# DG0834 Demo Guide Running Webserver and IAP Using TFTP on PolarFire Device





а 🔨 Міскоснір 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 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.



# Contents

1	Revisi	on History
	1.1	Revision 3.0
	1.2	Revision 2.0
	1.3	Revision 1.0
2	Runnii	ng Webserver and IAP Using TFTP on PolarFire Device
	2.1	Webserver and TFTP Server Demo Design Layers    3
		2.1.1 Application Layer
		2.1.2       Hansport Layer (with For /it Stack)         2.1.3       RTOS and Firmware Layer         3
	2.2	Design Requirements
	2.3	Prerequisites
	2.4	Demo Design
	o -	2.4.1 Design Implementation
	2.5	Clocking Structure
	2.0	
3	Libero	Design Flow
	3.1	Synthesize
	3.2	Place and Route
		3.2.1 Resource Utilization
	3.3	Verify Timing
	3.4 2.5	Configure Design Initialization Data and Memories
	3.5	Generate Bitstream
	3.7	Export FlashPro Express Job
	3.8	Run PROGRAM Action
	3.9	Program SPI Flash Image
4	Runnii	ng the Demo
	4.1	Tera Term Setup         27
	4.2	Running Webserver Demo
	4.3	Running TFTP Demo         29
		4.3.1 Running IAP Authentication
		4.3.2 Running IAP Program
5	Appen	idix 1: Enable TFTP Client
6	Appen	dix 2: Running the SoftConsole Project in Debug Mode from LSRAM or DDR
	Memo	ry
	6.1	Running the Design in Static IP Mode
7	Appen	dix 3: Programming the Device and External SPI Flash Using FlashPro Express



# **Figures**

Figure 1	Webserver and TFTP Server Applications on a PolarFire Device	. 3			
Figure 2	Demo Design High-level Block Diagram	. 5			
Figure 3	Top-Level Libero Implementation	. 6			
Figure 4	Mi-V Configurator				
Figure 5	PF_SRAM_0	. 8			
Figure 6	PF_LSRAM_1_0	. 9			
Figure 7	CoreAXI4Interconnect Configurator	. 9			
Figure 8	CoreAXI4Interconnect Configurator—Master Configuration	. 9			
Figure 9	CoreAXI4Interconnect Configurator—Slave Configuration	10			
Figure 10	CoreAXI4Interconnect Configurator—Crossbar Configuration	10			
Figure 11	DDR3 Configuration	11			
Figure 12	PF_CCC_0 Input Clock Configuration	12			
Figure 13	PF_CCC_0 Output Clock Configuration	12			
Figure 14	NWC_PLL_0 Input Clock Configuration	13			
Figure 15	NWC_PLL_0 Output Clock Configuration	14			
Figure 16	NWC_PLL_0 DLL Configuration	15			
Figure 17	CoreSPI_0 Configuration	15			
Figure 18	Mi-V Processor Bus Interface Memory Map	16			
Figure 19	CoreAHBLite_2 Configuration	17			
Figure 20	CoreAPB3 Configuration	18			
Figure 21	Clocking Structure	19			
Figure 22	Directory Structure of Webserver SoftConsole Project	20			
Figure 23	Directory Structure of TFTP IAP SoftConsole Project	20			
Figure 24	Libero Design Flow Options	21			
Figure 25	Configure Design Initialization Data and Memories Option	23			
Figure 26	Fabric RAMs Tab	23			
Figure 27	Start Address for SPI Flash Clients	24			
Figure 28	Export FlashPro Express Job	25			
Figure 29	Run Program Action	26			
Figure 30	SPI Flash Programming	26			
Figure 31	Select Serial as the Connection Type	27			
Figure 32	Tera Term Configuration	28			
Figure 33	Tera Term General Setup	28			
Figure 34	Tera Term with IP Address	28			
Figure 35	Webserver Demo Page	29			
Figure 36	Tera Term Window	29			
Figure 37	Erasing the SPI Flash Memory Location [0xA00000 - 0x13FFFFF]	30			
Figure 38	Acquiring IP Address	30			
Figure 39	Transfer Programming Image1	30			
Figure 40	Bytes Received for Image1	31			
Figure 41	Erasing the SPI Flash Memory Location [0x1400000 - 0x1DFFFFF]	31			
Figure 42	Iransfer IAP Image2	31			
Figure 43	Bytes Received for Image2	32			
Figure 44	SPI Directory	32			
Figure 45		32			
Figure 46		33			
Figure 47		33			
Figure 48	Successful IAP by Image2	34			
Figure 49	Control Panel—Programs and Features	35			
Figure 50	Selecting IFIP Client from Windows Features	35			
Figure 51		30 00			
Figure 52		30 07			
Figure 53	Add an app window	31 27			
Figure 54	Selecting Trivial File Transfer Protocol App in Allowed apps Window	3/ 20			
Figure 55		38			



Figure 56	Project Explorer Window	39
Figure 57	Project Properties	39
Figure 58	Properties for gbe_webserver	40
Figure 59	Host PC TCP/IP Settings	41
Figure 60	FlashPro Express Job Project	42
Figure 61	New Job Project from FlashPro Express Job	43
Figure 62	Programming the Device	43
Figure 63	FlashPro Express—RUN PASSED	44



# **Tables**

Table 1	Design Requirements	1
Table 2	I/O Signals	7
Table 3	Resource Utilization	2
Table 4	Jumper Settings	5



# 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 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.

### 1.2 **Revision 2.0**

The document was updated for Libero SoC v12.0 release.

## 1.3 Revision 1.0

The first publication of this document.



# 2 Running Webserver and IAP Using TFTP on PolarFire Device

Microsemi PolarFire<sup>®</sup> FPGAs support 1G Ethernet solutions for various networking applications. In PolarFire devices, 10/100/1000 Mbps (1G) Ethernet is implemented using the CoreTSE\_AHB Media Access Control (MAC) soft IP core. The CoreTSE\_AHB IP implements a Serial Gigabit Media-Independent Interface (SGMII or GMII) with an Ethernet PHY. This Ethernet interface can be implemented in the FPGA by using either a transceiver (PF\_XCVR IP) or a GPIO with clock and data recovery (PF\_IOD\_CDR IP) capability. In this demo, the 1G Ethernet solution is implemented in the FPGA design by using GPIOs with CDR capability and CoreTSE\_AHB IP.

The CoreTSE\_AHB IP core enables system designers to implement a broad range of Ethernet designs, from low cost 10/100 Ethernet to higher performance 1 gigabit ports. The CoreTSE\_AHB IP core is suitable for use in networking equipment such as switches, routers, and data acquisition systems.

The CoreTSE\_AHB IP has the following major interfaces:

- 10/100/1000 Mbps Ethernet MAC with a Gigabit Media Independent Interface (GMII) and Ten Bit Interface (TBI) to support Serial Gigabit Media Independent Interface (SGMII), 1000BASE-T, and 1000BASE-X.
- GMII or TBI physical layer interface connects to Ethernet PHY
- MAC data path interface

The CoreTSE\_AHB IP core is available in two different versions:

- CoreTSE\_AHB: Uses AHB interface for both the transmit and receive paths.
- CoreTSE\_AHB (Non-AMBA): Uses direct access to the MAC with a streaming packet interface.

For more information about CoreTSE\_AHB IP, see the CoreTSE\_AHB Handbook.

CoreTSE\_AHB IP core requires license for using in Libero<sup>®</sup> SoC design. For license request, contact *soc\_marketing@microsemi.com*.

This demo design implements a Webserver application and a Trivial File Transfer Protocol (TFTP) server using the PolarFire Evaluation Kit board. For more information about this board, see *UG0747: PolarFire FPGA Evaluation Kit User Guide*.

This demo design demonstrates the following:

- Use of Ethernet MAC connected to a serial gigabit media independent interface (SGMII) PHY.
- Integration of CoreTSE\_AHB MAC driver with IwIP TCP/IP stack and FreeRTOS operating system.
- Implementation of Webserver on the PolarFire Evaluation board.
- Implementation of TFTP server on the PolarFire Evaluation board.
- Procedure to run Webserver and TFTP server designs on the PolarFire Evaluation board.

This 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 Appendix 3: Programming the Device and External SPI Flash Using FlashPro Express, page 42.
- Using Libero SoC: To program the device using Libero SoC, see Libero Design Flow, page 21.



## 2.1 Webserver and TFTP Server Demo Design Layers

The Webserver and TFTP server demo design have the following layers.

- Application layer
- Transport layer (lwIP TCP/IP stack)
- RTOS and firmware layer

The following figure is a block diagram of the three layers in the Webserver and TFTP server applications on a PolarFire device.

#### Figure 1 • Webserver and TFTP Server Applications on a PolarFire Device



### 2.1.1 Application Layer

The Webserver handles the HTTP request from the client (host PC) browser and transfers the static pages to the client in response to its request. When the IP address (for example, http://10.60.3.25) is typed in the address bar of the browser, an HTTP request is sent to the port associated with the Webserver. The Webserver then interprets the request and responds to the client with the requested page or resource.

The TFTP client (the host PC) transfer files to the PolarFire device (the TFTP server) using the TFTP PUT command. Transferred files are stored in the PolarFire Evaluation board external flash memory, which is connected to the System Controller SPI interface.

### 2.1.2 Transport Layer (IwIP TCP/IP Stack)

The lwIP TCP/IP stack, developed by Adam Dunkels at the Swedish Institute of Computer Science (SICS), is suitable for embedded systems because of its low system resource usage. The lwIP stack can be used with or without an operating system. It consists of actual implementations of IP, ICMP, UDP, and TCP protocols, as well as the support functions such as buffer and memory management.

IwIP is available (under a BSD license) in C source-code format for download at http://download.savannah.gnu.org/releases/lwip/.

### 2.1.3 RTOS and Firmware Layer

FreeRTOS is an open-source, real-time operating system kernel. In this demo, FreeRTOS is used to prioritize and schedule tasks. For more information about FreeRTOS and the latest source code, see *http://www.freertos.org*.

The firmware provides software drivers to configure and control the following components.

- Ethernet MAC
- Core UART APB
- SPI



## 2.2 Design Requirements

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

### Table 1 • Design Requirements

Requirement	Version
Operating System	Windows 7, 8.1, or 10
Hardware	
PolarFire Evaluation Kit (MPF300-EVAL-KIT)	Rev D or later
Ethernet cable	RJ45
Software	
FlashPro Express	
Libero SoC	Note: Refer to the readme.txt file provided in the design files for the software versions used with this reference
SoftConsole	design.
A serial terminal emulation program	HyperTerminal, TeraTerm, or PuTTY
Browser	Mozilla Firefox, Internet Explorer

**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.

## 2.3 **Prerequisites**

Before you begin:

- For demo design files download link: http://soc.microsemi.com/download/rsc/?f=mpf\_dg0834\_df
- 2. Download and install Libero SoC (as indicated in the website for this design) on the host PC from the following location:

https://www.microsemi.com/product-directory/design-resources/1750-libero-soc#downloads

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



## 2.4 Demo Design

The following is the data flow of the demo design:

- 1. PF\_CCC\_0 provides the clock to the Mi-V processor and other APB peripherals.
- 2. NWC\_PLL\_0 drives the IOD CDR clocks SGMII\_CDR\_0: TX\_CLK\_G and HS\_IO\_CLK.
- 3. Mi-V performs the following functions:
  - Executes the application from LSRAM (PF\_SRAM IP)
  - Configures the ZL30364 clock generation hardware through the CoreSPI IP to generate reference clocks for the VSC PHY and the IOD CDR fabric module.
  - Configures the CoreTSE\_AHB IP MAC in TBI mode and initializes the MAC in 1000 Base-T.
  - Sends a request to the CoreTSE\_AHB IP to negotiate with the on-board VSC8575 PHY.
- 4. CoreTSE\_AHB IP implements the 1G Ethernet MAC and is configured to interface with the PF\_IOD\_CDR block in the SGMII mode. The CoreTSE IP has an inbuilt MDIO interface to exchange control and status information with the VSC PHY.
- 5. PF\_IOD\_CDR IP does the following:
  - Interfaces with the on-board VSC8575 PHY.
  - Recovers the data and clock from the incoming RX\_P and RX\_N ports. Deserializes the recovered data and sends 10-bit parallel data to the CoreTSE.
  - Receives Ethernet data from VSC PHY through the RX\_P and RX\_N input pads, gears down the receive data rate, and deserializes the data.
- 6. The deserialized data is sent from SGMII\_CDR\_0:RX\_DATA[9:0] to CoreTSE\_AHB IP: RCG[9:0]. The CoreTSE\_AHB IP MAC receives the Ethernet packet from the on-board Ethernet PHY through high-speed PF\_IOD\_CDR IP using the built-in DMA controller and the Mi-V processes the Ethernet packets.
- 7. The Ethernet packets from the Mi-V processor are sent to CoreTSE\_AHB IP, and CoreTSE\_AHB IP:TCG[9:0] is sent to SGMII\_CDR\_0:TX\_DATA[9:0].
- 8. SGMII\_CDR\_0 serializes the data, gears up the transmit data rate, and transmits the data to the onboard VSC PHY through the TX\_P and TX\_N output pads.

Following are the demo design features:

- Webserver
- IAP using TFTP server

The following figure shows the high-level demo design block diagram. In this demo design, CoreTSE\_AHB IP is instantiated in the FPGA fabric and connected to the on-board VSC PHY using the IOD CDR IP.

### Figure 2 • Demo Design High-level Block Diagram





## 2.4.1 Design Implementation

The following figure shows the top-level Libero implementation of the demo design. The libero project implementation is the same for both Webserver and IAP using TFTP but the application firmware is different.



### Figure 3 • Top-Level Libero Implementation



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

### Table 2 • I/O Signals

Signal	Direction	Description		
RX_P, RX_N	Input	IOD CDR receive signals connected to the VSC PHY transmit data signals		
REFCLK_N, REFCLK_P	Input	125 MHz input clock received from the on-board ZL30364 and fed to NWC_PLL_0.		
RESET_N	Input	Mi-V reset. Asserted by pressing the on-board K22 push-button		
REF_CLK_0	Input	50 MHz input clock received from the on-board 50 MHz oscillator and fed to PF_CCC_0.		
TCK, TDI, TMS, and TRSTB	Input	JTAG signals interfaced to the soft processor for debugging		
TDO	Output	-		
TX_P, TX_N	Output	IOD CDR transmit signals connected to the VSC PHY receive data signals.		
LINK_OK	Output	Link status indicator. Provides the link up or down status with the on-board PHY. This signal is mapped to on-board LED7. The LED ON condition indicates that the link is up.		
PHY_RST	Output	Reset signal to the on-board VSC8575 PHY		
PHY_MDC	Output	MDIO clock fed to the on-board VSC8575 PHY		
PHY_MDIO	Output	Management Data IO interface for accessing the on-board VSC8575 PHY registers		
coma_mode	Output	Signal held low (connected to ground) to keep the VSC PHY fully active when it is out of reset.		
REF_CLK_SEL	Output	Reference clock speed pin of the VSC PHY. Held high for selecting the 125 MHz reference clock speed		
RD_BC_ERROR	Output	CoreTSE receive error signal. Indicates the receive code group error. This signal is synchronous to RX_CLK_R and mapped to on-board LED4. The LED ON condition indicates an error in the received code group.		
SPISCLKO, SPISS, and SPISDO	Output	SPI controller signals to interface with the ZL30364 clock generation		
SPISDI	Input			



### 2.4.1.1 Mi-V Soft Processor

The Mi-V soft processor supports RISC-V processor-based designs. The Mi-V soft processor executes the application from the LSRAM mapped at 0x80000000. It configures the ZL30364 clock generation hardware through the CoreSPI IP and the VSC PHY through the CoreTSE\_AHB MDIO interface. It also configures the CoreTSE\_AHB registers using the AHB interface.

The following figure shows the Mi-V soft processor configuration, where the **Reset Vector Address** is set to 0x8000\_000. This is because in the Mi-V processor memory map, the memory range used for the AHB memory interface is 0x8000\_0000 to 0x8FFF\_FFC, and the memory range used for the AHB I/O interface is 0x6000\_0000 to 0x7FFF\_FFF.

#### Figure 4 • Mi-V Configurator

[	Configurator	-		×
	Mi-V RV32IMA_L1_AHB Configurator			
	Microsemi:MiV:MIV_RV32IMA_L1_AHB:2.3.100			
	Configuration			
	Reset Vector Address			
	Upper 16 bits (Hex) 0x8000 Lower 16 bits (Hex) 0x0		0	

### 2.4.1.2 PF\_SRAM\_AHBL\_AXI

This design uses two instances of PF\_SRAM\_AHBL\_AXI core—pf\_sram\_0 and PF\_LSRAM\_1\_0.

The pf\_sram\_0 IP is connected to Mi-V as an AHB slave using Core AXI4Interconnect. The LSRAM blocks are initialized with the user application code from the external SPI flash.

The processor uses the SRAM memory to execute the application. The following figure shows the LSRAM depth and the interface settings. The **Fabric Interface type** is selected AXI because the fabric interfaces with the Mi-V processor using Core AXI4Interconnect. The memory depth can be selected based on the application size. This design uses 512 KB RAM (131072 words).

Figure 5 •	PF SRAM	0
riguic o		_•

CrosemiSystemBuilder:PF_SRAM_AHBL_AXI:1.1.127  Port settings   Memory Initialization Settings    Memory Settings  SRAM type   SRAM	1
Memory Depth(in words)   131072	SRAM_AHB_AXI_UI_0

The PF\_LSRAM\_1\_0 is connected to the Mi-V MMIO interface using CoreAHBLite. This memory is used for Ethernet MAC transmit and receive buffers.



### Figure 6 • PF\_LSRAM\_1\_0

Memory Settings	
SRAM type SRAM   Memory Depth(in works) 2048	
Use Native Interface	
Interface Settings	SRAM AHB AXI UI (
Fabric Interface type AHBUte T Data Width 32 T	
AXI4 interface options	HRESETN
Address Width 32 Width of ID 8	AHBSlaveInterface
Write Interface	
Read Interface     Wrap Burst support	SRAM_AHB_AXI_UI

### 2.4.1.3 CoreAXI4Interconnect

The AXI interconnect bus must be configured to connect the Mi-V core with memory. The following figure shows the bus configuration and other configuration of CoreAXI4Interconnect.

### Figure 7 • CoreAXI4Interconnect Configurator

Configura	ation Mast	ter Configuration	Slave Config	uration	Crossbar Configu	ration	
🗆 Bus C	Configuration						
Numb	er of Masters: 1	•	Number of S	ilaves: 2	•		
ID W	idth: 3	<b>•</b>	Address Wid	ith: 32			
User	Width: 1						
🗆 Othe	r Configuration						
Numb	er of Threads:	1	•	Max Outstand	ing Transactions:	2	•
Slave	FIFO Address Dep	pth: 4		Slave FIFO Da	ta Depth:	4	
DWC	Address FIFO Dep	oth Ceiling 10		Read Arbitrati	on Enable:	<b>v</b>	

### Figure 8 • CoreAXI4Interconnect Configurator—Master Configuration

Configuration	Master Configuration	Slave Configuration Crossbar Configuration
Master0 Con	figuration	
M0 Type:	AHB-Lite 💌	M0 Data Width: 32 💌
M0 DWC Data	FIFO Depth: 16 💌	M0 Register Slice: 🔽
M0 Clock Doma	ain Crossing: 🗍	



Figure 9 • CoreAXI4Interconnect Configurator—Slave Configuration

	-	-	
Configuration Master C	Configuration Slave	Configuration Cro	ossbar Configuration
Slave0 Configuration			
S0 Type:	AXI4	S0 Data Width:	64
S0 DWC Data FIFO Depth:	16 💌	S0 Register Slice:	<b>v</b>
S0 SLAVE Start Address:	0x8000000	S0 SLAVE End Address:	0x800fffff
S0 Clock Domain Crossing:	Γ		
Slave1 Configuration			
S1 Type:	AXI4	S1 Data Width:	64
S1 DWC Data FIFO Depth:	16 💌	S1 Register Slice:	<b>v</b>
S1 SLAVE Start Address:	0x80100000	S1 SLAVE End Address:	0x80ffffff
S1 Clock Domain Crossing:	<b>v</b>		

### Figure 10 • CoreAXI4Interconnect Configurator—Crossbar Configuration

Configuration	Master Configuration	Slave Configuration	Crossbar Configuration	
Crossbar Architectur	e Configuration			
Crossbar Mode:	<b>v</b>			
Data Width Configura	ation			
Crossbar Data V	Vidth: 64 💌			



### 2.4.1.4 DDR3

The DDR3 subsystem is configured to access the 16-bit DDR3 memory through an AXI4 interface. The PolarFire evaluation kit DDR3 memory preset is applied to configure all of the memory initialization and timing parameters in the DDR configurator. The following figure shows general configuration settings for the DDR3 memory.

### Figure 11 • DDR3 Configuration

Configurator			— 🗆	X
PolarFire DDR3 (Pre-produ	ction)			
Microsemi:SystemBuilder:Pf_DDR3:2.3.201	General Momory Initialization	Momory Timing	Controllor Mins	
J	General Memory Initialization     Top	Memory Timing	Controller   Misc.	
<ul> <li></li></ul>	Protocol DDR3 💌			
MT41K1G8SN-125	Generate PHY only			
	Clock			
	Memory Clock Frequency (MHz)	666		
	CCC PLL Clock Multiplier	8 💌	]	
	CCC PLL Reference Clock Frequen	cy (MHz) 83.250		
	User Logic Clock Rate	QUAD -	]	
	User Clock Frequency	166.5	1	
	Topology			
	Memory Format			
	DQ Width	16 💌		
	SDRAM Number of Ranks	1 💌		
	Enable address mirroring on odd ra	anks 🗖		
	DQ/DQS group size	8 🔻		
Apply New preset	Row Address width	16		
	Column Address Width	11		
	Bank Address Width	3		
	Enable DM	DM		
	Enable Parity/Alert	Г		
	Enable ECC	Г		
	Number of clock outputs	1 .		
Help			OK Com	
incip.				4



### 2.4.1.5 PF\_CCC\_0

The PF\_CCC\_0 (PolarFire Clock Conditioning Circuitry) generates the fabric reference clock that drives the soft processor and the APB peripherals. The PF\_CCC\_0 IP is configured to generate one output fabric clock from a 50 MHz input.

The following figure shows the PF\_CCC\_0 input clock configuration.

*Figure 12* • **PF\_CCC\_0** Input Clock Configuration

Configuration PLL:Single Cock Options PLL Output Clocks Deput Frequency Input Frequency Delay Lines Finable Delay Line Finable Delay Line Fina	
Dack Options PLL Output Clocks     Input Frequency   50   Input Frequency   6   Badup Clock Delay Lines   6   Feedback Clock Delay   1 <t< th=""><th></th></t<>	
Input Frequency <td>: Clocks</td>	: Clocks
Input Frequency 50   Bandwidth Medum-Low     Index Lines   Index L	
Bandwidth Medium-Low     Delay Lines      Enable Delay Lines      Feedback Clock Delay Delay Steps: 1 ÷            Feedback Clock Delay     Bandwidth        Power / Jitter      Minimize Jitter      Minimize Jitter      Minimize Mode                       Power / Jitter           Minimize Mode <b>Power / Jitter Minimize Node Pre-CCC Pre-</b>	MHz 🗖 Backup Clock
Cleave Lines <ul> <li>Enable Delay Line</li> <li>Freedback Clock Delay</li> <li>Delay Steps: 1 ÷</li> <li>Beckup Clock Delay</li> </ul> <ul> <li>Power / Jitter</li> <li>Minimize Ntco*</li> <li>Minimize Noc*</li> </ul> <ul> <li>Feedback Mode</li> <li>Post-VCO ▼</li> </ul> <ul> <li>Features</li> <li>Integer Mode</li> </ul>	dium-Low 💌
Predback Clock Delay Delay Steps: 1 = =      PFCCCC_0      Power / Jitter     Minimize Jitter     Minimize Jitter     Minimize VC0*     Minimize Power      Feedback Hode      Post-VCO      Integer Mode	
I Enable Delay Line                  Freedback Clock Delay             Delay Steps: 1 ±	
C Feedback Clock Delay Steps: 1 ≤ C Backap Clock Delay OUTO_FABCLK Power / Jitter C Minimize Jitter C Minimize VC0* C Minimize Power Feedback Mode Post-VCO ▼ Integer Mode	PF CCC 0
© Badup Clock Delay     OUTO_FABCLK     Power / Jitter     © Minimize Itter   © Minimize VCO*   © Minimize Power     Pre-CCC      PFCCC	Delay Delay Steps: 1 🚊
Power / Jitter     Minimize Itter     Minimize VCO*     Minimize Power      Feedback Mode      Post-VCO      Features      Integer Mode	elay OUTO_FABCLK_
Power / Jitter  Minimize Jitter  Minimize VCO*  Minimize Power  Feedback Mode  Post-VCO  Feedback Mode  Integer Mode	
Image: Minimize Jitter   Image: Minimize VCO*   Image: Minimize Power     Image: Post-VCO ▼     Image: Mode     Image: Mode	
<ul> <li>∩ Minimize VCO*</li> <li>∩ Minimize Power</li> <li>□ Feedback Mode</li> <li>□ Post-VCO ▼</li> <li>□ Features</li> <li>□ Integer Mode</li> </ul>	∠ PF CCC
C Minimize Power ■ Feedback Mode Post-VCO ▼ ■ Features Thteger Mode	
Feedback Mode  Post-VCO  Features  Integer Mode	
Post-VCO   Features  Integer Mode	
Features Integer Mode	
Features      Integer Mode	
Integer Mode	
SSCG Modulation	

The following figure shows the PF\_CCC\_0 output clock configuration. The Mi-V processor supports up to 120 MHz. This design uses an 83.25 MHz system clock for configuring the APB peripherals.

Figure 13 • PF\_CCC\_0 Output Clock Configuration

Configuration PLL-Single	
Por best results, put the highest frequency inst.   Output Clock 0	PF_CCC_0
Requested Phase       0       Degrees       C       Actual Lower       0       Degrees       C       Actual Higher       0       Degree         Dynamic Phase Shifting       Expose Enable Port       Enable Bypass       REF_PREDIV         Image: Pabric Clock       Image: Pabric Clock (Gated)       Image: Higher Pabric Clock (Gated)       Image: Pabric Clock       Image: Pabric Clock (Gated)       Image: Pabric Clock (Gated)	
<ul> <li>Output Clock 1</li> <li>Output Clock 2</li> </ul>	
Output Clock 3	Symbol /



### 2.4.1.6 NWC\_PLL\_0

The NWC\_PLL\_0 block is used for IOD CDR. NWC\_PLL\_0 is configured in PLL-DLL cascaded mode to generate five output fabric clocks from a 125 MHz input. Four clocks are required for clock recovery and one clock for the fabric Tx interface of the CoreTSE block. The DLL is needed to control the clock position (delay) with DLL codes when the data is active on the Rx interface (RX\_P/RX\_N). A glitch-less DLL can adjust the clock delay setting when the data is active. Therefore, the HSIO clock frequency is selected as 625 MHz with four phases.

The following figure shows the NWC\_PLL\_0 input clock configuration.

*Figure 14* • NWC\_PLL\_0 Input Clock Configuration

Input Frequency       125       MHz       Badup Clock         Bandwidth       Medium-Low       Image: Control of the second	PF_CCC_0 orth orth: J.5.0 OX
	PF_CCC_0
Enable Delay Line	
<ul> <li>Feedback Clock Delay Delay Steps: 8</li></ul>	
G Backup Clock Delay	OUTO_HS_IO_CLK
	OUT1_HS_IO_0
	 -REF_CLK_0 OUT3_HS_JO_CL OLL_CODE_UPDATE PL_LOO
Power / Jitter	DUL_DELAY_DI
Minimize Jitter	PF_CCC
C Minimize VCO*	
C Minimize Power	
Feedback Mode	
Post-VCO 💌	
E Features	
I Integer Mode	
SSCG Modulation	



The following figure shows the NWC\_PLL\_0 output clock configuration. A bank clock is generated for 0, 90, 180, and 270 degrees, as shown in Figure 15, page 14. Output clock 2 is given as an input to the DLL. The output clock 0 (dedicated clock of 625 MHz) is sent to the clock divider to generate the 125 MHz frequency required for IOD CDR TX block and CoreTSE AHB block.

### Figure 15 • NWC\_PLL\_0 Output Clock Configuration

nguration  PLL-DLL Cascaded	•										
k Options Output Clocks	DLL										
best results, put the highest fre	equency i	first.									
Output Clock 0											
Enabled											
											1
Requested Frequency	625		MHz	C Act	ual Lower	625	MHz	۰	Actual Higher	625	MHz
Requested Phase	0		Degrees	O Act	ual Lower	0	Degrees	۰	Actual Higher	0	Degrees
Dynamic Phase Shiftin	g	Expose E	nable Port		🗌 Ena	able Bypas	s		REF_PRED	VIV	~
Fabric Clock		Fabric Clo	ck (Gated)		MS HS	I/O Clock			Dedication	ted Clock	
Output Clock 1											
✓ Enabled											
Requested Frequency	625		MH7	C Act	allower	625	MH7	¢	Actual Higher	625	MHz
Requested Phase	90		Degrees	C Act	ual Lower	90	Degrees	e	Actual Higher	90	Degrees
		<b>F</b> = -	-						-		
Dynamic Phase Shiftin     Eabric Clock	g	Expose E	nable Port		I Ena	able Bypas	s		REF_PREL	01V	
I duric Cluck		Example ( )	ck (Cated)		E HS	T/O Clock			Dedicat	ted Clock	
		Fabric Cio	ck (Gated)		I HS	I/O Clock				ted Clock	
<ul> <li>Output Clock 2</li> </ul>		Pabric Clo	ck (Gated)		₩ HS	I/O Clock			Dedica	ted Clock	
Output Clock 2     Enabled (Feeding DLL)			ick (Gated)		I HS	I/O Clock			C Dedica	ted Clock	
Output Clock 2  Enabled (Feeding DLL)			ck (Gated)		HS	I/O Clock			Dedicat	ted Clock	
Output Clock 2  Enabled (Feeding DLL)  Requested Frequency	625		MHz	C Acti	HS	I/O Clock	MHz	(•	C Dedical	625	MHz
Output Clock 2  Enabled (Feeding DLL)  Requested Frequency Requested Phase	625	1 Pabric Clo	MHz Degrees	C Acti	HS Jal Lower	I/O Clock 625 180	MHz Degrees	(° (°	C Dedical	625 180	MHz Degrees
Output Clock 2  Centre Enabled (Feeding DLL)  Requested Frequency Requested Phase  Dynamic Phase Shiftin	625 180	Expose Er	MHz Degrees	C Actu	HS Jal Lower Jal Lower	625 180 able Bypas	MHz Degrees	(° (°	Actual Higher	625 180	MHz Degrees
Output Clock 2  Enabled (Feeding DLL)  Requested Frequency Requested Phase  Dynamic Phase Shiftin Fabric Clock	625 180 g	Expose Er	MHz Degrees nable Port	C Actu C Actu	I HS Jal Lower Jal Lower I Ena I Ena I HS	625 180 I/O Clock	MHz Degrees	•	Actual Higher Actual Higher Actual Higher	625 180 DIV ted Clock*	MHz Degrees
Output Clock 2  Carlot Characteristics  Carlot Clock 2  Requested Frequency Requested Phase  Dynamic Phase Shiftin Fabric Clock	625 180 g	Expose Ei	MHz Degrees nable Port	C Actu	IF HS Jal Lower Jal Lower IF Ena IF HS	625 180 I/O Clock	MHz Degrees s	(° (°	C Dedical Actual Higher Actual Higher REF_PRED C Dedical	625 180 DIV ted Clock*	MHz Degrees
Output Clock 2  Enabled (Feeding DLL)  Requested Frequency Requested Phase Dynamic Phase Shiftin Fabric Clock Output Clock 3	625 180 g	Expose El	MHz Degrees nable Port ck (Gated)	C Actu	I HS ual Lower ual Lower I Ena I HS	625 180 able Bypas	MHz Degrees s	(° (°	C Dedicat	625 180 DIV ted Clock*	MHz Degrees 
Output Clock 2  For Enabled (Feeding DLL)  Requested Frequency Requested Phase  Dynamic Phase Shiftin Fabric Clock 3  Output Clock 3  For Enabled	625 180 g	Expose E	MHz Degrees nable Port ck (Gated)	C Actu	IF HS	625 180 able Bypas I/O Clock	MHz Degrees	(° (°	C Dedical	625 180 DIV ted Clock*	MHz Degrees
Output Clock 2  Carlot Clock 2  Requested Frequency Requested Phase  Dynamic Phase Shiftin Fabric Clock  Output Clock 3  Enabled	625 180 g	Expose Er	MHz Degrees nable Port ck (Gated)	C Actu	I HS Jal Lower Jal Lower I Ena I Ena I HS	625 180 able Bypas I/O Clock	MHz Degrees s	۰ ۲	C Dedicat	625 180 DIV ted Clock*	MHz Degrees
Output Clock 2  F Enabled (Feeding DLL)  Requested Frequency Requested Phase Dynamic Phase Shiftin Fabric Clock 3  Output Clock 3  Requested Frequency Requested Frequency	625 180 g	Expose El Fabric Clo	MHz Degrees nable Port ck (Gated) MHz	C Act	HS Jal Lower Jal Lower F Ena F HS Jal Lower	625 180 able Bypas I/O Clock	MHz Degrees S	•	Actual Higher Actual Higher REF_PRED Dedicat	625 180 DIV ted Clock*	MHz Degrees 
Output Clock 2	625 180 9 625 270	Expose El	MHz Degrees nable Port ck (Gated) MHz Degrees	C Acti	IF HS	625 180 able Bypas I/O Clock	MHz Degrees S MHz Degrees	•	Actual Higher Actual Higher REF_PRED Dedical Actual Higher Actual Higher	625 180 DIV ted Clock* 625 270	MHz Degrees T MHz Degrees
Output Clock 2  Carlot Enabled (Feeding DLL)  Requested Frequency Requested Phase Dynamic Phase Shiftin Fabric Clock 3  Cutput Clock 3  Requested Frequency Requested Frequency Requested Phase Dynamic Phase Shiftin	625 180 g 625 270 g	Expose El Expose El Expose El	MHz Degrees nable Port MHz Degrees nable Port	C Act	IF HS Jal Lower Jal Lower IF Ena Jal Lower Jal Lower Jal Lower	625 180 able Bypas 625 270 able Bypas	MHz Degrees s MHz Degrees s	• •	C Dedicat	625 180 DIV ted Clock* 625 270 DIV	MHz Degrees V MHz Degrees



The following figure shows the DLL configuration of NWC\_PLL\_0. The settings selected for DLL configuration are:

- Clock Modes: Phase Reference Mode.
- **Reference and Phase Shifting:** Output2. This indicates that the Output2 of the PLL is given as input to the DLL because this CCC was configured in the PLL-DLL cascaded mode.

Figure 16 • NWC\_PLL\_0 DLL Configuration

Configuration PLL-DLL Cascaded  Clock Options   Output Clocks DLL	
Clock Hodes  Chase Reference Mode  Phase Generation Mode  Dirjection Removal Mode*  Reference and Phase Shifting  Reference Clock 111 MHz Output2 Divide by 1  Outputs and Options	PF_CCC_0 OUT0_FABCLK_0- OUT2_FABCLK_0- PIL_LOCK_0- PIL_CODE_UPDATE DLL_CODE[7:0]- DLL_CODE_UPDATE DLL_CODE[7:0]- DLL_DELAY_DIFF- PF_CCC
Jitter Range     Low      pe Taps     0        □     Enable Dynamic Reconfiguration Interface (DRI)       □     Dynamic Code Mode	Symbol /

### 2.4.1.7 CORESPI\_0

The CORESPI0 (CoreSPI) block is a controller IP, which implements serial communication. Mi-V configures the ZL30364 clock generation hardware using the CORESPI\_0 block. The following points describe the CoreSPI configuration, as shown in the following figure.

- APB Data Width is selected as 32 because the design uses an APB data width of 32 bit.
- The default serial protocol mode, Motorola mode is retained to interface with ZL30364.
- Frame size is set to 16 to match the read/write cycles supported by ZL30364.
- FIFO depth is set to 32 to store maximum frames (TX and RX) in FIFO.
- The clock rate for the SPI master clock is selected as 7. This is used to generate the SPICLK, which
  is generated as PCLK/(2\*(clock rate+1) = 83.25/(2\*(7+1)).
- The Keep SSEL active checkbox is enabled to keep the slave peripheral active between back-toback data transfers.

The following figure shows the CoreSPI configuration.

### Figure 17 • CoreSPI\_0 Configuration

Configurator	-		$\times$
CoreSPI Configurator			
MicrosemiDirectCore:CORESPI:5.2.104			
Configuration			
APB Data Width: C 8 C 16 C 32			
-SPI Configuration			_
Mode:      Motorola Mode C TI Mode C NSC Mode			
Frame Size (4-32): 16	_		
FIFO Depth (1-32): 32			
Clock Rate (0-255): 7	_		
Motorola Configuration			-1
Mode:  Mode 0  Mode 1  Mode 2  Mode 2	: 3		
Keep SSEL active 🔽			
-TI/NSC Configuration			
Transfer Mode: C Normal C Custom			
Free running dock			
Jumbo frames			
NSC Specific Configuration Standard			
Testbench: User 💌			
License: RTL			
Help *	ок	Can	el



### 2.4.1.8 Core\_SPI\_FLASH\_0

This core is configured for accessing the external SPI flash for IAP. The configuration options are same as Figure 17, page 15.

### 2.4.1.9 CoreGPIO\_0\_0

This core is configured to control the on-board LEDs and switches.

### 2.4.1.10 PF\_SPI\_0

This macro is an interface between system controller SPI and CoreSPI controller.

### 2.4.1.11 Design Memory Map

The following figure shows the Mi-V processor bus interface memory map.

### Figure 18 • Mi-V Processor Bus Interface Memory Map

Modify Memory Map		×
Select Bus to View or Assign Peripheral(s)	A	ssign peripherals to addresses on bus:
AXI4Interconnect_0	Address	Peripheral
COREAHBLITE_2_0	0x00000000	COREAHBTOAPB3_0:AHBslave
	0x01000000	CORETSE_AHB_0_0:AHBS
	0x02000000	PF_LSRAM_1_0:AHBSIaveInterface
Help		OK Cancel
50 Modify Memory Man		
Select Bus to View or Assign Peripheral(s)	A	X
Select Bus to View or Assign Peripheral(s)	A Address	ssign peripherals to addresses on bus: Peripheral
Select Bus to View or Assign Peripheral(s) AXI4Interconnect_0 COREAHBLITE_2_0 CoreAPB3 0	A Address 0x0000000	xsign peripherals to addresses on bus: Peripheral CoreUARTapb_0:APB_bif
Select Bus to View or Assign Peripheral(s) AXI4Interconnect_0 COREAHBLITE_2_0 COREAHBLITE_2_0	A Address 0x0000000 0x00001000	Ssign peripherals to addresses on bus:  Peripheral CoreUARTapb_0:APB_bif CORESPI_0:APB_bif
Select Bus to View or Assign Peripheral(s) AXI4Interconnect_0 COREAHBLITE_2_0 CoreAPB3_0	A Address 0x00000000 0x00001000 0x00002000	Ssign peripherals to addresses on bus: Peripheral CoreUARTapb_0:APB_bif CORESPI_0:APB_bif core_spi_flash_0:APB_bif
Select Bus to View or Assign Peripheral(s) → AXI4Interconnect_0 ↔ COREAHBLITE_2_0 ← CoreAPB3_0	A Address 0x0000000 0x00001000 0x00002000 0x00003000	Sign peripherals to addresses on bus: Peripheral CoreUARTapb_0:APB_bif CORESPI_0:APB_bif core_spi_flash_0:APB_bif coreGPIO_0_0:APB_bif
Select Bus to View or Assign Peripheral(s) → COREAHBLITE 2_0 CoreAPB3_0	A Address 0x0000000 0x00001000 0x00002000 0x00002000 0x00004000	X ssign peripherals to addresses on bus: Peripheral CoreUARTapb_0:APB_bif CORESPI_0:APB_bif core_spi_flash_0:APB_bif coreGPIO_0_0:APB_bif PF_CORE_SYSTEM_SERVICES_0
Select Bus to View or Assign Peripheral(s) AXI4Interconnect_0 COREAHBLITE_2_0	A Address 0x0000000 0x0001000 0x00002000 0x00003000 0x00004000	× ssign peripherals to addresses on bus: Peripheral CoreUARTapb_0:APB_bif CORESPI_0:APB_bif core_spi_flash_0:APB_bif coreGPIO_0_0:APB_bif PF_CORE_SYSTEM_SERVICES_0:



### 2.4.1.12 CoreAHBLite\_2

CoreAHBLite\_2 is configured as shown in the following figure to interface the APB peripherals to the Mi-V processor at 0x6000\_0000.

### Figure 19 • CoreAHBLite\_2 Configuration

Memory space:			256MB addressable spa	ce apporti	oned into 16 slave slots, each of size 16	MB	<b>•</b>	
Address range seen by slave connecte	d to huge (2	GB) slot interface:	0x00000000 - 0x7FF	FFFFF	© 0x80000000 - 0xFFFFFFF	=		
cate memory space to combined region s	lave							
Slot 0: 🔽 Slot 1: 🖵	Slot 2:	Slot 3:						
Slot 4: 🗖 Slot 5: 🗖	Slot 6:	Slot 7:						
Slot 8: 🗖 Slot 9: 🗖	Slot 10: 🕅	Slot 11:						
Slot 12: 🔽 Slot 13: 🗖	Slot 14: 🕅	Slot 15:						
able Master access								
M0 can access slot 0:	<b>V</b>	M1 can access slot	o: r		42 can access slot 0:		M3 can access slot 0:	
M0 can access slot 1:	<b>V</b>	M1 can access slot	1:		M2 can access slot 1:		M3 can access slot 1:	
M0 can access slot 2:	<b>v</b>	M1 can access slot	2:	Z 1	12 can access slot 2:	<b>v</b>	M3 can access slot 2:	
M0 can access slot 3:		M1 can access slot	3: П		42 can access slot 3:		M3 can access slot 3:	
M0 can access slot 4:		M1 can access slot	4: Г		12 can access slot 4:		M3 can access slot 4:	
M0 can access slot 5:		M1 can access slot	5: Г		42 can access slot 5:		M3 can access slot 5:	
M0 can access slot 6:		M1 can access slot	6: Г		42 can access slot 6:		M3 can access slot 6:	
M0 can access slot 7:		M1 can access slot	7:		12 can access slot 7:		M3 can access slot 7:	
M0 can access slot 8:	Γ	M1 can access slot	8: Г		12 can access slot 8:		M3 can access slot 8:	
M0 can access slot 9:	Γ	M1 can access slot	9: Г		42 can access slot 9:		M3 can access slot 9:	
M0 can access slot 10:		M1 can access slot	10:		M2 can access slot 10:		M3 can access slot 10:	
M0 can access slot 11:		M1 can access slot	11:		M2 can access slot 11:		M3 can access slot 11:	
M0 can access slot 12:		M1 can access slot	12:		42 can access slot 12:		M3 can access slot 12:	
M0 can access slot 13:		M1 can access slot	13: Г		42 can access slot 13:		M3 can access slot 13:	
M0 can access slot 14:		M1 can access slot	14:		42 can access slot 14:		M3 can access slot 14:	
M0 can access slot 15:		M1 can access slot	15: Г		12 can access slot 15:		M3 can access slot 15:	
M0 can access slot 16 (combined/huge)	. E	M1 can access slot	16 (combined/huge): Γ		M2 can access slot 16 (combined/huge):	Г	M3 can access slot 16 (combined/huge):	Г



### 2.4.1.13 CoreAPB3\_0

CoreAPB3\_0 is configured as shown in the following figure to connect the peripherals CoreSPI, Core\_SPI\_Flash, CoreGPIO, PF\_SYSTEM\_SERVICES, and CoreUARTapb as slaves.

- APB Master Data bus width: 32 bit
- A number of address bits are driven by the master: 16. The Mi-V processor addresses slaves using 16-bit addressing, so the final address for these slaves translates to 0x6000\_0000, 0x6000\_1000, and 0x6000\_2000
- Enabled APB Slave Slots: S0, S1, S2, S3, and S4 (for CoreSPI, Core\_SPI\_Flash, CoreGPIO, PF\_SYSTEM\_SERVICES, and CoreUARTapb, respectively.

```
Figure 20 • CoreAPB3 Configuration
```

Configuration	
APB Master Data Bus Width       32-bit C 16-bit C 8-bit	
Audress configuration	_
Number of address bits driven by master: 16	<u>-</u>
Position in slave address of upper 4 bits of master address: [27:24] (Ignored if master address width >= 32 bits)	•
Indirect Addressing: Not in use	•
Allocate memory space to combined region slave	
Slot 0: 🔽 Slot 1: 🗖 Slot 2: 🗖 Slot 3: 🗖	
Slot 4: 🗖 Slot 5: 🗖 Slot 6: 🗖 Slot 7: 🗖	
Slot 8:  Slot 9:  Slot 10:  Slot 11:	
Slot 12: 🗖 Slot 13: 🗖 Slot 14: 🗖 Slot 15: 🗖	
Enabled APB Slave Slots	
Slot 0: 🔽 Slot 1: 🔽 Slot 2: 🔽 Slot 3: 🔽	
Slot 4: 🔽 Slot 5: 🗆 Slot 6: 🗔 Slot 7: 🗖	
Slot 8:  Slot 9:  Slot 10:  Slot 11:  Slot 11:	
Slot 12: 🗖 Slot 13: 🗖 Slot 14: 🗖 Slot 15: 🗖	
Testbench: User	
License: C Obfuscated C RTL	

### 2.4.1.14 COREAHBTOAPB3\_0

The COREAHBTOAPB3 IP connects to CoreAPB3. This IP retains the default configuration.



## 2.5 Clocking Structure

In the demo design, there are two clock domains—the on-board 50 MHz oscillator and the on-board ZL30364 clock generation hardware.

- On-board 50 MHz oscillator: This oscillator drives the PLL that generates an 83.25 MHz clock for the Mi-V soft processor and peripherals. The Mi-V soft processor can operate up to 120 MHz. In this design, the Mi-V processor runs at 83.25 MHz.
- On-board ZL 30364 clock generation hardware: This hardware generates the reference clocks for the VSC PHY and the IOD CDR fabric module.

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







## 2.6 SoftConsole Firmware Project

The following stacks available in the SoftConsole Project Explorer are used in this demo design.

- IwIP TCP/IP stack v1.4.1
- FreeRTOS

•

The following figure shows the directory structure of the Webserver SoftConsole project (located at <\$design\_file\_directory>\mpf\_dg0834\_df\webserver\libero\SoftConsole). It contains the Webserver application (which uses LWIP and FreeRTOS) and all the firmware and hardware abstraction layers that correspond to the hardware design.

#### Figure 22 • Directory Structure of Webserver SoftConsole Project



The following figure shows the directory structure of the TFTP\_IAP SoftConsole project (located at <\$design\_file\_directory>\mpf\_dg0834\_df\tftp\_iap\libero\SoftConsole). It contains the IAP application, TFTP server (which uses LWIP and FreeRTOS) and all the firmware and hardware abstraction layers that correspond to the hardware design.

#### Figure 23 • Directory Structure of TFTP IAP SoftConsole Project





# 3 Libero Design Flow

This chapter describes the Libero design flow for running this demo design, which includes:

- Synthesize, page 22
- Place and Route, page 22
- Verify Timing, page 22
- Generate FPGA Array Data, page 23
- Configure Design Initialization Data and Memories, page 23
- Generate Bitstream, page 24
- Export FlashPro Express Job, page 24
- Run PROGRAM Action, page 25
- Program SPI Flash Image, page 26

The Libero project is available at the following design files folder location:

- Webserver: mpf\_dg0834\_df\webserver\libero
- TFTP\_IAP: mpf\_dg0834\_df\tftp\_iap\libero

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

#### Figure 24 • Libero Design Flow Options

🖻 🕨 Constraints
🔤 🔂 Manage Constraints
🖌 🖻 🕨 Implement Design
🖓 Open Netlist Viewer
🖌 🚽 🔁 Synthesize
Verify Post-Synthesized Design
-• Generate Simulation File
🔤 Simulate
V Blace and Route
Verify Post Layout Implementation
🖌 🔤 🖓 Verify Timing
🗠 💩 Open SmartTime
🖹 Verify Power
🖓 Open SSN Analyzer
🖻 🕨 Configure Hardware
Programming Connectivity and Interface
- 🔊 Configure Programmer
- 🔊 Select Programmer
Program Design
🖌 🛶 🗍 Generate FPGA Array Data
Configure Design Initialization Data and Mem
✓ Generate Design Initialization Data
Configure I/O States During JTAG Programming
Configure Programming Options
🛛 🐼 Configure Security
- 🚱 Configure Permanent Locks (OTP)
V Generate Bitstream
V Run PROGRAM Action
🖌 🖻 🕨 Program SPI Flash Image
V Generate SPI Flash Image
V Run PROGRAM_SPI_IMAGE Action
🖻 🕨 Debug Design
<ul> <li>Generate SmartDebug FPGA Array Data</li> </ul>
SmartDebug Design
🖳 🔍 Identify Debug Design
Handoff Design for Production
🖌 📲 Export Bitstream
🖌 📲 Export FlashPro Express Job



## 3.1 Synthesize

To synthesize the design, perform the following steps:

- On the Design Flow tab, double-click Synthesize. When the synthesis is successful, a green tick mark appears next to Synthesize, as shown in the preceding figure.
- 2. Right-click **Synthesize** and select **View Report** to view the synthesis report and log files in the **Reports** tab.

## 3.2 Place and Route

The demo project includes the IO PDC file and the floor planner PDC constraint files. The Place and Route process uses these PDC files to place the I/Os.

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

- On the Design Flow tab, double-click Place and Route. When place and route is successful, a green tick mark appears next to Place and Route, as shown in Figure 24, page 21.
- 2. Right-click **Place and Route** and select **View Report** to view the place and route report and the log files in the **Reports** tab.

### 3.2.1 Resource Utilization

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

Туре	Used	Total	Percentage
4LUT	58856	299544	19.65
DFF	44408	299544	14.83
I/O register	0	1536	0.00
User I/O	87	512	16.99
- Single-ended I/O	75	512	14.65
– Differential I/O pairs	6	256	2.34

### Table 3 • Resource Utilization

## 3.3 Verify Timing

To verify timing, perform the following steps:

 On the Design Flow tab, double-click Verify Timing. When the design successfully meets the timing requirements, a green tick mark appears next to Verify Timing, as shown in Figure 24, page 21.



## 3.4 Generate FPGA Array Data

To generate FPGA array data, perform the following step:

1. On the **Design Flow** tab, double-click **Generate FPGA Array Data**.

When the FPGA array data is successfully generated, a green tick mark appears next to **Generate FPGA Array Data**, as shown in Figure 24, page 21.

## 3.5 Configure Design Initialization Data and Memories

The **Configure Design Initialization Data and Memories** option creates the LSRAM initialization client. When the PolarFire device powers up, the LSRAM memory is initialized with the sNVM contents.

To create the LSRAM initialization client, perform the following steps:

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

#### Figure 25 • Configure Design Initialization Data and Memories Option



2. In the **Configure Design Initialization Data and Memories** window, select the **Fabric RAMs** tab, and then select the **pf\_sram** file to import the memory information, as shown in the following figure.

Figure 26 • Fabric RAMs Tab

Jsage statistics LSRAM Memory Available Memory(Bytes): 2437120	Clents Load design configuration Edit Initialize all clents from: User Selection  Filter Inferred RAMs
Used Memory(Bytes): 1423360 Free Memory(Bytes): 1013760	Logical Instance Name
	33 iog.cdr.test/ddr3_subsystem_0/DDRPHY_BLK_0000_TRAINING occorpond_training and automative substantion and and a substantiation of the substantiation of
	35 log_cd_test/dd3_subsystem_0/DDRPHY_BLC0 Clent name: log_cd_test/dd3
	36 iog_cdr_test/ddr3_subsystem_0/DDRPHY_BLK_0 RAM Initialization Options RAM Initialization Options RAM Initialization Options
Used space Free space	Kog_cot_ceteded_cetes/tetrophot_cetes/cetes
uSRAM Memory Available Memory(Bytes): 266112	39 jog_cdr_test/ddr3_subsystem_0/MSC_i_0/MSC_i_ 40 jog_cdr_test/ddr3_subsystem_0/MSC_i_0/MSC_i_ C Content filed with 0s
Used Memory(Bytes): 8928 Free Memory(Bytes): 257184	log_cdr_test/ddr3_subsystem_0/MSC_i_0/MSC_i
	42 iog_cdr_test/ddr3_subsystem_0/MSC_i_0/MSC_i 0ptimize for: C High Speed C Low power 43 iog_cdr_test/ddr3_subsystem_0/MSC_i_0/MSC_i C terms Tage T Total T
	4 iog_cdr_test/ddr3_subsystem_0/MSC_i_0/MSC_i_
	45 jog_cdr_test/ddr3.subsystem_0/MSC_i_0/MSC_i_ 46 jog_cdr_test/ddr3.subsystem_0/MSC_i_0/MSC_i_10/MSC_i_4/MSC_i_8/MSC_i_12/mam/690]
Used space Free space	47 iog_cdr_test/pf_sram

3. Import the hex file (gbe\_webserver.hex for Webserver design and gbe\_tftp\_iap.hex for TFTP\_IAP design) provided with the design files from

mpf\_dg0834\_df\tftp\_iap\libero\gbe\_webserver.hex or mpf\_dg0834\_df\tftp\_iap\libero\gbe\_tftp\_iap.hex. The gbe\_webserver.hex or gbe\_tftp\_iap.hex file is a application file generated using SoftConsole that configures the ZL clock generation hardware, the CoreTSE\_AHB registers, and the VSC PHY. The application code is initially stored in an external SPI flash. On device power-up, the system controller copies the code to LSRAM from external SPI flash, and the Mi-V processor executes the code from LSRAM. To ensure that the fabric LSRAM contents are stored in external SPI flash, select Storage Type SPI flash, as shown in Figure 26, page 23.

4. Click Apply.



5. Select **Start address for SPI-FLash clients** and **SPI Clock divider value**, in the **Design Initialization** tab as shown in the following figure.

Figure 27 • Start Address for SPI Flash Clients

Design Initialization uPROM SNVM SPI Flash Fabric RAMs	
Apply         Discard         Help           In design initialization, user design blocks such as LSRAM, µSRAM, transceivers, and PCIe can be initialized as an option using data stored in the non-volatile storage memory.           The initialization data can be stored in µPROM, sNVM, or an external SPI Flash.	
Follow the below steps to program the initialization data: 1. Set up your fabric RAMs initialization data, if any, using the 'Fabric RAMs' tab 2. Define the storage location of the initialization data 3. Generate the initialization clients 4. Generate or export the bitstream 5. Program the device — Design bitalization specification	
Trirst stage (sww)	
-Second stage (sNVM)	
In the second stage, the initialization sequence initializes the PCIe and XCVR blocks present in the design.	
Start address for second stage initialization client: 0x 00000000	
Third stage (sNVM/uPROM/SPI-Flash)	
In the third stage, the initialization sequence initializes the Fabric RAMs present in the design	
To save the initialization instructions in sNVM/uPROM/SPI-Flash, please use 'Fabric RAMs' tab to make your selection for each RAM client.	
Start address for sNVM clients: 0x 00000000 sNVM start page: 0	
Start address for uPROM clients: 0x 0000000	
V Start address for SPI-Flash clients: 0x 00002000	
SPI-Hash binding: SPI-Hash - No-binding Plaintext	
Time Out (s): 128	
Auto Calibration Time Out (ms): 3000	
	1

- **Note:** The default start address for SPI-Flash clients 0x400 is used for Webserver design. The start address for TFTP design is modified to 0x2000. This is required to support flash erase of 4 KB while writing the SPI directory into initial SPI flash 1 KB memory using design firmware.
  - 6. On the Design Flow tab, double-click Generate Design Initialization Data. When the LSRAM initialization client is successfully generated in sNVM, a green tick mark appears next to Generate Design Initialization Data, as shown in Figure 24, page 21. When the device is programmed, the LSRAM block is initialized from the sNVM.

### 3.6 Generate Bitstream

To generate the bitstream, perform the following steps:

- On the Design Flow tab, double-click Generate Bitstream. When the bitstream is successfully generated, a green tick mark appears next to Generate Bitstream, as shown in Figure 24, page 21.
- 2. Right-click **Generate Bitstream** and select **View Report** to view the corresponding log file in the **Reports** tab.

## 3.7 Export FlashPro Express Job

To generate .job file, perform the following steps:

On the **Design Flow** tab, double-click Export FlashPro Express Job and select Design and SPI Flash as shown in figure. The exported job file contains the data contents to be programmed into PolarFire FPGA and external SPI flash. This Job file is utilized in FlashPro Express software to program both Device and external SPI flash as shown in Appendix 3: Programming the Device and External SPI Flash Using FlashPro Express, page 42.



### Figure 28 • Export FlashPro Express Job

Export FlashPro Express Job	×
Design settings Programming options set with the Configure Programming Options tool: Design version - 1 Back Level version - 0 Security options set with the Configure Security tool: Encrypt bitstream with default key. No user keys or security settings are enabled.	
FlashPro Express Job file Configured device chain with bitstream files and programmer settings will be included in the programming job.	
Name:         gbe_tftp_iap_v1         Location:         osocv12p2_df\tftp_iap\libero\designer\top\export	
Existing files: gbe_tftp_iap_v1.job top.job	
Program (JTAG programming interface): ↓ Design ↓ SPI Flash	
Design bitstream file format: PPD (*.ppd) 💌 🕕	
File to program at trusted fadility	
Zeroization actions:  Like New (Erases all user data; device can be immediately reprogrammed by user)  Unrecoverable (Erases all data and destroys reprogrammability; device must be scrapped)	
Help OK Cano	el

## 3.8 Run PROGRAM Action

After generating the bitstream, the PolarFire device must be programmed. Follow these steps to program the PolarFire device using the Libero design flow:

- **Note:** If you want to program the PolarFire FPGA using the .job file instead, see Appendix 3: Programming the Device and External SPI Flash Using FlashPro Express, page 42.
  - 1. Ensure that the jumper settings on the board are as listed in the following table.

Jumper	Setting
J18, J19, J20, J21, and J22	Close pins 2 and 3 for programming through FTDI
J28	Close pins 1 and 2 for programming through the on-board FlashPro5
J4	Close pins 1 and 2 for switching the power manually using SW3
J12	Close pins 3 and 4 for 2.5 V

#### Table 4 • Jumper Settings

- 2. Connect the power supply cable to the **J9** connector on the board.
- 3. Connect the USB cable from the host PC to **J5** (FTDI port) on the board.
- 4. Connect any one of the open network 1G Ethernet capable ports to the **J15** connector (RJ45-PORT 0) on the board.
- 5. Power up the board using the **SW3** slide switch.
- 6. On the Libero **Design Flow** tab, double-click **Run PROGRAM Action**.
- When the device is successfully programmed, the LEDs 6 and 7 on the board glow, and a green tick mark appears, as shown in the following figure.



### Figure 29 • Run Program Action



7. Right-click **Run Program Action** and select **View Report** to view the corresponding log file in the **Reports** tab.

## 3.9 Program SPI Flash Image

To program SPI Flash Image, perform the following steps:

- 1. Double-click **Generate SPI Flash Image** and double-click **Run PROGRAM\_SPI\_IMAGE Action** to get the SPI flash programmed with the application as shown in the following figure.
- **Note:** If you want to program the external SPI flash using the .job file instead, see Appendix: Programming the Device Using FlashPro Express, page 41.

### Figure 30 • SPI Flash Programming



 Power-cycle the board once you program the PolarFire device and external SPI flash. The demo is ready to be run. For information about how to run the demo, see Running the Demo, page 27.



# 4 Running the Demo

To run the demo design, perform the following steps:

- 1. For demo design files download link: http://soc.microsemi.com/download/rsc/?f=mpf\_dg0834\_df
- 2. Power up the board using the SW3 slide switch.
- 3. Start a serial terminal emulation program such as HyperTerminal, PuTTY, or TeraTerm.
- Note: For this demo, TeraTerm is used.

For more information about configuring serial terminal emulation programs, see *Configuring Serial Terminal Emulation Programs Tutorial*.

### 4.1 Tera Term Setup

The user application provides a user interface on the Tera Term terminal through the UART interface.

To set up the Tera Term program, perform the following steps:

- 1. Ensure that the USB cable connects the host PC to the **J5** (USB) port on the PolarFire Evaluation board.
- 2. Start Tera Term.
- 3. Select Serial as the Connection type.
- 4. Set the Serial **Port** to the second highest COM port number from the drop-down list as shown in the following figure. For example, **COM33: FlashPro5 Port [COM33]** in this instance.

#### Figure 31 • Select Serial as the Connection Type

SSH	SSH version: SSH2
Service: O Telnet	TCP port#: 22

5. In the Tera Term window, go to Setup > Serial port..., set Baud rate to 115200



Rest of the serial port settings must be at default state as shown in the following figure and click OK.

### Figure 32 • Tera Term Configuration

Tera Term: Serial port setup				Х
Port:	COM33	$\sim$	ОК	
Baud rate:	115200	~		
Data:	8 bit	$\sim$	Cancel	
Parity:	none	$\sim$		
Stop:	1 bit	$\sim$	Help	
Flow control:	none	$\sim$		
Transmit delay	:har O	mse	c/line	

6. In the **Tera Term** window, go to **Setup > General...**, set the **Language** to **English** and click **OK**, as shown in the following figure. This setup is required for running the Tera Term macro script.

#### Figure 33 • Tera Term General Setup

Tera Term: General set	tup	×
Default port:	COM	33 ~
Language:	English	~
LanguageUI:	Default.Ing	~
ОК	Cancel	Help

This completes the Tera Term program setup.

### 4.2 Running Webserver Demo

This section describes how to run the Webserver. The following procedure assumes that the serial terminal is setup, for more information about setting up the serial terminal, see Tera Term Setup, page 27.

Before you begin:

- 1. Connect the power supply cable to the **J9** connector on the board.
- 2. Connect the USB cable from the host PC to **J5** (FTDI port) on the board.
- 3. Open pin 1 and 2 of the **J23** jumper.
- 4. Connect any one of the open network 1G Ethernet capable ports to the **J15** connector (RJ45-PORT 0) on the board.
- 5. Power-up the board using the **SW3** slide switch.
- 6. Ensure that the device is programmed with the gbe\_webserver.job file and external SPI flash is programmed with the application. See Program SPI Flash Image, page 26 to program the external SPI flash.

After the device is programmed, power cycle the board. The application prints a welcome message with an IP address on the Tera Term program through the UART interface, as shown in following figure.

#### Figure 34 • Tera Term with IP Address

SCOM65:115200baud - Tera Term VT	_	×
File Edit Setup Control Window Help		
/*CoreTSE WebServer using Mi-V SoftProcessor*/ Acquiring IP address 10.60.132.61 D		^

Open a web browser, and enter the IP address displayed on the address bar of the browser. The PolarFire Webserver demo page appears, as shown in the following figure. To use the design in static IP mode, see Running the Design in Static IP Mode, page 40.



### Figure 35 • Webserver Demo Page

C (I) Not secure 1	0.60.132.61					\$	<b>8</b>
۸ 🌍	licrosemi		RISC-V + ( FreeRTOS	Core] + Iw	ГSE IP		
Ethernet	Receive statistics		CoreTSE Stat TX-RX statist	istics	Transmit statistics		
Port Parameter	RX_BYTE_CNT:	386416	FRAME_CNT_64:	1863	TX_BYTE_CNT:	241774	
Address: c0.b1:3c:61	50:50 RX_PKT_CNT:	3637	FRAME_CNT_127:	1732	TX_PKT_CNT:	896	
Address: 10.60.132	2.61 RX_FCS_ERR_CNT:	0	FRAME_CNT_255:	369	TX_MULTICAST_PKT_CNT:	0	
Speed: 1000Mb	PS RX_MULTICAST_PKT_CNT:	1209	FRAME_CNT_511:	300	TX_BROADCAST_PKT_CNT:	10	
Mode: full dupl	RX_BROADCAST_PKT_CNT:	1454	FRAME_CNT_1K:	268	TX_PAUSE_PKT_CNT:	0	
	RX_CTRL_PKT_CNT:	0	FRAME_CNT_MAX:	1	TX_DEFFERAL_PKT_CNT:	0	
	RX_PAUSE_PKT_CNT:	0	FRAME_CNT_VLAN:	0	TX_EXC\$_DEFFERAL_PKT_CNT:	0	
	RX_UNKNOWN_OPCODE_CNT:	0			TX_SINGLE_COLL_PKT_CNT:	0	
	RX_ALIGN_ERR_CNT:	0			TX_MULTI_COLL_PKT_CNT:	0	
	RX_FRAMELENGTH_ERR_CNT:	0			TX_LATE_COLL_PKT_CNT:	0	
	RX_CODE_ERR_CNT:	0			TX_EXC\$\$_COLL_PKT_CNT:	0	
	RX_CS_ERR_CNT:	1			TX_TOTAL_COLL_PKT_CNT:	0	
	RX_UNDERSIZE_PKT_CNT:	0			TX_PAUSE_HONORED_CNT:	0	
	RX_OVERSIZE_PKT_CNT:	0			TX_DROP_CNT:	0	
	RX_FRAGMENT_CNT:	0			TX_JABBER_CNT:	0	
	RX_JABBER_CNT:	0			TX_FC\$_ERR_CNT:	0	
	RX_DROP_CNT:	1311			TX_CNTRL_PKT_CNT:	0	
					TX_OVERSIZE_PKT_CNT:	0	
					TX_UNDERSIZE_PKT_CNT:	0	
					TX_FRAGMENT_CNT:	0	

## 4.3 Running TFTP Demo

This section describes how to run the IAP using TFTP. The following procedure assumes that the serial terminal is setup, for more information about setting up the serial terminal, see Tera Term Setup, page 27.

Before you begin:

- 1. Connect the power supply cable to the **J9** connector on the board.
- 2. Connect the USB cable from the host PC to **J5** (FTDI port) on the board.
- 3. Open pin 1 and 2 of the J23 jumper.
- 4. Power-up the board using the **SW3** slide switch.
- 5. Ensure that the device is programmed with the gbe\_tftp\_iap\_v1.job file and external SPI flash is programmed with the application. See Program SPI Flash Image, page 26 to program the external SPI flash.
- 6. Enable TFTP client in Host PC. To enable the TFTP client in Host PC, see Appendix 1: Enable TFTP Client, page 35.

After power-up, Tera Term displays the options as shown in the following figure. Observe the design version **01** in the device.

#### Figure 36 • Tera Term Window

<u></u>	COM6	5:115200	)baud - Te	ra Term VT						_	$\times$
File	Edit	Setup	Control	Window	Help						
Des 32b *** 1.C 2.C 3.R 4.R 5.P 6.P	ign Ve it USI «User opy If opy If in IAI in IAI rograf rograf	ersion ERCODI AP ima AP ima P imag P imag n Devi n Devi	ACMSB f SCSILIC DAS MGE1 to MGE2 to JE1 aut JE2 aut JE2 aut LCE wit	irst): on Sign SPI fl. SPI fl. hentica hentica h IAP i h IAP i	00 01 ature (MS) ash addres tion tion mage1 mage2	B first) ss 0xA00 ss 0x140	: 1 000 0000	L234567 using Jusing	8 TFTP TFTP		^

- 1. Press 1 to load IAP Image1 to SPI flash address 0xA00000 using TFTP.
- 2. Press e to erase the SPI flash memory location (0xA00000 0x13FFFFF).



#### Figure 37 • Erasing the SPI Flash Memory Location [0xA00000 - 0x13FFFFF]

🜉 COM65:115200baud - Tera Term VT	_		×
File Edit Setup Control Window Help			
			^
Design Version(MSB first): 00 01 32bit USEBCODE/Silicon Signature (MSB first): 12345678			
****User options****** 1.Copy IAP image1 to SPI flash address 0xA00000 using TFTP 2.Copy IAP image2 to SPI flash address 0x1400000 using TFTP			
3.Run IAP image1 authentication 4.Run IAP image2 authentication 5.Program Device with IAP image1 6.Program Device with IAP image2			
Press 'e' to erase spi flash memory			
SPI Flash Memory [0xA00000 - 0x13FFFFF] erase is in progress	please	wait.	
SPI Flash Memory is erased successfully.			-

3. After completion of the SPI flash erase operation, the Ethernet link is up, and the IP address is displayed on the Tera Term terminal. In this example, the IP address is 10.60.132.61. The TFTP command uses this IP address to transfer the file to the external SPI flash. The LED G1 on the PolarFire Evaluation Kit board starts blinking. To use the design in static IP mode, see Running the Design in Static IP Mode, page 40.

Figure 38 • Acquiring IP Address

🚇 COM65:115200baud - Tera Term VT	_		$\times$
File Edit Setup Control Window Help			
Design Version(MSB first): 00 01 32bit USERCODE/Silicon Signature (MSB first): 12345678 ****User options*****			^
1.Copy IAP image1 to SPI flash address 0xA00000 using TFTP 2.Copy IAP image2 to SPI flash address 0x1400000 using TFTP 3.Run IAP image1 authentication 4.Run IAP image2 authentication 5.Program Device with IAP image1 6.Program Device with IAP image2			
Press 'e' to erase spi flash memory			
SPI Flash Memory [0xA00000 - 0x13FFFFF] erase is in progress	please	wait.	
SPI Flash Memory is erased successfully.			
/*CoreTSE IFIP Server using Mi-U Soft Processor*/ Acquiring IP address			~

- 4. On the Host PC command prompt, browse to the folder <\$design file directory>\mpf\_dg0834\_df\tftp\_iap\iap\iap\_images
- 5. Type the tftp -i 10.60.132.61 PUT iog\_cdr\_tftp\_iap\_v2.spi command to transfer the iog\_cdr\_tftp\_iap\_v2.spi programming file to the SPI flash as shown in the following figure.

Figure 39 • Transfer Programming Image1

#### Command Prompt - C × Microsoft Windows [Version 10.0.15063] (c) 2017 Microsoft Corporation. All rights reserved. C:\Users\ >cd C:\Users\divyesh.patel\Documents\mpf\_dg0834\_liberosocv12p0\_df\tftp\_iap\iap\_images C:\Users\ \mpf\_dg0834\_liberosocv12p0\_df\tftp\_iap\iap\_images>tftp -i 10.60.132.61 PUT iog\_cdr\_tftp\_ iap\_v2.spi Transfer successful: 9478672 bytes in 178 second(s), 53250 bytes/s

 Wait until total bytes received message is displayed on the Tera Term, to ensure that the programming image1 is transferred to the SPI flash. On completion of the Image1 transfer, the user options are displayed.



### Figure 40 • Bytes Received for Image1

CONICO: 115200baud - Tera Term V I			
File Edit Setup Control Window Help			
SPI Flash Memory [0xA00000 - 0x13FFFFF] erase is in progress	. please	wait	^
SPI Flash Memory is erased successfully.			
/*CoreTSE TFTP Server using Mi-U Soft Processor*/ Acquiring IP address			
Bytes received=9478672 ****Mlser options****** 1.Copy IAP image1 to SPI flash address 0xA00000 using TFTP 2.Copy IAP image1 to SPI flash address 0x1400000 using TFTP 3.Run IAP image1 authentication 4.Run IAP image2 authentication			
5.Program Device with IAP image1			
D.Frugram Device with the image2			$\checkmark$

- 7. Press 2 to load IAP Image2 to SPI flash address 0x1400000 using TFTP.
- 8. Press e to erase the SPI flash memory location (0x1400000 0x1DFFFFF).

Figure 41 • Erasing the SPI Flash Memory Location [0x1400000 - 0x1DFFFFF]

📒 co	M65:115200baud - Tera Term VT	_		$\times$
File Ed	lit Setup Control Window Help			
Acquir 10.60	ing IP address			^
				:::
	~			
		• • • • • •		••••
Bytes ****Us 1.Copy 2.Copy 3.Run 4.Run 5.Prog 6.Prog	received=9478672 er options=xxxxxx IAP image1 to SPI flash address 0xA00000 using TFTP IAP image2 to SPI flash address 0x1400000 using TFTP IAP image1 authentication IAP image1 authentication ram Device with IAP image1 ram Device with IAP image2			
Press	'e' to erase spi flash memory			
SPI F1	ash Memory [0x1400000 - 0x1DFFFF] erase is in progress ;	please	e wait	
ŠPI FI □	ash Memory is erased successfully.			· 🗸

- 9. On the Host PC command prompt, make sure to browse the folder <\$design file directory>\mpf\_dg0834\_df\tftp\_iap\iap\_images
- 10. Type the tftp -i 10.60.132.61 PUT iog\_cdr\_tftp\_iap\_v3.spi command to transfer the programming image2 as shown in the following figure.

Figure 42 • Transfer IAP Image2

Command P	Prompt	_		$\times$
Microsoft Wi (c) 2017 Mic	indows [Version 10.0.15063] crosoft Corporation. All rights reserved.			^
C:\Users\	<pre>&gt;cd C:\Users\divyesh.patel\Documents\mpf_dg0834_liberosocv12p0_df\tftp_iap\iap_images</pre>			
C:\Users\ iap_v2.spi	\mpf_dg0834_liberosocv12p0_df\tftp_iap\iap_images>tftp -i 10.60.132.61 PUT i	log_c	dr_tft	tp_
Transfer suc	ccessful: 9478672 bytes in 178 second(s), 53250 bytes/s			
C:\Users\ iap_v3.spi	\mpf_dg0834_liberosocv12p0_df\tftp_iap\iap_images>tftp -i 10.60.132.61 PUT i	log_c	dr_tft	tp_
Transfer suc	ccessful: 9478672 bytes in 186 second(s), 50960 bytes/s			
C:\Users\	<pre>\mpf_dg0834_liberosocv12p0_df\tftp_iap\iap_images&gt;</pre>			
	11. Wait until total bytes received message is displayed on the Tera Term, to ensure that	t the		

programming image2 is transfered to the SPI flash. On completion of the Image2 transfer, the user options are displayed.



#### Figure 43 • Bytes Received for Image2

🚇 COM65:115200baud - Tera Term VT 🦳		$\times$
File Edit Setup Control Window Help		
5.Program Device with IAP image1 6.Program Device with IAP image2		^
Press 'e' to erase spi flash memory		
SPI Flash Memory [0x1400000 - 0x1DFFFFF] erase is in progress please	wait.	
SPI Flash Memory is erased successfully.		•
Bytes received=9478672 ****User options****** 1.Copy IAP image1 to SPI flash address 0xA00000 using TFTP 2.Copy IAP image2 to SPI flash address 0x1400000 using TFTP		
3.Run IAP image1 authentication 4.Run IAP image2 authentication 5.Program Device with IAP image1		1
6.Program Device with IAP image2		¥

The images are transferred successfully to external SPI flash using TFTP. The firmware application takes care of external SPI flash programming with SPI directory as shown in the following figure.

#### Figure 44 • SPI Directory



### 4.3.1 Running IAP Authentication

To run the IAP authentication, perform the following steps:

1. Press 3 to initiate IAP image1 authentication. The IAP authentication with image at index 2 is executed successfully. Tera Term displays the status code as shown in the following figure.

#### Figure 45 • Successful IAP Image1 Authentication

🚇 COM65:115200baud - Tera Term VT	_		$\times$
File Edit Setup Control Window Help			
Butes received=9478672			
****User options****** 1_Conv_IAP_image1_to_SPI_flash_address_0xA00000_using_TFTP			
2.Copy IAP image2 to SPI flash address 0x1400000 using TFTP			
4. Run IAP image2 authentication			
5.Program Device with IAP imagel 6.Program Device with IAP image2			
IAP image authentication for image at index 2(address 0xA00000) is	in	progress	
Authentication status: SUCCESS			
****User options******			
1.Copy IAP image1 to SPI flash address 0xA00000 using TFTP 2.Copy IAP image2 to SPI flash address 0x1400000 using TFTP			
3.Run IAP image1 authentication			
5.Program Device with IAP image1			- 11
6.Program Device with IAP image2		_	~



2. Press 4 to initiate the IAP image2 authentication. The IAP authentication with image at index 3 is executed successfully. Tera Term displays the status code, as shown in the following figure.

Figure 46 • Successful IAP Image2 Authentication

		-	
🜉 COM65:115200baud - Tera Term VT		- 🗆	$\times$
File Edit Setup Control Window Help			
IAP image authentication for image at index 2(address 0xA	00000) is :	in progre	ss 🔨
Authentication status: SUCCESS			
****User options****** 1.Copy IAP image1 to SPI flash address 0xA00000 using IFT 2.Copy IAP image2 to SPI flash address 0x1400000 using IF 3.Run IAP image1 authentication 4.Run IAP image2 authentication 5.Program Device with IAP image1 6.Program Device with IAP image2	P TP		
IAP image authentication for image at index 3(address Øx1 Authentication status: SUCCESS	.400000) is	in progr	ess.
*****User options****** 1.Copy IAP image1 to SPI flash address 0xA00000 using IFT 2.Copy IAP image2 to SPI flash address 0x1400000 using IF 3.Run IAP image1 authentication 4.Run IAP image2 authentication 5.Program Device with IAP image1 6.Program Device with IAP image1	™ TP I		
			~

This concludes the IAP image authentication.

### 4.3.2 Running IAP Program

To run the IAP with programming images, perform the following steps:

1. Press 5, **Program Device with IAP image1**. The IAP program with image1 is executed successfully and the design version **02** with different silicon signature is displayed as shown in the following figure. This operation takes few seconds.

#### Figure 47 • Successful IAP with Image1

🜉 COM65:115200baud - Tera Term VT	_	$\times$
File Edit Setup Control Window Help		
Authentication status: SUCCESS		^
****User options****** 1.Copy IAP image1 to SPI flash address 0xA00000 using TFTP 2.Copy IAP image2 to SPI flash address 0x1400000 using TFTP 3.Run IAP image1 authentication 4.Run IAP image2 authentication 5.Program Device with IAP image1 6.Program Device with IAP image2 IAP is in progress I		
Design Version(MSB first): 00 02 32bit USERCODE/Silicon Signature (MSB first): 23456789 *****User options****** 1.Copy IAP image1 to SPI flash address 0xA00000 using TFTP 2.Copy IAP image2 to SPI flash address 0x1400000 using TFTP 3.Run IAP image1 authentication 4.Run IAP image1 authentication		
5.Program Device with IAP image1 6.Program Device with IAP image2		~



 Press 6, Program Device with IAP image2. The IAP program with image2 is executed successfully and the design version 03 with different silicon signature is displayed as shown in the following figure. This operation takes few seconds.

#### Figure 48 • Successful IAP by Image2

```
    COM65:115200baud - Tera Term VT - C
    File Edit Setup Control Window Help
    Design Uersion(MSB first): 00 02
32bit USERCODE/Silicon Signature (MSB first): 23456789
    *****User options*******
    1.Copy IAP image1 to SPI flash address 0x1400000 using TFTP
    2.Copy IAP image2 authentication
    5.Program Device with IAP image1
    1.AP image2 authentication
    32bit USERCODE/Silicon Signature (MSB first): 3456789A
    *****User options*******
    1.Copy IAP image1 to SPI flash address 0x1400000 using TFTP
    2.Copy IAP image2 to Signature (MSB first): 3456789A
    *****User options******
    1.Copy IAP image1 authentication
    S.Program Device VI flash address 0x1400000 using TFTP
    2.Copy IAP image1 authentication
    5.Program Device VI flash address 0x1400000 using TFTP
    3.Run IAP image1 authentication
    6.Program Device with IAP image1
    6.Program Device with IAP image1
    6.Program Device with IAP image2
```

This concludes running the IAP Program with images.



# 5 Appendix 1: Enable TFTP Client

The following steps describe how to enable TFTP client:

1. Navigate to **Control Panel > Programs**. Click **Turn Windows features on or off** as shown in the following figure.

Figure 49 • Control Panel—Programs and Features



2. Select the TFTP Client check box from Windows Features as shown in the following figure.

#### Figure 50 • Selecting TFTP Client from Windows Features





3. Browse through Control Panel > System and Security and click Allow an app through Windows Firewall.

Figure 51 • System and Security Window



- **Note:** If the System and Security option is not available, then enter the firewall in the search window to perform step 3.
  - 4. Click Change settings and choose Allow another program...

Figure 52 • Allow Programs Window

Allow apps to communicate through Window	ws Firew	all				
To add, change, or remove allowed apps and ports, click C	hange setti	ings.			_	
What are the risks of allowing an app to communicate?				😯 Change setti	ings	
For your security, some settings are managed by you	r system ad	Iministrato	or.			
Allowed apps and features:						
Name	Domain	Private	Public	Group Policy	^	
BranchCache - Hosted Cache Server (Uses HTTPS)				No		
BranchCache - Peer Discovery (Uses WSD)				No		
Canon Office Printer Utility	✓	✓	$\checkmark$	No		
Captive Portal Flow	✓	✓	$\checkmark$	No		
Cast to Device functionality	✓	✓	$\checkmark$	No		
Connect	✓	✓	✓	No		
Core Networking	✓	✓	✓	No		
✓ Cortana	✓	$\checkmark$	$\checkmark$	No		
Delivery Optimization	✓	◄	✓	No		
☑ DiagTrack	✓	$\checkmark$	✓	No		
DIAL protocol server	✓	✓		No		
Distributed Transaction Coordinator				No	$\checkmark$	
			Details	. Remove	e	
			AI	low another app	p	



- 5. The Add an app window is displayed and click Browse...
- 6. Browse through C: \ -> Windows->System32 and choose TFTP.exe and click Open.
- 7. Ensure that the TFTP.EXE path (C:\Windows\System32\TFTP.EXE) is selected correctly and click Add.

### Figure 53 • Add an app Window

Add an ap	'P	×
Select the listed, and	app you want to add, or click Browse to find one that is not then click OK.	
Apps:		
Trivial	l File Transfer Protocol App	
Path:	C:\Windows\System32\TFTP.EXE Browse	
What are th	ne risks of unblocking an app?	
You can cho	oose which network types to add this app to.	
Netw	vork types Add Cancel	

8. Ensure that the **Trivial File Transfer protocol App** is added and also select all the check boxes (Domain, Home/Work, and Public) as shown in the following figure.

Figure 54 • Selecting Trivial File Transfer Protocol App in Allowed apps Window

Allow apps to communicate through Window	vs Firew	all				
To add, change, or remove allowed apps and ports, click C	hange setti	ings.				
What are the risks of allowing an app to communicate?				😯 Change sett	ings	
For your security, some settings are managed by your	r system ad	Iministrato	or.			
Allowed apps and features:						
Name	Domain	Private	Public	Group Policy	^	
System Center Configuration Manager		✓		No		
✓ Take a Test	$\checkmark$	✓	☑	No		
TechSmith Snagit	✓	✓	✓	No		
TPM Virtual Smart Card Management				No		
Trivial File Transfer Protocol App		<ul><li>✓</li></ul>	☑	No		
Virtual Machine Monitoring				No		
✓ Wallet	$\checkmark$	✓	✓	No		
Wi-Fi Direct Network Discovery			$\checkmark$	No		
Windows Calculator		✓	✓	No		
Windows Collaboration Computer Name Registra				No		
✓ Windows Default Lock Screen		⊻	⊻	No		
✓ Windows Defender Security Center		✓	✓	No	~	
			Details	. Remov	e	
			AI	low another app	р	

9. Click **OK**.

Appendix 2: Running the SoftConsole Project in Debug Mode from LSRAM or DDR Memory



# Appendix 2: Running the SoftConsole Project in Debug Mode from LSRAM or DDR Memory

The following steps describe how to run the SoftConsole project in debug mode.

- 1. Open the Webserver or TFTP\_IAP application project using SoftConsole.
  - Webserver SoftConsole project location: mpf\_dg0834\_df\webserver\libero\SoftConsole
  - TFTP\_IAP SoftConsole project location: mpf\_dg0834\_df\tftp\_iap\libero\SoftConsole
- In SoftConsole, select Run > Debug Configurations. The Debug Configurations dialog box displayed. To debug the Webserver project, select gbe\_webserver Debug, as shown in the following figure.



6

SC Debug Configurations		×
Create, manage, and run configurations		
Image: Second system         Image: Second system <th>Name:       gbe_webserver Debug         Main</th> <th>mon ₽ SVD Path Project Browse Variables Search Project Browse O Disable auto build Configure Workspace Settings</th>	Name:       gbe_webserver Debug         Main	mon ₽ SVD Path Project Browse Variables Search Project Browse O Disable auto build Configure Workspace Settings
Filter matched 3 of 10 items		vexert Appix
?		<u>D</u> ebug Close

3. Click **Debug**. The tool copies the code to LSRAM memory and launches the debug session. This SoftConsole project is configured to debug from LSRAM.



Debugging the application from DDR memory:

1. In the SoftConsole Project Explorer window, right-click **gbe\_webserver** project, and select **Properties**, as shown in the following figure.

Figure 56 • Project Explorer Window



Change the linker script file setting to microsemi-riscv-ram\_ddr.ld and re-build the project.
 Figure 57 • Project Properties



3. In SoftConsole, select Run > Debug Configurations to debug the application from DDR memory.

Appendix 2: Running the SoftConsole Project in Debug Mode from LSRAM or DDR Memory



## 6.1 Running the Design in Static IP Mode

The following steps describe how to run the design in static IP mode.

- **Note:** This procedure provide steps to run the Webserver design. To run the IAP using TFTP design, perform the same steps by opening the IAP\_TFTP project in SoftConsole.
  - 1. In SoftConsole Project Explorer window, right-click the Webserver (**gbe\_webserver**) project, and select **Properties**, as shown in Figure 56, page 39.
  - 2. In **Properties for gbe\_webserver** window, remove the **NET\_USE\_DCHP** symbol listed under **Defined symbols (-D)**, and click **Apply**, as shown in the following figure.

Figure 58 • Properties for gbe\_webserver

SC Properties for gbe_webserver			— 🗆 X
type filter text	Settings		↓ ↓ ↓ ▼
type tilter text > Resource Builders • C/C++ Build Build Variables Environment Logging Settings Tool Chain Editor > C/C++ General > cppcheclipse > MCU Project References Refactoring History Run/Debug Settings	Settings Configuration: Debug [Active] Tool Settings Toolchains Devices Target Processor Optimization Warnings Debugging Software Settings Software Settings Miscellaneous Software Settings Software Settings Sof	Build Steps Build Artifact Binary Parsers Erro Do not search system directories (-nostdinc) Preprocess only (-E) Defined symbols (-D) INTER USE DHAP LWIP_COMPAT_MUTEX LWIP_PROVIDE_ERRNO Undefined symbols (-U)	<ul> <li></li></ul>
			Restore Defaults Apply
?		A	pply and Close Cancel

Appendix 2: Running the SoftConsole Project in Debug Mode from LSRAM or DDR Memory



3. Change the host TCP/IP settings to connect with the board which has static IP address, **169.254.1.23**. The following figure shows the host PC TCP/IP settings.

### Figure 59 • Host PC TCP/IP Settings

Internet Protocol Version 4 (TCP/IPv4) Properties		
General		
You can get IP settings assigned autor this capability. Otherwise, you need to for the appropriate IP settings.	matically if your network supports ask your network administrator	
O Obtain an IP address automatical	ly	
• Use the following IP address:		
IP address:	169.254.1.22	
Subnet mask:	255.255.255.0	
Default gateway:		
Obtain DNS server address autor	natically	
• Use the following DNS server add	resses:	
Preferred DNS server:	169.254.1.23	
Alternate DNS server:		
Validate settings upon exit	Ad <u>v</u> anced	
	OK Cancel	

- 4. Connect the PolarFire Evaluation board port J15 to Host PC using RJ45 Ethernet cable.
- 5. After configuring the settings, compile the design, load it into memory, and run it using SoftConsole. The Serial terminal shows board static IP:

/\*CoreTSE WebServer using Mi-V SoftProcessor\*/

Acquiring IP address. 169.254.1.23

6. Use the IP address in web browser to display the Microsemi web page.



# 7 Appendix 3: Programming the Device and External SPI Flash Using FlashPro Express

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

- Webserver: mpf\_dg0834\_df\webserver\programming\_job
- **TFTP\_IAP**: mpf\_dg0834\_df\tftp\_iap\programming\_job

To program the device, perform the following steps:

1. Ensure that the jumper settings on the board are the same as listed in Table 4, page 25. **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. To create a new job project, click New or

In the Project menu, select **New Job Project from FlashPro Express Job**, as shown in the following figure.

Figure 60 • FlashPro Express Job Project

🔛 FlashPro Express	_	E FlashPro Express	
Project Edit View Programmer <u>H</u> elp		Project Edit View Programmer <u>H</u> elp	
Joh Projects		New Job Project from FlashPro Express Job	Ctrl+N
		🚰 Open Job Project	Ctrl+O
New		× Close Job Project	
Open		🔚 Save Job Project	Ctrl+Shift+A
Recent Projects		Set Log File	•
	or	Export Log File	
		<u>P</u> references	
		Execute Script	Ctrl+U
		Export Script File	
		Recent Projects	۶.
		Exit	Ctrl+Q



- 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\_dg0834\_df\webserver\programming\_job or
    - <download\_folder>\mpf\_dg0834\_df\tftp\_iap\programming\_job
  - FlashPro Express job project location: Click Browse and navigate to the location where you
    want to save the project.

#### Figure 61 • New Job Project from FlashPro Express Job

Rew Job Project from FlashPro Express Job	×
Programming job file:	
atel\Documents\mpf_dg0834_liberosocv12p0_df\tftp_iap\programming_job\gbe_tftp_iap_v1.job	Browse
FlashPro Express job project name:	
gbe_tftp_iap_v1	
FlashPro Express job project location:	
C: \webserver	Browse
	l cont l
	Cancel

- 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 62 • Programming the Device

Project Edit View Programmer <u>H</u> elp			
Refresh/Rescan Programmers			
Programmer	MPF300TS         Ш         Ц           Ф ТДО         ТДІ Ф		
1 E2003G9SLI IDLE	IDLE		
PROGRAM/PROGRAM_SPI_IMAGE			
RUN		IDLE	
_og			& ×
🔳 Messages 🔞 Errors 🗼 Warnings 🏮 Info			
Embedded FlashPro5 programmer detected. programmer 'E2003G9SLI' : FlashPro5			
Created new project 'E:\fpexp\gbe_webserve PPD file 'E:\fpexp\gbe_webserver\gbe_webserver	r\gbe_webserver.pro' rver.ppd' has been loaded su	ccessfully.	
Creating folder: E:\fpexp\gbe_webserver\pr Software Version: 12.700.0.21	ON : I.U ojectData		
PPD file 'E:\fpexp\gbe_webserver\gbe_webse DESIGN : top; CHECKSUM : 9134; PDB_VERSI	rver.ppd' has been loaded su ON : 1.0	ccessfully.	
Embedded FlashPro5 programmer detected. programmer 'E2003G95LI' : FlashPro5			T



10. Click **RUN**. When the device is programmed successfully, a **RUN PASSED** status is displayed as shown in the following figure. See Running the Demo, page 27 to run the webserver and TFTP\_IAP demo.

Project Edit View Programmer <u>H</u> elp		
Defrech Descap Dragrammere		
Keiresn/Kescari Programmers		
Programmer	1 MPF300TS 🖲 📱	
Programmer	Ф ТОО ТОІ Ф	
1 E2003G9SLI RUN PASSED	PASSED	
1		
PROGRAM/PROGRAM_SPI_IMAGE		
RUN	I PROGRAMMER(S) PASSED	
97		8 ×
Messages 🔞 Frrors 🗼 Warnings 📦 Info		
programmer E200303511 . device MF5001.	) - EAFORI SNYM COMPONENT DITSTIERM GIGEST[200] -	
programmer 'E2003G9SLI': device 'MPF300T 6fe26ele98096e7386ed53ec6c207a02d0d93e238	zzeriosodiodosoficioso 5' : EXPORT EOB component bitstream digest[256] = 6669d57er0b31cef897657	
programmer 'E2003G9SLI' : device 'MPF300T	S' :	
programmer 'E2003G9SLI' : device 'MPF300T	S' : EXFORT DSN[128] = 757d2c0809548721d599c75ff31f3052	
programmer 'E2003G95LI' : device 'MPF3001	5': Finished: Thu Oct 24 18:22:42 2019 (Elapsed time 00:02:13)	
programmer 'E2003G9SLI' : device 'MPF300T	S': Executing action PASSED.	1
programmer 'E2003G9SLI' : device 'MPF300T	5' : Executing action PROGRAM PASSED.	
Chain Programming Finished: Thu Oct 24 18	:22:42 2019 (Elapsed time 00:03:07)	-
1		_

### Figure 63 • FlashPro Express—RUN PASSED

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