

**DG0843**  
**Demo Guide**  
**PolarFire FPGA CPRI**



---

a  **MICROCHIP** company



a  MICROCHIP company

**Microsemi Headquarters**

One Enterprise, Aliso Viejo,  
CA 92656 USA

Within the USA: +1 (800) 713-4113

Outside the USA: +1 (949) 380-6100

Sales: +1 (949) 380-6136

Fax: +1 (949) 215-4996

Email: [sales.support@microsemi.com](mailto:sales.support@microsemi.com)

[www.microsemi.com](http://www.microsemi.com)

©2021 Microsemi, a wholly owned subsidiary of Microchip Technology Inc. All rights reserved. Microsemi and the Microsemi logo are registered trademarks of Microsemi Corporation. All other trademarks and service marks are the property of their respective owners.

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

**About Microsemi**

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

# Contents

---

1	Revision History	1
1.1	Revision 4.0	1
1.2	Revision 3.0	1
1.3	Revision 2.0	1
1.4	Revision 1.0	1
2	Common Public Radio Interface Demo	2
2.1	Design Requirements	3
2.2	Prerequisites	3
2.3	Demo Design	3
2.3.1	Design Implementation	5
2.4	Port Description	10
2.5	Register Configuration	11
2.6	Clocking Structure	12
2.7	Reset Structure	13
3	Libero Design Flow	15
3.1	Synthesis	16
3.2	Place and Route	17
3.3	Verify Timing	18
3.4	Design and Memory Initialization	18
3.5	Generate Bitstream	20
3.6	Run Program Action	21
4	Running the Demo	22
5	Appendix 1: Programming the Device Using FlashPro Express	25
6	Appendix 2: Running the TCL Script	28
7	Appendix 3: References	29

# Figures

---

Figure 1	Radio Base Station System	2
Figure 2	CPRI Demo Design Block Diagram	4
Figure 3	CPRI Master Subsystem SmartDesign	5
Figure 4	Ethernet Data Pattern	6
Figure 5	VSS Data Pattern	6
Figure 6	Antenna Carrier Data Pattern	6
Figure 7	IQ Data Pattern	6
Figure 8	CPRI Slave System	7
Figure 9	Transceiver Subsystem	8
Figure 10	Mi-V Subsystem	8
Figure 11	Test Interface SmartDesign	9
Figure 12	Clocking Structure	12
Figure 13	Reset Structure	14
Figure 14	Libero Design Flow Options	15
Figure 15	I/O Editor Option—XCVR View	17
Figure 16	Place and Route	17
Figure 17	Verify Timing	18
Figure 18	Design and Memory Initialization Window	19
Figure 19	Fabric RAM	19
Figure 20	Edit Fabric RAM Initialization Client	20
Figure 21	Design Initialization	20
Figure 22	PolarFire Evaluation Board Setup	21
Figure 23	Selecting COM Port and Connecting	22
Figure 24	Master Checker Status—Pass	23
Figure 25	Payload Error Enabled—Master	24
Figure 26	Payload Error Disabled—Master	24
Figure 27	FlashPro Express Job Project	25
Figure 28	New Job Project from FlashPro Express Job	26
Figure 29	Programming the Device	26
Figure 30	FlashPro Express—RUN PASSED	27

# Tables

---

Table 1	Design Requirements .....	3
Table 2	System Memory Map .....	9
Table 3	Port Description .....	10
Table 4	Register Configuration .....	11
Table 5	Clocks .....	12
Table 6	Resource Utilization—Synthesis .....	16
Table 7	Resource Utilization—Place and Route .....	17
Table 8	PolarFire Evaluation Board Jumper Settings .....	21

# 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 4.0

Added Appendix 2: Running the TCL Script, page 28.

## 1.2 Revision 3.0

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

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

## 1.3 Revision 2.0

Updated the document for Libero SoC v12.0.

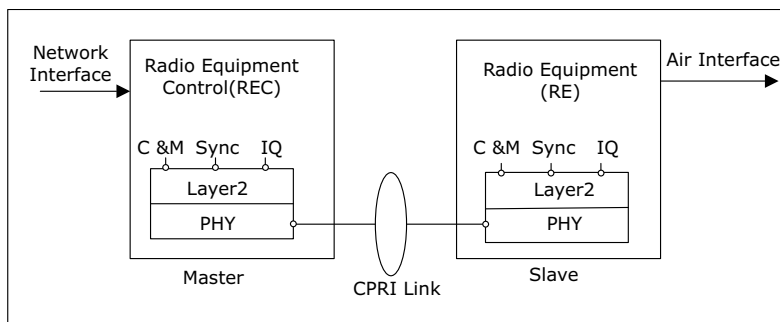
## 1.4 Revision 1.0

The first publication of this document.

## 2 Common Public Radio Interface Demo

The Common Public Radio Interface (CPRI) is an industry standard which defines the publicly available specification for the key internal interface of radio base stations between the Radio Equipment Control (REC) and the Radio Equipment (RE) as shown in following Figure 1, page 2. Microsemi provides the CPRI slave IP core that implements the transmitter and receiver interfaces of the CPRI standard. These IP cores are easy to integrate with CPRI-based data converters to develop high-bandwidth applications. Microsemi CPRI IP supports from line rate1 (614.4 Mbps) to line rate 7 (9830.4 Mbps). The transceiver supports link rate from 250 Mbps to 12.5 Gbps per lane. The transceiver supports up to line rate 9 for CPRI protocol. The transceiver (PF\_XCVR) module integrates several functional blocks to support multiple high-speed serial protocols within the FPGA.

**Figure 1 • Radio Base Station System**



This demo design is created using the PolarFire high-speed transceiver blocks and CPRI Slave IP core. The design operates in the loopback mode by sending the CPRI Master data to the CPRI Slave IP core through the transceiver lanes, which are looped back on the board. This loopback setup facilitates a standalone CPRI interface demo that does not require CPRI testers and other devices to validate the design.

The demo design shows CPRI loopback using transceiver on the Evaluation board. The CPRI Slave IP is configured at:

- Line Rate 5: 4.9152 Gbps
- Number of Antenna Carriers: 4
- The Transceiver is configured in 8b10b mode running at 4.9152 Gbps data rate.
- 32-bit PCS fabric interface using a 122.88 MHz reference clock.

## 2.1 Design Requirements

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

**Table 1 • Design Requirements**

Requirement	Version
Operating system	Windows 7, 8.1, 10
<b>Hardware</b>	
PolarFire Evaluation Kit (MPF300-EVAL-KIT)	Rev D or later
<b>Software</b>	
Libero SoC	<b>Note:</b> Refer to the <code>readme.txt</code> file provided in the design files for the software versions used with this reference design.
FlashPro Express	

**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:

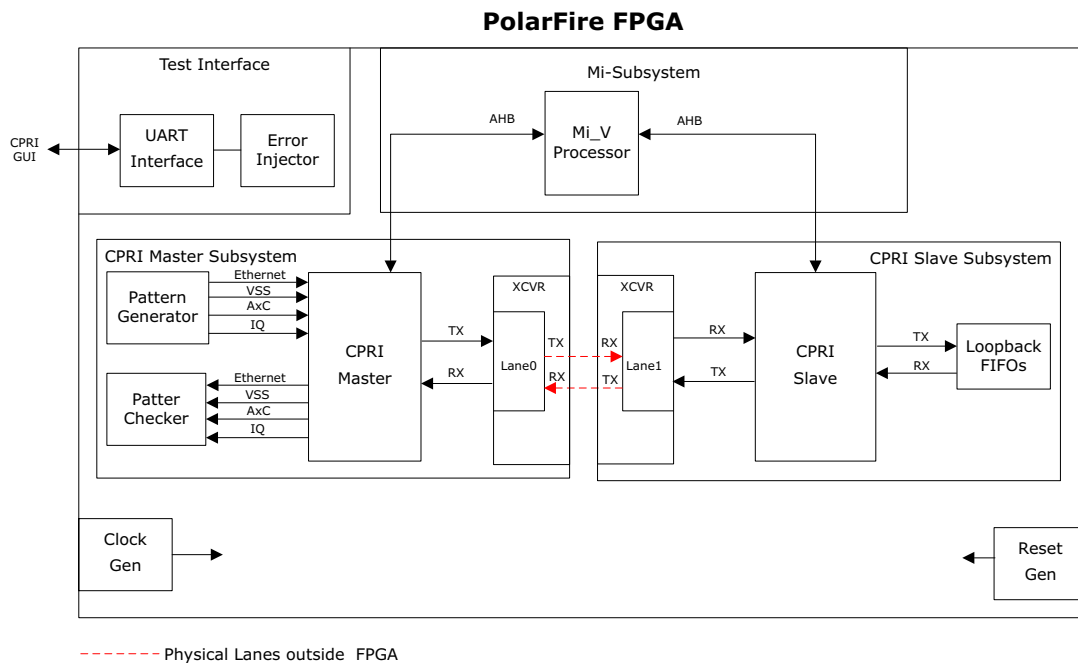
- For demo design files download link:  
[http://soc.microsemi.com/download/rsc/?f=mpf\\_dg0843\\_df](http://soc.microsemi.com/download/rsc/?f=mpf_dg0843_df)
- Download and install Libero SoC (as indicated in the website for this design) on the host PC from the following location:  
<https://www.microsemi.com/product-directory/design-resources/1750-libero-soc#downloads>
- Download and install the SoftConsole on the host PC from the following location:  
<https://www.microsemi.com/product-directory/design-tools/4879-softconsole#downloads>

## 2.3 Demo Design

The PolarFire CPRI Loopback demo design is developed for demonstrating CPRI IP in Slave mode. The pattern generator in this demo design generates CPRI protocol data such as IQ data, VSS data, Ethernet data, and Antenna carrier control data which is provided to the CPRI master module. The CPRI master module frames the data according to the CPRI protocol. The generated frames from CPRI master are loopbacked to CPRI slave IP using the XCVR. The CPRI slave IP unpacks the incoming data into IQ data, VSS data, Ethernet data, and Antenna carrier control data. This data is then loopbacked from CPRI slave IP to CPRI master module, therefore demonstrates Full duplex transmission of CPRI IP in Slave mode.

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

**Figure 2 • CPRI Demo Design Block Diagram**



The following steps describe the data flow in the demo design:

1. The Mi\_V\_module configures the registers of the CPRI Master module and CPRI slave IP blocks.
2. The reference design uses a transceiver interface (PF\_XCVR\_ERM) configured in 8b10b mode running at 4.9152 Gbps data rate, 32-bit PCS fabric interface, and 122.88 MHz reference clock.
3. The CPRI master module and CPRI IP are configured at Line Rate 5: 4.9152 Gbps with 4 Antenna Carriers.
4. The CPRI master module receives the IQ data from the TX IQ Data Generator and control information such as Ethernet data, Vendor specific data and antenna carrier control from the respective pattern generators.
5. CPRI master module then frames the incoming data and transmits a 32-bit CPRI frame to the transceiver.
6. The differential serial data of TX and RX is looped back using an on-board loopback.
7. The CPRI slave IP receives the incoming frames and segregates the incoming data according to the CPRI protocol.
8. The segregated IQ data, Vendor specific data, Ethernet data, and Antenna carrier control information are written to the respective loop back FIFO's.
9. CPRI slave IP then reads the data from loopback FIFO's frames the data and transmits the a 32-bit CPRI frame to CPRI master module through the transceiver.
10. The CPRI master module receives the incoming frames, segregates and sends the incoming data to respective pattern checkers.
11. Incoming control information is compared with the control data for Vendor specific, Fast Ethernet, IQ data and Antenna carrier control data. When the data is matched, the respective lock signals are asserted.
12. The IQ data and control information such as Ethernet, Vendor specific and antenna carrier control lock signals from CPRI Slave IP and Master module along with transceiver RX\_Valid and RX\_ready for both the lanes are sent to the UART\_interface block.
13. The UART\_interface block forwards these status signal and locks information on its TX interface to the GUI for display.

## 2.3.1 Design Implementation

Each CPRI link connects two ports that have asymmetrical functions and roles: a master and a slave. The CPRI Demo design shows the CPRI point to point link between CPRI REC Master and CPRI RE Slave.

The Top-Level design includes the following SmartDesign components:

- CPRI Master Subsystem, page 5
- CPRI Slave Subsystem, page 7
- Transceiver Subsystem, page 7
- Mi-V Subsystem, page 8
- Test Interface, page 9

### 2.3.1.1 CPRI Master Subsystem

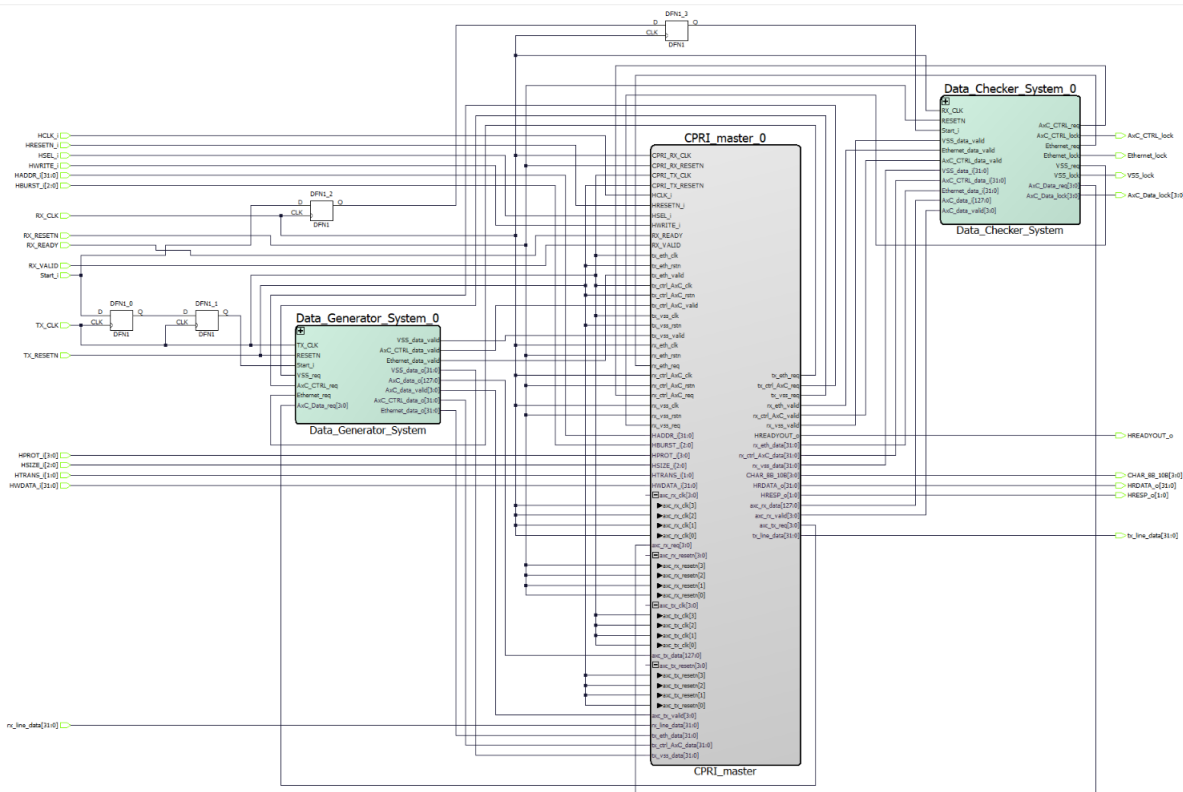
This sections gives the brief description about the CPRI REC Master.

The CPRI REC Master subsystem consists of the following modules.

- CPRI Master
- Pattern data generator
- Pattern checker modules

The following figure shows the implementation of CPRI master subsystem SmartDesign.

Figure 3 • CPRI Master Subsystem SmartDesign



### 2.3.1.1.1 CPRI Master

When the CPRI master is configured, it receives IQ data, Vendor specific data, Antenna carrier control data and Ethernet data from the respective pattern generator blocks. It then frames incoming data into the CPRI frame format at the rate of 4.9152 Gbps and transmits it to the serial lines through the XCVR interface.

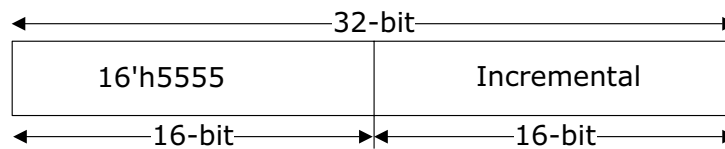
On the receive side, the CPRI master receives the data from the serial lines through the XCVR interface in the CPRI frame format, it then unpacks the data and sends the received IQ, VSS, and Ethernet data to the respective pattern checker block. For more information about register configuration, see [Register Configuration](#), page 11.

### 2.3.1.1.2 Pattern Data Generator

The pattern generator generates the following data:

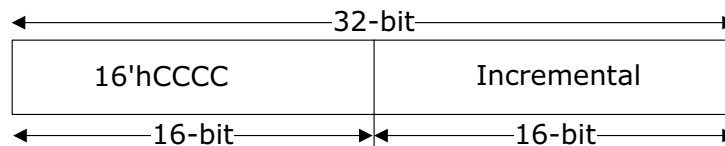
- 32-bit Ethernet data. The MSB 16-bit are constant and LSB 16-bits are incremental data.

**Figure 4 • Ethernet Data Pattern**



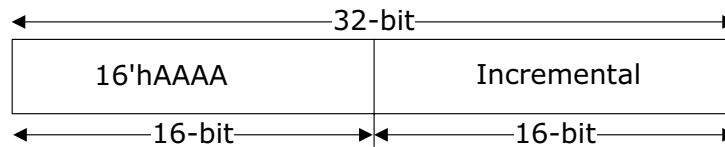
- 32-bit VSS data. The MSB 16-bit are constant and the LSB 16-bits are incremental data.

**Figure 5 • VSS Data Pattern**



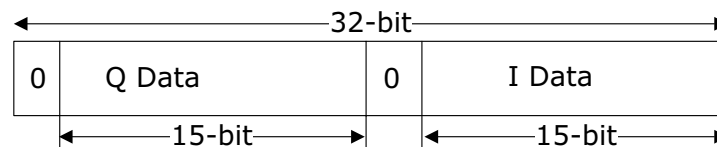
- 32-bit Antenna Carrier Control data. The MSB 16-bit are constant and the LSB 16-bits are incremental data.

**Figure 6 • Antenna Carrier Data Pattern**



- In this demo four antenna carriers are used, each antenna generates 32-bit data. The IQ data samples of each Antenna carrier control are fixed at 15-bits. The user-interface width is fixed at 32-bit, with zero appending on MSB. 15-bit IQ data samples are bit-stuffed in the 32-bit user interface for each antenna carrier.

**Figure 7 • IQ Data Pattern**



### 2.3.1.1.3 Pattern Checker

The pattern checker has individual pattern checker modules for VSS, Ethernet, Antenna carrier control, and IQ data, which checks for incremental, data and if the received data is matched, then the respective lock signals are asserted.





The address map of Mi-V processor is described in the following table.

**Table 2 • System Memory Map**

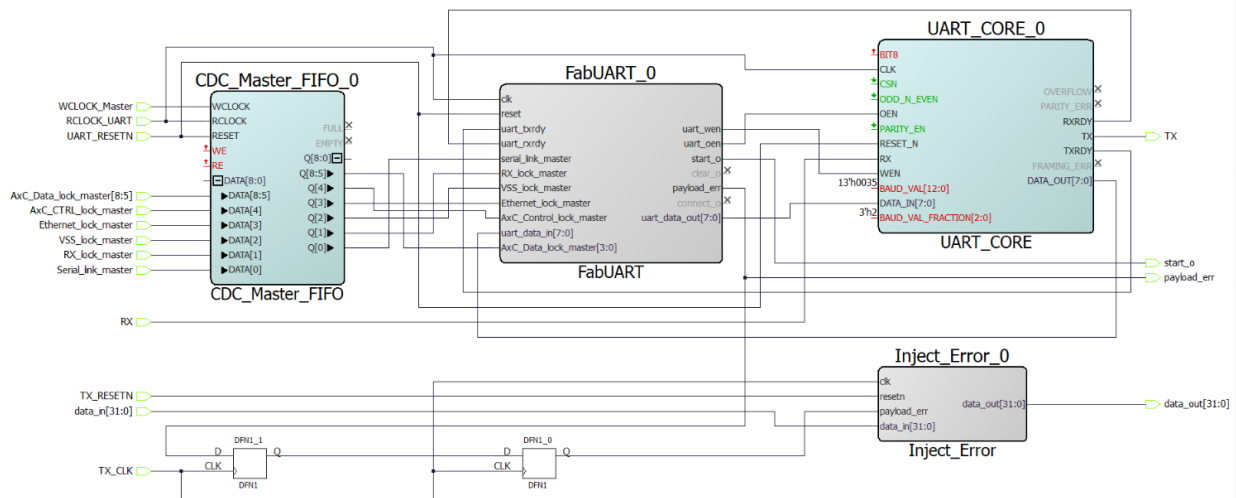
Component	Memory Map	Description
CoreGPIO	0x60051000	This bus interface is used to access the GPIO's through APB interface.
CPRI Slave IP	0x60060000	This bus interface is used to access CPRI Slave IP configuration registers through the AHB interface.
CPRI Master module	0x60070000	This bus interface is used to access CPRI Master configuration registers through the AHB interface.

### 2.3.1.5 Test Interface

The test interface system contains the UART interface and Inject error module.

The following figure shows the test interface SmartDesign implementations.

**Figure 11 • Test Interface SmartDesign**



#### 2.3.1.5.1 UART Interface

The UART\_Interface SmartDesign interfaces Fabric UART logic with the GUI, which displays the lock signal.

#### 2.3.1.5.2 Inject Error

The Inject error module induces an error in the CPRI frame which propagates through the serial interface. The presence of an error is validated using a pattern checker module.

## 2.4 Port Description

The following table describes the input and output ports of the design.

**Table 3 • Port Description**

Signal	Direction	Description
RESETN	Input	External reset
REF_CLK_PAD_P_0 and REF_CLK_PAD_N_01	Input	This is the differential reference clock generated from the on-board 122.88 MHz oscillator.
LANE0_RXD_P and LANE0_RXD_N	Input	Transceiver Receiver differential input of Lane0
LANE1_RXD_P and LANE1_RXD_N	Input	Transceiver Receiver differential input Lane1
RX	Input	This is the input signal received by the UART interface from the GUI.
LANE0_TXD_P and LANE0_TXD_N	Output	Transceiver Receiver differential output of Lane0
LANE1_TXD_P and LANE1_TXD_N	Output	Transceiver Receiver differential output of Lane1
Config_done	Output	Indicates CPRI master and slave register configuration is completed by Mi-V processor
VSS_lock	Output	Indicates the Received Vendor specific data is correct
Ethernet_lock	Output	Indicates the Received Ethernet data is correct
AxC_Control_lock	Output	Indicates the received Antenna Control data is correct
AxC_Data_lock	Output	Indicates the received Antenna IQ data is correct
TX	Output	This is the output data received by the GUI from the UART interface.

## 2.5 Register Configuration

The CPRI Master module and CPRI Slave IP are connected to the Mi-V processor using an AHB interface. The base address of CPRI master is 0x60070000 and the base address of CPRI Slave IP is 0x60060000.

**Note:** It is recommended to follow the order of register configuration as described in the Table 4, page 11.

The following table describes register configuration and these registers are configured in the SoftConsole project: mpf\_dg0843\_df\SoftConsole\CPRI\_config\main.c

**Table 4 • Register Configuration**

Offset	Value	Description
0x04	0x1F	TX Interface Sequence Number Register Line Rate 5: TX_SEQ_NUM: 0x1f
0x08	0x1F	RX Interface Sequence Number Register Line Rate 5: RX_SEQ_NUM: 0x1f
0x0C	0xBC	TX Framer K character Register TX_Z_0_0_SYNC: 0xBC which is K28.5 character
0x10	0xBC	RX Framer K character Register RX_Z_0_0_SYNC: 0xBC which is K28.5 character
0x14	0x50	TX Framer D character Register TX_Z_0_1_SYNC: 0x50 which is D16.2 character
0x18	0x50	RX Framer D character Register RX_Z_0_1_SYNC: 0x50 which is D16.2 character
0x1C	0x50	TX Framer D character Register TX_Z_0_Y_SYNC: 0x50 which is D16.2 character
0x20	0x50	RX Framer D character Register RX_Z_0_Y_SYNC: 0x50 which is D16.2 character
0x2C	0x14	TX Fast Ethernet pointer register TX_Z_194_0: 0x14 the Fast Ethernet data to be written from this location
0x24	0x01	TX Protocol Version Register TX_Z_2_0_SYNC: 0x01, the current version of the IP supports only protocol version 1
0x30	0x14	RX Fast Ethernet pointer register RX_Z_194_0: 0x14 the Fast Ethernet data to be written from this location
0x100	0xF4240	Start Timer Register Time out value for L1 Sync state machine
0x104	0x1FFF	HFN Sync Counter Register Line 5: HFN_COUNT: 1FFF This value indicates the duration of one CPRI Hyperframe for the selected Line rate
0x00	0xF	CPRI Control Register Enables IQ mapper, demapper, and L1 sync state machine and CPRI IP functionality

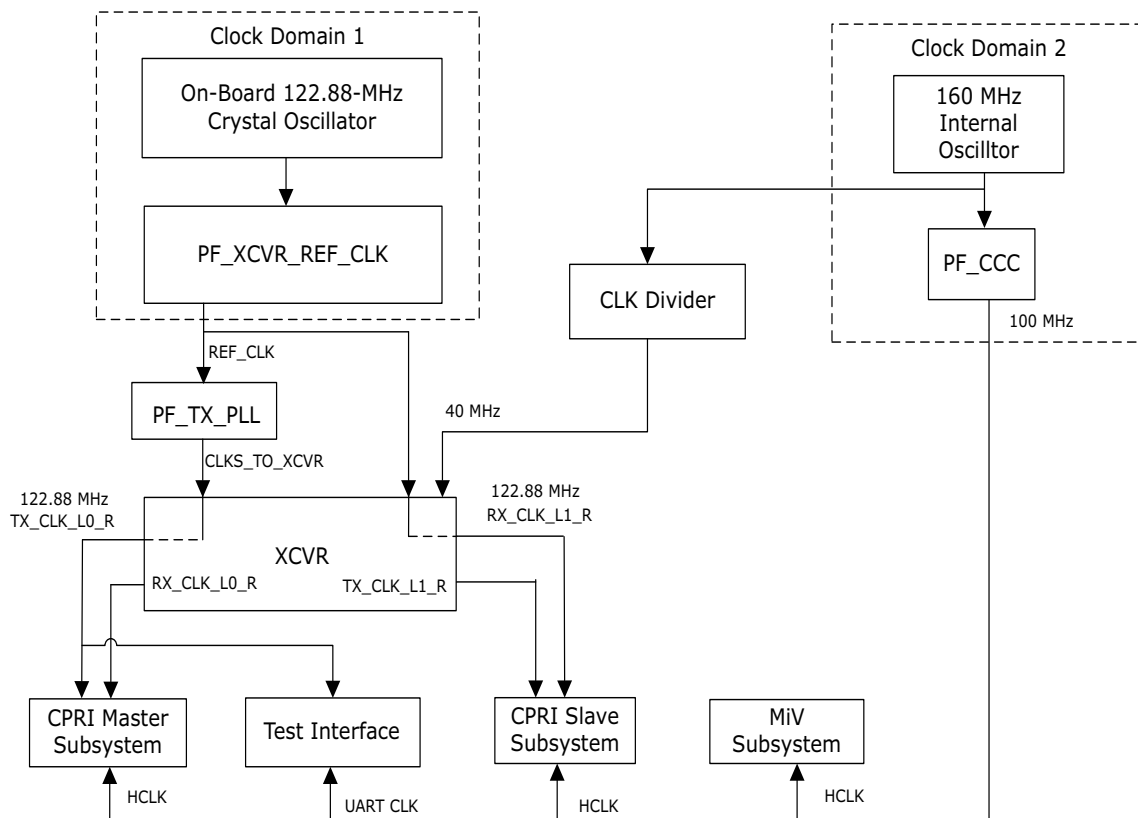
## 2.6 Clocking Structure

In this demo design, there are two clock domains. The on-board 122.88 MHz crystal oscillator drives the XCVR reference clock in 8b10b mode. This generates Lane0/1 RX clock and Lane0/1 TX clock. CPRI Master system, CPRI Slave Subsystem, and HDL modules use TX and RX clock (122.88 MHz).

The on-chip 160 MHz RC oscillator drives the CCC which generates 100 MHz clock. The UART\_Interface, Mi-V system and the AHB interface of the CPRI Master Subsystem and CPRI Slave Subsystem use 100 MHz clock. The clock divider generates 40 MHz clock for the XCVR\_ERM.

The following figure shows the clocking structure in the reference design.

**Figure 12 • Clocking Structure**



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

**Table 5 • Clocks**

Clock Name	Source	Frequency
Mi-V Clock	CCC_0	100 MHz
RX_CLK_L0_R	Transceiver RX recovered clock (Lane0)	122.88 MHz
RX_CLK_L1_R	Transceiver RX recovered clock (Lane1)	122.88 MHz
TX_CLK_L0_R	Transceiver TX PLL clock (Lane0)	122.88 MHz
TX_CLK_L1_R	Transceiver TX PLL clock (Lane1)	122.88 MHz
CTRL_CLK	CLK Divider	40 MHz

## 2.7 Reset Structure

In the demo design, the reset signal is generated using the Reset\_Block module. CoreReset\_FF (CoreReset\_PF) module releases active low reset signal of TEST\_INTERFACE, Mi-V subsystem and CPRI Master and Slave subsystem, when PLL\_lock output from PF\_CCC block, RESETN (External active low signal) and DEVICE\_INIT\_DONE signal from INIT\_MONITOR block are asserted.

The CoreReset\_L0\_TX (CoreReset\_PF) module releases an active low reset signal of the CPRI Master subsystem and Test Interface when RESETN (External active low signal) and DEVICE\_INIT\_DONE signal from INIT\_MONITOR block are asserted.

The CoreReset\_L0\_RX (CoreReset\_PF) module releases an active low reset signals of the CPRI Master subsystem when RESETN (External active low signal) and DEVICE\_INIT\_DONE signal from INIT\_MONITOR block are asserted.

The CoreReset\_L1\_TX (CoreReset\_PF) module releases an active low reset signal of the CPRI Slave subsystem when RESETN (External active low signal) and DEVICE\_INIT\_DONE signal from INIT\_MONITOR block are asserted.

The CoreReset\_L1\_RX (CoreReset\_PF) module releases an active low reset signal of the CPRI Slaven subsystem when RESETN (External active low signal) and DEVICE\_INIT\_DONE signal from INIT\_MONITOR block is asserted.

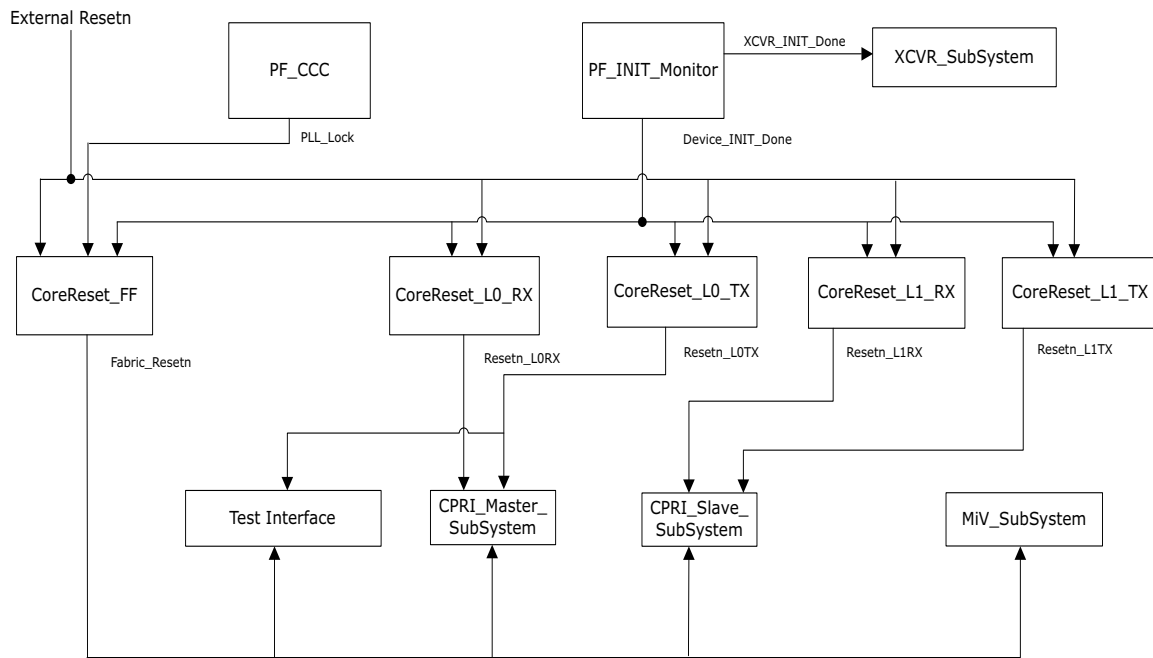
The INIT\_MONITOR releases the active low signal of XCVR\_subsystem reset signals (PMA\_ARST\_N and PCS\_ARST\_N) when XCVR\_INIT\_DONE signal from INIT\_MONITOR block is asserted.

DEVICE\_INIT\_DONE and XCVR\_INIT\_DONE signals are asserted when the device initialization is complete. 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.

The following figure shows the reset structure in the demo design.

**Figure 13 • Reset Structure**



## 3 Libero Design Flow

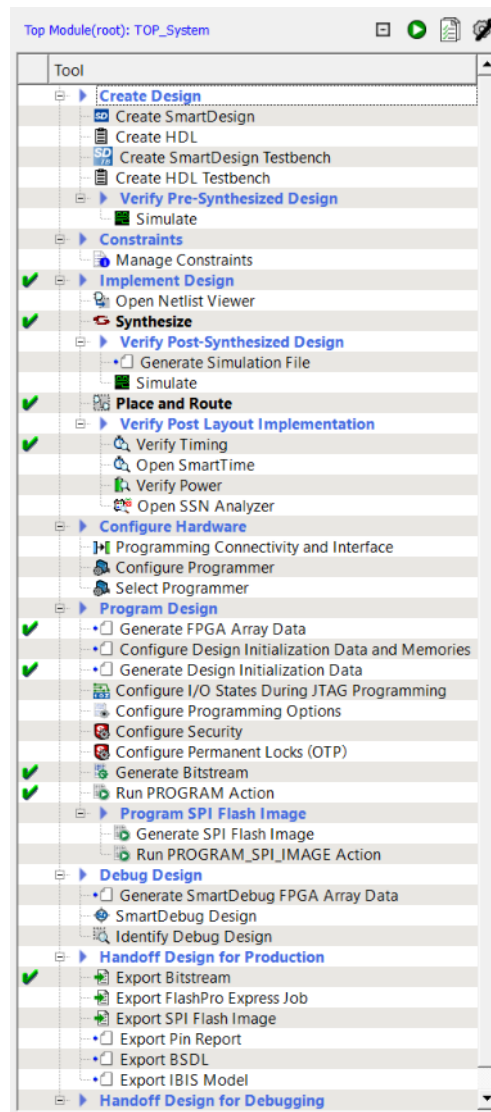
This chapter describes the Libero design flow, which involves the following processes:

- Synthesis, page 16
- Place and Route, page 17
- Verify Timing, page 18
- Design and Memory Initialization, page 18
- Generate Bitstream, page 20
- Run Program Action, page 21

**Note:** To initialize the TCM in PolarFire using the system controller, a local parameter `I_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 window.

**Figure 14 • Libero Design Flow Options**



## 3.1 Synthesis

To synthesize the design, perform the following step:

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

When the synthesis is successful, a green tick mark appears next to **Synthesize**, as shown in Figure 14, page 15.

The following table lists the resource utilization of the CPRI loopback design. These values may vary slightly for different Libero runs, settings, and seed values.

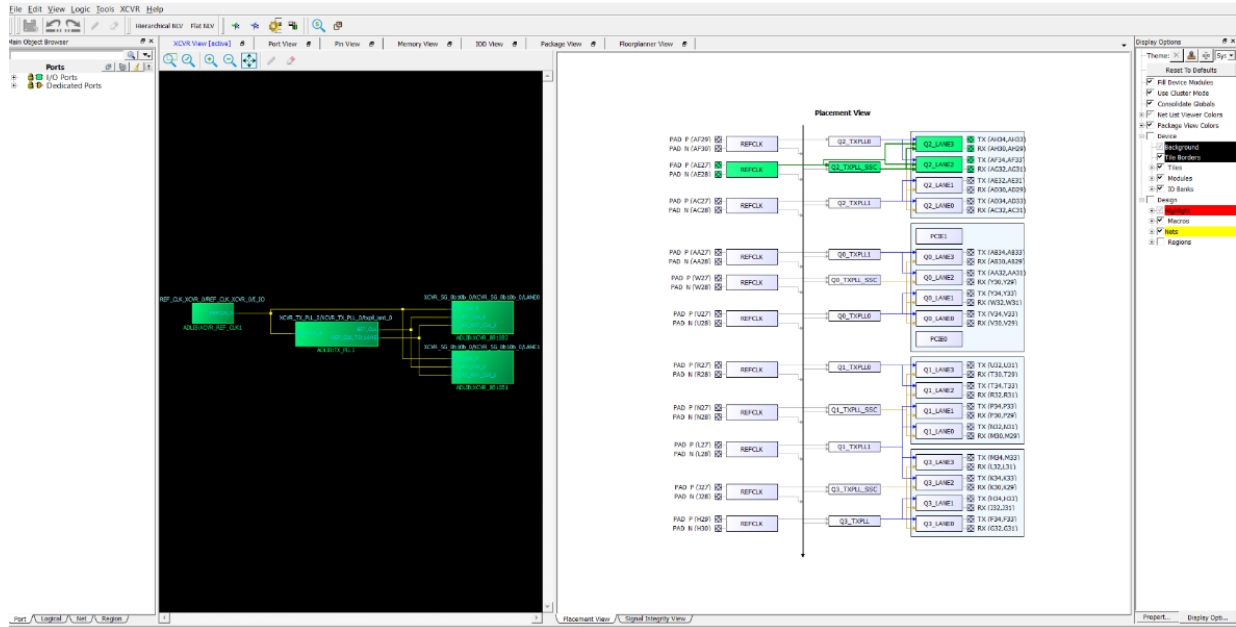
**Table 6 • Resource Utilization—Synthesis**

Type	Used	Total	Percentage
4LUT	31871	299544	10.64
DFF	20214	299544	6.75
I/O	0	1536	0.00
User I/O	11	512	2.15
-Single-ended I/O	11	512	2.15
-Differential-ended I/O	0	256	0.00
μSRAM	12	2772	0.43
LSRAM	97	952	10.19
Math	2	924	0.22
H-Chip Global	10	48	20.83
Local Global	4	1008	0.40
PLL	1	8	12.50
DLL	0	8	0.00
CRN_INT	1	24	4.17
UJTAG	1	1	100.00
INIT	1	1	100.00
OSC_RC1600 MHz	1	1	100.00
Transceiver Lanes	2	16	12.50
Transceiver PCIe	0	2	0.00
TX_PLL	1	11	9.09
XCVR_REF_CLK	1	11	9.09
ICB_CLKINT	4	72	5.56
ICB_CLKDIV	1	24	4.17
ICB_INT	1	12	8.33

## 3.2 Place and Route

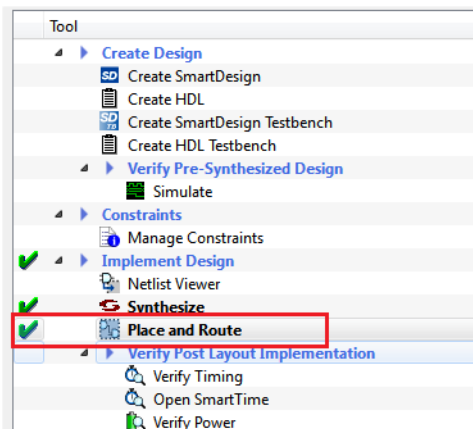
To place and route the design, TX\_PLL, XCVR\_REF\_CLK, and PF\_XCVR, must be configured using the I/O Editor. For On-board transceiver loop back, the Lane2 and Lane3 of Quad2 are used, as shown in the following figure.

Figure 15 • I/O Editor Option—XCVR View



1. On the **Design Flow** window, double-click **Place and Route**. When place and route is successful, a green tick mark appears as shown in the following figure.

Figure 16 • Place and Route



The following table lists the resource utilization after place and route.

Table 7 • Resource Utilization—Place and Route

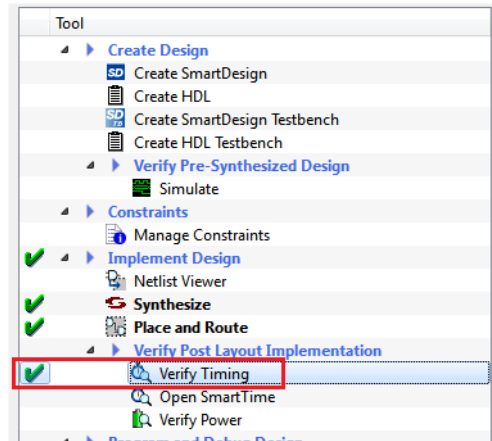
Type	Used	Total	Percentage
4LUT	32019	299544	10.69
DFF	20284	299544	6.77
I/O Register	0	510	0.00
Logic Elements	38062	299544	12.71

### 3.3 Verify Timing

To verify timing, perform the following steps:

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

Figure 17 • Verify Timing



### 3.4 Design and Memory Initialization

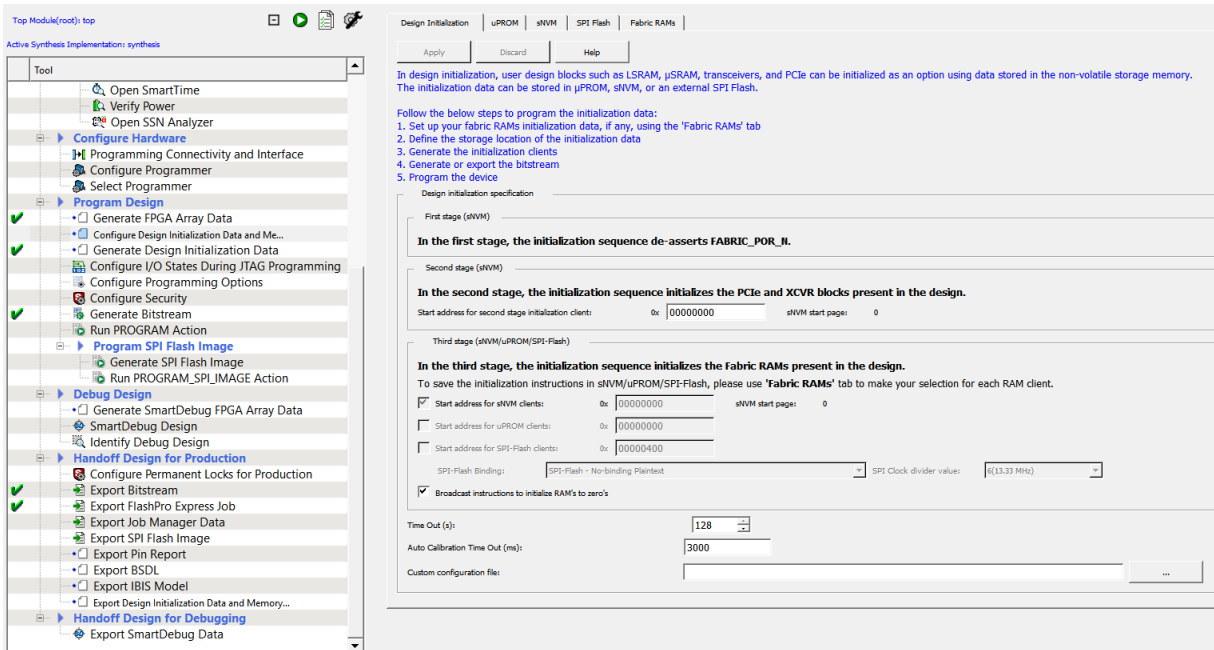
The **Configure Design Initialization Data and Memories** generates the LSRAM initialization client and adds it to sNVM,  $\mu$ PROM, or an external SPI flash, based on the type of non-volatile memory selected. In this demo, the LSRAM initialization client is stored in the sNVM.

This process requires the user application executable file (hex file) to initialize the LSRAM blocks on device power-up. The hex file (CPRI\_config.hex) is available in the `DesignFiles_Directory\mpf_dg0843_df\Libero_Project` folder. When the hex file is imported, a memory initialization client is generated for LSRAM blocks.

To initialize the memory, perform the following steps:

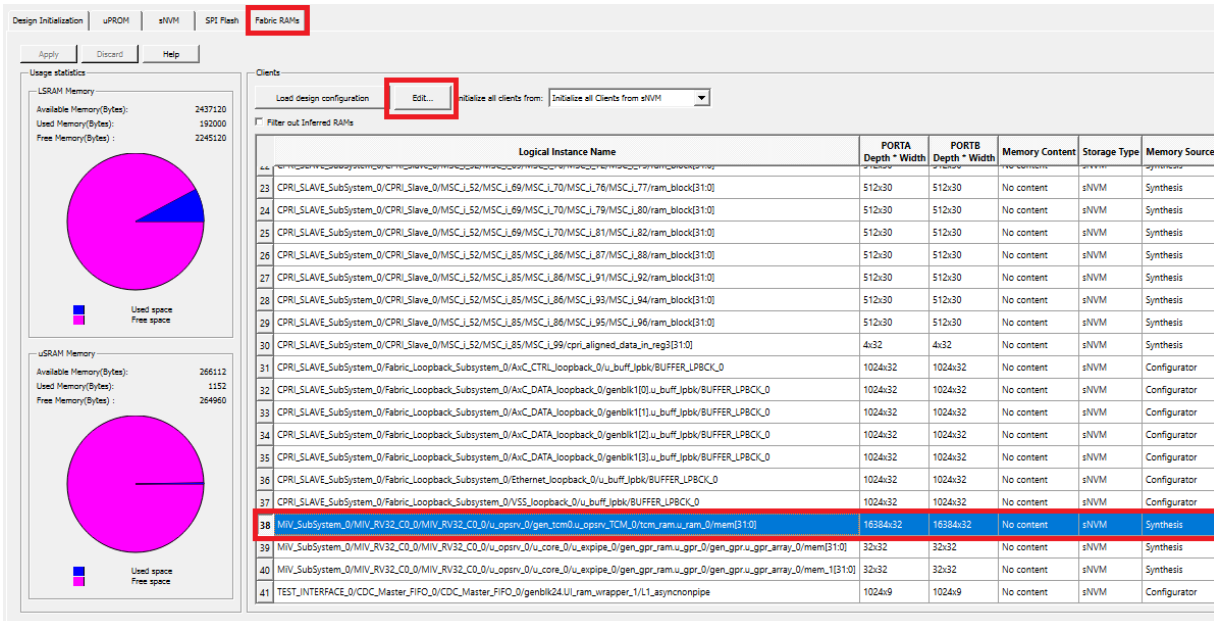
1. On the Design Flow window, double-click **Configure Design Initialization Data and Memories**. The Design and Memory Initialization window opens as shown in the following figure.

**Figure 18 • Design and Memory Initialization Window**



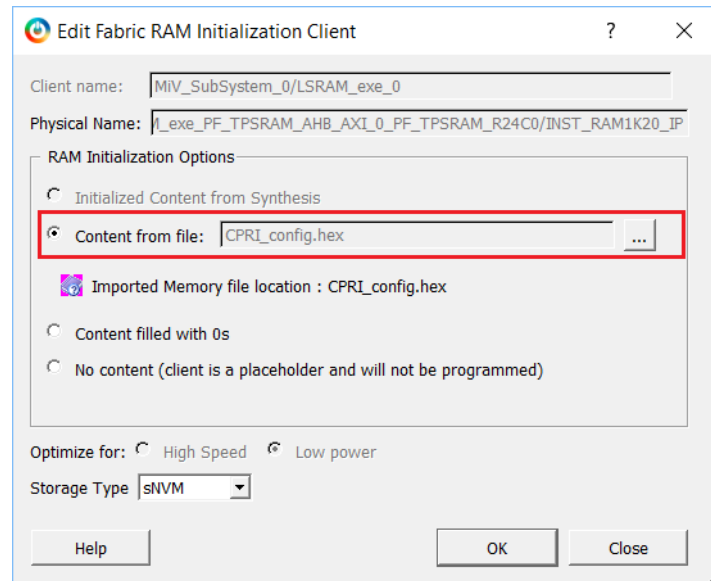
2. In the **Fabric RAMs** tab, select the LSRAM client from the list, and then click **Edit**, as shown in the following figure.

**Figure 19 • Fabric RAM**



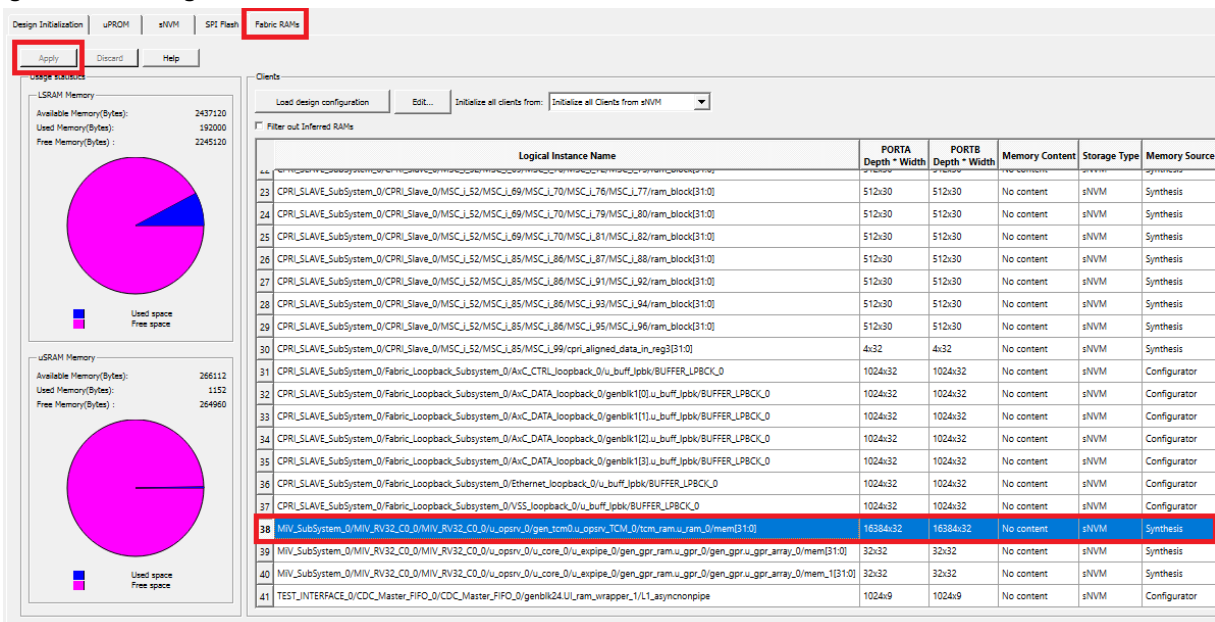
3. In the **Edit Fabric RAM Initialization Client** dialog box, select the **Content from file** option, and locate the CPRI\_config.hex file from DesignFiles\_directory\mpf\_dg0843\_df\Libero\_Project folder and then click **OK**, as shown in the following figure.

Figure 20 • Edit Fabric RAM Initialization Client



4. Click **Apply** as shown in the following figure.

Figure 21 • Design Initialization



5. To generate design initialization data, click **Generate Initialization Data** on the Design Flow window. After successful generation of the Initialization data, a green tick mark appears next to Generate Initialization Data option as shown in the Figure 21, page 20.

### 3.5 Generate Bitstream

To generate bitstream, perform the following steps:

- On the Design Flow window, double-click **Generate Bitstream**.  
When the bitstream is successfully generated, a green tick mark appears next to **Generate Bitstream**, as shown in Figure 14, page 15.
- Right-click **Generate Bitstream** and select **View Report** to view the corresponding log file in the **Reports** tab.

## 3.6 Run Program Action

To program the PolarFire device, perform the following steps.

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

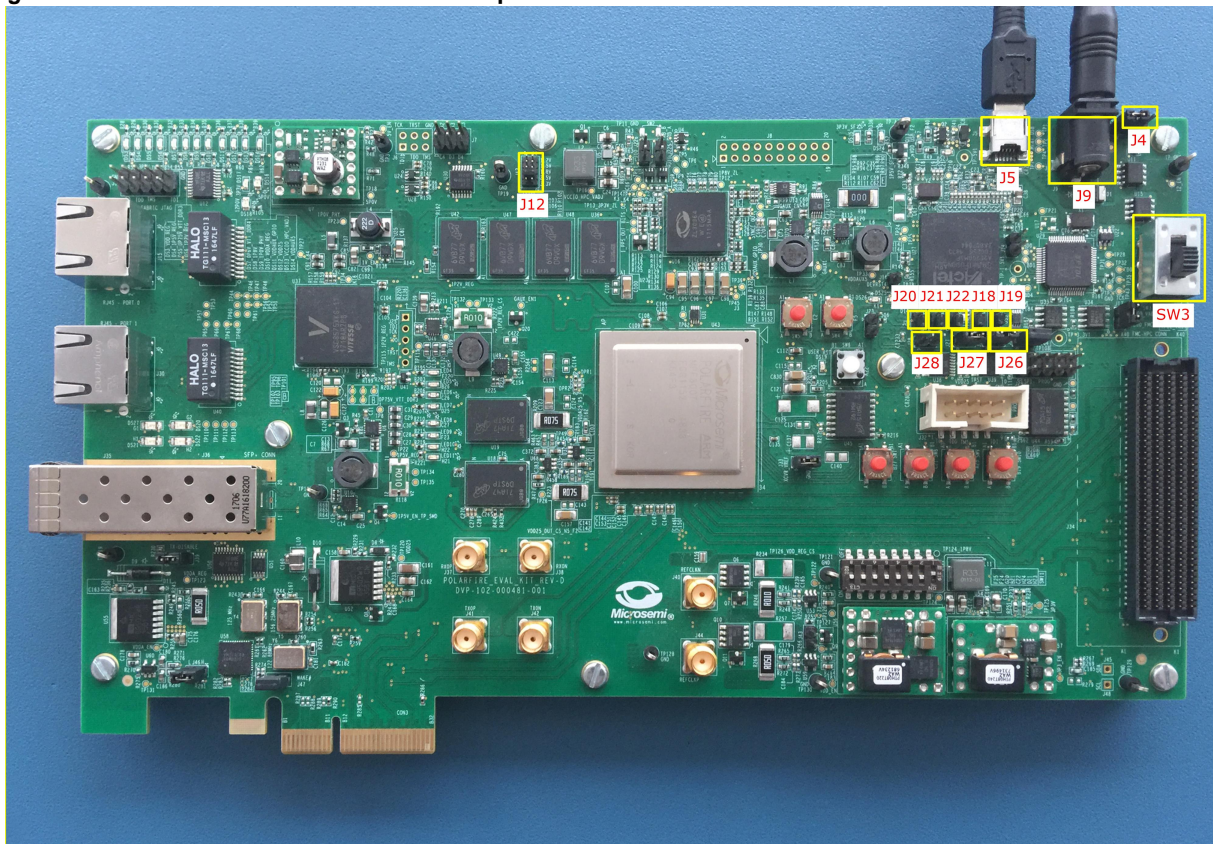
**Table 8 • PolarFire Evaluation Board Jumper Settings**

Jumper	Description
J18, J19, J20, J21, and J22	Short pin 2 and 3 for programming the PolarFire FPGA through FTDI
J28	Short the pin 2 and 3 for programming through the on-board FlashPro5
J4	Short pin 1 and 2 for manual power switching using SW3
J12	Short pin 3 and 4 for 2.5 V
J46	Short pin 1 and 2 for routing 125 MHz differential clock oscillator output to the side. Open pin 1 and 2 for routing 122.88 MHz differential clock oscillator to the line side.

2. Connect the power supply cable to the J9 connector.
3. Connect the USB cable from the host PC to the J5 (FTDI port).
4. Power on the board using the SW3 slide switch.

The following figure shows the board setup.

**Figure 22 • PolarFire Evaluation Board Setup**



5. On the Design Flow window, double-click **Run PROGRAM Action**.

The device is successfully programmed and a green tick mark appears next to **Run PROGRAM Action** as shown in Figure 14, page 15. LED 4 is asserted once the device is programmed.

## 4 Running the Demo

This section describes how to install and use the CPRI Demo application Graphic User Interface (GUI). The PolarFire CPRI demo application is a simple GUI that runs on the host PC to communicate with the PolarFire Device.

### Installing CPRI Demo Application

To install CPRI demo application, perform the following steps:

1. Install the GUI (setup.exe) from the following design files folder `mpf_dg0843_df/GUI`

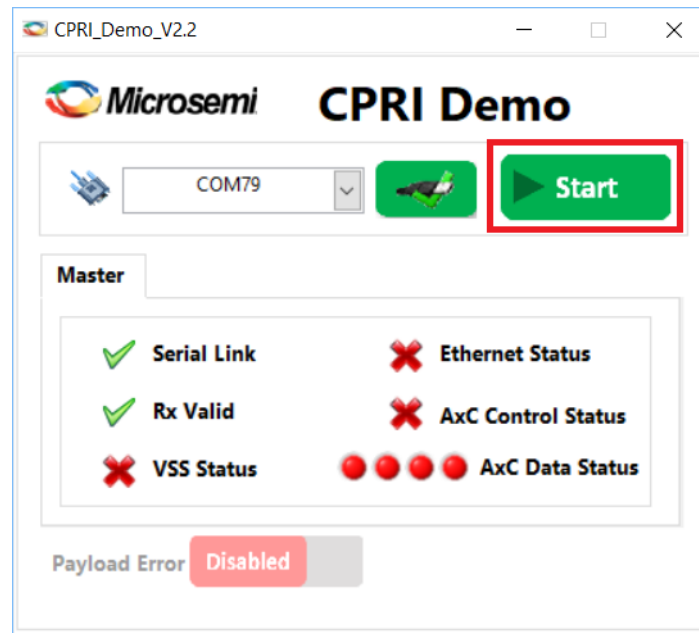
The following steps describes how to install and use the GUI to run the CPRI Loopback demo. The following procedure assumes that:

- The PolarFire Evaluation board is connected.
- The PolarFire FPGA is programmed with the CPRI Loopback design.

To run the CPRI Loopback demo, perform the following steps:

1. Extract the contents of the `mpf_dg0843_df.zip` file.
2. From the `mpf_dg0843_df\GUI_Installer` folder, double-click the `setup.exe` file.
3. Follow the instructions displayed on the installation wizard.
4. After successful installation, `CPRI_GUI` appears on the **Start** menu of the host PC desktop.
5. From the **Start** menu, click **CPRI\_GUI** to launch the application.
6. The GUI detects the COM port number and automatically connects to the board, as shown in the following figure. COM Port numbers may vary.

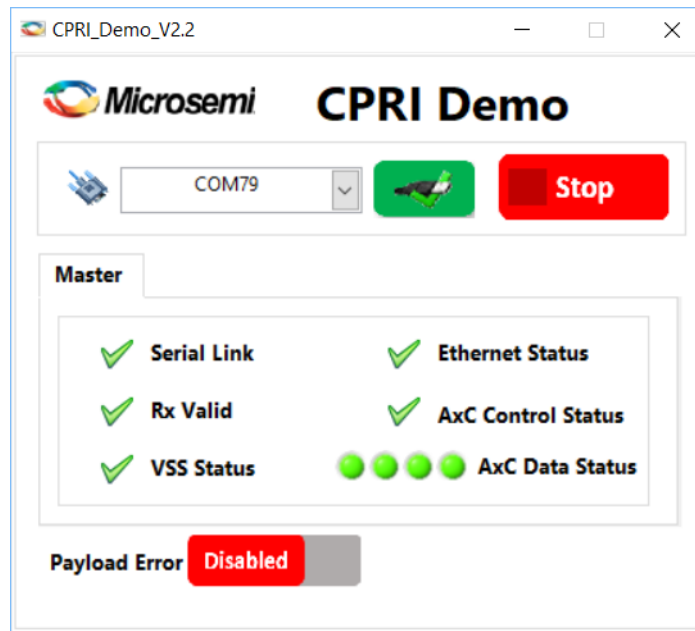
**Figure 23 • Selecting COM Port and Connecting**



7. Click **Start**. The data is generated and framed using the CPRI master module and sent over the serial transmit link. It is then received by the receiver, and CPRI slave IP unpacks the data and loops back the data to the CPRI master module. The checker in the Master system checks the incoming data for any errors. The status can be monitored using the status signals on the GUI at any time, as shown in the following figure. The following are the status signals:
- Serial Link: Indicates transceiver link status
  - Rx Valid: Indicates if the transmitter and receiver data are locked
  - VSS status: Indicates the received VSS data is valid.
  - Ethernet status: Indicates the received Ethernet data is valid.
  - AxC Control status: Indicates the received Antenna carrier control data is valid.
  - AxC Data Status: Indicates the received Antenna carrier data is valid.

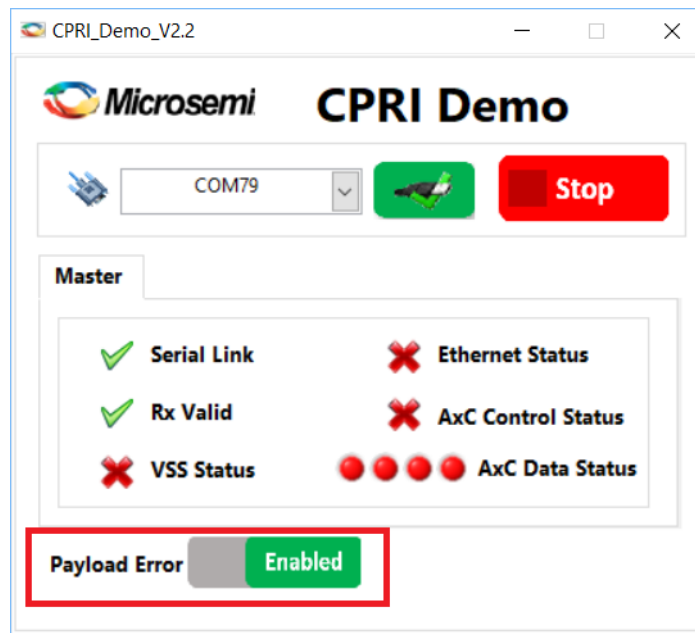
The following shows the status signals. The LED4, LED5, LED6, LED7, LED8, LED9, LED10, and LED11 is asserted at the same time.

**Figure 24 • Master Checker Status—Pass**



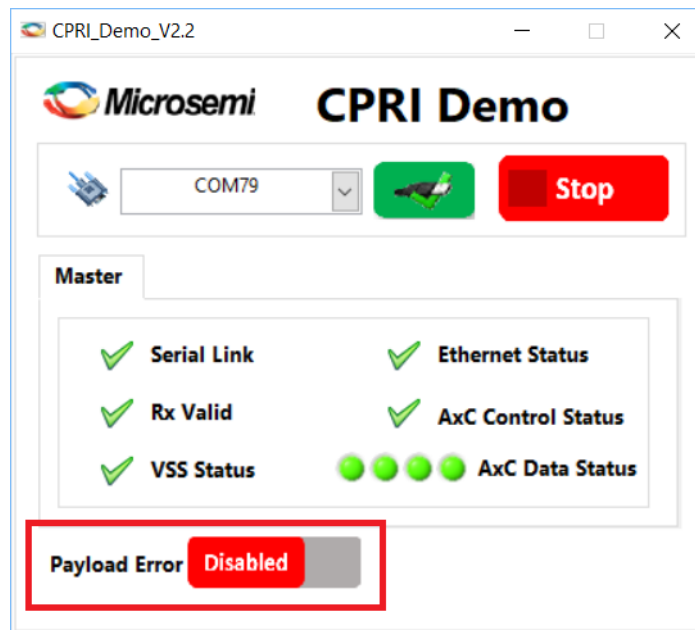
- Click **Payload Error** to induce error in VSS data, Ethernet data, Antenna carrier control data and Antenna carrier Data. Observe the error status of master as shown following figure. The LED5, LED6, LED7, LED8, LED9, LED10, and LED11 is deasserted at the same time.

**Figure 25 • Payload Error Enabled—Master**



- Disable Payload Error to stop generating an error and observe that the Serial Link, Rx Valid, and all the status signals of master turn green, as shown in the following figure. When the error is cleared, the LED5 to LED11 is asserted at the same time.

**Figure 26 • Payload Error Disabled—Master**



The CPRI demo is successfully run.

## 5 Appendix 1: Programming the Device Using FlashPro Express

This chapter describes how to program the PolarFire device with the Job programming file using a FlashPro programmer. The default location of the .stp file is: mpf\_dg0843\_df\Programming\_Job

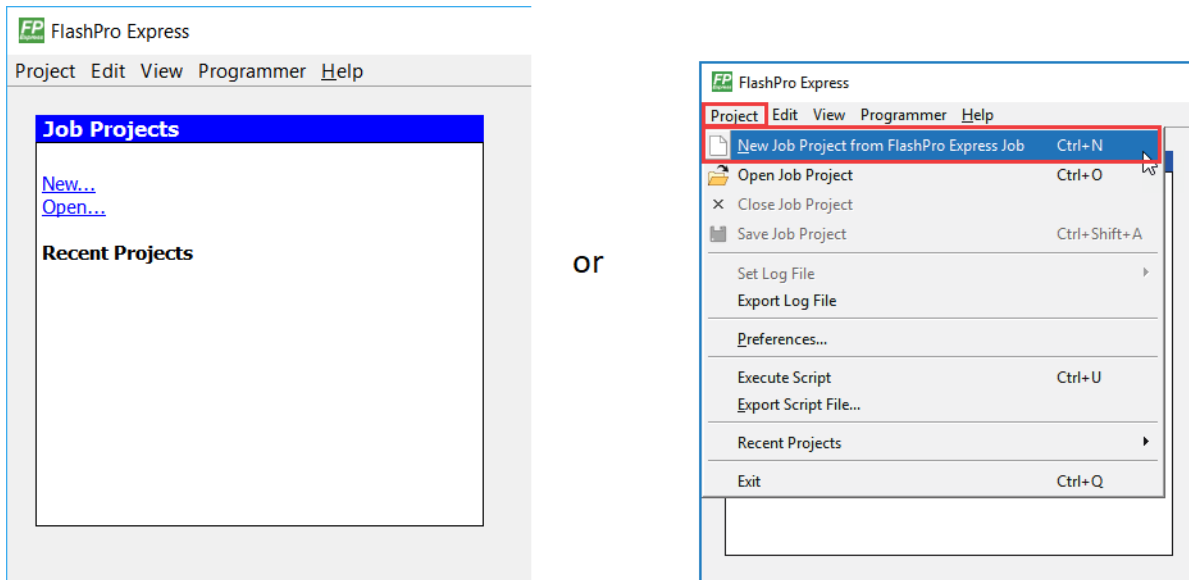
To program the PolarFire device using FlashPro Express, perform the following steps:

1. Ensure that the jumper settings on the board are the same as listed in Table 8, page 21.

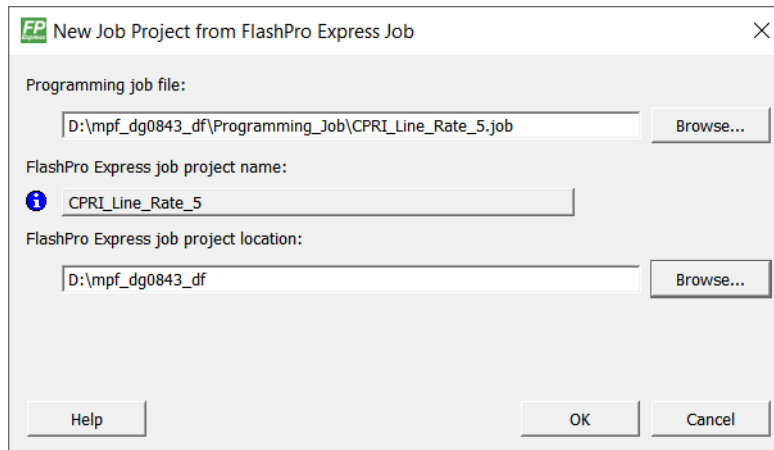
**Note:** The power supply switch must be switched off while making the jumper connections.

2. Connect the power supply cable to the **J9** connector on the board.
3. Connect the USB cable from the Host PC to the **J5** (FTDI port) on the board.
4. Power on the board using the **SW3** slide switch.
5. On the host PC, launch the FlashPro Express software.
6. Click **New** or select **New Job Project** from FlashPro Express Job from Project menu to create a new job project, as shown in the following figure.

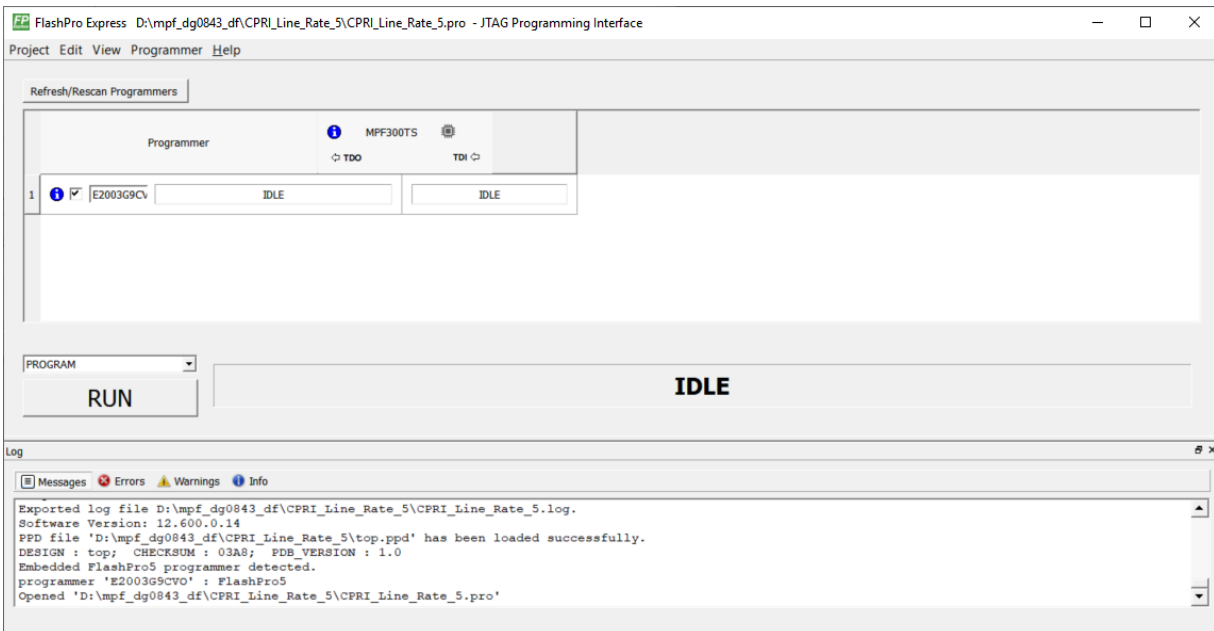
**Figure 27 • FlashPro Express Job Project**



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\_dg0843\_df\Programming\_Job.
  - FlashPro Express job project location: Click **Browse** and navigate to the location where you want to save the project.

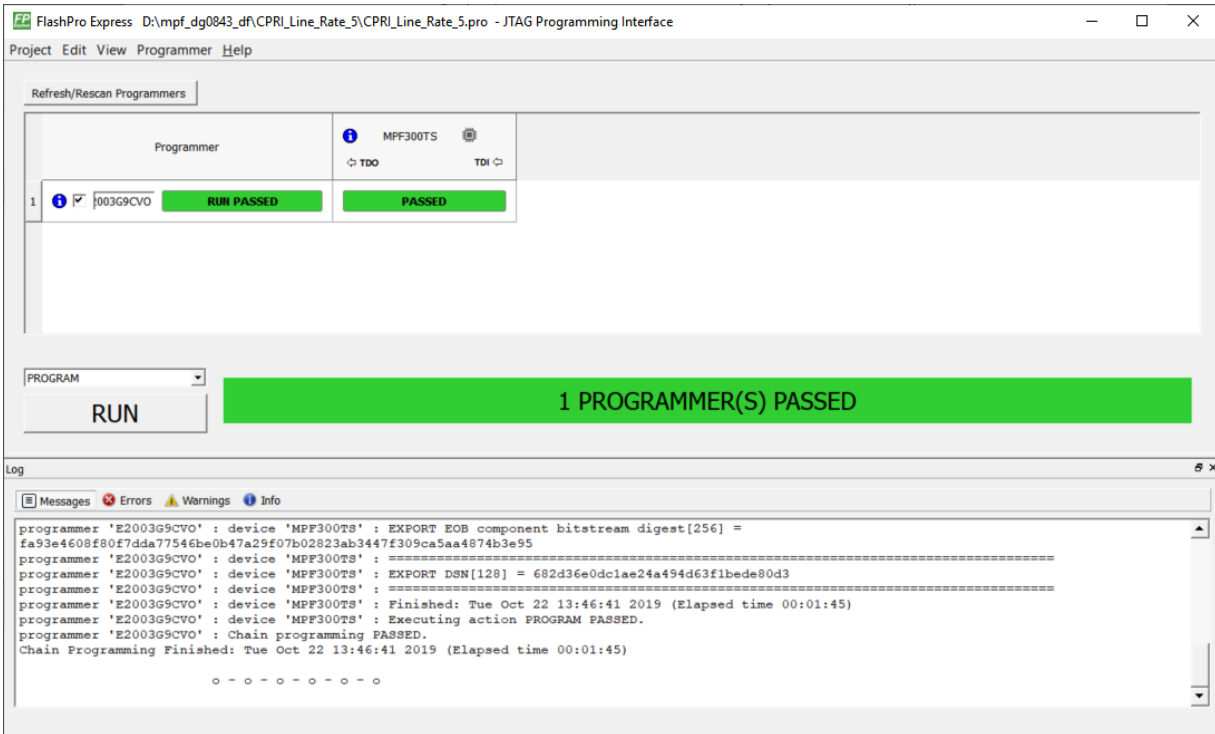
**Figure 28 • New Job Project from FlashPro Express Job**

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

**Figure 29 • Programming the Device**

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

**Figure 30 • FlashPro Express—RUN PASSED**



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

## 6 Appendix 2: Running the TCL Script

---

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_dg0843_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.

## 7 Appendix 3: References

---

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

- For more information about PolarFire transceiver blocks, see *UG0677: PolarFire FPGA Transceiver User Guide*.
- For more information about Libero, ModelSim, and Synplify, refer *Microsemi Libero SoC webpage*.
- For more information about CPRI IP, see **Libero Catalog -> Solution-Wireless->Microsemi CPRI User Guide**
- For more information about PolarFire Evaluation kit, see *UG0747: PolarFire FPGA Evaluation Kit User Guide*
- For more information about Power-Up and Reset, see *UG0725: PolarFire FPGA Device Power-Up and Resets User Guide*
- For more information about the CoreJTAGDEBUG IP core, see *CoreJTAGDebug\_HB.pdf* from **Libero->Catalog**.
- For more information about the MIV\_RV32 IP core, see *MIV\_RV32 Handbook* from the Libero SoC Catalog.
- For more information about the CoreAHBtoAPB3 IP core, see *CoreAHBtoAPB3\_HB.pdf*.
- For more information about the CoreUARTapb IP core, see *CoreUARTapb\_HB.pdf*.
- For more information about the CoreAHBLite IP core, see *CoreAHBLite\_HB.pdf*.
- For more information about the CoreAPB3 IP core, see *CoreAPB3\_HB.pdf*.
- For more information about the CoreGPIO IP core, see *CoreGPIO\_HB.pdf*