DG0885
Demo Guide
PolarFire FPGA USXGMII Design

Microsemi
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
www.microsemi.com

©2019 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.

50200885. 1.0 12/19

Contents

1 Revision History 4
1.1 Revision 1.0 4
2 PolarFire FPGA USXGMII Design 5
2.1 Demo Requirements 6
2.2 Prerequisites 6
2.3 Demo Design Architecture 7
2.3.1 I/O Ports 7
2.3.2 Subsystem Components 8
2.3.3 Clocking Structure 12
2.3.4 Reset Structure 13
2.3.5 Resource Utilization 13
3 Setting Up the Demo 14
3.1 Setting Up the Hardware 14
3.2 Programming the PolarFire Device 15
3.2.1 Programming the Device Using FlashPro Express 15
3.2.2 Programming the Device Using Libero SoC 17
4 Running the Demo 20

Figures

Figure 1 USXGMII Design Architecture . . . 7
Figure 2 Mi-V Subsystem . . . 9
Figure 3 Mi-V Soft Processor - DRI - PF_CCC Interface . . . 11
Figure 4 Clocking Structure . . . 12
Figure 5 Board Setup . . . 15
Figure 6 FlashPro Express New Job . . . 15
Figure 7 New Job Project from FlashPro Express Job . . . 16
Figure 8 Programming the Device . . . 16
Figure 9 Run Passed . . . 16
Figure 10 Libero Design Flow . . . 17
Figure 11 Design and Memory Initialization Option . . . 17
Figure 12 Selecting the Logical RAM Instance . . . 18
Figure 13 Edit Fabric RAM Client . . . 18
Figure 14 Import Memory File . . . 19
Figure 15 Apply Configuration . . . 19
Figure 16 Generate Design Initialization Data . . . 19
Figure 17 Initialization Data Generated . . . 19
Figure 18 10GBASE-T Advertisement . . . 20
Figure 19 TeraTerm Configuration . . . 21
Figure 20 UART Message - 1 . . . 21
Figure 21 10G Traffic Report . . . 21
Figure 22 5GBASE-T Advertisement . . . 22
Figure 23 UART Message -2 . . . 22
Figure 24 5G Traffic Report . . . 22
Figure 25 2.5GBASE-T Advertisement . . . 23
Figure 26 UART Message - 3 . . . 23
Figure 27 2.5G Traffic Report . . . 23
Figure 28 1000BASE-T Advertisement . . . 24
Figure 29 UART Message - 4 . . . 24
Figure 30 1000BASE-T Traffic Report . . . 24

Tables

Table 1 Design Requirements 6
Table 2 I/O Ports 7
Table 3 Core10GMAC Registers 9
Table 4 CoreUSXGMII Registers 10
Table 5 PHY Registers 10
Table 6 USXGMII and Core10GMAC Clock Sources 11
Table 7 USXGMII Resource Utilization 13
Table 8 Jumper and Switch Settings 14

1 Revision History

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

1.1 Revision 1.0

The first publication of this document.

2 PolarFire FPGA USXGMII Design

Microchip's PolarFire® FPGAs and Ethernet IPs enable quick development of Ethernet solutions. In the 10G Ethernet segment, the Universal Serial 10G Media Independent Interface (USXGMII) IP core from Microchip enables building 10GBASE-R solutions on PolarFire FPGAs, the IP is complaint with IEEE802.3ae.

USXGMII provides the support for variable data rates of 10G, 5G, 2.5G, and 1G based on the corresponding data rate change at the downstream PHY. Implementing the solution on PolarFire FPGAs offer low-power advantages which include the low-power transceiver and FPGA fabric. The low-power advantage helps saving the power budget of the system.

The USXGMII Ethernet solution is implemented using the CORE10GMAC soft IP media access control (MAC) core configured in XGMII mode, and CoreUSXGMII IP used to carry single network port over a single SERDES between MAC and PHY (Aquantia PHY AQR107).

This document describes the Microchip PolarFire USXGMII design and how to run the demo using the PolarFire Video Kit, Microchip Daughter Card with Aquantia PHY (AQR107), and a USXGMII compliant network module.

The PolarFire USXGMII demo design features:

- 10G Ethernet MAC IP.
- USXGMII IP that provides an XGMII interface with the MAC IP.
- Transceiver connected to a PHY daughter card via FMC at the system side.
- USXGMII Compliant network module at the line side.

The PolarFire Video Kit (DVP-102-000512-001) features:

- A 300K LE FPGA (MPF300T, FCG1152)
- HDMI 1.4 transmitter (ADV7511) chipset and corresponding connector
- HDMI 2.0 with rail clamps, ReDrivers and corresponding connectors
- Dual camera sensor featuring IMX334 Sony image sensor
- Image sensor interface to support upto two MIPI CSI-2 cameras
- DSI Interface
- NVIDIA Jetson Interface (MIPI CSI-2 TX connector)
- A High Pin Count (HPC) FMC connector to connect to high-speed interfaces (like 12G-SDI and USXGMII)

For more information about this video kit, see https://www.microsemi.com/existing-parts/parts/150747.

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

- Using the pre-generated .job file: To program the device using the .job file provided along with the demo design files, see Programming the Device Using FlashPro Express.
- Using Libero SoC: To program the device using Libero SoC, see Programming the Device Using Libero SoC.

2.1 Demo Requirements

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

Table 1 • **Design Requirements**

Design Requirement	Description
-Host PC	A host PC with USB port
Design Software	
Libero SoC	v12.2
Hardware	
-PolarFire VIDEO KIT	DVP-102-000512-001 REV 1.0
-Aquantia PHY daughter card	
-Spirent TestCenter	-An Ethernet test module for 10 GbE USXGMII compliant traffic generation and error analysis -Chassis Model: SPT - N4U
-USB A to mini-B cable	For FPGA programming (Available with the PolarFire Video kit)
-Category 6 (Cat 6) cables with RJ45 connectors	Two cables required for: -Connecting PHY daughter card and test module -Connecting Host PC to LAN
-Power Adapter	12V, 5A (Available with the PolarFire Video kit)
Software	
-FlashPro Express	v12.2
-Test module Software	4.69.9486
-Serial Emulation Terminal	TeraTerm or PuTTY

2.2 Prerequisites

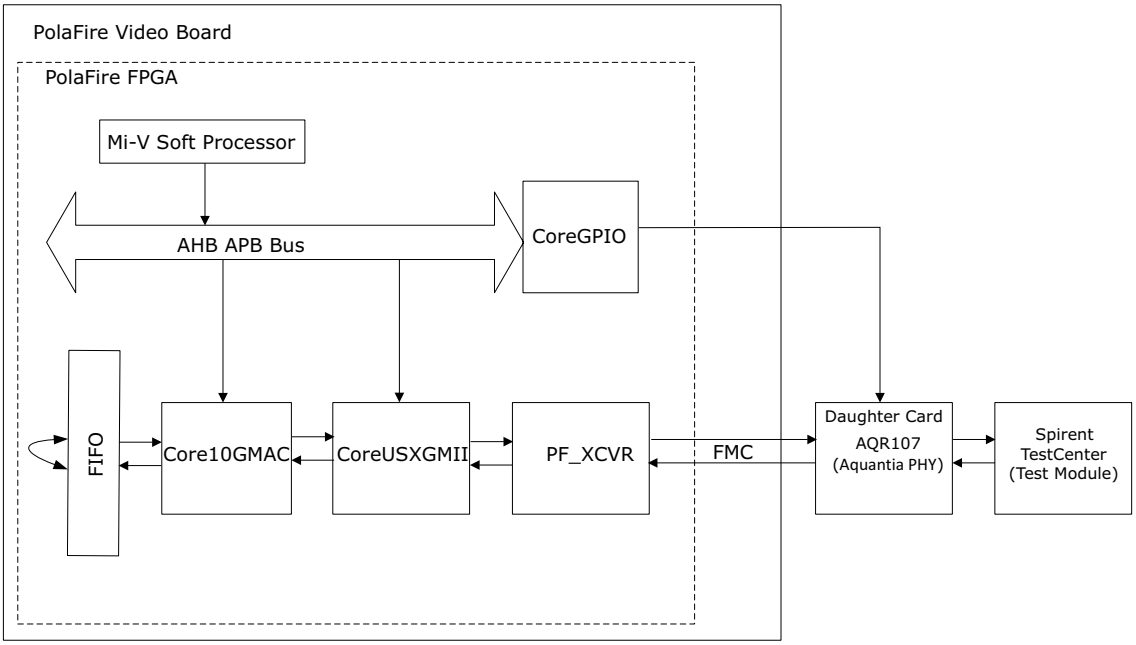
Before you start:

1. Download the design files from:
   http://soc.microsemi.com/download/rsc/?f=mpf_dg0885_liberosoc_df
2. Download FlashPro Express from:
   https://soc.microsemi.com/portal/default.aspx?r=3&p=f=ProgramDebug_v12_2_WIN
3. Download the test module software from:
   link?
4. Download and install Libero SoC v12.2 from:
   https://www.microsemi.com/product-directory/design-resources/1750-libero-soc#downloads
5. Download and install SoftConsole v6.1 from:
   https://www.microsemi.com/product-directory/design-tools/4879-softconsole#downloads

2.3 Demo Design Architecture

The following figure shows the architecture of the demo design.

Figure 1 • **USXGMII Design Architecture**



The design loops back the XGMII traffic generated by the test module as per the following steps:

1. The data generated by the test module passes through the Aquantia PHY(AQR107) and is received by the PolarFire transceiver inside the FPGA via FMC.
2. The PolarFire transceiver RX converts the serial data stream in to parallel data and clock, which is sent to the CoreUSXGMII RX interface. The data is downscaled at the CoreUSXGMII RX interface based on the data rate set during Auto-Negotiation.
3. CoreUSXGMII RX sends the data to the Ethernet MAC RX (Core10GMAC), which loops back the XGMII data and RX control signals using a FIFO logic implemented in RTL.
4. The looped back data passes through Core10GMAC TX, CoreUSXGMII TX (data upscaling), PF_XCVR TX and the Aquantia PHY, and finally received by the test module.
5. The received packets are analyzed for throughput rate and errors using the test module software.

2.3.1 I/O Ports

The following table lists the important I/O ports of the USXGMII Libero hardware design.

Table 2 • **I/O Ports**

Port Name	Direction	Description
TMS TRSTB TDI TCK TDO	Input Input Input Input Output	JTAG signals interfaced to the Mi-V soft processor for debugging.
REF_CLK_PAD_P_0 REF_CLK_PAD_N_0	Input	148.5 MHz reference clock received from the on-board LVDS oscillator. This reference clock is used to generate clocks for the Fabric and DRI interface.

Table 2 • **I/O Ports *(continued)***

Port Name	Direction	Description
PHY_RSTN_OUT	Input	Active-high reset signal from external PHY. This signal indicates that the external PHY is powered-up.
LANE0_RXD_P LANE0_RXD_N	Input	Receive lane of the Transceiver to receive the serial data via FMC from the external PHY. These pads are connected to the receive pins of the FMC.
LANE0_TXD_P LANE0_TXD_N	Output	Transmit lane of the Transceiver to transmit the serial data via FMC to the external PHY. These pads are connected to the transmit pins Transceiver.
REF_CLK_PAD_P REF_CLK_PAD_N	Input	Reference clocks received from the external PHY card.
SYSRESTN	Input	Active-low system reset. Asserted by pressing the on-board AL27 push-button.
TX	Input	UART interface to the FPGA from the Host PC
RX	Output	UART interface to the Host PC from the FPGA.
PHY_MDIO	Input/Output	Management Data IO Interface for accessing the external PHY PHY registers.
PHY_MDC	Output	Management Data IO clock fed to the external PHY.
PHY_RST	Output	Active-high reset signal to the external PHY.

2.3.2 Subsystem Components

The following sections describe the subsystems used in the design.

- CORE10GMAC
- CoreUSXGMII
- PF_XCVR_ERM
- Mi-V Processor Subsystem
- FIFO Logic
- PF_TX_PLL
- PF_XCVR_REF_CLK
- PF_CCC

2.3.2.1 CORE10GMAC

The CORE10GMAC IP is the 10-Gbps Ethernet MAC that transmits and receives the Ethernet packets.

Core10GMAC is configured for XGMII mode with a core data width of 64 bits. Core data width is the width of the data path connected to the USXGMII IP. The system data width, that is, the width of the interface to the user logic, is configured as 64 bits. In this demo, the FiFo_wrapper_top module provides this interface. The TX and RX Pause features are disabled, and both the MAC TX FIFO depth and MAC RX FIFO depth are set to 256.

The Core10GMAC IP is configured using Mi-V soft processor and is covered in-detail in the Mi-V subsystem section. For information about the features and registers of Core10GMAC, see **Libero SoC -> Catalog -> Core10GMAC Handbook**.

2.3.2.2 CoreUSXGMII

The CoreUSXGMII IP provides an XGMII interface between MAC and transceiver, and adapts the Ethernet frames to/from MAC based on the Auto-Negotiated data rate using data downscaling or upscaling logic in the RX and TX interface.

CoreUSXGMII is configured using the Mi-V soft processor. For information about the features and registers of CoreUSXGMII, see
Libero SoC -> **Catalog** -> **CoreUSXGMII Handbook**.

2.3.2.3 PF_XCVR_ERM

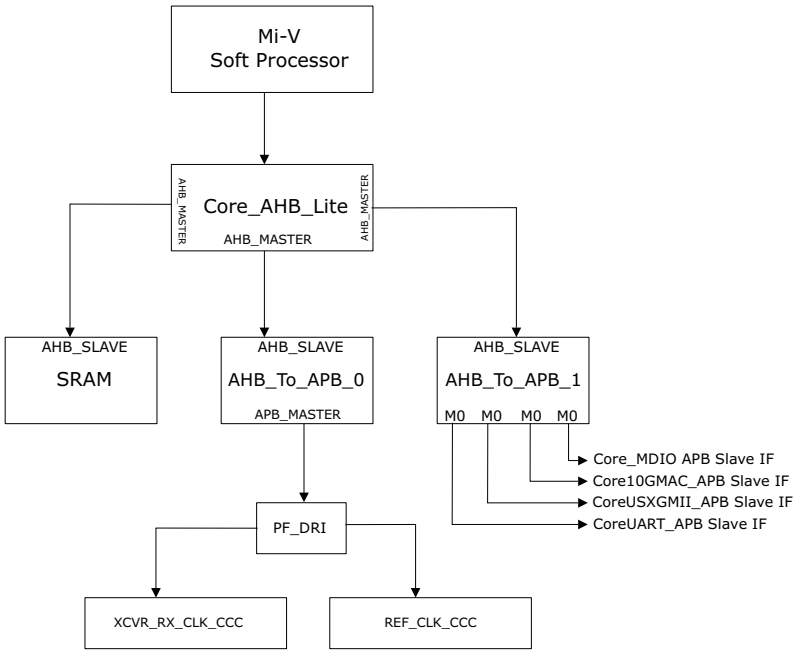
The PF_XCVR IP provides the 10GBASE-R physical interface for data transfers. PF_XCVR is configured for 64b/66b encoding/decoding with scrambler/descrambler enabled with a PCS interface width of 64 bits to the COREUSXGMII.

The PolarFire high-speed transceiver (PF_XCVR) is a hard IP block and supports data rates from 250 Mbps to 12.5 Gbps. In this demo, PF_XCVR is configured for the data rate of 10312.5 Mbps. It is configured with a CDR reference clock of 156.25 MHz with Lock to data selected as the CDR lock mode.

2.3.2.4 Mi-V Processor Subsystem

The Mi-V processor subsystem configures Core10GMAC, CoreUSXGMII through AHB-APB bus interface, and also communicates with external AQR107 PHY through CoreGPIO which provides the MDIO clause 45 interface implemented in the firmware.

Figure 2 • **Mi-V Subsystem**



MIV_RV32IMA_L1_AHB Configurator, set the Reset Vector Address to 0x80000000, as shown in the following figure. This is the address the processor will start executing from after a reset. The processor's main memory must be accessible to Mi-V AHB memory interface whose memory mapped address ranges from 0x80000000 to 0x8FFFFFFF. The Mi-V memory interface supports cached transactions, whereas Mi-V MMIO interface does not support them.

Mi-V subsystem communicates to the IP block using the following interfaces:

- Core10GMAC: Mi-V (AHB master) -> CoreAHBlite (AHB master) -> CoreAHB_to_APB3 (APB master) -> Core10GMAC (APB slave). Registers configured by Mi-V subsystem are as follows.

Table 3 • **Core10GMAC Registers**

Register	Address	Offset	Bit	Binary value
MAC TX Config Register (0xA)		0x3	cfg_sys_mac_tx_en	1
		0x4	sys_mac_tx_fcs_ins	1

Table 3 • **Core10GMAC Registers**

Register	Address	Offset	Bit	Binary value
MAC RX Config Register (0xB)		0x0	mac_rx_fcs_remove	1
		0x3	cfg_sys_mac_rx-en	1

- CoreUSXGMII: Mi-V (AHB master) -> CoreAHBlite (AHB master) -> CoreAHB_to_APB3 (APB master) -> CoreUSXGMII (APB slave). Registers configured by Mi-V subsystem are listed in Table 4.

Table 4 • **CoreUSXGMII Registers**

Register	Offset	Description
USXGMII-CONTROL_REG	0x0	Control: Enables/Disables the USXGMII Auto negotiation function
USXGMII_STATUS_REG	0x4	Status: Indicates the Link status along with Auto negotiation status
USXGMII_AN_ADV	0x8	Auto Negotiation Advertise: configures the mode of operation and configures the speed selection
USXGMII_AN_LP_ADV	0xC	Auto Negotiation Link Partner Base Page Ability: Read Only register indicates the link partners USXGMII configuration

- CoreUARTAPB: Mi-V (AHB master) -> CoreAHBlite (AHB master) -> CoreAHB_to_APB3 (APB master) -> CoreUARTAPB (APB slave). Registers configured by Mi-V subsystem and the user requests are as follows.
- Aquantia PHY (AQR107): Mi-V (AHB master) -> CoreAHBlite (AHB master) -> CoreAHB_to_APB3 (APB master) -> CoreGPIO (APB slave). Registers configured by Mi-V subsystem are as follows:

Table 5 • **PHY Registers[1]**

Register	Offset	Description
PHY REGISTER	0x4	Enables / Disables / re-start Auto Negotiation

1. For information about the features and registers of Aquantia PHY, see AQR107 Handbook.

- PF_SRAM: The PolarFire system controller initializes the LSRAM's with user application and releases the system reset.

2.3.2.5 FIFO Logic

The FIFO interface logic loops back the CORE10GMAC RX data to TX data. FiFo_wrapper_Top is a user-defined RTL module, which uses the CoreFIFO IP to loop the MAC RX PACKET INTERFACE to the MAC TX PACKET INTERFACE.

2.3.2.6 PF_TX_PLL

The PF_TX_PLL IP generates the bit clock required for the transceiver.

The PolarFire transmit PLL (PF_TX_PLL) is a hard IP block that provides a bit clock and a reference clock to the transceiver block. The transmit PLL is configured with a reference clock of 156.25 MHz and generates an output clock of 10312.5 Mbps.

2.3.2.7 PF_XCVR_REF_CLK

PF_XCVR_REF_CLK generates the fabric clock and the reference clock for the transceiver and the TX_PLL. The transceiver reference clock (PF_XCVR_REF_CLK) is a hard IP block that provides a reference clock (REF_CLK) of 156.25 MHz to the transmit PLL and a fabric reference clock

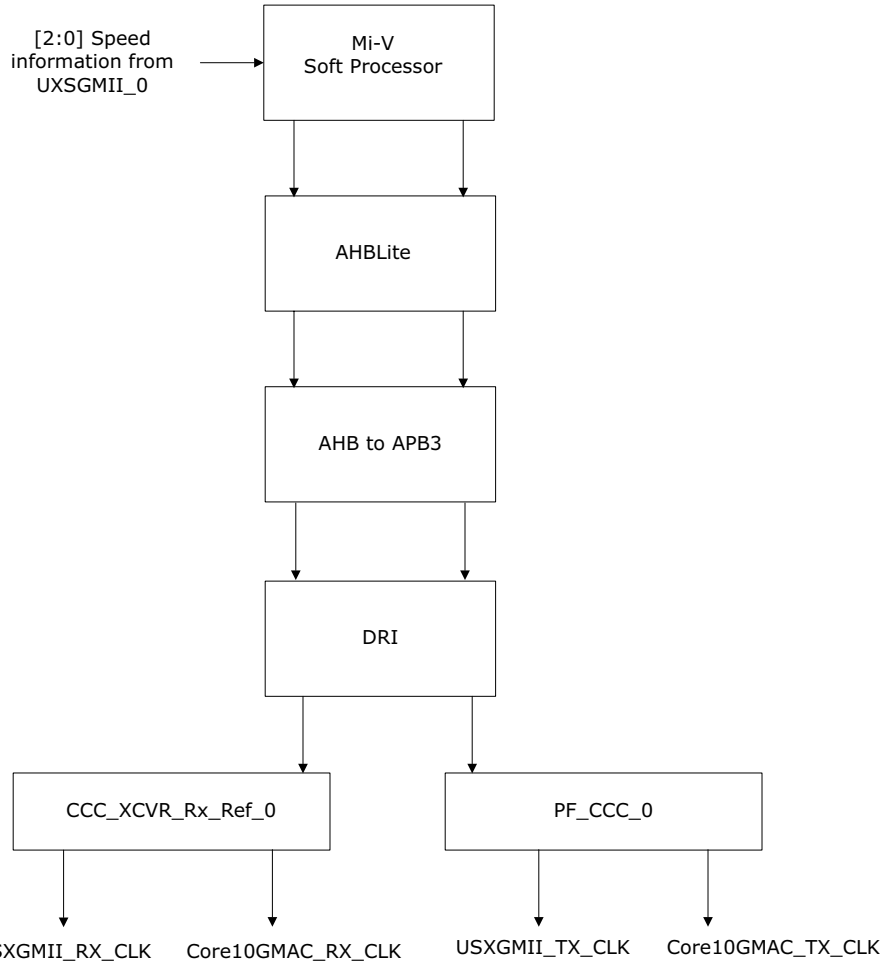
(FAB_REF_CLK) which is provided as input to the Clock Conditioning circuit (CCC) to generate the PCLK (for configuration) and I_SYS_CLK of CORE10GMAC.

2.3.2.8 PF_CCC

The PF_CCC IP instances provide the required clock frequency for CoreUSXGMII, Core10GMAC, and FIFO logic. The PolarFire Clock Conditioning Circuitry (CCC) block sources an input clock of 148.5 MHz from the FAB_REF_CLK signal (output of PF_XCVR_REF_CLK) and generates a 50 MHz clock at OUT0 and 156.25 at OUT1. The OUT0 port of CCC is used for the configuration and OUT1 is used for the user logic in the design.

Mi-V soft processor receives speed information from configures the PF_CCC instantiations via the DRI interface as shown in the following figure.

Figure 3 • **Mi-V Soft Processor - DRI - PF_CCC Interface**



The following table shows the clock generated by PF_CCC instantiations used for generating RX and TX clocks for USXGMII and Core10GMAC.

Table 6 • USXGMII and Core10GMAC Clock Sources

PF_CCC Instance	Input Source	Output Clocks
CCC_XCVR_Rx_Ref_0	XCVR_RX_CLK	USXGMII_core_rx_clk
		Core10GMAC_lcore_rx_clk

Table 6 • USXGMII and Core10GMAC Clock Sources ***(continued)***

PF_CCC Instance	Input Source	Output Clocks
PF_CCC_0	PF_XCVR_REF_CLK_1	USXGMII_core_tx_clk
		Core10GMAC_Icore_tx_clk

For more information, see Clocking Structure.

2.3.2.9 PF_INIT_MONITOR

The PF_POWER_INIT block ensures the device is powered up in a systematic way. The process of powering up the device includes three steps:

1. Power-on reset
2. Programmed device boot
3. Design initialization

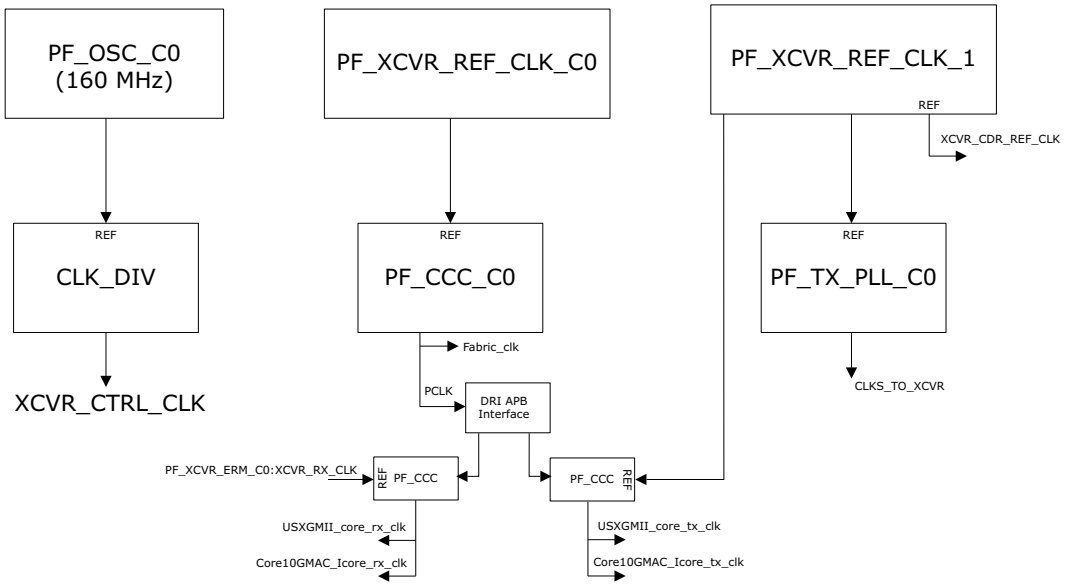
During design initialization, the transceiver configuration is initialized using the data stored in the non-volatile memory. The output of the PF_POWER_INIT block is ANDed with the resets used in the design to reset entire logic.

2.3.3 Clocking Structure

Figure 4 shows the clocking structure of the PolarFire USXGMII design. As shown below, the clocking structure consists of the following clock domains:

- **On-chip 160 MHz RC Oscillator**: Drives PF_XCVR_ERM_C0:XCVR_CTRL_CLK.
- **PF_XCVR_REF_CLK_C0**: Generates reference clock required for the PF_DRI_C0 block.
- **PF_XCVR_REF_CLK_1:** Generates reference clocks required for:
  - Driving Transceiver CDR reference clock used to derive the RX clocks for Transceiver, USXGMII, and Core10GMAC blocks.
  - Driving TX clocks for USXGMII and Core10GMAC blocks.

Figure 4 • **Clocking Structure**



2.3.4 Reset Structure

The Reset structure is implemented in the `Clock_Reset_Subsystem` SmartDesign file in the Libero design. This SmartDesign module generates the following reset signals:

- **FABRIC_RESET_N**: To reset the Mi-V_Subsystem. FABRIC_RESET_N is asserted when the SYSRESET_N, PHY_RST_OUT, DEVICE_INIT_DONE, and PLL_LOCK signals are asserted. PHY_RST_OUT is asserted when the external PHY is powered up.
- **XCVR_PCS_PMA_RESET** and **PHY_RST**: To reset the PolarFire Transceiver (PF_XCVR_ERM) PMA and PCS. XCVR_PCS_PMA_RESET and PHY_RST are asserted when SYSRESETN and DEVICE_INIT_DONE signals are asserted.

The Mi-V Subsystem generates the following reset signals after reset:

- **EXT_RST**: To reset FIFO, USXGMII blocks.
- **MAC_RST**: To reset Core10GMAC block using the CoreGPIO APB interface.

2.3.5 Resource Utilization

Table 7 lists the resource utilization of the UXSGMII design on the MPF300T device.This report is derived after Place and Route.

Table 7 • **USXGMII Resource Utilization**

Element	Used	Total	Percentage
4LUT	28006	299544	9.35
DFF	22730	299544	7.59
Logic Elements	32744	299544	10.93

3 Setting Up the Demo

This section describes steps to successfully set up the hardware and program the FPGA.

Setting up the demo involves the following steps:

- Setting Up the Hardware
- Programming the Device Using FlashPro Express

3.1 Setting Up the Hardware

This section describes how to connect all of the components required to run the demo.

1. Connect the test module device to a LAN using the Cat 6 cable.
2. Connect the Host PC to the same LAN using the Cat 6 cable.
3. Connect the host PC and the video kit through **J12** of the video kit using the USB mini cable.
4. Insert the Aquantia PHY daughter card on the FMC connector of video kit.
5. Connect the Aquantia PHY daughter card and the test module using the Cat 6 cable. At the test module side use the 10Gbe variable data rate port (for example port 9).
6. Connect the power supply cable to **J20** of the video kit.
7. Ensure that the following jumper settings are set on the video kit.

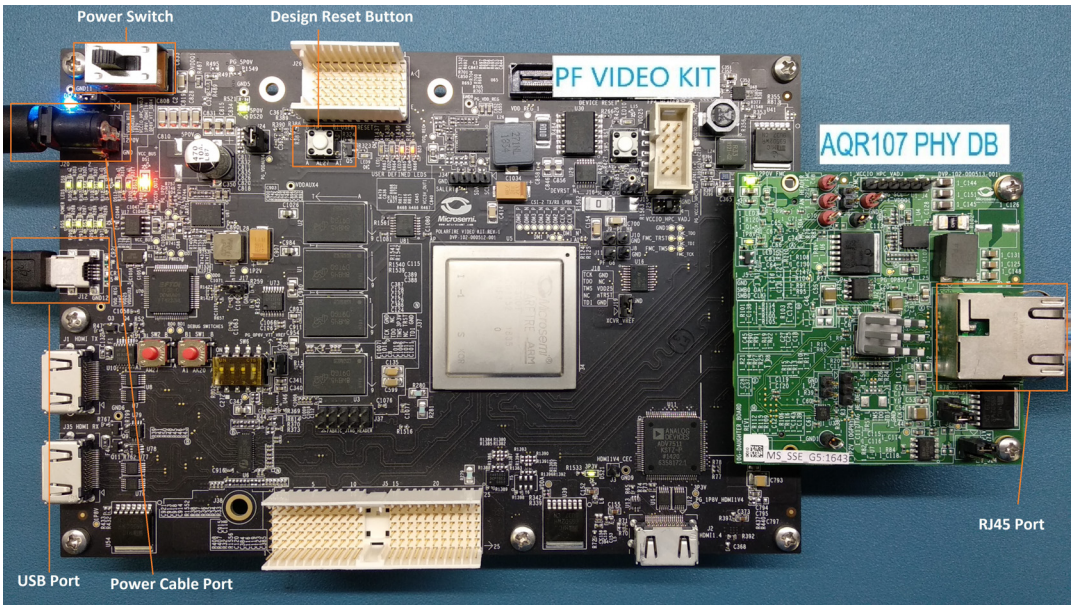
Table 8 • **Jumper and Switch Settings**

Jumper	Default Position	Functionality
J15	Open	SPI Slave and Master mode selection. By default, SPI master.
J17	Open	100K PD for TRSTn. By default,1K PD is connected.
J19	Pin 1&2	Default: XCVR_VREF is connected to GND.
J28	Pin 1&2	Default: Programming through the FTDI.
J24	Pin 2&4	Default: VDDAUX4 voltage is set to 3V3.
J25	Pin 5&6	Default: Bank4 voltage is set to 1V8.
J36	Pin 1&2	Default: Board power up through the SW4.
SW4	OFF (Pin 2-3,5-6 Positions)	Power ON\OFF switch.
SW6	OFF	User slide switch. Default position: OFF.
J20	12 Volts Input	12V input to the board.

8. Power-up the Host PC and test module.
9. Power-up the video kit using the **SW4** slide switch.

The PolarFire USXGMII hardware is set up as shown in Figure 5. See the following section to program the PolarFire device.

Figure 5 • **Board Setup**



3.2 Programming the PolarFire Device

The PolarFire device can be programmed using any of the following methods:

- Programming the Device Using FlashPro Express
- Programming the Device Using Libero SoC

3.2.1 Programming the Device Using FlashPro Express

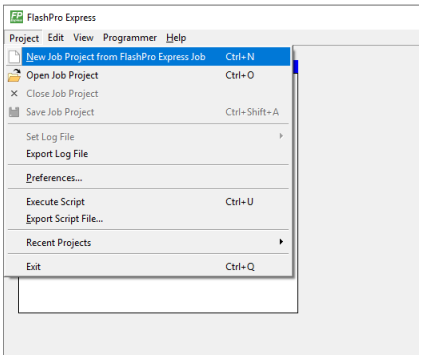
This section describes how to program the PolarFire device with the job file using Flashpro Express. The job file can be downloaded using the following link:

http://soc.microsemi.com/download/rsc/?f=mpf_dg0885_liberosoc_jb

Follow these steps:

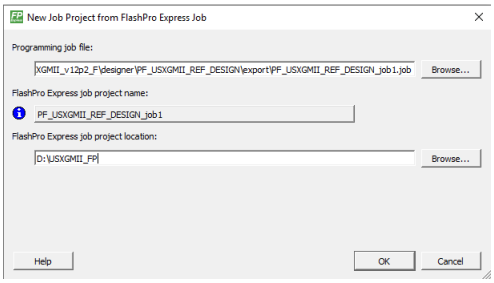
1. On the host PC, start the FlashPro Express software from its installation directory.
2. Select **New** or **New Job Project from FlashPro Express Job** from Project menu to create a new job project, as shown in Figure 6.

Figure 6 • **FlashPro Express New Job**



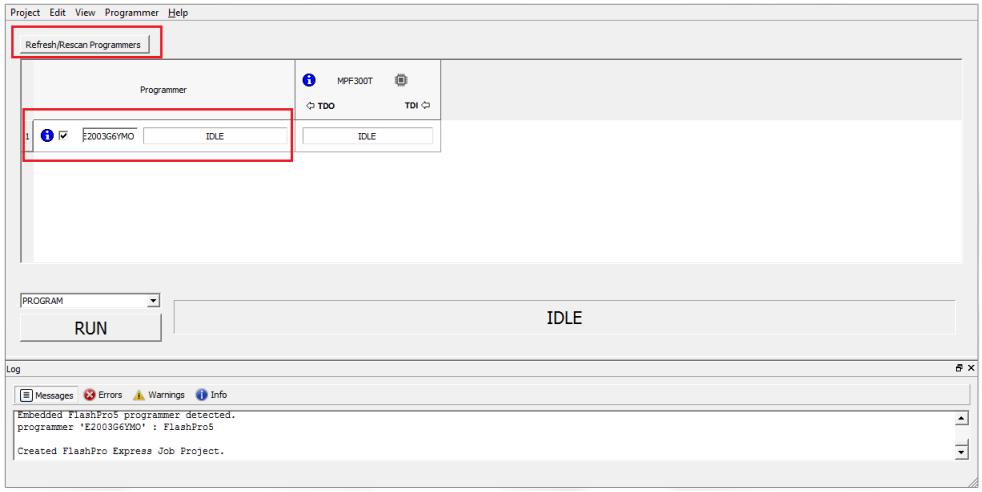
3. Enter the following in the New Job Project from FlashPro Express Job dialog box:
   - Programming job file: Click **Browse** and navigate to the location where the job file is located and select the file. The default location is:
     `<$Download_Directory>\mpf_dg0885_liberosoc_jb\Programming_job`
   - **FlashPro Express job project location**: Select **Browse** and navigate to the location where you want to save the project.

Figure 7 • **New Job Project from FlashPro Express Job**



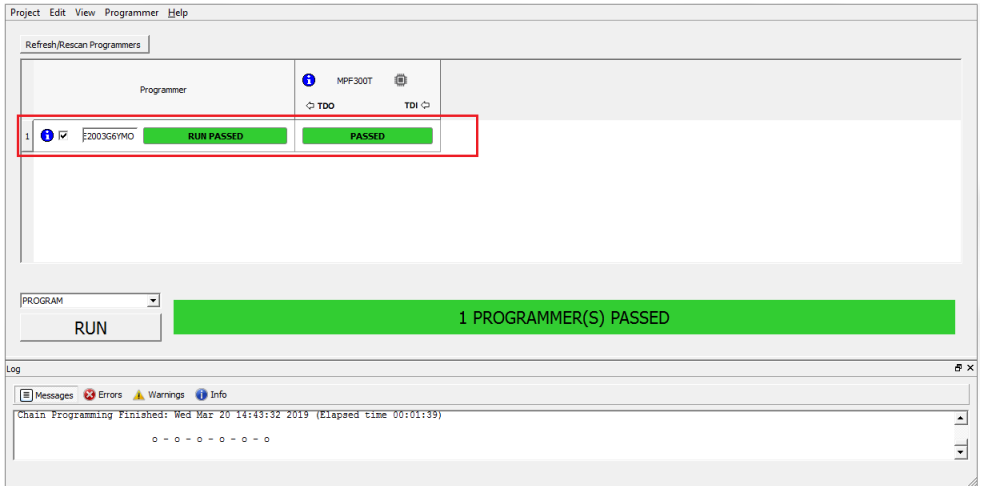
4. Click **OK**. The required programming JOB file is selected and ready to be programmed in the device.
5. The FlashPro Express window appears as shown in Figure 8. Confirm that a programmer number appears in the Programmer field. If it does not, confirm the board connections and click **Refresh/Rescan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

Figure 8 • **Programming the Device**



6. Click **RUN** to program the device. When the device is programmed successfully, a **RUN PASSED** status is displayed as shown in Figure 9. See Running the Demo.

Figure 9 • **Run Passed**

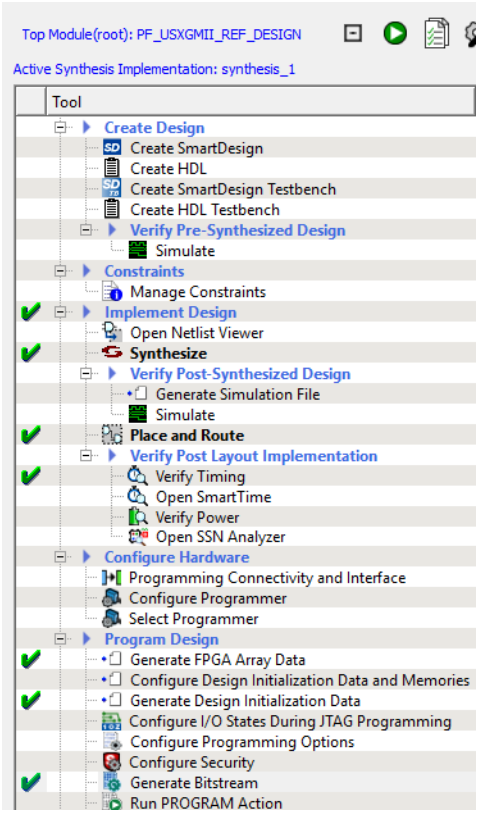


3.2.2 Programming the Device Using Libero SoC

The design files include the Libero SoC project for USXGMII. The PolarFire device can be programmed using Libero SoC. The Libero SoC project is completely built and run from Synthesis, Place and Route, Timing Verification, FPGA Array Data Generation, Design and Memory Initialization, Bitstream Generation, FPGA Programming.

The Libero design flow is shown in the following figure.

Figure 10 • **Libero Design Flow**



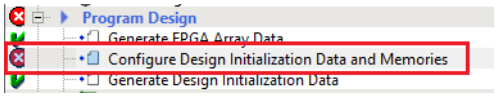
To program the PolarFire device, the USXGMII project must be opened in Libero SoC and the following steps must be re-run:

- **Design and Memory Initialization**: In this step, the following options are selected:
  - Storage type (sNVM, µPROM, or SPI Flash) for the initialization client to initialize the designated fabric RAM block.
  - Generating the initialization client by selecting the user application file (`.hex`).
- **Bitstream Generation**: In this step, the STAPL file is generated for the PolarFire device.
- **FPGA Programming**: In this step, the PolarFire device is programmed using the STAPL file.

Follow these steps:

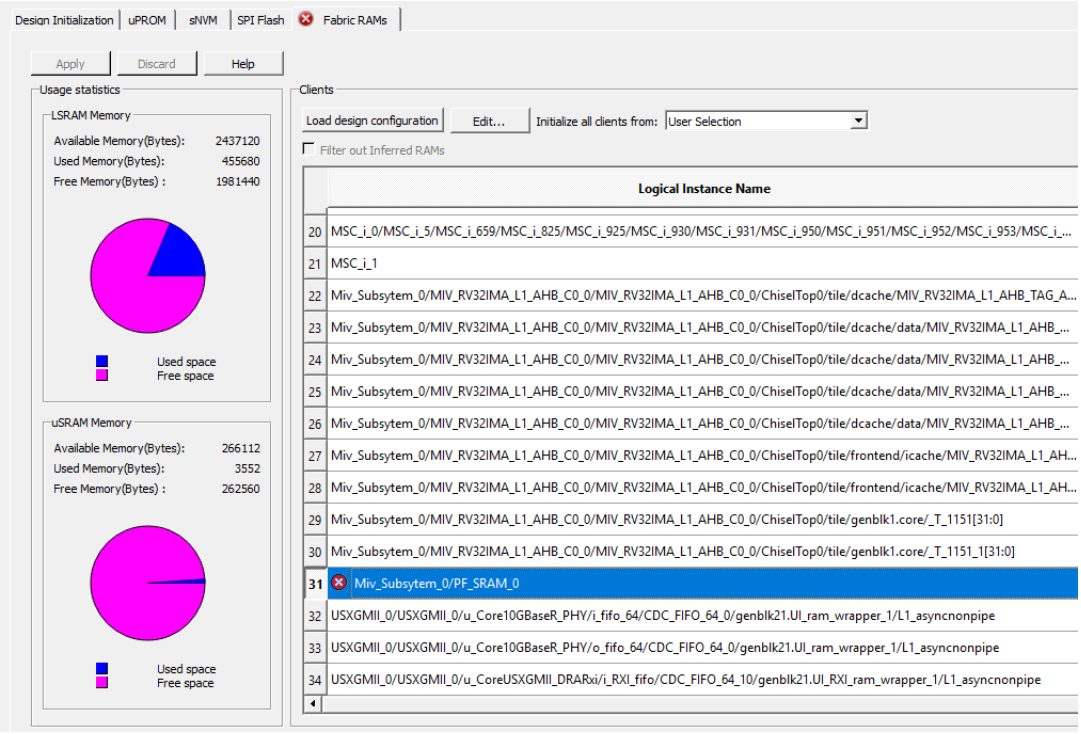
1. Launch Libero SoC.
2. Open the USXGMII project by selecting the `USXGMII_Demo.prjx` file from following location: `<$Design_Files_Directory>:\mpf_dg0885_liberosoc_df\Solution\Libero_Project`
3. Select the Design Initialization Data and Memories as shown in Figure 11.

Figure 11 • **Design and Memory Initialization Option**



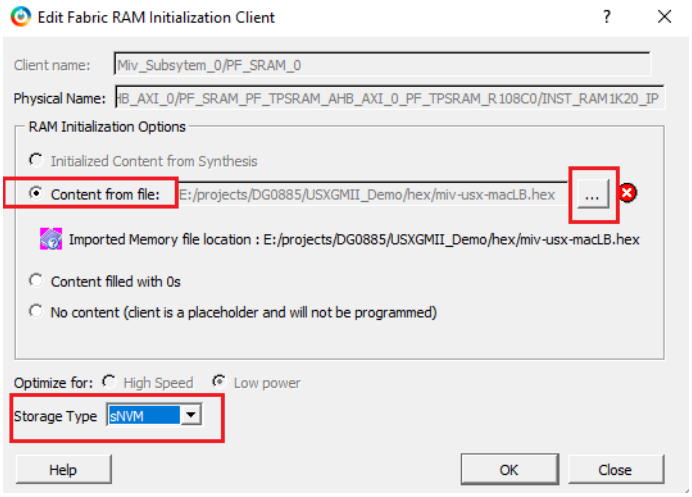
4. Select the logical RAM instance as shown in Figure 12.

Figure 12 • **Selecting the Logical RAM Instance**



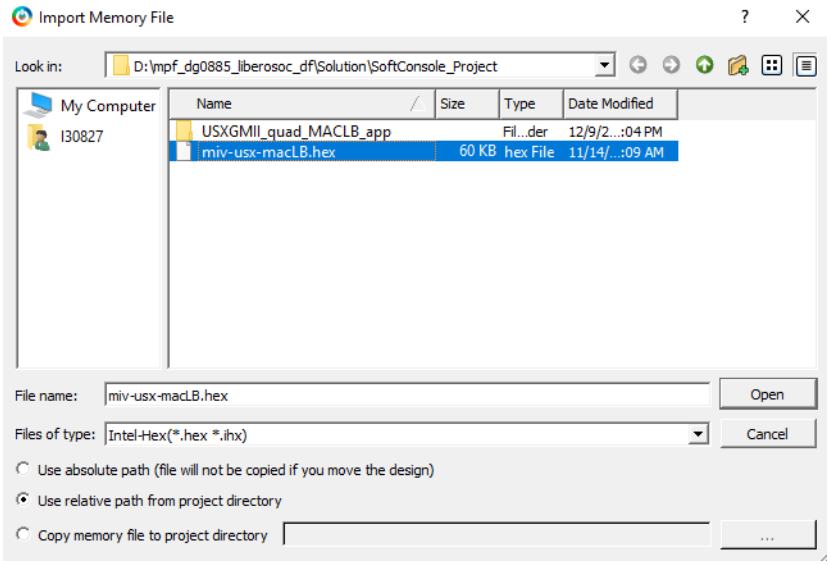
5. Select the storage type and select the Import option to import user application file (Figure 13).

Figure 13 • **Edit Fabric RAM Client**



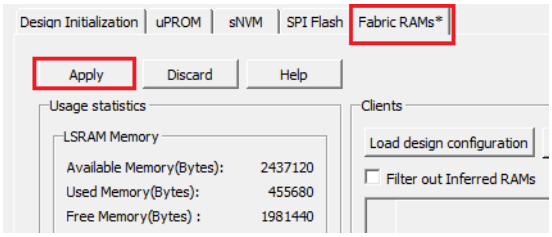
6. Select the application file as shown in Figure 14.

Figure 14 • **Import Memory File**



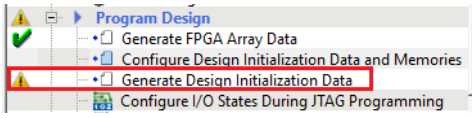
7. Apply the configuration as shown in Figure 15.

Figure 15 • **Apply Configuration**



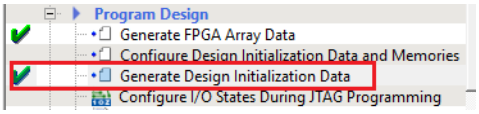
8. Select the Generate Design Initialization Data option as Figure 16.

Figure 16 • **Generate Design Initialization Data**



The design initialization data is generated as shown in Figure 17.

Figure 17 • **Initialization Data Generated**



9. Select the Generate Bitstream option to the generate bitstream for the PolarFire device
10. Select the Run PROGRAM Action to program the PolarFire device.

This concludes the programming of the PolarFire device.

4 Running the Demo

This section describes steps to successfully run the demo and observe the Ethernet packets transmitted and received by the Ethernet test module. The following points describe the overview of the demo:

1. The test module initiates the Ethernet traffic on the line. At the system side, the FPGA configures the AQR107 PHY. Then, AN packets are passed to CoreUSXGMII via AQR107 PHY and the auto negotiation is completed at the system side.
2. The Ethernet traffic is received at USXGMII through XCVR lane which is connected to AQR107 PHY on 10G daughter board. Ethernet Packets are looped back at the USER FIFO located at CORE10GMAC system interface.
3. The test module receives the packets from AQR107 PHY through Cat6 cable and checks for CRC errors. It displays the number of packets transmitted, received, errors received and the line throughput.

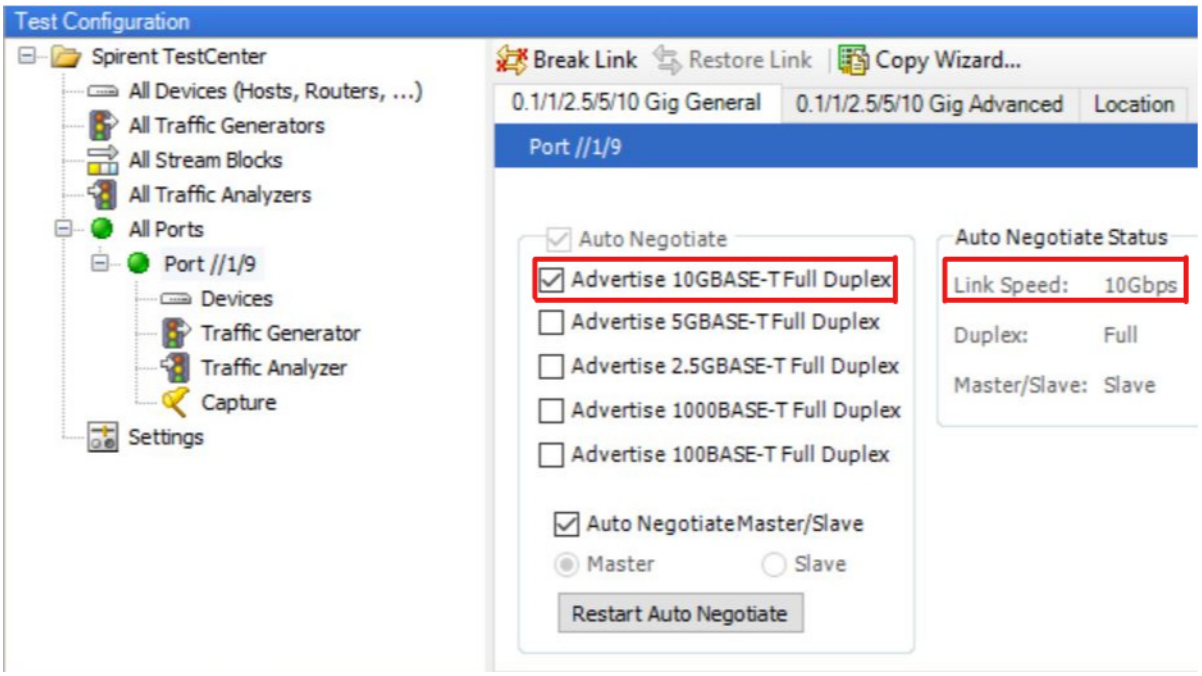
Note: Before running the demo:

- Ensure that the demo hardware is set up as described in Setting Up the Demo.
- The user must know how to launch the test module software on the Host PC, discover the test module, and use the test module software.

Follow these steps to run the USXGMII demo:

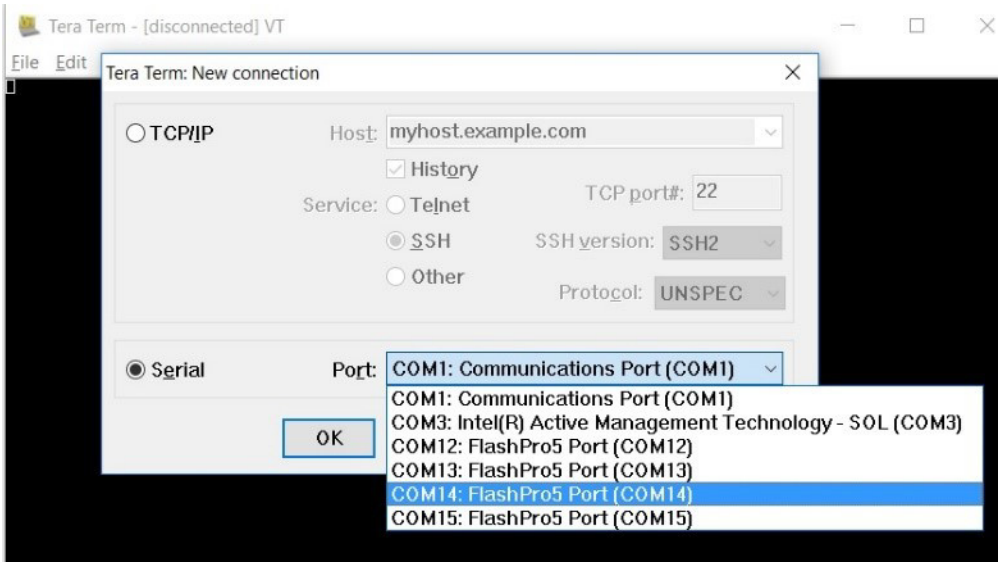
1. Configure the test module for 10GBASE-T advertisement using the test module software.

Figure 18 • **10GBASE-T Advertisement**



2. Launch TeraTerm with the 3rd FlashPro5 Port and 115200 Baud rate.

Figure 19 • **TeraTerm Configuration**

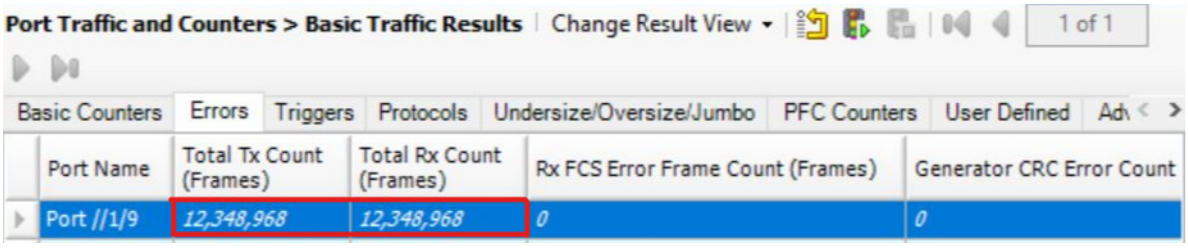


3. Reset the design or power cycle the video board.
4. Observe the UART messages for the completion of PHY initialization, AN enabled between the external PHY and USXGMII, MAC configuration, and the 10G clock configured message.

Figure 20 • **UART Message - 1**

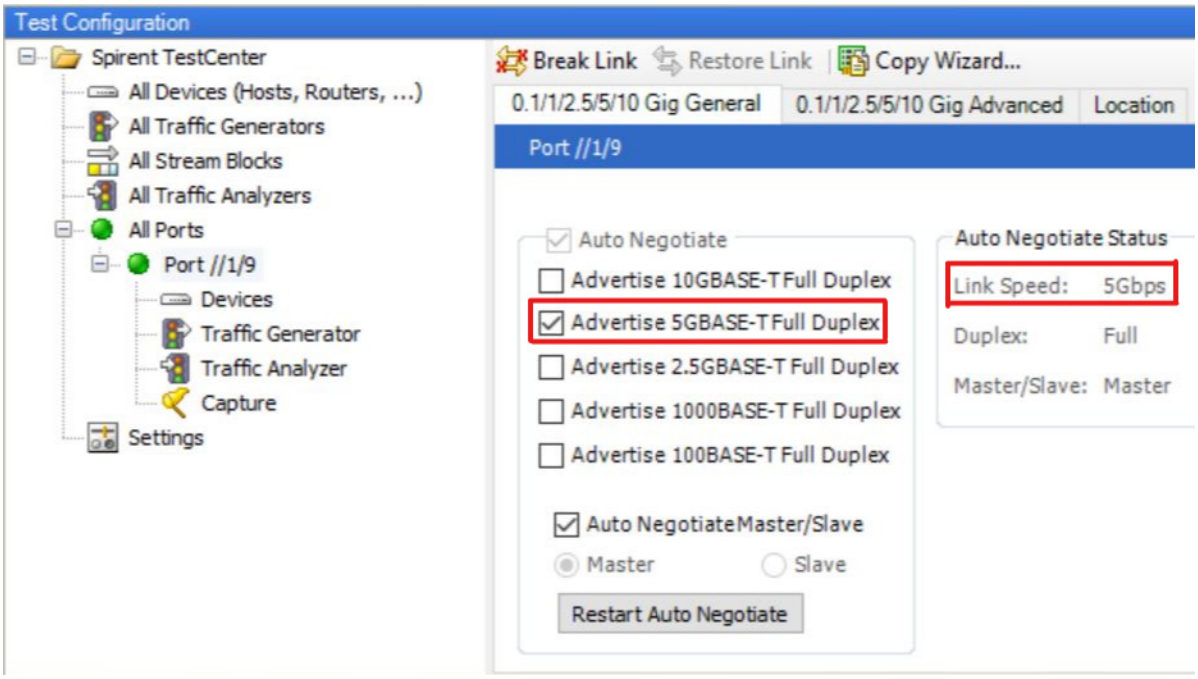
5. Observe the 10G traffic transmitted and received by the test module.

Figure 21 • **10G Traffic Report**



6. Configure the test module for 5GBASE-T advertisement.

Figure 22 • **5GBASE-T Advertisement**

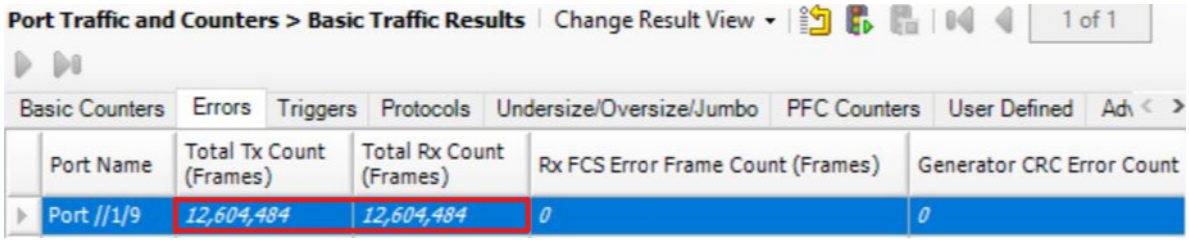


7. Observe the clocks configured message for 5G data rate on the TeraTerm.

Figure 23 • **UART Message -2**

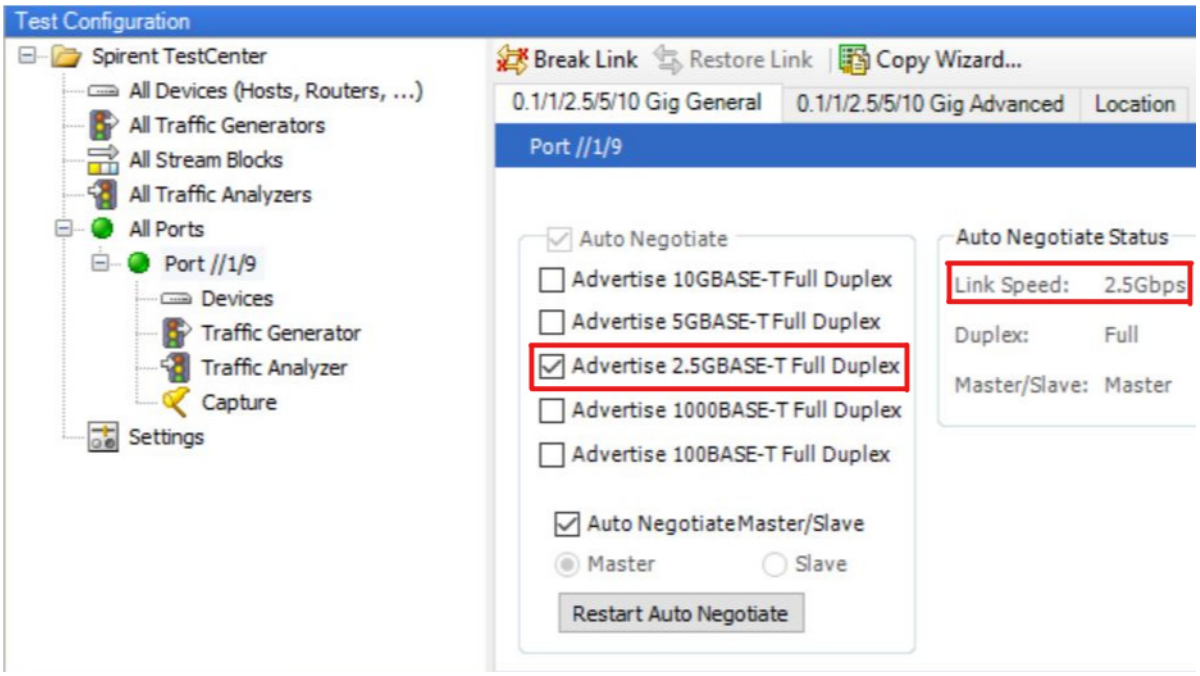
8. Observe the 5G traffic transmitted and received by the test module.

Figure 24 • **5G Traffic Report**



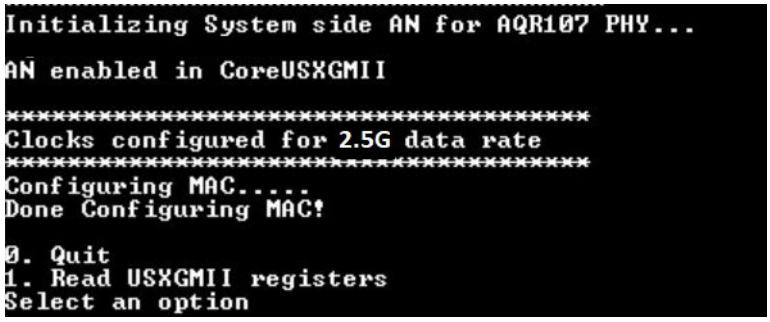
9. Configure the test module for 2.5BASE-T advertisement.

Figure 25 • **2.5GBASE-T Advertisement**



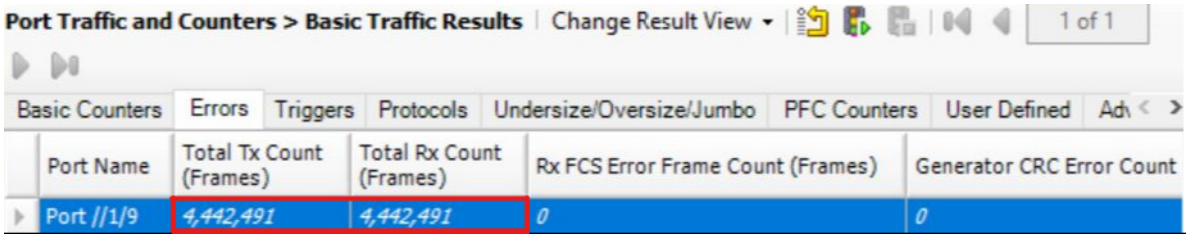
10. Observe the clocks configured message for 2.5G data rate on the TeraTerm.

Figure 26 • **UART Message - 3**



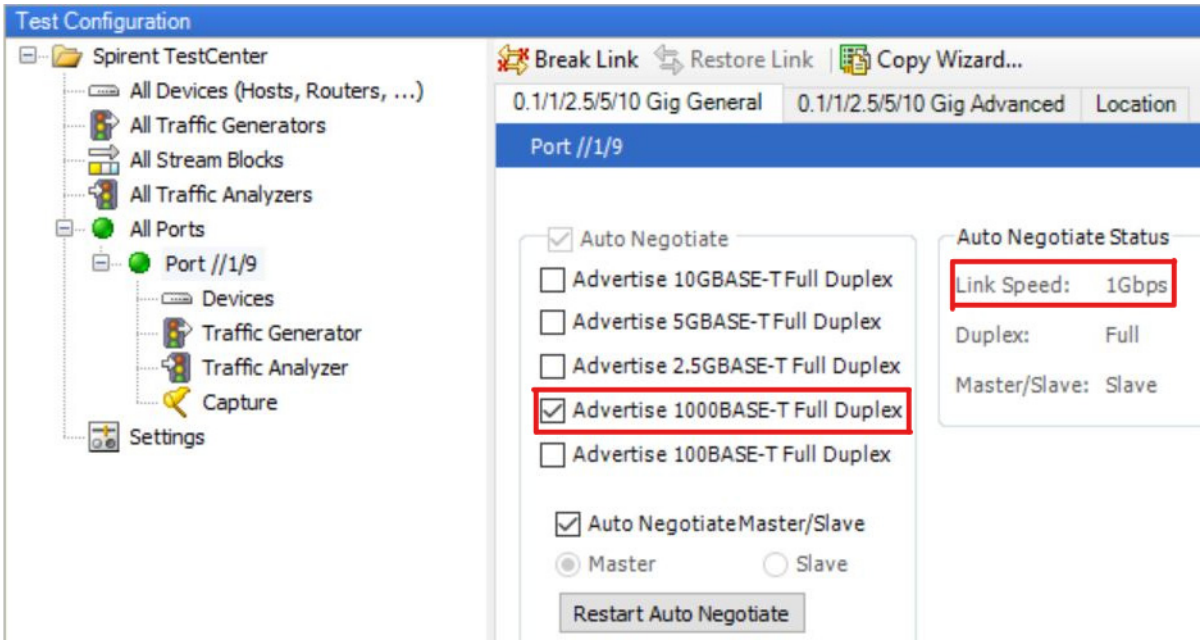
11. Observe the 2.5G traffic transmitted and received by the test module.

Figure 27 • **2.5G Traffic Report**



12. Configure the test module for 1000BASE-T advertisement.

Figure 28 • **1000BASE-T Advertisement**

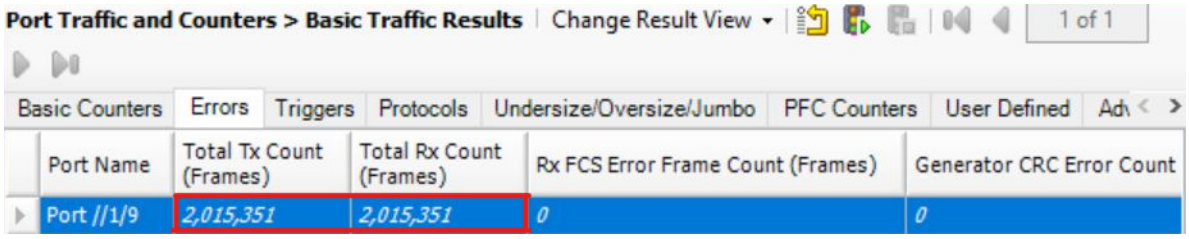


13. Observe the clocks configured message for 1000BASE-T data rate on the TeraTerm.

Figure 29 • **UART Message - 4**

14. Observe the 1000BASE-T traffic transmitted and received by the test module.

Figure 30 • **1000BASE-T Traffic Report**



This concludes the demo.