PolarFire FPGA Dynamic Reconfiguration Interface Application Note

AN4592

Introduction (Ask a Question)

The PolarFire® FPGA family includes multiple embedded low-power and performance-optimized transceivers. Each transceiver has Physical Medium Attachment (PMA) and Physical Coding Sub-layer (PCS) logics, and interfaces to the FPGA fabric.

The transceiver has a multi-lane architecture with each lane natively supporting serial data transmission rates from 250 Mbps to 12.7 Gbps.

This document describes how to perform the dynamic reconfiguration of Clock Conditioning Circuit (CCC) and transceivers in a PolarFire FPGA by changing the output clock frequency in a glitch-free way.

Important: The SmartDebug and APB DRI accesses must not be initiated at the same time.

Important: The DRI protocol standard uses “Master” and “Slave.” The equivalent Microchip terminology used in this document is **Initiator** and **Target**, respectively.

Table of Contents

1. PolarFire FPGA Dynamic Reconfiguration Interface (Ask a Question)

Each CCC and transceiver has a Dynamic Reconfiguration Interface (DRI) that can be enabled to configure its parameters without reprogramming the device. The volatile configuration registers control CCC and transceiver reconfiguration that are loaded with values from the flash configuration bits at power-up. An APB3 bus initiator must be interfaced to the CCC and transceiver using a DRI macro for dynamic configuration. The APB3 bus initiator is required to dynamically modify the CCC and transceiver configuration register values as per the design needs.

Any of the configuration registers can be accessed dynamically using the APB3 interface. This document does not discuss all the Fabric CCC and transceiver registers that can be dynamically configured. It describes how to dynamically change the output clock frequency in CCC and the data rate of transceivers.

1.1 Design Requirements (Ask a Question)

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

Table 1-1. Design Requirements

Requirement	Version
Operating System	64-bit Windows® 10 and 11
Hardware	
PolarFire® Evaluation Kit (MPF300TS-EVAL-KIT)	Rev D or later
2 SMA-to-SMA cables with 10 Gbps support (not provided with the kit)	—
Oscilloscope	—
Software	
FlashPro Express	See the `readme.txt` file provided in the design files for the software versions used with this reference design.
Libero® System-on-Chip (SoC)	

Important: Libero® SmartDesign and configuration screen shots shown in this guide are for illustration purpose only. Open the Libero design to see the latest updates.

1.2 Prerequisites (Ask a Question)

Before you start, perform the following steps:

- Download the demo design files from AN4592: PolarFire FPGA Dynamic Reconfiguration Interface Application Note.
- Download and install Libero® SoC on the host PC from Libero SoC Documentation.
- Download the transceiver register map from PolarFire Device Register Map.

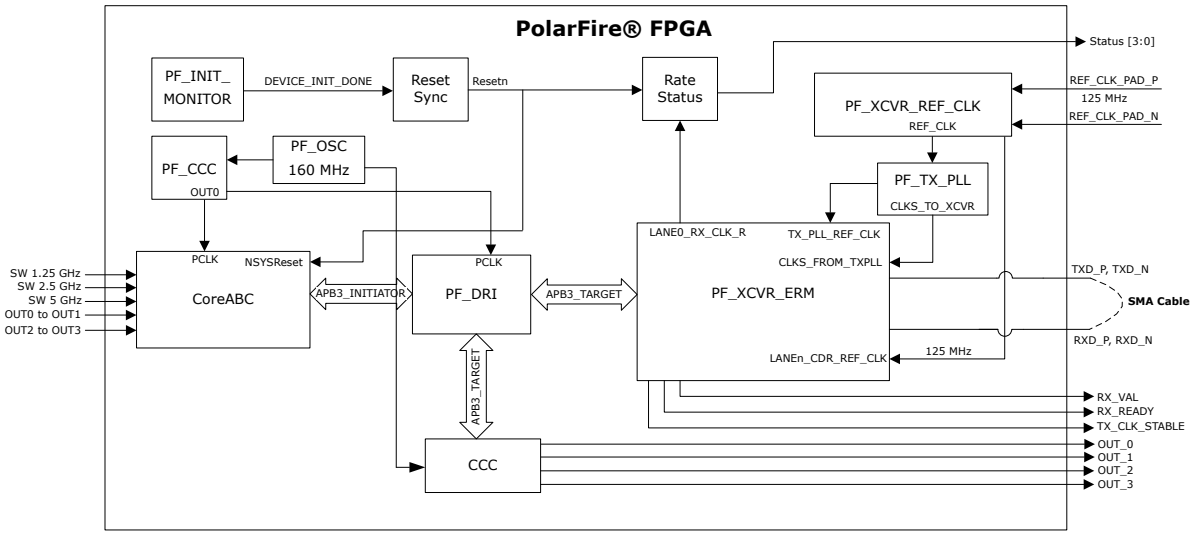
1.3 Demo Design (Ask a Question)

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

1. OSC_160 MHz provides 160 MHz clock source to PF_CCC_50 block and CCC block.
2. The PF_CCC_50 block provides 50 MHz clock for the fabric.
3. The 50 MHz fabric clock drives Reset-Synchronizer, CoreABC, and PF_DRI modules.
4. The transceiver (PF_XVCR) block instantiates the transceiver in 8b10b mode. This block receives clock from the REF_CLK signal of PF_XCVR_REF_CLK_0. The PF_TX_PLL_0 block also derives its reference clock from REF_CLK of PF_XCVR_REF_CLK_0.

5. The TX and RX lanes of the transceiver are externally looped back using SMA cables.
6. After CoreABC instruction is executed, the RX_VAL and RX_READY outputs must be monitored for link status.
7. The Rate_Status block indicates the rate at which the transceiver and the Dynamic CCC are configured.

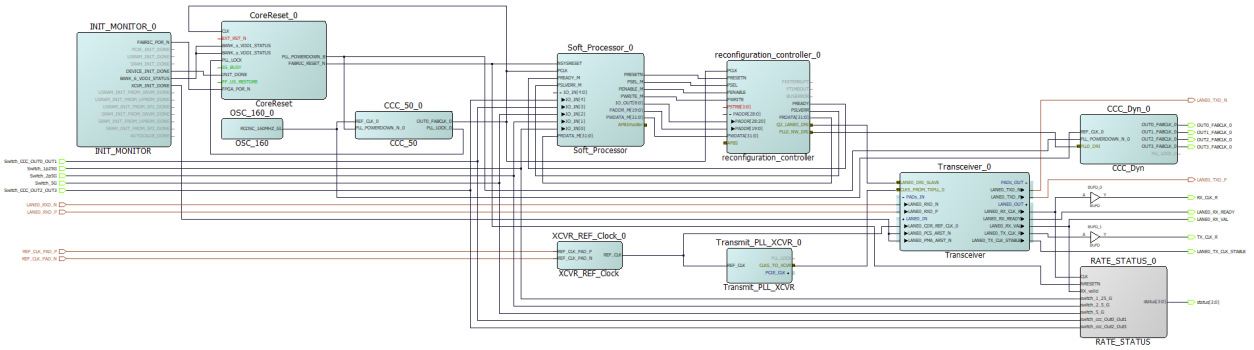
Figure 1-1. DRI Block Diagram



1.3.1 Design Implementation (Ask a Question)

The following figure shows the top-level Libero design of the PolarFire Transceiver Dynamic Reconfiguration Interface design.

Figure 1-2. Top Level Libero Design



Important: For the latest SmartDesign, see the www.microchip.com/en-us/application-notes/AN4592.

1.3.2 IP Configuration (Ask a Question)

The following sections describe the IP cores used in the design and their configurations.

Important: The IP cores which are not described in the following section keep the default configuration.

1.3.2.1 PF_CCC_0 Configuration (Ask a Question)

The PF_CCC block provides a clock for CoreABC and Dynamic Configuration interface. The input for the CCC is from 160 MHz on-chip RC oscillator. The output clock of CCC is configured at 50 MHz.

1.3.2.2 Oscillator Configuration (Ask a Question)

The RC oscillator runs on two configurations; 160 MHz RC oscillator and 2 MHz RC oscillator. The Enable RCOSC_160 MHz to Global configuration is selected.

1.3.2.3 PF_CCC_Dyn_0 (Ask a Question)

This CCC is dynamically reconfigured using DRI interface. The input for the CCC is from 160 MHz on-chip oscillator. Four Output clocks are configured at 100 MHz, 50 MHz, 100 MHz, and 50 MHz and their respective Output enables are used.

The Output enables are used to switch the Output clock frequencies in glitch-free way.

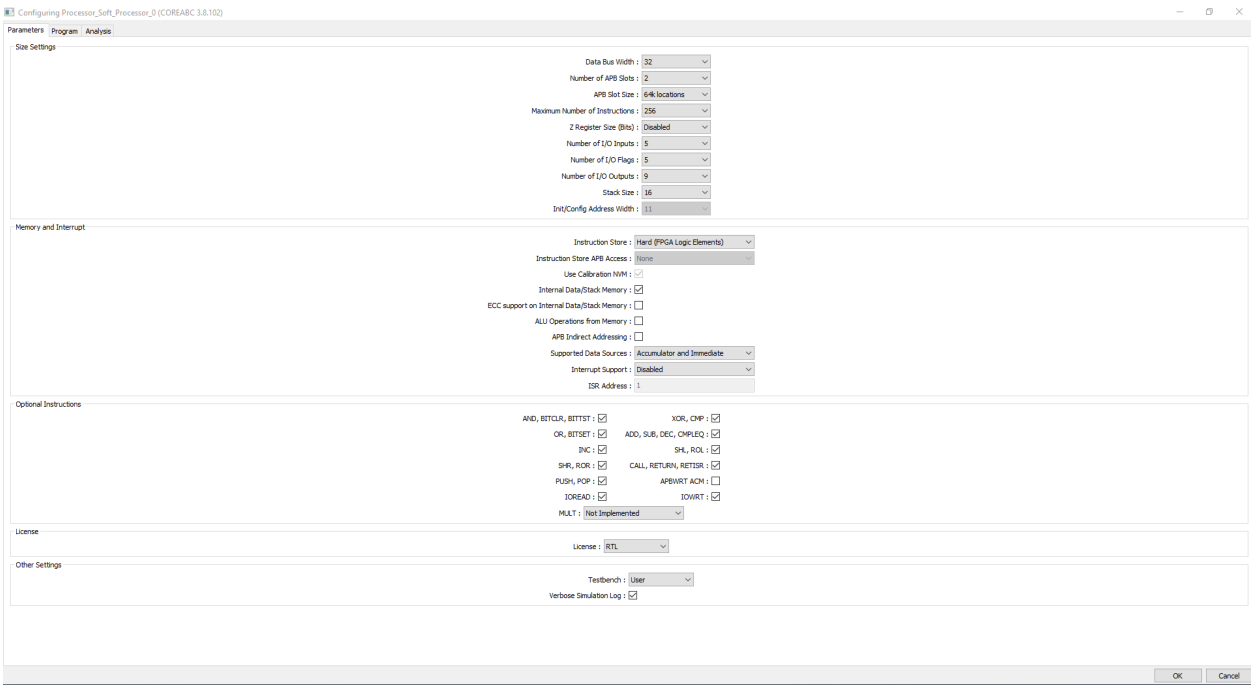
1.3.2.4 CoreABC (Ask a Question)

The CoreABC is a programmable soft-controller targeted for implementing Advanced Microcontroller Bus Architecture (AMBA) based designs.

CoreABC in this design is connected to DRI as an APB3 initiator. The APB3 target of DRI are connected to the transceiver and CCC. CoreABC initiates the instruction sequence and dynamically performs Read/Write operation on Transceiver and CCC registers. The number of APB slots, APB slot size, and maximum number of instructions are configured depending on the number of peripherals and address size used. The CoreABC uses the 20-bit address bus and the DRI uses 29-bit address bus. Figure 1-2 shows how the remaining 9-bit address bus can be connected using IOs from CoreABC.

The following figure shows the parameter configuration of CoreABC interface.

Figure 1-3. CoreABC Configuration



1.3.2.4.1 CoreABC Program (Ask a Question)

The following figure shows the register settings required for performing DRI instructions. The DIP switches are connected to the inputs of CoreABC and are used to control the switching between the rates.

Figure 1-4. CoreABC Program

```
$Rate_select

    IOWRT 0x600
    IOREAD
    NOP
    NOP
    NOP
    JUMP IF INPUT0 $Rate_change_1_25_G
    NOP
    NOP
    JUMP IF INPUT1 $Rate_change_2_5_G
    NOP
    NOP
    JUMP IF INPUT2 $Rate_change_5_G
    NOP
    NOP
    JUMP IF INPUT3 $Clock_change_out0_out1
    NOP
    NOP
    JUMP IF INPUT4 $Clock_change_out2_out3

JUMP $Rate_select
```

Read Input DIP Swtiches

```
$Rate_change_1_25_G

IOWRT 0x611
//SER_RSTPD
APBWRT DAT 0 0x1078 0x00000003
//DES_RSTPD
APBWRT DAT 0 0x104C 0x00000032

//TXPLL_DIV_1
    APBWRT DAT 1 0x0010 0x0140000
//TXPLL_DIV_2
    APBWRT DAT 1 0x0014 0x1000000
//SER_CLK_CTRL
    APBWRT DAT 0 0x1074 0x0000073
//DES_RXPLL_DIV
    APBWRT DAT 0 0x1040 0x000420A

JUMP $Reset_release
```

XCVR Configuration Setting for 1.25 G

```
$Rate_change_2_5_G

IOWRT 0x611
//SER_RSTPD
APBWRT DAT 0 0x1078 0x00000003
//DES_RSTPD
APBWRT DAT 0 0x104C 0x00000032

//TXPLL_DIV_1
    APBWRT DAT 1 0x0010 0x0140000
//TXPLL_DIV_2
    APBWRT DAT 1 0x0014 0x1000000
//SER_CLK_CTRL
    APBWRT DAT 0 0x1074 0x0000071
//DES_RXPLL_DIV
    APBWRT DAT 0 0x1040 0x0002214

JUMP $Reset_release
```

XCVR Configuration Setting for 2.5 G

Figure 1-5. CoreABC Program-Continued

```
$Rate_change_5_G

IOWRT 0x611
//SER_RSTPD
APBWRT DAT 0 0x1078 0x00000003
//DES_RSTPD
APBWRT DAT 0 0x104C 0x00000032

//TXPLL_DIV_1
APBWRT DAT 1 0x0010 0x0140000
//TXPLL_DIV_2
APBWRT DAT 1 0x0014 0x1000000
//SER_CLK_CTRL
APBWRT DAT 0 0x1074 0x0000070
//DES_RXPLL_DIV
APBWRT DAT 0 0x1040 0x0000228

JUMP $Reset_release
```

XCVR Configuration Setting for 5 G

```
$Reset_release

//SER_RSTPD
APBWRT DAT 0 0x1078 0x00000001
//DES_RSTPD
APBWRT DAT 0 0x104C 0x00000010
IOWRT 0x601
//LRST
APBWRT DAT 0 0x1068 0x00000D0D
NOP
//LRST
APBWRT DAT 0 0x1068 0x00000404


JUMP $Rate_Change_Done
```

Release PMA and PCS Reset

```
$Clock_change_out0_out1

IOWRT 0x481
//PLL_OUT0_AND_OUT1_DIVIDER
APBWRT DAT 0 0x0010 0x08001800
IOWRT 0x681

JUMP $Rate_Change_Done

$Clock_change_out2_out3

IOWRT 0x281
//PLL_OUT2_AND_OUT3_DIVIDER
APBWRT DAT 0 0x0014 0x08001800
IOWRT 0x681

JUMP $Rate_Change_Done
```

CCC Configuration for OUT0, OUT1, OUT2, and OUT3

```
$Rate_Change_Done

NOP
NOP

JUMP $Rate_select

HALT
```

Jump Back to I/O Read

1.3.2.4.2 Instruction Flow for Transceiver Rate Change (Ask a Question)

The transceiver reconfiguration occurs in the following sequence:

1. Assert Reset for the serializer
2. Assert Reset for the deserializer

3. Change TX_PLL_DIV_1 register
4. Change TX_PLL_DIV_2
5. Change SER_CLK_CNTRL
6. Change DES_RXPLL_DIV
7. De-Assert reset for the serializer
8. De-assert reset for the deserializer
9. Assert and de-assert the soft PCS reset

Important: For more information about transceiver configuration register, see PolarFire Device Register Map.

1.3.2.4.3 Instruction for CCC Frequency Change (Ask a Question)

In this demo design, only the configuration of CCC post dividers is changed dynamically. To change the configuration of CCC post dividers, the output clocks must be stopped using corresponding OUT#_EN signals. The CoreABC instructions are used to perform the following steps:

1. Change PLL_DIV_0_1
2. Change PLL_DIV_2_3

1.3.2.5 Dynamic Reconfiguration Interface (Ask a Question)

DRI performs the runtime configuration of transceiver PMA/PCS, PCIe, CCC, and Transmit PLLs after initialization. DRI ports are routed through hardwired connections.

In this demo design, DRI is used to perform the runtime calibration of transceiver and CCC. Q2_LANE0 is enabled to expose target to the Transceiver interface and PLL0_NW is enabled to expose target to the CCC interface, as shown in the following figure.

Figure 1-6. PF_DRI XCVR Configuration

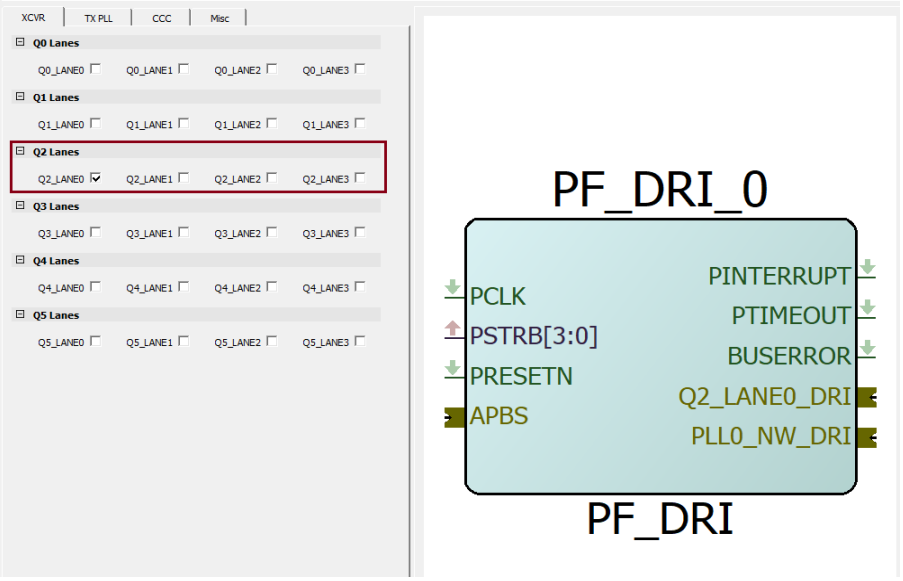
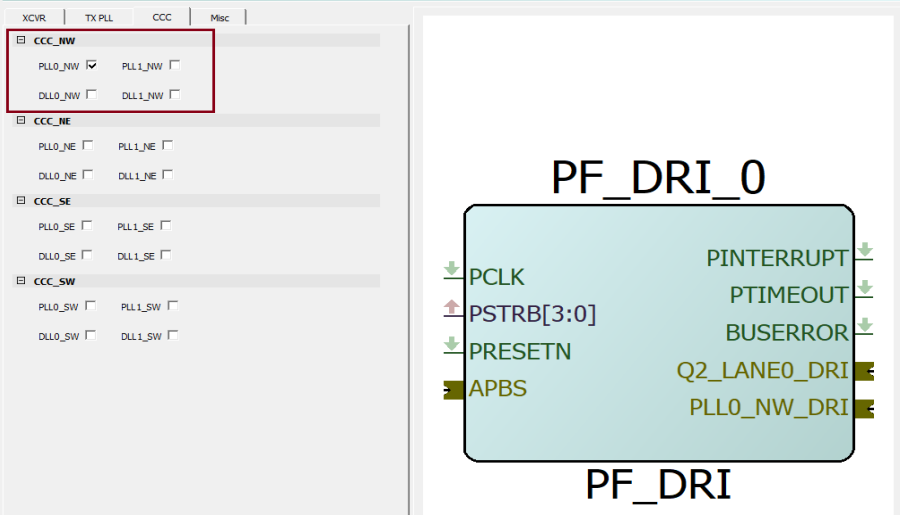


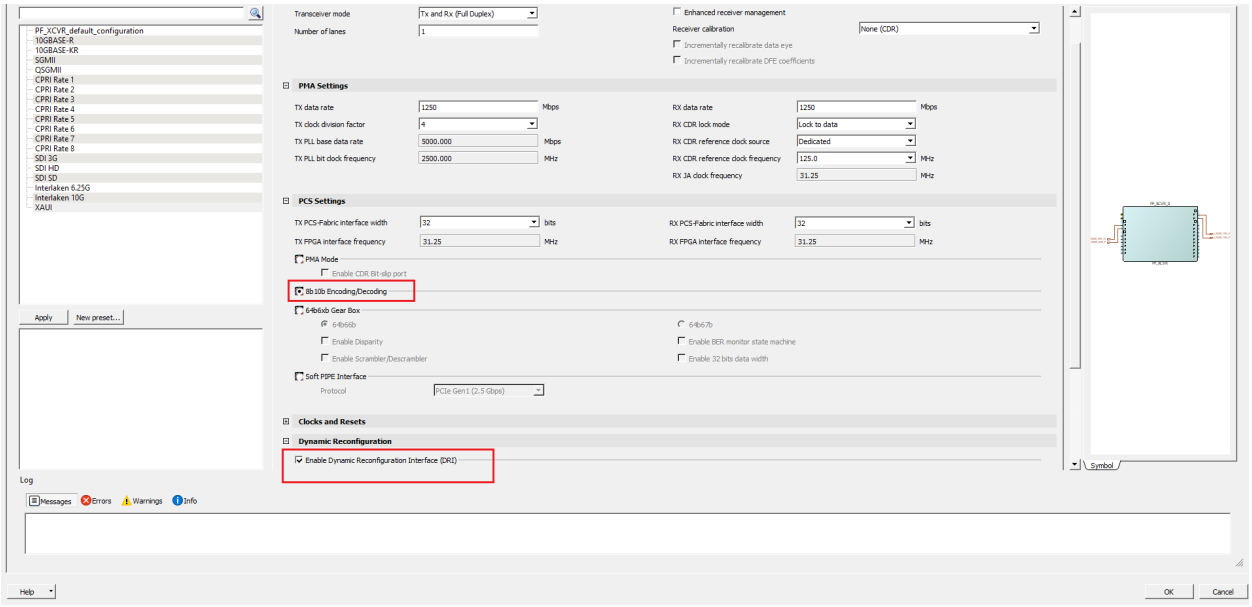
Figure 1-7. PF_DRI CCC Configuration



1.3.2.6 Transceiver Interface Reconfiguration (Ask a Question)

The PolarFire transceiver interface configurator is set to 1.25 Gbps, 32-bit PCS-Fabric interface width. The following figure shows the PolarFire Transceiver Interface configurator settings and how to enable DRI.

Figure 1-8. PolarFire Transceiver Reconfiguration GUI



1.4 Port Description (Ask a Question)

The following table lists the key signals for this design.

Table 1-2. Port Description

Signal	Direction	Description
REF_CLK_PAD_P and REF_CLK_PAD_N	—	Differential reference clock is generated from the on-board 125 MHz oscillator
LANE0_RXD_N	Input	Transceiver receiver differential input
LANE0_RXD_P	Input	Transceiver receiver differential input

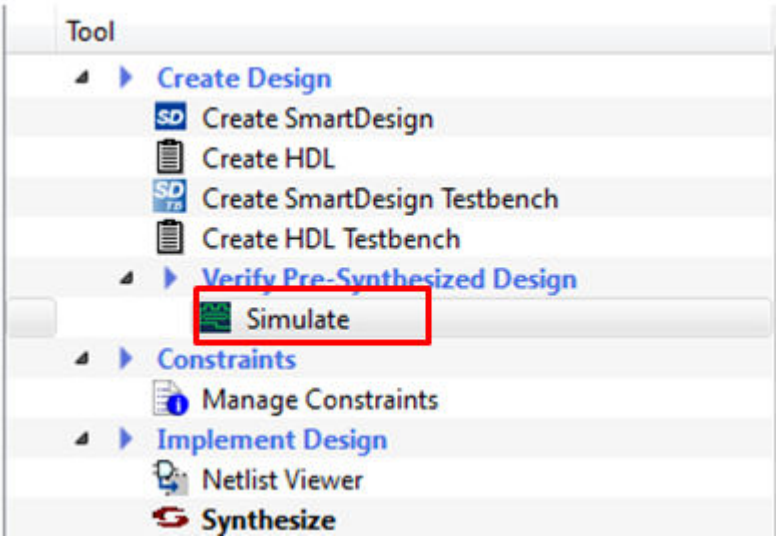
.....continued		
Signal	**Direction**	**Description**
LANE0_TXD_N	Output	Transceiver transmitter differential output
LANE0_TXD_P	Output	Transceiver transmitter differential output
LANE0_RX_READY	Output	Asserted when the CDR is phase-locked to the incoming data transitions and the de-serializer is powered-up.
LANE0_TX_CLK_STABLE	Output	Transmit transceiver/PCS lane ready flag. This flag is asserted when the transmit PLL is locked to the reference clock.
LANE0_RX_VAL	Output	RX_VAL indicates that the XCVR data path is initialized.
RX_CLK_R	Output	Global or regional receive clock to the fabric
TX_CLK_R	Output	Global or regional transmit clock to the fabric
OUT0_FABCLK_0	Output	Dynamic CCC OUT0 Fabric Clock
OUT1_FABCLK_0	Output	Dynamic CCC OUT1 Fabric Clock
OUT2_FABCLK_0	Output	Dynamic CCC OUT2 Fabric Clock
OUT3_FABCLK_0	Output	Dynamic CCC OUT3 Fabric Clock

1.5 Simulating the Design (Ask a Question)

For simulation, perform the following the steps:

1. Open Libero Soc design built through the tcl scripts.
2. To run the simulation, go to **Stimulus Hierarchy**, right click on the `DRI_top_tb` file and select **Set as active stimulus**.
3. In the **Design Flow** tab, double click **Simulate** under **Verify Pre-Synthesized Design** to simulate the design.

Figure 1-9. Simulating the Design



1.6 Simulation Flow (Ask a Question)

The following steps describe the simulation flow:

1. At the start, the transceiver is kept at reset.
2. Once the device initialization is done, the transceiver is brought out of reset.
3. Constant K28.5 character is provided to transceiver.
4. The transmitter lanes are connected to receiver lanes internally in the testbench stimulus.

5. The transceiver is reconfigured at 1.25G by providing a pulse on Switch_1_25_G signal.
6. The transceiver is reconfigured at 2.5G by providing a pulse on Switch_2_5_G signal. See Figure 1-11.
7. The transceiver is reconfigured at 5G by providing a pulse on Switch_5_G signal.
8. The OUT0 and OUT1 of Dynamic CCC are reconfigured by providing a pulse on Switch_CCC_OUT0_OUT1. See Figure 1-12.
9. The OUT2 and OUT3 of Dynamic CCC are reconfigured by providing a pulse on Switch_CCC_OUT2_OUT3. See Figure 1-12.
10. The status signal denotes which reconfiguration is used currently.

Important: When the pulse is provided on Switch_1_25_G, Switch_2_5_G, and Switch_5_G signals, the RX_valid and RX_READY signals are deasserted and asserted once the reconfiguration is complete where as RX_IDLE (active-low) is asserted indicating inactivity on the RX data path and deasserted once the reconfiguration is complete.

Figure 1-10. Top Level Simulation

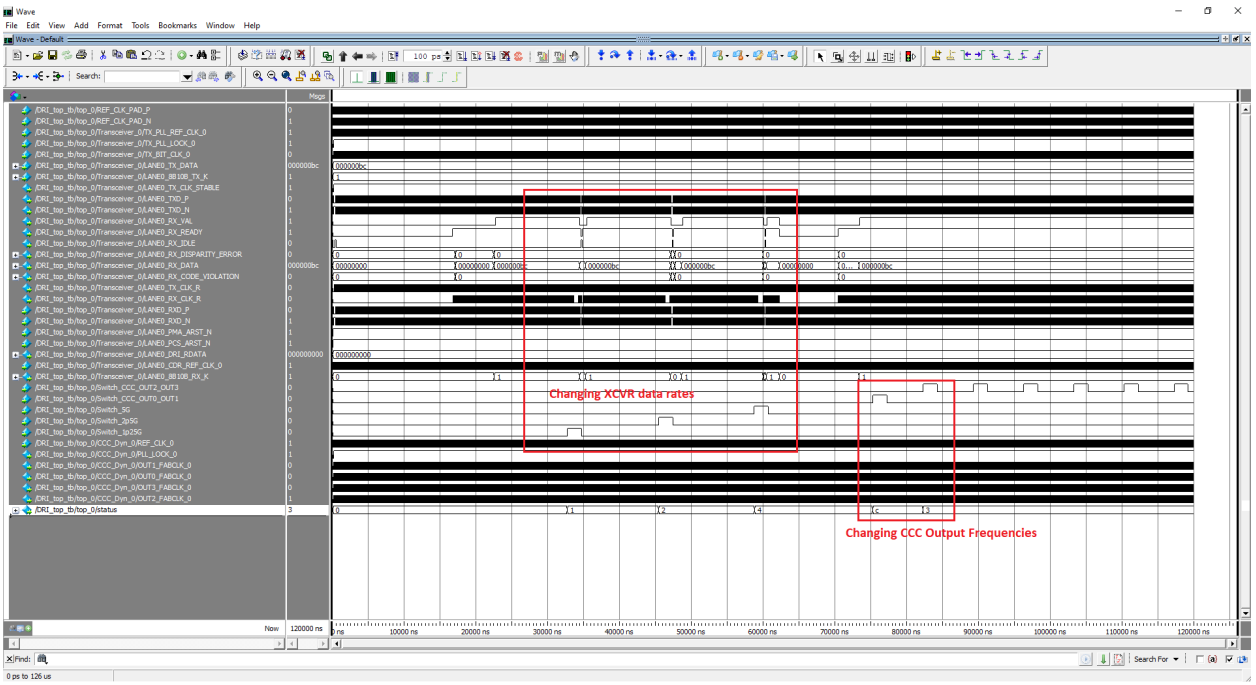


Figure 1-11. 2.5G Transceiver Simulation

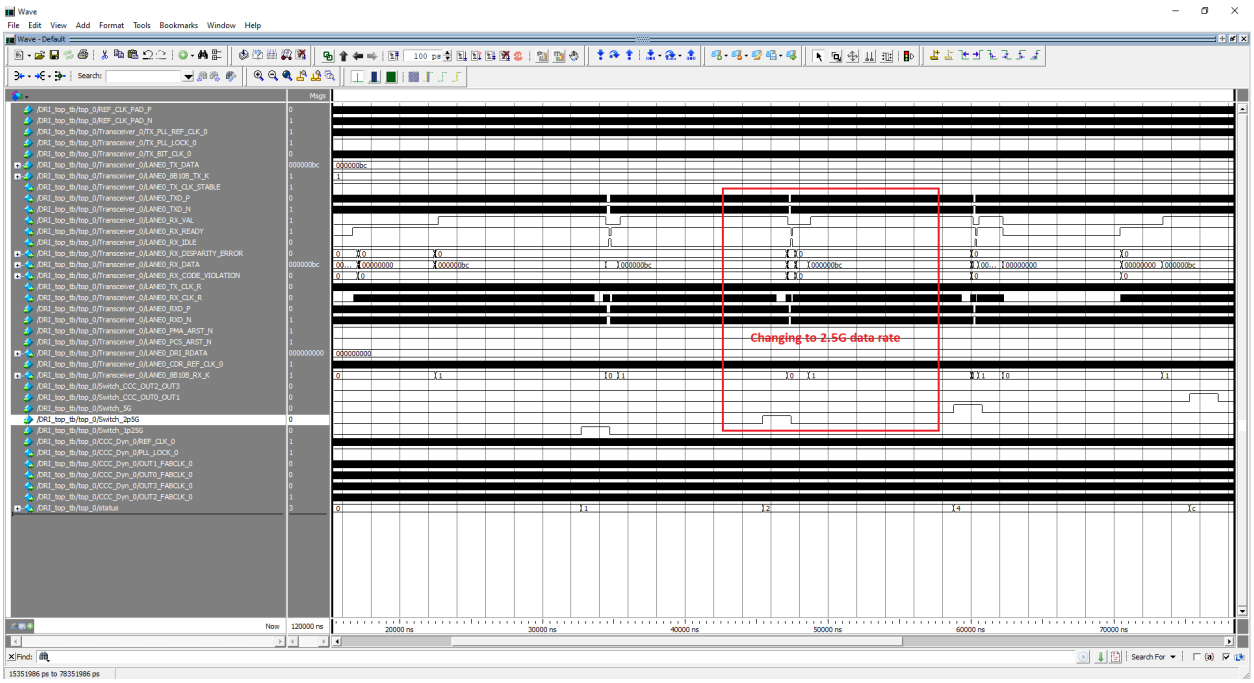
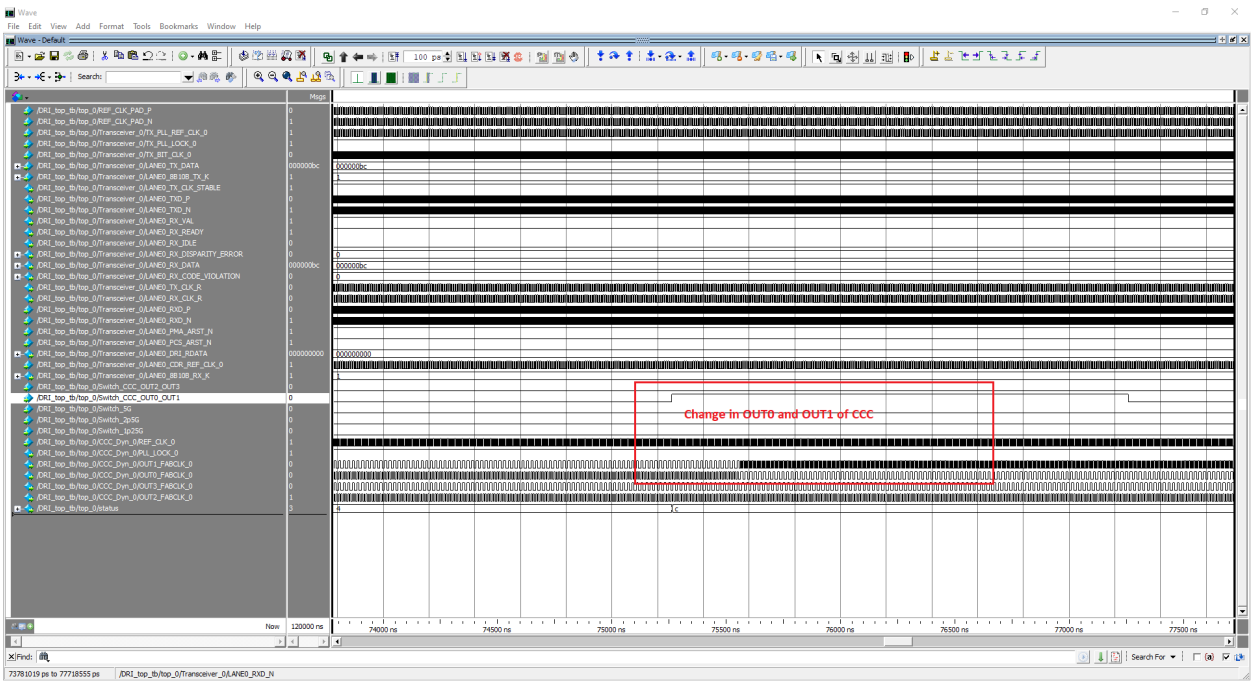


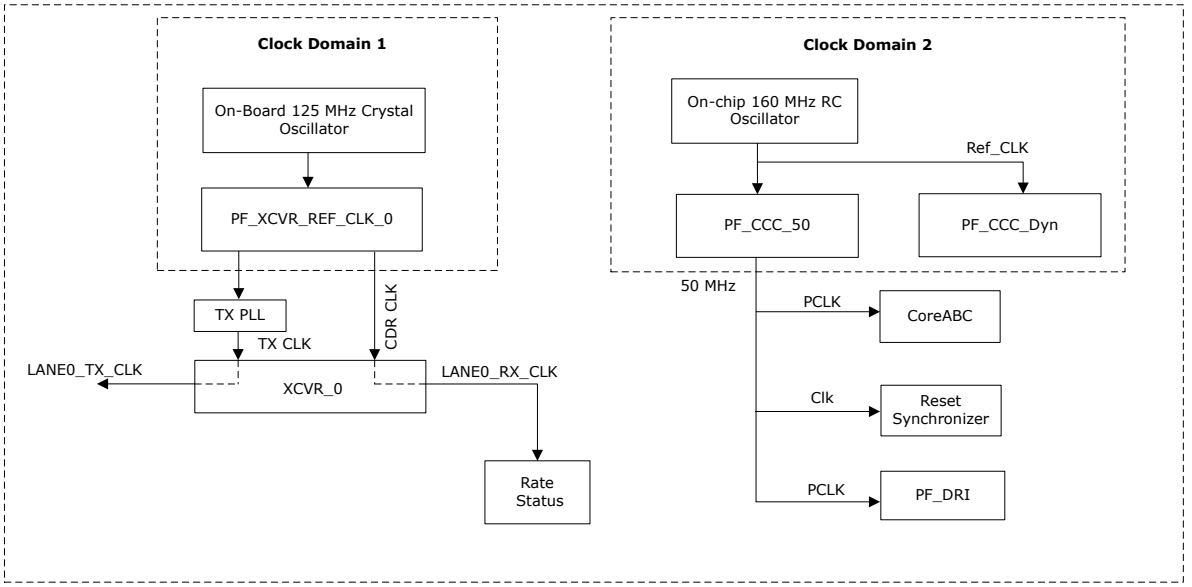
Figure 1-12. CCC Output Frequency Simulation



1.7 Clocking Structure (Ask a Question)

In this demo design, there are two clock-domains. The on-board 125 MHz crystal oscillator drives the XCVR reference clock. The XCVR REFCLK source the transceivers and global clock network. The on-chip 160 MHz RC oscillator drives the CoreABC and PF_DRI block. The following figure shows the clocking structure of the design.

Figure 1-13. Clocking Structure



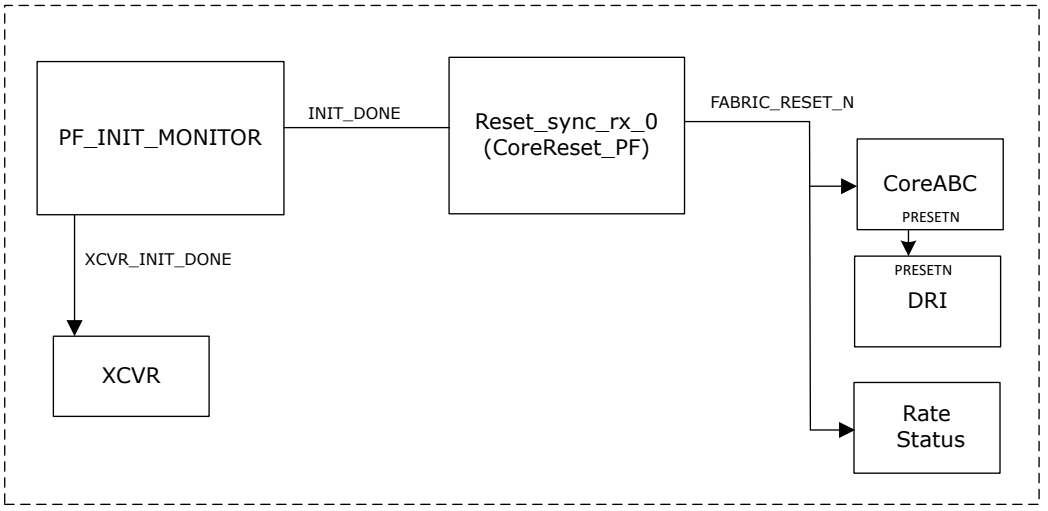
1.8 Reset Structure (Ask a Question)

When PLL_LOCK output from CCC_50 block and DEVICE_INIT_DONE signal from PF_INIT_MONITOR block are asserted, the Reset_Synchronizer_0 (CoreReset_PF) module releases active low reset of Soft_Processor_0 (CoreABC), reconfiguration_controller_0 (PF_DRI), and the Rate_status_0 modules.

DEVICE_INIT_DONE signal is asserted when the device initialization is complete. For more information about device initialization, see PolarFire Family Power-Up and Resets User Guide. For more information on CoreReset_PF IP core, see the *CoreReset_PF handbook* from the Libero catalog.

The following figure shows the reset structure of the design.

Figure 1-14. Reset Structure



Important:

- The DRI PCLK must be gated off until a valid source clock is present and device initialization is completed. It is recommended to use an RGCLKINT macro when external clock source is used. The gate enable signal must be qualified by the DEVICE_INIT_DONE and AUTO_CALIB_DONE outputs of the PF_INIT_MONITOR IP.
- In this design, INIT MONITOR and CoreReset_PF are used for stable clock and reset generation.

2. Run PROGRAM Action (Ask a Question)

Once the design is created through TCL scripts, as mentioned in Appendix B: Running the TCL Script, open the design in Libero® SoC software.

To program the PolarFire® device, perform the following steps:

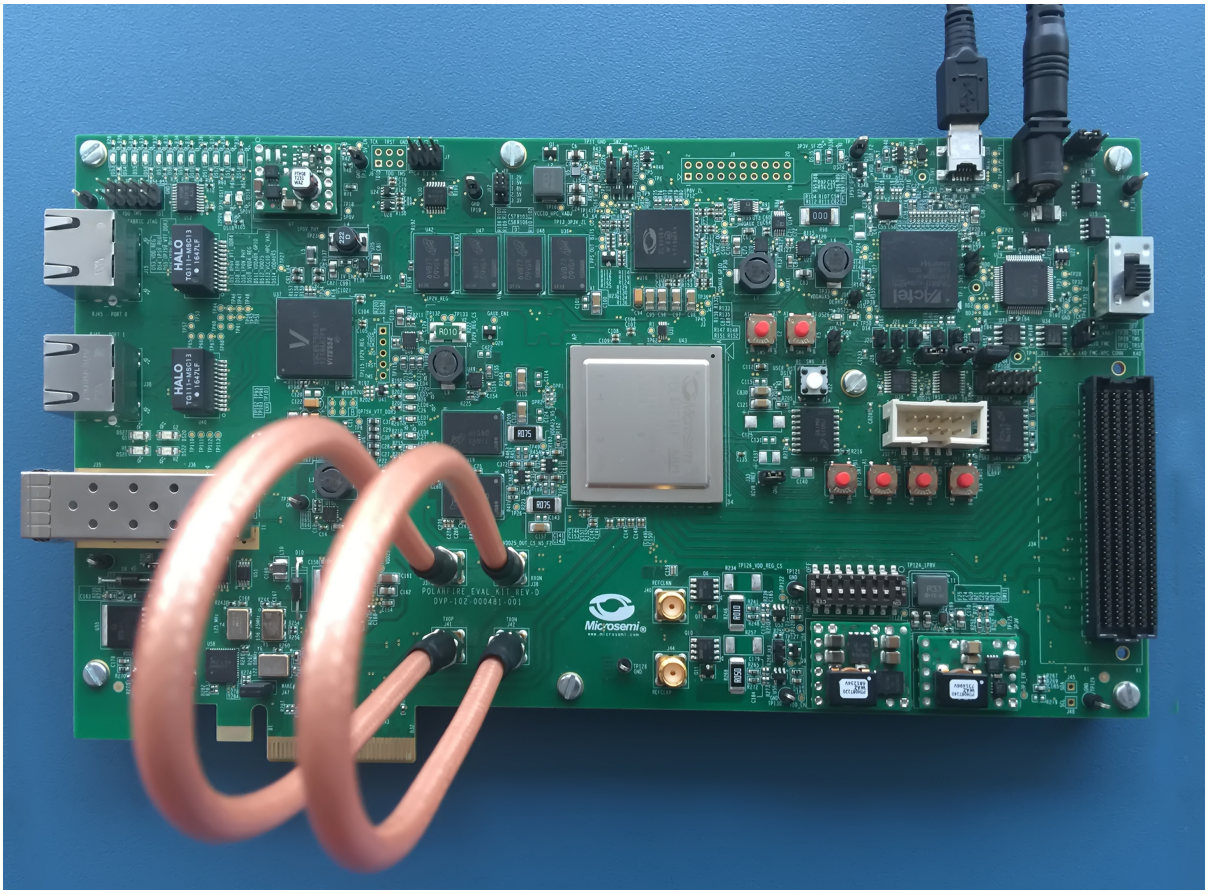
1. Ensure that the following jumper settings are set on the board.

Table 2-1. Jumper Settings on Evaluation Kit

Jumper	Description
J18, J19, J20, J21, and J22	Short pins 2 and 3 for programming the PolarFire FPGA through FTDI
J28	Short pins 1 and 2 for programming through the onboard FlashPro5
J26	Short pins 1 and 2 for programming through the FTDI SPI
J27	Short pins 1 and 2 for programming through the FTDI SPI
J4	Short pins 1 and 2 for manual power switching using SW3
J12	Short pins 3 and 4 for 2.5V
J46	• Short pin 1 and 2 for routing 125 MHz differential clock oscillator output to the line side • Open pin 1 and 2 for routing 122.88 MHz differential clock oscillator output to the line side

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. Connect TXN to RXN and TXP to RXP using the two SMA to SMA cables, as shown in the following figure.
   The following figure shows the PolarFire Evaluation kit board setup. For information about PolarFire Evaluation kit, see PolarFire FPGA Evaluation Kit User Guide.

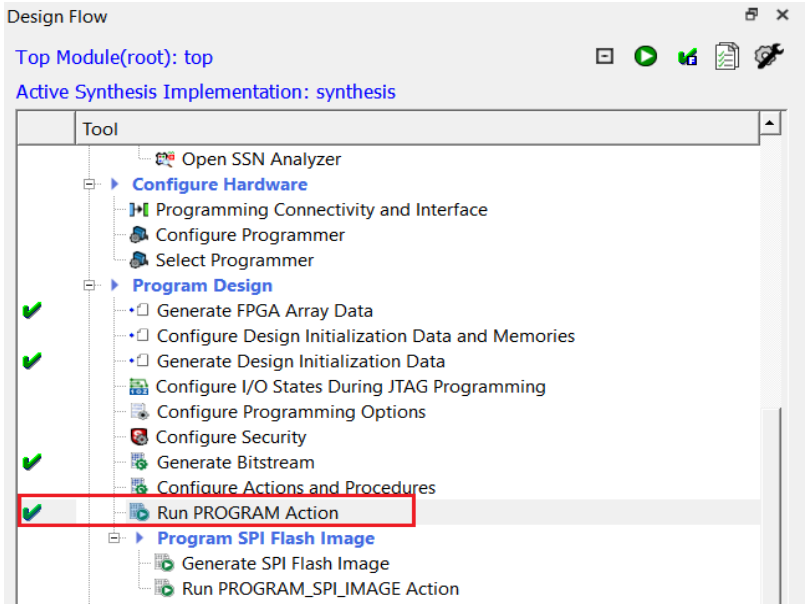
Figure 2-1. Board Setup



6. Double click **Run PROGRAM Action** from the **Libero Design** Flow.

The device is now successfully programmed and the onboard LEDs 4, 5, 6, and 7 start glowing. A green tick mark appears next to **Run PROGRAM Action**, as shown in the following figure.

Figure 2-2. Run PROGRAM Action

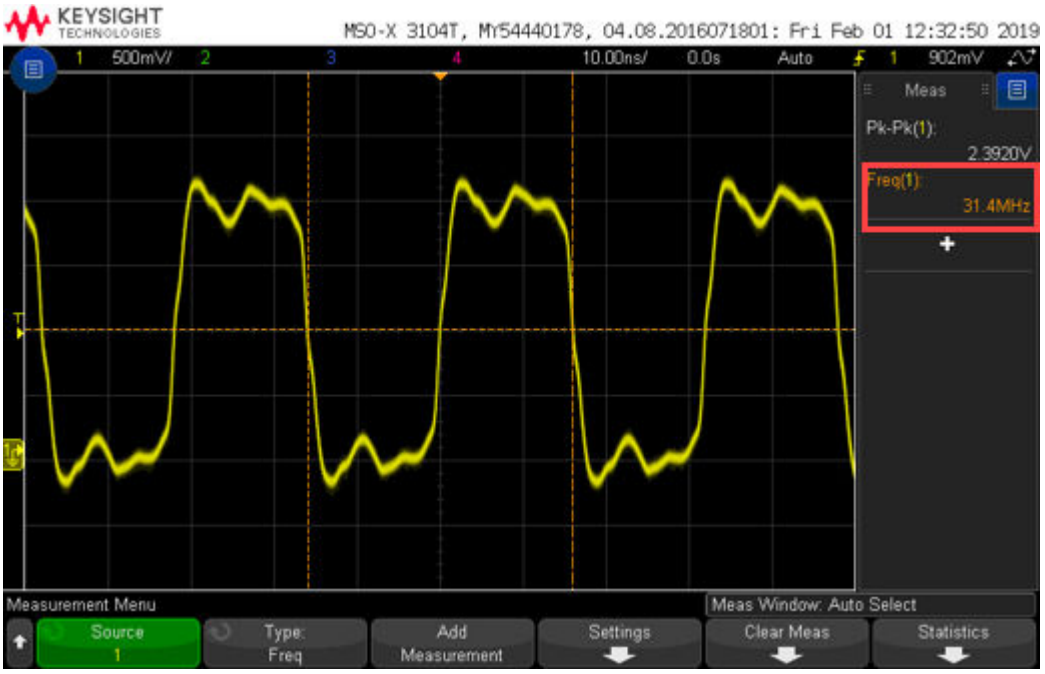


3. Running the Demo (Ask a Question)

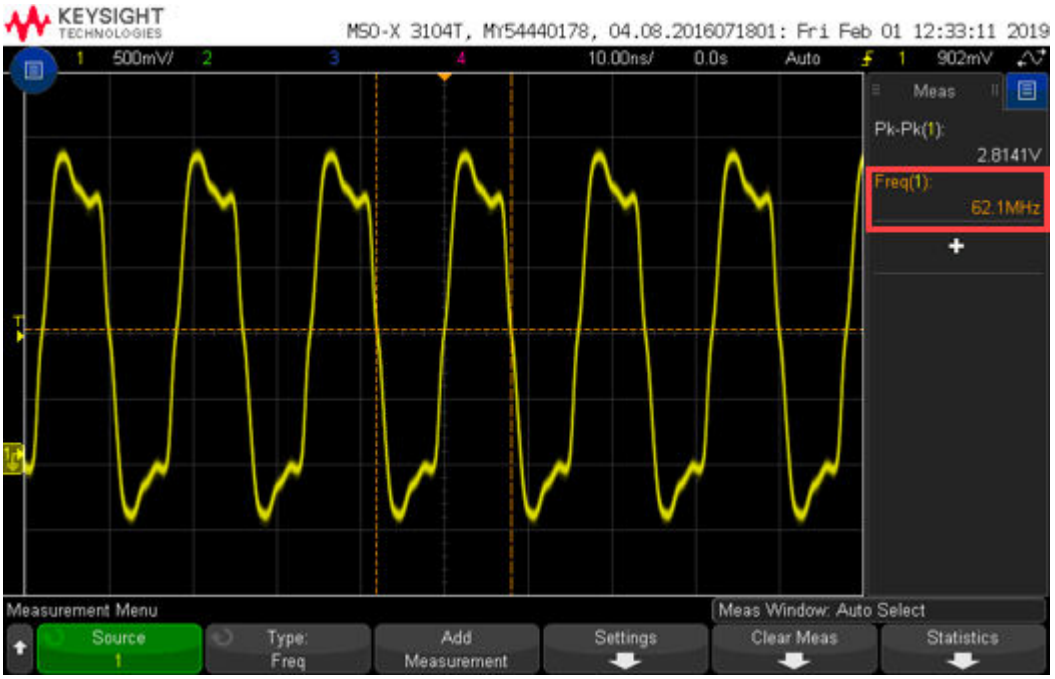
This section describes how to run the DRI demo, check the results on the board, and observe the frequency on an oscilloscope.

To run the demo, setup the PolarFire Evaluation board as explained in Run PROGRAM Action. This programs the evaluation kit with DRI design and ensures that all DIP switches of SW11 are ON. Execute the following tests:

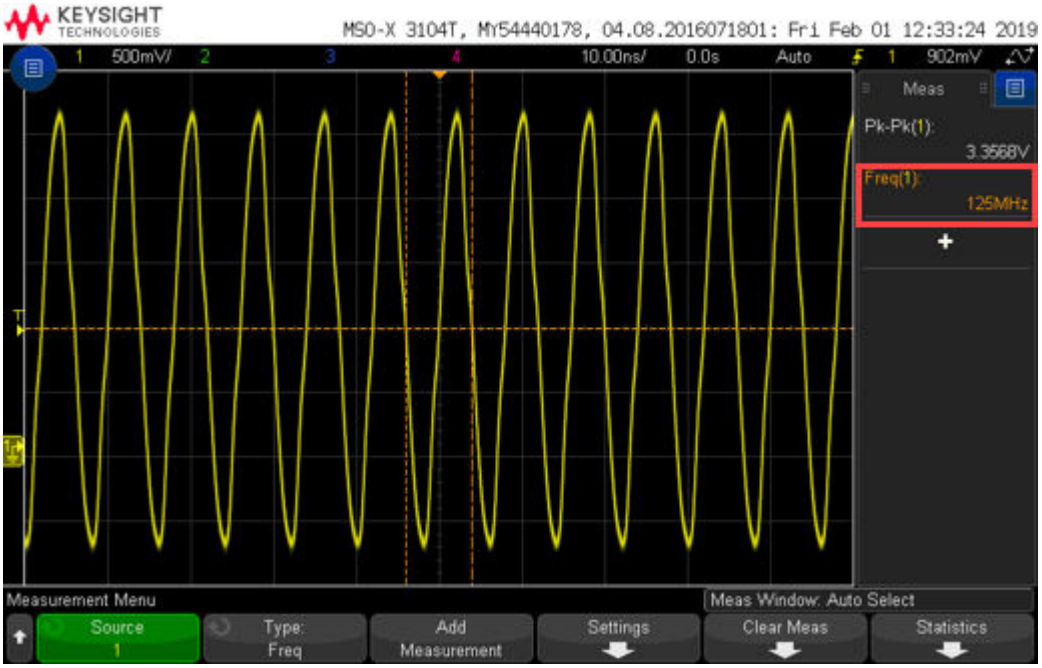
TEST 1: Switch OFF **SW11 DIP1** and change it back to ON to configure the transceiver in 1.25G at 31.25 MHz of TX and RX Clock frequency.



TEST 2: Switch OFF **SW11 DIP2** and change it back to ON to configure the transceiver in 2.5G mode at 62.5 MHz of TX and RX Clock frequency.



TEST 3: Switch OFF **SW11 DIP3** and change it back to ON to configure the transceiver in 5G mode at 125 MHz of TX and RX Clock frequency.



TEST 4: Switch **OFF DIP5** and change it back to ON to configure OUT0 and OUT1 frequencies of Dynamic CCC.

TEST 5: Switch **OFF DIP6** and change it back to ON to configure OUT2 and OUT3 frequencies of Dynamic CCC.

Important: The DIP switches are active-low on the Evaluation Kit. If more than one DIP switch is high, DIP1 has higher precedence over DIP2 and the XCVR ise configured for 1.2G, and so on.

The following table lists the status of LEDs as per selected inputs.

Table 3-1. Output Status-LED

Test No.	Input/Output	LED4	LED5	LED7	LED8	LED9	LED10	LED11
		TX_CLK_STABLE	RX_READY	RX_VAL	Status			
TEST 1	DIP1-1.25G	ON	ON	ON	ON	OFF	OFF	OFF
TEST 2	DIP2-2.5G	ON	ON	ON	OFF	ON	OFF	OFF
TEST 3	DIP3-5G	ON	ON	ON	OFF	OFF	ON	OFF
TEST 4	DIP5-CCC OUT0-OUT1	ON	ON	ON	OFF	OFF	ON	ON
TEST 5	DIP6-CCC OUT2-OUT3	ON	ON	ON	ON	ON	OFF	OFF

Probe internal signals of the design from the PolarFire FPGA through test points to get the output frequency. Table 3-2 lists the output signals and probing points to observe the clock changes on an oscilloscope.

Important: For more information about the Jumper locations on the board, see the silkscreen in PolarFire FPGA Evaluation Kit User Guide.

Table 3-2. Output Status-Probe Test Points

Signal	Header	Probe Test Point
CC-Out2	J7	Pin-4
CC-Out3		Pin-6
CC-Out0	J8	Pin-5
CC-Out1		Pin-3
RX_CLK_R	J8	Pin-9
TX_CLK_R	J8	Pin-13

3.1 PowerMonitor (Ask a Question)

PolarFire Evaluation Kit board comes with power monitoring solution, implemented using the on-board SmartFusion A2F 200 device and the PowerMonitor application. The PowerMonitor application connects to the power monitoring program running on the A2F 200 device to measure power. For more information about PowerMonitor, see PolarFire FPGA Evaluation Kit User Guide.

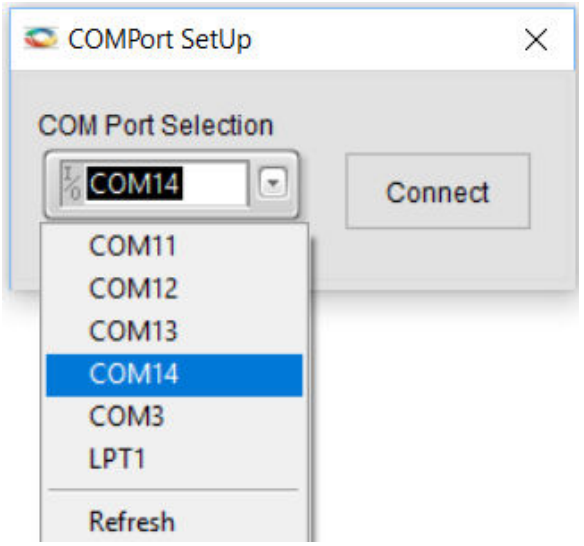
On the host PC, download the Microchip PowerMonitor application from the following location: PolarFire Evaluation Kit PowerMonitor.

After the download is completed, follow the instructions in the installation wizard to install the PowerMonitor application.

To measure the power, perform the following steps:

1. On the host PC desktop, click **Start** and select **PowerMonitor**.
2. The following figure shows the **COMPort SetUp** dialog box. Select the highest COM port from the drop-down and click **Connect**.

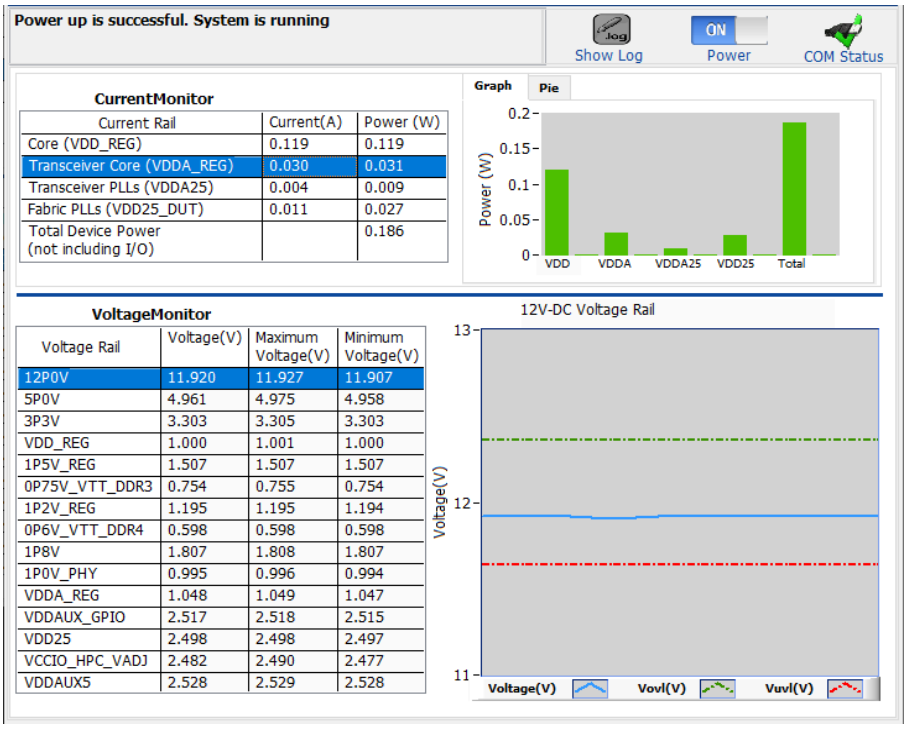
Figure 3-1. COMPort Setup



The PowerMonitor application successfully connects to the board and starts displaying the Core Fabric (VDD) power, Fabric PLL (VDD25) power, Transceiver Core (VDDA) power, and Transceiver PLL (VDDA25) power.

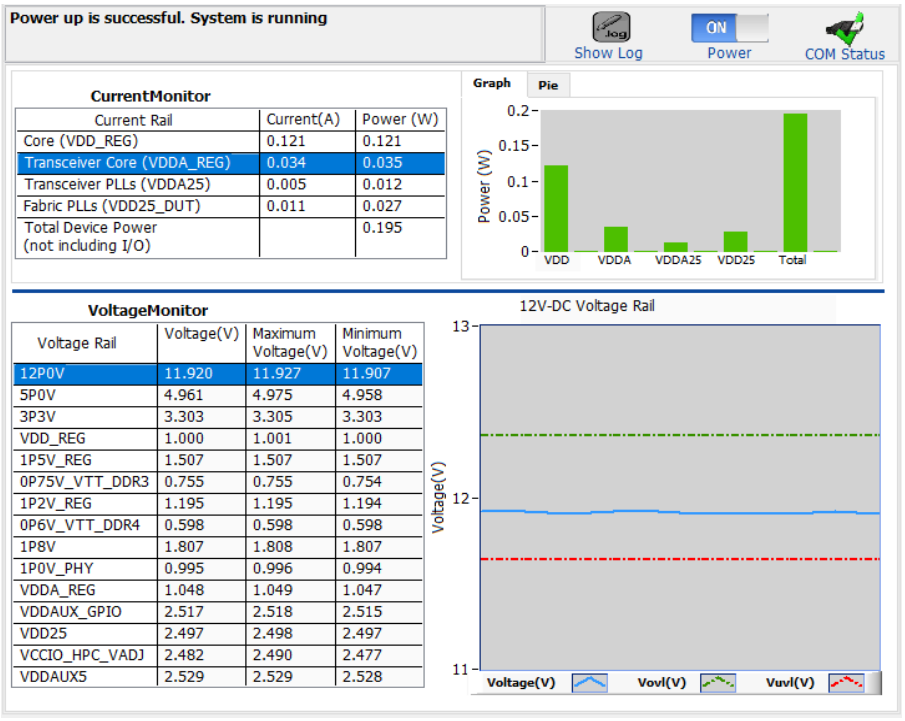
The following figure shows the total power consumed by the device is at 1.25G.

Figure 3-2. Power Usage at 1.25G



The following figure shows the total power consumed by the device is at 2.5G.

Figure 3-3. Power Usage at 2.5G



The following figure shows the total power consumed by the device is at 5G.

Figure 3-4. Power Usage at 5G

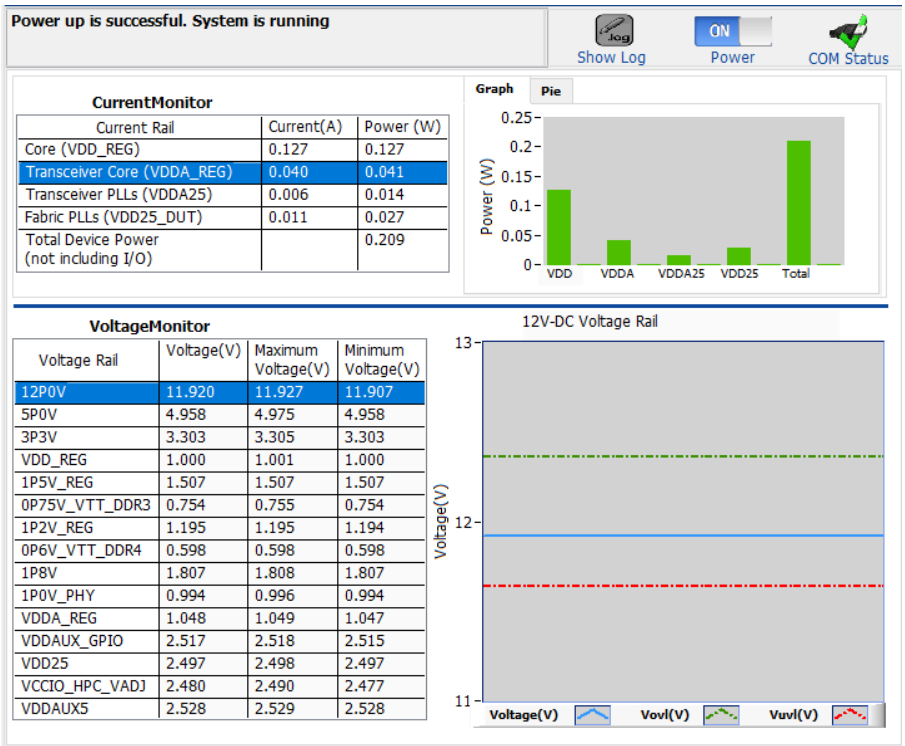


Table 3-3. Transceiver Power

S.No	Lane Rate	Transceiver Power (W)	Total Power (W)
1	1.25	0.031	0.186
2	2.5	0.035	0.195
3	5	0.041	0.209

This concludes Dynamic Reconfiguration of XCVR and CCC.

4. Appendix A: Programming the Device Using FlashPro Express (Ask a Question)

This section describes how to program the PolarFire device with the `.job` programming file using FlashPro Express. The `.job` file is available at the following design files folder location.

`mpf_an4592_df\Programming_Files`

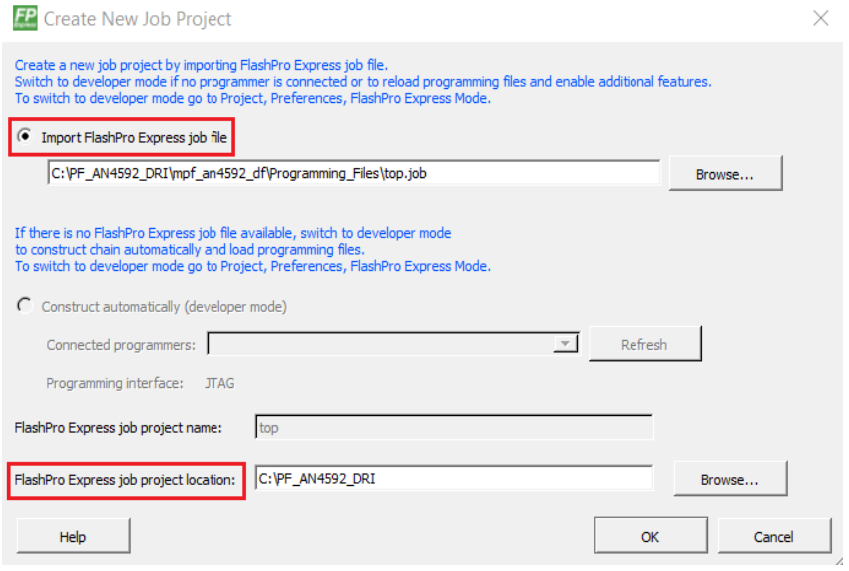
To program the device, perform the following steps:

1. Ensure that the jumper settings on the board are the same as listed in Table 2-1.

> **Important:** 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. Connect TXN to RXN and TXP to RXP using 2 SMA to SMA cables.
6. On the host PC, launch the **FlashPro Express software**.
7. Click **New** or select **New Job Project from FlashPro Express Job** from **Project** menu to create a new job project.
8. Enter the following in the **New Job Project from FlashPro Express Job** dialog box:
   - **Programming job file**: Click **Browse**, navigate to the location where the `.job` file is located, and select the file. The default location is: `<download_folder>\mpf_an4592_df\Programming_Files`.
   - **FlashPro Express job project location**: Click **Browse** and navigate to the location to save the project.

Figure 4-1. New Job Project from FlashPro Express Job



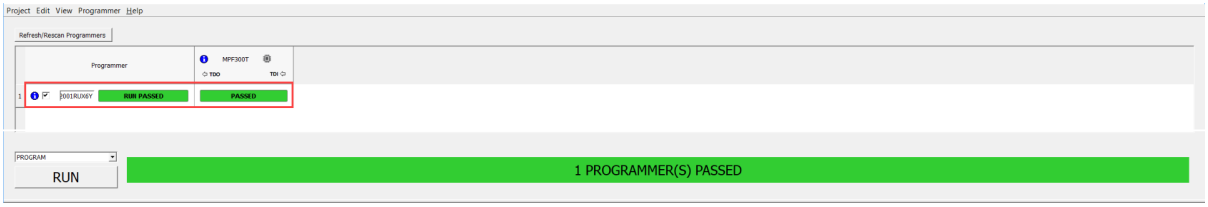
9. Click **OK**. The required programming file is selected and ready to be programmed in the device.

10. The FlashPro Express window appears. Confirm that a programmer number appears in the Programmer field. If it does not, then confirm the board connections and click **Refresh/Rescan Programmers**.

Figure 4-2. Programming the Device

11. Click **RUN**. When the device is programmed successfully, a **RUN PASSED** status is displayed and the on-board LEDs 4, 5, 6, and 7 glow. See Running the Demo to run the TVS demo.

Figure 4-3. FlashPro Express—RUN PASSED



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

5. Appendix B: Running the TCL Script (Ask a Question)

TCL scripts are provided in the design files folder under directory HW. If required, the design flow can be reproduced from Design Implementation till generation of job file.

To run the TCL, perform the following steps:

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

After successful execution of TCL script, Libero project is created within HW directory. For more information about TCL scripts, see `mpf_an4592_df\HW\TCL_Script_readme.txt`.

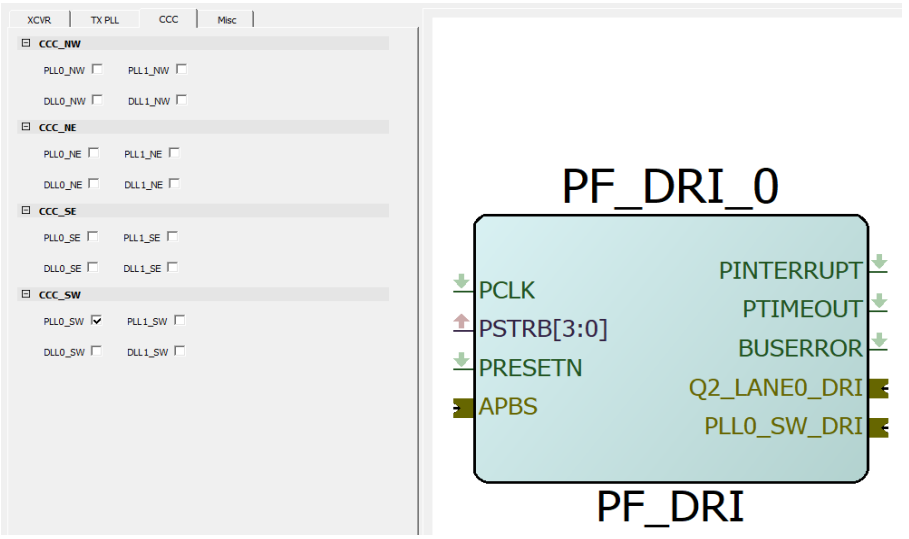
For more details on TCL commands, see Tcl Commands Reference Guide. Contact Technical Support for any queries encountered when running the TCL script.

6. Appendix C: PLL or DLL Placement with DRI (Ask a Question)

To configure the PLL and DLL placement with Dynamic Reconfiguration Interface (DRI) Configurator, follow these steps:

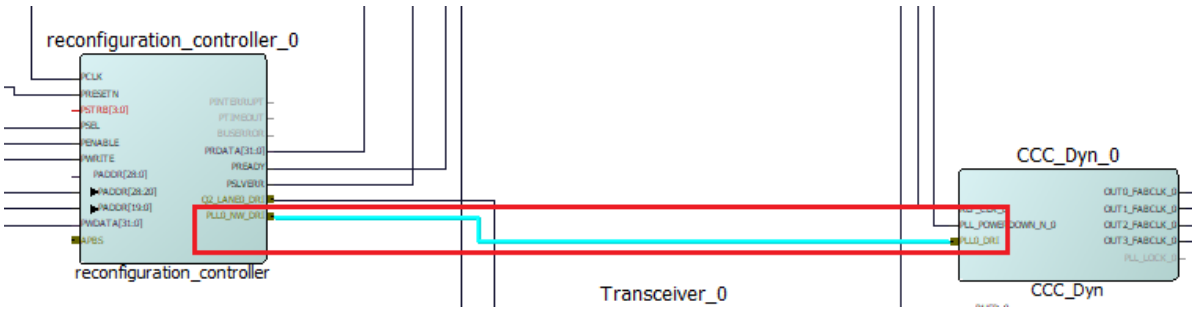
1. Select the PLL location in the **CCC** tab of the DRI configurator, as shown in the following figure. In this case, PLL is selected in the southwest corner of the CCC.

Figure 6-1. Dynamic Reconfiguration Interface Configurator—CCC Tab



Based on PLL location selected in the configurator, the connection between PF_DRI (reconfiguration controller) and PF_CCC (CCC_Dyn) is updated in SmartDesign, as shown in following figure.

Figure 6-2. Connection between PF_CCC and PF_DRI

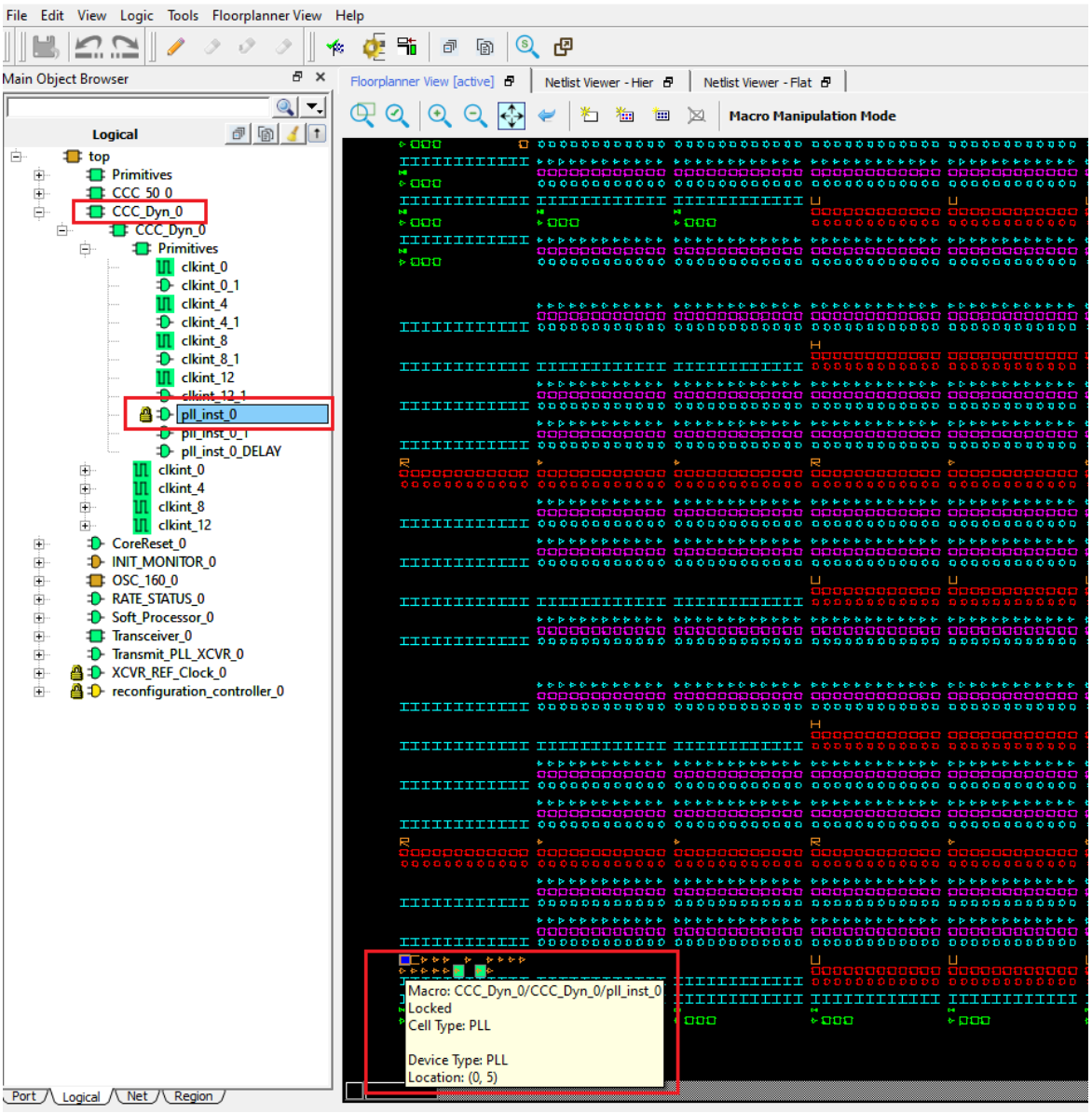


2. To place **PLL**, add fp constraints with the selected PLL location using the following PDC command: `set_location -inst_name CCC_Dyn_0/CCC_Dyn_0/pll_inst_0 -location PLL0_SW`
3. To place **DLL**, use the following PDC command: `set_location -inst_name PF_CCC_0/dll_inst_0 -location DLL0_SE`
4. These PDC commands must be added to the `.PDC` file and included in the design through Constraint Manager. To run place and route, these fp constriants must be enabled from Constraint Manager.

Important: After the placement of PLL and DLL using preceding PDC commands, the PDC file can be viewed from **Constraint Manager** > **Floor Planner** at the following location: `HW\src\constraints\fp_constraints.pdc`.

5. Run **Place and Route**. After place and route is completed, open the chip planner to verify the location of the selected PLL. In this case, click **CCC_Dyn_0** in the left pane to verify the location of the PLL, as shown in the following figure.

Figure 6-3. CCC_Dyn_0 PLL Location



Important: If DRI is not used, then the user need not manually place the PLL or the DLL.

7. Revision History (Ask a Question)

The revision history describes the changes that were implemented in the document. The changes are listed by revision, starting with the current publication.

Table 7-1. Revision History

Revision	Date	Description
D	11/2024	The following is the list of changes made in revision D of the document: • Updated the document for Libero v2024.1. • Updated the `.job` filepath and TCL script filepath throughout the document. • Updated Figure 1-6 and Figure 1-7 in the Dynamic Reconfiguration Interface section. • Added a Note in the Reset Structure section. • Added Figure 2-2 and updated Run PROGRAM Action section. • Updated the link for downloading PowerMonitor in the PowerMonitor section. • Updated Figure 6-1 in the Appendix C: PLL or DLL Placement with DRI section.
C	02/2024	Added section.
B	03/2023	Added a note in the Introduction about SmartDebug and API DRI accesses.
A	07/2022	The following is the list of changes in revision A of the document: • The document was migrated to the Microchip template. • The document number was updated from 51900475 to DS00004592A. • The document ID was updated to AN4592 from AC475. • Updated Figure 1-1. • Updated Figure 1-2. • Updated Figure 1-3. • Updated Figure 1-6 and Figure 1-7. • Updated Figure 1-8. • Updated Figure 1-10, Figure 1-11, and Figure 1-12. • Updated Figure 1-14. • Updated Figure 3-2, Figure 3-3, and Figure 3-4. • Updated Table 3-3. • Updated the simulation flow steps. • Updated Simulating the Design.
6.0	—	Added Appendix B: Running the TCL Script.
5.0	—	The following is thw summary of the changes made in this revision. • Updated the document for Libero SoC v12.2. • Removed the references to Libero version numbers.
4.0	—	Updated the document for Libero SoC v12.1 release.
3.0	—	Updated the document for Libero SoC v12.0 release.
2.0	—	Updated the document for Libero SoC PolarFire v2.3 release.
1.0	—	The first publication of this document.

Microchip FPGA Support

Microchip FPGA products group backs its products with various support services, including Customer Service, Customer Technical Support Center, a website, and worldwide sales offices. Customers are suggested to visit Microchip online resources prior to contacting support as it is very likely that their queries have been already answered.

Contact Technical Support Center through the website at www.microchip.com/support. Mention the FPGA Device Part number, select appropriate case category, and upload design files while creating a technical support case.

Contact Customer Service for non-technical product support, such as product pricing, product upgrades, update information, order status, and authorization.

- From North America, call **800.262.1060**
- From the rest of the world, call **650.318.4460**
- Fax, from anywhere in the world, **650.318.8044**

Microchip Information

Trademarks

The “Microchip” name and logo, the “M” logo, and other names, logos, and brands are registered and unregistered trademarks of Microchip Technology Incorporated or its affiliates and/or subsidiaries in the United States and/or other countries (“Microchip Trademarks”). Information regarding Microchip Trademarks can be found at https://www.microchip.com/en-us/about/legal-information/microchip-trademarks.

ISBN: 979-8-3371-0090-6

Legal Notice

This publication and the information herein may be used only with Microchip products, including to design, test, and integrate Microchip products with your application. Use of this information in any other manner violates these terms. Information regarding device applications is provided only for your convenience and may be superseded by updates. It is your responsibility to ensure that your application meets with your specifications. Contact your local Microchip sales office for additional support or, obtain additional support at www.microchip.com/en-us/support/design-help/client-support-services.

THIS INFORMATION IS PROVIDED BY MICROCHIP “AS IS”. MICROCHIP MAKES NO REPRESENTATIONS OR WARRANTIES OF ANY KIND WHETHER EXPRESS OR IMPLIED, WRITTEN OR ORAL, STATUTORY OR OTHERWISE, RELATED TO THE INFORMATION INCLUDING BUT NOT LIMITED TO ANY IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY, AND FITNESS FOR A PARTICULAR PURPOSE, OR WARRANTIES RELATED TO ITS CONDITION, QUALITY, OR PERFORMANCE.

IN NO EVENT WILL MICROCHIP BE LIABLE FOR ANY INDIRECT, SPECIAL, PUNITIVE, INCIDENTAL, OR CONSEQUENTIAL LOSS, DAMAGE, COST, OR EXPENSE OF ANY KIND WHATSOEVER RELATED TO THE INFORMATION OR ITS USE, HOWEVER CAUSED, EVEN IF MICROCHIP HAS BEEN ADVISED OF THE POSSIBILITY OR THE DAMAGES ARE FORESEEABLE. TO THE FULLEST EXTENT ALLOWED BY LAW, MICROCHIP’S TOTAL LIABILITY ON ALL CLAIMS IN ANY WAY RELATED TO THE INFORMATION OR ITS USE WILL NOT EXCEED THE AMOUNT OF FEES, IF ANY, THAT YOU HAVE PAID DIRECTLY TO MICROCHIP FOR THE INFORMATION.

Use of Microchip devices in life support and/or safety applications is entirely at the buyer’s risk, and the buyer agrees to defend, indemnify and hold harmless Microchip from any and all damages, claims, suits, or expenses resulting from such use. No licenses are conveyed, implicitly or otherwise, under any Microchip intellectual property rights unless otherwise stated.

Microchip Devices Code Protection Feature

Note the following details of the code protection feature on Microchip products:

- Microchip products meet the specifications contained in their particular Microchip Data Sheet.
- Microchip believes that its family of products is secure when used in the intended manner, within operating specifications, and under normal conditions.
- Microchip values and aggressively protects its intellectual property rights. Attempts to breach the code protection features of Microchip products are strictly prohibited and may violate the Digital Millennium Copyright Act.
- Neither Microchip nor any other semiconductor manufacturer can guarantee the security of its code. Code protection does not mean that we are guaranteeing the product is “unbreakable”. Code protection is constantly evolving. Microchip is committed to continuously improving the code protection features of our products.