DG0841
Demo Guide
PolarFire Burst Mode Receiver
## Contents

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

2 PolarFire FPGA Burst Mode Receiver ........................................ 7
   2.1 Prerequisite ................................................................. 7
   2.2 Design Requirement ................................................. 7
   2.3 Demo Design ............................................................. 8
      2.3.1 Design Implementation .......................................... 9
   2.4 Port Description ....................................................... 13
      2.4.1 Clocking Structure ............................................. 14
   2.5 Simulating the PolarFire BMR Design ......................... 15
      2.5.1 Simulation Flow .............................................. 15

3 Libero Design Flow .............................................................. 18
   3.1 Synthesize ............................................................... 18
   3.2 Place and Route ....................................................... 19
   3.3 Verify Timing .......................................................... 21
   3.4 Generate Bitstream .................................................. 21
   3.5 Run PROGRAM Action .............................................. 22

4 Programming the Device Using FlashPro ............................. 23

5 Running the Demo ............................................................... 25
   5.1 Installing the GUI ..................................................... 25
   5.2 Running the Demo Design ......................................... 25
## 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>7</td>
</tr>
<tr>
<td>Table 2</td>
<td>CDR Modes</td>
<td>11</td>
</tr>
<tr>
<td>Table 3</td>
<td>I/O Signals</td>
<td>13</td>
</tr>
<tr>
<td>Table 4</td>
<td>Clocks Used</td>
<td>14</td>
</tr>
<tr>
<td>Table 5</td>
<td>Resource Utilization</td>
<td>20</td>
</tr>
<tr>
<td>Table 6</td>
<td>Jumper Settings</td>
<td>22</td>
</tr>
</tbody>
</table>
## Figures

| Figure 1 | Demo Design | 8 |
| Figure 2 | BM Transmitter Subsystem—SmartDesign | 9 |
| Figure 3 | BM Upstream Burst Data Format | 9 |
| Figure 4 | BM Receiver —SmartDesign | 10 |
| Figure 5 | UART—SmartDesign | 12 |
| Figure 6 | Clocking Structure—UART | 14 |
| Figure 7 | Use Automatic DO File Option Selected | 15 |
| Figure 8 | Simulate Option in Libero Design Flow | 15 |
| Figure 9 | Testbench and BMR Demo Design Interaction | 16 |
| Figure 10 | Simulation Waveform | 17 |
| Figure 11 | Libero Design Flow Options | 18 |
| Figure 12 | Edit with I/O Editor Option | 19 |
| Figure 13 | I/O Editor Transceiver View | 19 |
| Figure 14 | Component Locations Updated in user.pdc File | 20 |
| Figure 15 | Verify Timing | 21 |
| Figure 16 | Timing Report | 21 |
| Figure 17 | PolarFire Evaluation Board Setup | 22 |
| Figure 18 | FlashPro New Project Window | 23 |
| Figure 19 | Programming File Details in FlashPro | 24 |
| Figure 20 | Programming Passed | 24 |
| Figure 21 | Before Connection | 25 |
| Figure 22 | After Connection | 26 |
| Figure 23 | Packed Received and Checked | 26 |
| Figure 24 | Inject Error | 27 |
| Figure 25 | Error Report | 27 |
| Figure 26 | Clear Error | 28 |
| Figure 27 | No Error Status | 28 |
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

The first publication of this document.
2 PolarFire FPGA Burst Mode Receiver

Microsemi PolarFire FPGA enables high-speed, point-to-multipoint communication technologies such as XGSPON by implementing a fast Burst Mode Receiver (BMR). The Microsemi PolarFire transceiver is uniquely equipped with the Burst Mode receiver, which can handle Burst data with inter-packet gaps. This feature enables the high speed transceivers to stay in lock when there is no data present and resynchronize to the incoming data when the data is present on the receive serial lines.

This document describes how to run a 10Gbps BMR demo design on the PolarFire® Evaluation kit hardware. The reference design is created using the PolarFire high-speed transceiver blocks. It operates in loopback mode by sending the BM transmitter (TX) data to the BM Receiver (RX) through the transceiver lanes. A GUI application is packaged along with the design files to run the demo. The demo uses on-board transceiver loopback traces and hence external high speed cables are not required to run the demo.

2.1 Prerequisite

Before you start:

- Download the demo design .stp file and the GUI installer from the following location: http://soc.microsemi.com/download/rsc/?f=mpf_dg0841_liberosocpolarfirev2p3
  
  Note: For Libero Design source files contact SOC Tech-support.

- Download and install Libero SoC PolarFire v2.3 on the host PC from the following location: https://www.microsemi.com/product-directory/design-resources/3863-libero-soc-polarfire#downloads

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

2.2 Design Requirement

The following table lists the resources required to run the demo.

<table>
<thead>
<tr>
<th>Requirement</th>
<th>Version</th>
</tr>
</thead>
<tbody>
<tr>
<td>Operating System</td>
<td>Windows 7, 8.1, or 10</td>
</tr>
<tr>
<td>Hardware</td>
<td></td>
</tr>
<tr>
<td>PolarFire Evaluation Kit (MPF300-EVAL-KIT):</td>
<td>Rev C or later with MPF300T Production devices¹</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 programming and interfacing with GUI</td>
<td></td>
</tr>
<tr>
<td>Software</td>
<td></td>
</tr>
<tr>
<td>Libero SoC PolarFire</td>
<td>v2.3</td>
</tr>
<tr>
<td>FlashPro</td>
<td>v2.3</td>
</tr>
<tr>
<td>ModelSim</td>
<td>ME 10.5 Pro</td>
</tr>
</tbody>
</table>

1. Note: This demo is not supported on MPF300TES or MPF300XT devices.
2.3 Demo Design

The PolarFire BMR demo design is developed to showcase the various CDR modes of transceiver operating at 10Gbps. In this reference design:

1. BM Transmitter subsystem interfaces with the GUI. The GUI enables/disables the data transfer through BM transmitter packet generator. The frame format of the packet is as shown in Figure 3, page 9.
2. BM Receiver subsystem receives the looped back burst data and passes the same to BM packet checker.
3. BM Packet checker detects the delimiter patterns and aligns the data to check for any sequence, CRC or payload errors.
4. The BM receiver subsystem also controls the CDR lock modes as described in CDR Lock Mode Controller, page 11.
5. GUI interacts with the design through UART subsystem interface to log and display the link and error status of the BMR design.

The following figure shows the top-level block diagram of the BMR demo design.

*Figure 1 • Demo Design*
2.3.1 Design Implementation

The Top-Level design includes the following SmartDesign components.

- BM Transmitter Subsystem, page 9
- BM Receiver Subsystem, page 10
- UART Subsystem, page 12

2.3.1.1 BM Transmitter Subsystem

The BM_TRANSMITTER_SUBSYSTEM generates the Burst data using the inputs provided by the user Interface and converts the parallel data into serial using a PF_XCVR.

The Subsystem has two main components:

- BM Packet Generator
- BM Transmitter(PF_XCVR)

Figure 2 •  BM Transmitter Subsystem—SmartDesign

2.3.1.1.1 BM Packet Generator

The BM packet generator consists of the following sub-modules:

- **Framer**: Frames a packet using the input preamble length, delimiter, burst length, burst sequence number, CRC-8 and payload.
- **Inter packet gap generator**: generates inter-packet gap using the variable inter-packet gap value from GUI
- **PRBS-31 generator**: PRBS-31 pattern is used as payload for the packet
- **CRC-8 calculator**: calculates CRC-8 based on the burst length and burst sequence number

A finite state machine in BM packet generator constructs a packet as described in following figure.

Figure 3 •  BM Upstream Burst Data Format

<table>
<thead>
<tr>
<th>Preamble (64- 8000)</th>
<th>D</th>
<th>S</th>
<th>L</th>
<th>CRC</th>
<th>Payload (100-50000)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Preamble: Configurable [64-8000] bits of alternating 1's and 0's pattern
IPG : Configurable (1-100000) clock cycles
D: 32-bit Delimiter fixed to 32'hA37670C9
S: 8-bit Sequence number, this value increments or each burst
L: 16-bit Sequence length, Input from GUI
CRC: 8-bit crc value computed using the Sequence number and the Burst length
Payload : Configurable (100-50000) words, where one word is 64 bits
When \texttt{start\_traffic\_i} is set to 1, the packet generator finite state machine starts framing a packet using the preamble, delimiter and payload data.

The finite state machine also generates the inter-packet gap using the \texttt{int\_pkt\_gap\_i} input from the GUI which is used to enable/disable the \texttt{TX\_ELEC\_IDLE} signal of the \texttt{PF\_XCVR}. During the IPG the \texttt{PF\_XCVR} sends an electrical idle condition to emulate the output of squelched optics.

BM transmitter packet generator can induce the errors in sequence number generation using the \texttt{inject\_seq\_err\_i} signal. This error injection is provided to demonstrate a logical error from the transmitter to the receiver when the demo is running and passing error free data.

2.3.1.2 BM Transmitter

The PolarFire high-speed transceiver (\texttt{PF\_XCVR}) is a Hard IP block and supports data rates from 500 Mbps to 12.5 Gbps. The transceiver is configured for:

- 10Gbs data rate
- 64-bit fabric interface at 156.25MHz
- PMA mode (no data encoding)
- \texttt{TX\_ELEC\_IDLE} port provided

Transmitter generates electrical idle during IPG using the \texttt{TX\_ELEC\_IDLE} port.

The transmitter transceiver is a unique physical lane from the receiver transceiver.

2.3.1.2 BM Receiver Subsystem

The BM\_RECEIVER\_SUBSYSTEM receives the burst data from \texttt{PF\_XCVR} interface configured in Burst mode receiver mode. It then sends the received parallel data to burst mode packet checker to perform data integrity checks and to cdr lock mode controller to set the cdr lock modes, for more info on CDR lock modes, see CDR Lock Mode Controller, page 11.

The BM\_RECEIVER\_SUBSYSTEM contains three main components.

- BM Receiver (\texttt{PF\_XCVR} configured as Burst mode receiver)
- Burst mode packet checker
- CDR lock mode controller

\textbf{Figure 4 • BM Receiver —SmartDesign}
2.3.1.2.1 BM Packet Checker

The packet checker receives the parallel data from PF_XCVR and detects the delimiter pattern. Packet checker also performs PRBS-31 rx data integrity and crc checks.

The BM Receiver packet checker consists of the following logic blocks:

1. **Pattern detector**: detects the delimiter pattern in the incoming rx_data.
2. **Word aligner**: aligns the rx_data with respect to the detected pattern location.
3. **Data extractor**: Extracts the sequence number, CRC value and payload data and passes to respective checkers.
4. **Internal Sequence number generator**: generates sequence number internally to compare it with the received sequence number.
5. **Internal CRC generator**: generates CRC data using sequence number and burst length to compare it with the received CRC data.
6. **Payload checker**: generates PRBS-31 data to compare it with the received payload.

2.3.1.2.2 CDR Lock Mode Controller

CDR Lock Mode Controller switches the transceiver CDR mode by driving the LANE0_CDR_LOCKMODE[1:0] signal.

The following table describes the various CDR modes.

<table>
<thead>
<tr>
<th>LANE0_CDR_LOCKMODE[1:0]</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>2'b11</td>
<td><strong>Normal mode</strong>: This is the normal CDR behavior for continuous data. The CDR is locking to the received data stream.</td>
</tr>
<tr>
<td>2'b01</td>
<td><strong>High Gain mode</strong>: The CDR gain is set to 4x. This allows the CDR to react faster to phase differences of the incoming data. It is useful to quickly acquire a phase lock, but will increase jitter on the recovered clock. Used during preamble/delimiter detection phase. Once the delimiter has been identified the CDR should be switched to the normal mode for the remainder of the burst. The high gain mode has a minimum time to lock which can be found in the PolarFire datasheet. This is the time from when data is received till the CDR locks to the phase. The high gain mode also has a maximum time. The maximum time is the amount of time the CDR can remain in high gain mode without phase locking to the received data. The CDR will eventually unlock causing the recovered clock to stop if the CDR is held in high gain mode too long without phase locking. Once the CDR unlocks it will take some time to regain lock. The high gain max time and CDR PLL lock time can be found in the PolarFire datasheet.</td>
</tr>
<tr>
<td>2'b10</td>
<td><strong>Lock to Reference mode</strong>: Demo design default at power-up. The CDR locks to the local reference clock, not the incoming receive data. This mode is used during the IPG to keep the CDR locked.</td>
</tr>
</tbody>
</table>

In the demo design, LANE0_CDR_LOCKMODE[1:0] changes its state as follows:

1. When tx_disable_i is asserted the CDR Lock Mode Controller forces the PF_XCVR to switch to Lock to reference mode.
2. When tx_disable_i is de-asserted the CDR Lock Mode Controller forces the PF_XCVR to switch to High Gain mode and remains in High Gain mode till the delimiter is detected in the BM packet checker.
3. When delimiter_det_i signal is asserted the CDR Lock Mode Controller forces the PF_XCVR to switch to Normal mode and remains in Normal mode till the next time tx_disable_i is asserted.
If CDR lock mode controller does not receive delimiter_det_i signal from BM packet checker, then it utilizes a fail-safe feature to stay in High gain mode till the high gain timeout counter reaches its threshold.

**Note:** When CDR lock mode controller is utilizing the high gain timeout, the lock modes will be switching between High gain and Lock to reference to help the CDR PLL retain its lock.

### 2.3.1.2.3 BM Receiver

The PolarFire high-speed transceiver (PF_XCVR) is a Hard IP block and supports data rates from 500 Mbps to 12.5 Gbps. The transceiver is configured for:

- 10Gbps data rate
- CDR using Burst Mode Receiver
- PMA mode (no data decoding)
- 64-bit fabric interface at 156.25MHz

### 2.3.1.3 UART Subsystem

The UART subsystem implements fabric UART logic to interface with GUI, which is used as a user interface for control and display status of the running design. The UART subsystem has three sub modules:

- UART RX interface
- UART TX interface
- CORE UART

The following figure shows the SmartDesign implementation of the UART system.

#### Figure 5 • UART—SmartDesign

### 2.3.1.3.1 UART RX Interface

The UART RX finite state machine receives the write request and write data from CORE_UART and provides the following data to the BM Packet Generator.

- Preamble length
- Payload length
- Inter-packet gap length
- Start/Stop traffic control signal
- Inject sequence error signal
2.3.1.3.2 UART TX Interface

The UART TX interface finite state machine receives the following design status and provides the status to CORE_UART upon receiving read request:

- Preamble enable status
- PRBS/Payload enable status
- Preamble/delimiter check status
- PRBS/Payload check status
- Sequence Error status and CRC error status
- Also read backs the use configured preamble length and interpacket gap values

2.3.1.3.3 CoreUART

CORE UART configured at baud value of 92100 and a 125Mhz clock is provided as reference which is generated from a on-chip 160 MHz RC oscillator.

2.4 Port Description

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

<table>
<thead>
<tr>
<th>Port name</th>
<th>Direction</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>LANE3_RXD_N</td>
<td>IN</td>
<td>BM Receiver XCVR inverted input</td>
</tr>
<tr>
<td>LANE3_RXD_P</td>
<td>IN</td>
<td>BM Receiver XCVR non-inverted input</td>
</tr>
<tr>
<td>LANE2_TXD_N</td>
<td>IN</td>
<td>BM Transmitter XCVR inverted input</td>
</tr>
<tr>
<td>LANE2_TXD_P</td>
<td>IN</td>
<td>BM Transmitter XCVR non-inverted input</td>
</tr>
<tr>
<td>REF_CLK_PAD_N</td>
<td>IN</td>
<td>Inverted reference clock obtained from on-board 125 MHz oscillator</td>
</tr>
<tr>
<td>REF_CLK_PAD_P</td>
<td>IN</td>
<td>Non-Inverted reference clock obtained from on-board 125 MHz oscillator</td>
</tr>
<tr>
<td>RESET_N_I</td>
<td>IN</td>
<td>Reset signal obtained from the SW10 push-button on the board</td>
</tr>
<tr>
<td>UART_RX_IF_I</td>
<td>IN</td>
<td>UART receiver interface</td>
</tr>
<tr>
<td>DEBUG_PIN_A2_O</td>
<td>OUT</td>
<td>Pin A2 of J7 header used for debug</td>
</tr>
<tr>
<td>DEBUG_PIN_A3_O</td>
<td>OUT</td>
<td>Pin A3 of J7 header used for debug</td>
</tr>
<tr>
<td>DEBUG_PIN_C3_O</td>
<td>OUT</td>
<td>Pin C3 of J7 header used for debug</td>
</tr>
<tr>
<td>DEBUG_PIN_C4_O</td>
<td>OUT</td>
<td>Pin C4 of J7 header used for debug</td>
</tr>
<tr>
<td>DEBUG_PIN_D3_O</td>
<td>OUT</td>
<td>Pin D3 of J7 header used for debug</td>
</tr>
<tr>
<td>DEBUG_PIN_D4_O</td>
<td>OUT</td>
<td>Pin D4 of J7 header used for debug</td>
</tr>
<tr>
<td>LANE3_TXD_N</td>
<td>OUT</td>
<td>BM Transmitter XCVR Inverted output (Not Used)</td>
</tr>
<tr>
<td>LANE3_TXD_P</td>
<td>OUT</td>
<td>BM Transmitter XCVR Non-Inverted output (Not Used)</td>
</tr>
<tr>
<td>LANE2_RXD_N</td>
<td>OUT</td>
<td>BM Receiver XCVR Inverted output (Not Used)</td>
</tr>
<tr>
<td>LANE2_RXD_P</td>
<td>OUT</td>
<td>BM Receiver XCVR Non-Inverted output (Not Used)</td>
</tr>
<tr>
<td>UART_TX_IF_O</td>
<td>OUT</td>
<td>UART transmitter Interface</td>
</tr>
</tbody>
</table>
2.4.1 Clocking Structure

In the reference design there are three clock domains RX_CLK (156.25 Mhz) TX_CLK(156.25 Mhz) and UART_SYS_CLK(125 Mhz).

Figure 6 • Clocking Structure—UART

The following tables describes the clocks used in the demo design.

Table 4 • Clocks Used

<table>
<thead>
<tr>
<th>Clock Name</th>
<th>Source</th>
<th>Frequency</th>
</tr>
</thead>
<tbody>
<tr>
<td>TX_CLK_R</td>
<td>Transceiver TX PLL clock (Lane2)</td>
<td>156.25 MHz</td>
</tr>
<tr>
<td>RX_CLK_R</td>
<td>Transceiver RX recovered clock (Lane2)</td>
<td></td>
</tr>
<tr>
<td>TX_CLK_R</td>
<td>Transceiver TX PLL clock (Lane3)</td>
<td>156.25 MHz</td>
</tr>
<tr>
<td>RX_CLK_R</td>
<td>Transceiver RX recovered clock (Lane3)</td>
<td></td>
</tr>
<tr>
<td>UART</td>
<td>Sys Clock From UART_REFCLKGEN_0</td>
<td>125 MHz</td>
</tr>
</tbody>
</table>
2.5 Simulating the PolarFire BMR Design

2.5.1 Simulation Flow

The design can be simulated using ModelSim ME 10.5C Pro provided with the Libero SoC PolarFire installation. The following sections describe the simulation flow.

2.5.1.1 Initiating Simulation with ModelSim

To simulate the design using ModelSim:

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.

   ![Use Automatic DO File Option Selected](image1)

2. Open Waveforms to ensure the Include DO File check box is selected.

3. Open Simulate under Verify Pre-Synthesized Design in the Design Flow tab of Libero SoC PolarFire, as shown in following figure.

   ![Simulate Option in Libero Design Flow](image2)

When simulation is initialized, ModelSim compiles all the design source files, runs the simulation and configures the waveform viewer to show simulation signals. The reference design can be simulated from the Libero SoC PolarFire.
2.5.1.2 Simulation Results

When the simulation is initiated, ModelSim compiles all the design source files, runs the simulation, and launches the waveform viewer to show the simulation signals.

At 0 ns, the testbench drives the 125-MHz system clock to the DUT. The following figure shows the interaction between the testbench and the design.

Figure 9 • Testbench and BMR Demo Design Interaction

Steps to Simulate the PolarFire BMR demo design.

1. At '0' ns the BM Transmitter packet generator module starts generating the packets as the default value of start_traffic_i is '1' after reset.
2. At 16000 ns the Testbench sets the inject_seq_err_i signal to '1' to observe the sequence and CRC related errors in BM packet checker module (Observe the rx_crc_err_o and rx_seq_err_o in the waveform).
3. At 32000 ns the Testbench sets the start_traffic_i signal to '0' to show that BM transmitter drives electrical idle when there is no traffic (observe the LANE0_TXD_N and LANE0_TXD_P in the waveform).
4. At 48000ns the Burst length is changed from 500 words to 2000 words and the change can be clearly observed on the highlighted LANE0_TXD_N/P and LANE0_RXD_N/P lines in the waveform.
5. At 56000ns the Inter packet gap is changed from 96 words to 256 words and the change can be clearly observed on the highlighted Electrical IDLE on LANE0_RXD_N/P in the waveform.
6. The simulation finishes at 80000 ns.
The following figure shows the simulation results of the reference design.

*Figure 10* • Simulation Waveform
3 Libero Design Flow

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

- Synthesize, page 18
- Place and Route, page 19
- Verify Timing, page 21
- Generate Bitstream, page 21
- Run PROGRAM Action, page 22

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

Figure 11 • Libero Design Flow Options

3.1 Synthesize

To synthesize this design:

1. Open Synthesize from the Design Flow tab.
   When the synthesis is successful, a green tick mark appears 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.
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.

2. On the I/O Attributes tab, click Edit with I/O Editor, as shown in the following figure.

*Figure 12 • Edit with I/O Editor Option*

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

*Figure 13 • 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.
4. On the Design Flow tab, open Place and Route.
When place and route is successful, a green tick mark appears next to Place and Route, as shown in Figure 11, page 18.
5. Right-click Place and Route and select View Report to view the place and route report and log files in the Reports tab. View the PF_10GBMR_place_and_route_constraint_coverage.xml file for place and route constraint coverage.

The resource utilization report is written to the PF_10GBMR_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>4769</td>
<td>299544</td>
<td>1.59</td>
</tr>
<tr>
<td>DFF</td>
<td>1446</td>
<td>299544</td>
<td>0.48</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>5156</td>
<td>299544</td>
<td>1.72</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 the following figure.

*Figure 15 • Verify Timing*

Right-click Verify Timing and select View Report to view the verify timing report and log files in the Reports tab.

*Figure 16 • Timing Report*

3.4 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 11, page 18.

Right-click Generate Bitstream and select View Report to view the corresponding log file in the Reports tab.
3.5 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>
</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 17 • PolarFire Evaluation Board Setup*
4 Programming the Device Using FlashPro

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_dg0841_liberosocpolarfirerev2p3\STPL\PF_10GBMR.stp
```

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 22.
2. On the host PC, start the FlashPro software.
3. Click New Project, and in the New Project window, enter the project name.
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 is enabled.
7. Click Configure Device.
8. In the Programming File section, click Browse, navigate to the location where the PF_10GBMR.stp file is located, and select the file. Details of the selected file are displayed, as shown in the following figure.

*Figure 19 • 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 20 • Programming Passed*

11. Power cycle the board.
5 Running the Demo

This chapter describes how to use the BMR Demo GUI to run the PolarFire BMR demo on the PolarFire Evaluation Board.

5.1 Installing the GUI

To run the demo, you must first install the BMR Demo GUI. To install the BMR Demo GUI:

1. Extract the contents of the `mpf_dg0841_liberoscopolarfirev2p3.rar` file.
2. From the GUI folder of the extracted RAR file, double-click the `setup.exe` file.
3. Follow the instructions displayed by the installation wizard to complete the installation.

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

5.2 Running the Demo Design

Follow these steps to run the BMR demo.

1. The following figure shows the GUI before connecting to the design.

*Figure 21 • Before Connection*
2. Connect the GUI to the design. The following figure shows the GUI after connecting to the design.

*Figure 22 • After Connection*

3. Click the **START** button to start the traffic from BM transmitter. The **PASS** indicates that the number of packets generated matches with the number of packet received and checked, as shown in the following figure.

*Note:* Hover the mouse over the parameters (Preamble, Inter Packet Gap and Payload Length) to see the range of the values.

*Figure 23 • Packed Received and Checked*
4. Inject the sequence number error by selecting the **Inject Sequence Error** check box in the GUI and then click **Set** as shown in the following figure.

*Figure 24 • Inject Error*

![Inject Error](image)

5. The GUI reports the sequence and CRC errors as shown in the following figure.

*Figure 25 • Error Report*

![Error Report](image)
6. Clear the sequence number error by de-selecting the **Inject Sequence Error** check box in the GUI and then click **Set** as shown in the following figure.

*Figure 26 • Clear Error*

![Clear Error](image)

7. Status is displayed in the GUI, after clearing sequence and CRC error as shown in the following figure.

*Figure 27 • No Error Status*

![No Error Status](image)