reset Vector Address of 0x80000000. After reset, the processor starts executing the instructions from this address. The main memory of the processor address ranges from 0x80000000 to 0x8FFFFFFF.

The AXI_Interconnect_0 and Core_APB_0 block connects the MIV_RV32_C0 block to:

- The PF_PCIE_0 block through PCIe_APB slave interface. The MIV_RV32_C0 block accesses the PCIe control registers through the PCIe_APB slave interface.
- The PF_PCIE_0 block through PCIe_AXI slave interface.
- The Core_UART_0 block through a APB slave interface.
- The CoreGPIO_C0_0 block through the PERSTn_CTRL_APB slave interface of CoreGPIO_C0_0.

These slaves are connected at the following addresses:

- PCIe_APB slave: 0x63000000
- PCIe_AXI: 0x70000000
- UART_APB slave: 0x64000000
- PERSTn_CTRL_APB slave: 0x65000000
2.3.2.2 PCIe Rootport Subsystem

The sub-blocks of PCIe_RP_0 are shown in the following figure.

Figure 4 • PCIe_RP SmartDesign

2.3.2.2.1 PF_PCIE_0

The PF_PCIE_0 IP block is used to configure the PCIe subsystem (PCIESS) as a Root Port (PCIe 1). PCIESS block is configured for x4 lanes, 5 Gbps data rate, and APB interface for PCIe Controller access.

2.3.2.2.2 PCIe_Tx_PLL_0

The PCIe_Tx_PLL_0 (Transmit PLL) is configured for a 100 MHz Reference Clock and a 5000 Mbps Desired Output Bit Clock.

The PolarFire FPGA transceiver is a half-rate architecture that is the internal high-speed path that uses both edges of the clock to keep the clock rates down. Therefore, the clock can run at half of the data rate, thereby consuming less dynamic power. The transceiver in PCIe mode requires a 2500 MHz bit clock.

2.3.2.2.3 PCIe_TL_CLK_0 SmartDesign

The PCIe_TL_CLK SmartDesign implements PCIe TL CLK for PolarFire devices as shown in the following figure. PCIe TL CLK needs to be connected to CLK_125 MHz of Tx PLL. In PolarFire devices, TL CLK is available only after PCIe initialization. The 80 MHz clock is derived from the on-chip 160 MHz oscillator to drive the TL CLK during PCIe initialization. The NGMUX is used to switch this clock to the required CLK_125 MHz after PCIe initialization.
2.3.2.2.4 PCIeM_AXI4Connect_0

The PCIeM_AXI4Connect_0 (CoreAXI4Interconnect) bus is configured for a single master and two slaves and used to connect the PF_PCIE_0 with PCIe_AXI_SRAM_0 and DDR4 for DMA operations.

2.3.2.3 DDR4 Subsystem

The DDR4 subsystem is configured to access the 32-bit DDR4 memory through an AXI4 64-bit interface. The DDR4 memory initialization and timing parameters are configured as per the DDR4 memory on the PolarFire Evaluation kit.

2.4 Clocking Structure

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

**Figure 5** • PCIe_TL_CLK_0 SmartDesign

**Figure 6** • Clocking Structure
2.5 Reset Structure

The following figure shows the reset structure of the PCIe Root port demo design.

*Figure 7 • Reset Structure*

The Reset_AXI_IF_0(CoreReset_PF) block synchronizes "PLL_LOCK" signal of PF_DDR4_SS_0 IP with the DDR4 system clock(200MHz) to generate FABRIC_RESET_N signal, which drives the PCIe_RP_0 and MIV_SS_0 blocks.

The Reset_MIV_0(CoreReset_PF) block synchronizes the external user_resetn (SW6 on the PolarFire Evaluation board) and DEVICE_INIT_DONE(PF_INIT_MONITOR) together with the RISCV system clock (100 MHz) to generate the SYS_RESET_N, which drives the PF_DDR4_SS block.

The CORERESET_PF_C0_0(CoreReset_PF) block synchronizes "CTRLR_READY" signal of PF_DDR4_SS_0 IP with the RISCV system clock (100 MHz) to generate FABRIC_RESET_N signal, which drives the PCIe_RP_0,MIV_SS_0 and CoreGPIO_C0_0 blocks.

For more information about device initialization, see *UG0725: PolarFire FPGA Device Power-Up and Resets User Guide*.

For more information on CoreReset_PF IP core, see CoreReset_PF handbook from the Libero catalog.
This chapter describes the Libero design flow of the demo design. The Libero design flow involves the following steps:

- Synthesize
- Place and route
- Verify Timing
- Configure Design Initialization Data and Memories
- Generate Bitstream
- Run PROGRAM Action

**Note:** To initialize the TCM in PolarFire using the system controller, a local parameter `l_cfg_hard_tcm0_en`, in the `miv_rv32_opsrv_cfg_pkg.v` file should be changed to `1'b1` prior to synthesis. See the 2.7 TCM section in the [MIV_RV32 Handbook](#).

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

**Figure 8 • Libero Design Flow Options**
3.1 Synthesize

To synthesize the design, perform the following steps:

1. From the Design Flow window, double-click Synthesize.
   When the synthesis is successful, a green tick mark appears as shown in Figure 8, page 10.

2. Right-click Synthesize and select View Report to view the synthesis report and log files in the Reports tab.
   We recommend viewing the RP_Demo_Top.srr and the RP_Demo_Top_compile_netlist.log files for debugging synthesis and compile errors.

3.2 Place and Route

To place and route the design, the Transmit PLL (TX_PLL), XCVR_REF_CLK, PF_XCVR TX and RX lane, and the PF_DDR4_SS_0 must be placed using the I/O Editor.

To place and route the design, perform the following steps:

1. From the Constraints Manager window, place the Transmit PLL, XCVR_REF_CLK, and PF_XCVR TX and RX lanes using I/O Editor as shown in the following figure.

   Figure 9 • I/O Editor—XCVR View

2. Place the PF_DDR4_SS_0 at NORTH_NW location as shown in the following figure.

   Figure 10 • PF_DDR3_SubSystem_0 Placement

3. From the Design Flow window, double-click Place and Route.
   When place and route is successful, a green tick mark appears as shown in Figure 8, page 10.

4. Right-click Place and Route and select View Report to view the place and route report and log files in the Reports tab.
   We recommend viewing the RP_Demo_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 `RP_Demo_Top_layout_log.log` file in the Reports tab -> RP_Demo_Top report -> Place and Route. Table 3, page 12 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>30938</td>
<td>299544</td>
<td>10.33</td>
</tr>
<tr>
<td>DFF</td>
<td>23206</td>
<td>299544</td>
<td>7.75</td>
</tr>
<tr>
<td>I/O Register</td>
<td>0</td>
<td>510</td>
<td>0</td>
</tr>
</tbody>
</table>

3.3 Verify Timing

To verify timing, perform the following steps:

1. From the Design Flow window, double-click Verify Timing.
2. When the design successfully meets the timing requirements, a green tick mark appears as shown in Figure 8, page 10.
3. 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

To generate FPGA array data, in the Design Flow window, double-click Generate FPGA Array Data. A green tick mark is displayed after the successful generation of the FPGA array data as shown in Figure 8, page 10.

3.5 Configure Design Initialization Data and Memories

The Configure Design Initialization Data and Memories step generates an TCM initialization client and adds it to sNVM, μPROM, or an external SPI flash, based on the type of non-volatile memory selected. In this tutorial, the TCM is initialized from μPROM.

This process requires the user application executable file (hex file) as input to initialize the TCM after device power-up. The hex file is provided with the design files.

To select the non-volatile memory and generate the initialization client, perform the following steps:

1. On the Design Flow tab, double-click Configure Design Initialization Data and Memories. The Design and Memory Initialization window opens.
2. Under Third stage (uPROM/sNVM/SPI-Flash), select μPROM, as shown in Figure 11, page 13. In the Third Stage pane, select uPROM as the non-volatile memory, and retain the default start address (0x00000000).

Note: The default start address 0x00000000 is retained because there are no other initialization clients specified in μPROM.
3. On the **Fabric RAMs** tab, select
   MiV_S0/MiV_RV32_C0_0/MiV_RV32_C0_0/u_opsrv_0/gen_tcm0.u_opsrv_TCM_0/tcm_ram_macro.u_ram_0 from the list of logical instances, and click **Edit**, as shown in *Figure 12*, page 13. The MiV_S0/MiV_RV32_C0_0/MiV_RV32_C0_0/u_opsrv_0/gen_tcm0.u_opsrv_TCM_0/tcm_ram_macro.u_ram_0 instance is the Mi-V processor’s main memory. The System Controller initializes this instance with the imported client at power-up.

*Figure 12 • Fabric RAMs Tab*

4. In the **Edit Fabric RAM Initialization Client** dialog box, click the **Import** button next to **Content from file**, as shown in the following figure.
Figure 13 • Edit Fabric RAM Initialization Client Dialog Box

1. In the following dialog box, browse the mpf_dg0802_df\Libero_Project\PCIe_RP_Demo.hex file and double-click it.

2. If any changes are applied to the Mi-V application code, rebuild the SoftConsole project in the release mode.

5. In the Edit Fabric RAM Initialization Client window, click **OK**.

6. On the Fabric RAMs tab, click **Apply**, as shown in the following figure.

Figure 14 • Applying Fabric RAM Content

7. The initialization client for
   MIV_SS_0/MIV_RV32_C0_0/MIV_RV32_C0_0/u_opsrv_0/gen_tcm0.u_opsrv_TCM_0/tcm_ram_macro.u_ram_0 instance is generated and stored in µPROM. This step can be verified by viewing the third stage client created in the µPROM tab as shown in the following figure.
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 as shown in Figure 8, page 10.
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. To program the PolarFire device, perform the following steps:

1. Ensure that the following Jumper Settings are set on the board, which will be used as the Root Port device.

<table>
<thead>
<tr>
<th>Jumper</th>
<th>Description</th>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td>J18, J19, J20, J21, and J22</td>
<td>Short pin 2 and 3 for programming the PolarFire FPGA through FTDI</td>
<td>Closed</td>
</tr>
<tr>
<td>J28</td>
<td>Short pin 1 and 2 for programming through the on-board FlashPro5</td>
<td>Open</td>
</tr>
<tr>
<td>J26</td>
<td>Short pin 1 and 2 for programming through the FTDI SPI</td>
<td>Closed</td>
</tr>
<tr>
<td>J4</td>
<td>Short pin 1 and 2 for manual power switching using SW3</td>
<td>Closed</td>
</tr>
<tr>
<td>J12</td>
<td>Short pin 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. Power on the board using the **SW3** slide switch.
5. On the **Libero in the Design Flow** tab, double-click **Run PROGRAM Action**.

When the device is programmed successfully, a green tick mark appears as shown Figure 8, page 10. The device is successfully programmed, see **Setting Up the Demo**, page 17.
Figure 16 • Board Setup—Evaluation Kit
4 Setting Up the Demo

Setting up the demo involves the following steps:

1. Programming the PolarFire devices on the two evaluation boards
2. Connecting the two PolarFire Evaluation boards though the PCIe Adapter card

Throughout this chapter, the two boards are referred using the following labels for simplicity:

- Board A—board running the Root Port design
- Board B—board running the Endpoint design

4.1 Connecting the Two Boards

This section describes how to connect the two boards though the Microsemi PCIe Adapter Card.

To connect the boards, perform the following steps:

1. Ensure that the pins 1 and 2 of the J1 jumper on the PCIe adapter card are closed.
2. Ensure that the pins 1 and 3 of the J2 jumper on the PCIe adapter card are open.
3. Connect CON1 of the adapter card to CON3 (PCIe slot) of Board A.
4. Connect CON2 of the adapter card to CON3 (PCIe slot) of Board B.
5. Connect the USB cable from the Host PC to J5 (FTDI port) on Board A.
6. Connect the USB cable from the Host PC to J5 (FTDI port) on Board B.
7. Connect the power supply cable to the J3 connector of the PCIe adapter card.
8. Power on Board A and B using the SW3 slide switch.
9. Power-up the PCIe adapter card using the SW1 slide switch.

Board A and Board B power-up using the PCIe adapter card. After successfully connecting the two boards, the demo setup looks like the following figure:

Figure 17 • Demo Setup
5 Running the Demo

This chapter describes how to install and use the GUI to run the PCIe Root Port demo. This chapter is divided into the following sections:

- Installing the GUI
- Viewing the Enumeration Data
- Running the Control Plane Commands
- Running the Data Plane Commands

5.1 Installing the GUI

To install the GUI, perform the following steps:

1. Extract the contents of the mpf_dg0802_df.rar file. From the mpf_dg0802_df\GUI_Installer folder, double-click the setup.exe file.
2. Follow the instructions displayed on the installation wizard.

After successful installation, PCIe_Root_Port_GUI appears on the Start menu of the host PC desktop.

5.2 Viewing the Enumeration Data

Before you begin, ensure that:

1. The PolarFire FPGA on one board is programmed with the PCIe Root Port design and the PolarFire FPGA on the other board is programmed with the PCIe Endpoint design.
2. The two boards are connected through a Microsemi PCIe adapter card and powered-up.
3. LED 9, 10, and 11 are glowing on the Root Port board. This indicates that the PCIe link is up. Otherwise, power-cycle the boards again.

To start the GUI and view the enumeration data, perform the following steps:

1. From the task bar, click the Start button and select PCIe_ROOT_PORT_GUI.
2. Click Connect to connect the GUI to the Root Port board as shown in the following figure.

Figure 18 • PCIe Root Port GUI
The GUI starts detecting the UART COM Port of the Root Port device.

3. After successfully connecting to the COM port, the Mi-V soft processor enumerates the PCIe EP device and sends the configuration space data to the GUI.

4. Click Device Info tab to view the Endpoint device information as shown in the following figure.

**Figure 19 • Endpoint Device Information**

5. Click Config Space tab to view the basic Type 0 Configuration Settings of the Endpoint as shown in the following figure.

**Figure 20 • Endpoint Config Space-Basic**
6. Click **Advanced** tab to view the MSI Capabilities of the Endpoint as shown in the following figure.

**Figure 21 • Endpoint Config Space-Advanced**

7. Similarly, click **Power Management Capability** and **PCIe Capability** tabs to view the relevant data.

### 5.3 Running the Control Plane Commands

In this demo, the Root Port initiates the following control plane operations:

- Control Endpoint LEDs
- Read DIP SW Status
- Read MSI count values
- BAR2 Memory read/write commands

#### 5.3.1 Controlling Endpoint LEDs

Root Port can initiate the Endpoint LED glowing and walk through.

To issue LED Commands, perform the following steps:

1. Click **Demo Controls** tab.
2. Select any single LED. For example, select LED3 as shown in the following figure.

**Figure 22 • Single LED Control**

The GUI initiates the LED glow request to the RISC-V processor, which passes this request to the PF_PCIE_0 block. PF_PCIE_0 sends the BAR2 MWr packet to the Endpoint.

As a result, LED6 on the Endpoint board glows.
3. Click **Start LED ON/OFF Walk** button as shown in Figure 23, page 21.

**Figure 23 • LED ON/OFF Walk**

The GUI initiates the LED ON/OFF walk request. As a result, LED ON/OFF is executed from the first to the last LED and in the reverse order.

### 5.3.2 Reading Endpoint DIP SW Status

To read the DIP SW Status, perform the following steps:

1. Click **Enable DIP SW Session** button as shown in the following figure.

**Figure 24 • DIP SW Status Option**
The GUI initiates the DIP switch status read request. As a result, the DIP SW status on the Endpoint board is displayed as shown in Figure 25, page 22. Change the DIP SW positions on the Endpoint board and observe the same in GUI.

**Figure 25** •  Endpoint DIP SW Status

### 5.3.3 Reading MSI Count Values

In the demo, Root Port can read the MSI count values for push-button interrupts on the Endpoint board. To read the MSI count values:

1. Click *Enable Interrupt Session* button as shown in Figure 26, page 22.

**Figure 26** •  Enable Interrupt Session Option
When the Interrupt session is enabled, the GUI sends the Enable Interrupt session request to the RISC-V processor. PF_PCIE_0 receives the number of MSI requested by the Endpoint. In the reference design, the Root Port allocates 4 types of MSI as shown in the following table.

<table>
<thead>
<tr>
<th>MSI Number</th>
<th>Interrupt Type on the Endpoint Board</th>
<th>Mapped Interrupt Counter on the GUI</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>SW10</td>
<td>Interrupt Counter1</td>
</tr>
<tr>
<td>2</td>
<td>SW9</td>
<td>Interrupt Counter2</td>
</tr>
<tr>
<td>3</td>
<td>SW8</td>
<td>Interrupt Counter3</td>
</tr>
<tr>
<td>4</td>
<td>SW7</td>
<td>Interrupt Counter4</td>
</tr>
</tbody>
</table>

2. Press switch and observe interrupt count.

3. Click **Clear Interrupt Count** button to clear all of the Interrupt counters on the GUI.

### 5.3.4 Running BAR2 Memory Read/Write Commands

In the demo, the Root port can initiate BAR2 memory read/write commands for reading/writing to Endpoint LSRAM/DDR3/DDR4.

The **PCIe Read/Write** tab on the GUI is used to initiate these commands. The Endpoint LSRAM/DDR3/DDR4 memory is first read, and then a value can be entered in a specific location to initiate the write command.

To run BAR2 read/write, perform the following steps:

1. Select **BAR2-LSRAM** option and click **Read** button as shown in the following figure.
2. Select any memory location and edit the value of that location. For example: See the following figure.

*Figure 29* • BAR2-LSRAM Write

![Image of BAR2-LSRAM Write](image)

3. The edited memory location turns green and the value entered is written to the Endpoint LSRAM memory location as shown in *Figure 30*, page 24.

*Figure 30* • BAR2-LSRAM Write Successful

![Image of BAR2-LSRAM Write Successful](image)

4. Similarly change any other memory location also.
5. Click **Read** button to check whether the memory locations contain the latest values or not.
6. Similarly, run the BAR2-DDR3 and BAR2-DDR4 memory read/write.
5.4 Running the Data Plane Commands

In the demo, the Root port initiates the Endpoint DMA engines to perform the following data plane commands:

- Running DMA operations
- Running memory test

5.4.1 Running DMA Operations

When the Root Port initiates the DMA operation, the Mi-V soft processor activates the Endpoint DMA registers through BAR0. The Endpoint DMA engines can perform the following DMA operations:

- Root Port LSRAM/DDR4 to Endpoint LSRAM/DDR3/DDR4
- Endpoint LSRAM/DDR3/DDR4 to Root Port LSRAM/DDR4

To run the DMA operations, perform the following steps:

1. Click **DMA Operations** tab as shown in Figure 31, page 25.
2. Do the following:
   - Select the **RP LSRAM -> EP LSRAM** from the drop-down list.
   - Select 64K from the Transfer Size (Bytes) drop-down.
   - Set the **Loop Count** to 20
   - Click **Start transfer**.

*Figure 31* • Initiating RP LSRAM to EP LSRAM DMA
The GUI displays the corresponding throughput details and graph as shown in the following figure.

Figure 32 • RP LSRAM to EP LSRAM Throughput

3. Do the following to initiate another DMA transaction:
   • Select Both RP LSRAM <-> EP LSRAM from the drop-down list.
   • Select 64K from the Transfer Size (Bytes) drop-down.
   • Set the Loop Count to 20.
   • Click Start Transfer.

4. Similarly, select the RP LSRAM to EP DDR3 and RP LSRAM to EP DDR4 from the drop-down and observe the throughputs.

5. Select DDR4 as the Root Port Memory Type and perform DMA operations by selecting the Endpoint destination memory type.

5.4.2 Running Memory Test

The Memory Test tab provides the memory test feature, which is also a DMA operation. The Memory Test tab enables DMA transactions between Root Port and Endpoint memory type (LSRAM, DDR3, and DDR4). This feature provides data pattern options with which the Root Port memory is initialized and then DMA operation is performed.

In memory testing, the user application performs the following sequence of operations:

1. Initializes the Root Port memory with the specified data pattern
2. Performs the DMA from Root Port memory to Endpoint memory
3. Erases the data pattern in the Root Port memory
4. Performs the DMA from Endpoint memory to Root Port memory
5. Compares the data in Root Port memory with the selected data pattern
To run the memory test, perform the following steps:

1. Select the DMA parameters like Transfer Size(Bytes), Pattern Type, Endpoint Memory Type, RootPort Memory Type, EndPoint Offset Address, and RootPort Offset Address as shown in the following figure.

   **Note:** The Root Port slave ATR3 is configured for 1 MB. Hence, the maximum Endpoint offset address is F80000 and the maximum Root Port address is 0x80000.

   **Figure 33 • Memory Test Feature**

2. Click Memory Test.
3. Select option DDR4 from the Root Port memory type and click View Memory as shown in the following figure to read the Root Port DDR4.

   **Figure 34 • The View Memory Option**
4. The GUI displays the data pattern written to the Root port DDR4 as shown in Figure 35, page 28.

Figure 35 • Root port DDR4 Memory Content

5. Select the PCIe Read/Write tab and click Read to view the data pattern written to the Endpoint LSRAM.

Figure 36 • Endpoint LSRAM Memory Content
### 5.5 PolarFire DMA Throughput Summary

The following table lists the throughput values observed during the continuous DMA mode.

**Table 6 • Throughput Summary**

<table>
<thead>
<tr>
<th>DMA Transfer Type</th>
<th>DMA Size</th>
<th>Throughput (MBps)</th>
<th>Throughput Average (MBps)</th>
</tr>
</thead>
<tbody>
<tr>
<td>RP LSRAM to EP LSRAM</td>
<td>512 K</td>
<td>1022</td>
<td>1022</td>
</tr>
<tr>
<td>EP LSRAM to RP LSRAM</td>
<td>512 K</td>
<td>779</td>
<td>779</td>
</tr>
<tr>
<td>RP LSRAM to EP DDR3</td>
<td>512 K</td>
<td>773</td>
<td>773</td>
</tr>
<tr>
<td>EP DDR3 to RP LSRAM</td>
<td>512 K</td>
<td>328</td>
<td>328</td>
</tr>
<tr>
<td>RP LSRAM to EP DDR4</td>
<td>512 K</td>
<td>998</td>
<td>998</td>
</tr>
<tr>
<td>EP DDR4 to RP LSRAM</td>
<td>512 K</td>
<td>391</td>
<td>391</td>
</tr>
<tr>
<td>RP DDR4 to EP LSRAM</td>
<td>512 K</td>
<td>540</td>
<td>540</td>
</tr>
<tr>
<td>EP LSRAM to RP DDR4</td>
<td>512 K</td>
<td>779</td>
<td>779</td>
</tr>
<tr>
<td>RP DDR4 to EP DDR3</td>
<td>512 K</td>
<td>540</td>
<td>540</td>
</tr>
<tr>
<td>EP DDR3 to RP DDR4</td>
<td>512 K</td>
<td>328</td>
<td>328</td>
</tr>
<tr>
<td>RP DDR4 to EP DDR4</td>
<td>512 K</td>
<td>540</td>
<td>540</td>
</tr>
<tr>
<td>EP DDR4 to RP DDR4</td>
<td>512 K</td>
<td>391</td>
<td>391</td>
</tr>
</tbody>
</table>
Appendix 1: Programming the Devices Using FlashPro Express

The Root Port design must be programmed on Board A and the Endpoint design must be programmed on Board B.

To program, perform the following steps:

1. Take Board A and ensure that the Jumper Settings are set as listed in Table 4, page 15.
2. Connect the power supply cable to the J9 connector on Board A.
3. Connect the USB cable from the Host PC to J5 (FTDI port) on Board A.
4. Power-up Board A using the SW3 slide switch.
5. On the host PC, launch the FlashPro Express software.
6. To create a new job, click New or
   in the Project menu, select New Job Project from FlashPro Express Job as shown in the following figure.

Figure 37 • 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_dg0802_df\Programming_Job`.
   - **FlashPro Express job project location**: Click **Browse** and navigate to the location where you want to save the project.

![New Job Project from FlashPro Express Job](image)

**Figure 38 • 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**.

![Programming the Device](image)

**Figure 39 • Programming the Device**
10. Click **RUN**. When the device is programmed successfully, a **RUN PASSED** status is displayed as shown in the following figure. See Running the Demo, page 18 to run the PCIe Root Port demo.

![FlashPro Express—RUN PASSED](image)

11. Close **FlashPro Express** or in the **Project** tab, click **Exit**. Root port design is successfully programmed on Board A.

12. Similarly, program Board B with the Endpoint design. Browse **PCIe_EP_Demo_EvalKit.job** file from **mpf_dg0756_df\Programming_Job** location.
Appendix 2: DDR4 Configuration

The DDR4 subsystem is configured to access the 32-bit DDR4 memory through an AXI4 64-bit interface. The DDR4 memory initialization and timing parameters are configured as per the DDR4 memory on the PolarFire Evaluation kit. The following figure shows the general configuration settings for the DDR4 memory.

Figure 41 • PF_DDR4 Configurator—General
The following figure shows the initialization configuration settings for the DDR4 memory.

*Figure 42 • PF_DDR4 Configurator—Memory Initialization*
The following figure shows the controller configuration settings for the DDR4 memory.

**Figure 43 • PF_DDR4 Configurator—Controller**
TCL scripts are provided in the design files folder under directory TCL_Scripts. If required, the design flow can be reproduced from Design Implementation till generation of job file.

To run the TCL, follow the steps below:

1. Launch the Libero software
2. Select Project > Execute Script....
3. Click Browse and select script.tcl from the downloaded TCL_Scripts directory.
4. Click Run.

After successful execution of TCL script, Libero project is created within TCL_Scripts directory.

For more information about TCL scripts, refer to \mpf_dg0802_df\TCL_Scripts\readme.txt.

Refer to Libero® SoC TCL Command Reference Guide for more details on TCL commands. Contact Technical Support for any queries encountered when running the TCL script.
Appendix 4: References

This section lists documents that provide more information about the PCIe Endpoint and IP cores used in the reference design.

- For more information about PolarFire transceiver blocks, PF_TX_PLL, and PF_XCVR_REF_CLK, see UG0677: PolarFire FPGA Transceiver User Guide.
- For more information about PF_PCIE, see UG0685: PolarFire FPGA PCI Express User Guide.
- For more information about PF_CCC, see UG0684: PolarFire FPGA Clocking Resources User Guide.
- For more information about DDR3 memory, see UG0676: PolarFire FPGA DDR Memory Controller User Guide.
- For more information about Libero, ModelSim, and Synplify, see the Microsemi Libero SoC PolarFire web page.
- For more information about PolarFire FPGA Evaluation Kit, see UG0747: PolarFire FPGA Evaluation Kit User Guide.
- For more information about CoreAHBLite, see CoreAHBLite Handbook.
- For more information about CoreAHBtoAPB3, see CoreAHBtoAPB3 Handbook.
- For more information about CoreUART, see CoreUART Handbook.