AN4737

Migrating Software to PolarFire® SoC from Zynq®-7000 SoC

Introduction

Microchip's PolarFire® SoC is the first System-on-Chip (SoC) Field-Programmable Gate Array (FPGA) with a deterministic, coherent RISC-V® CPU cluster with 64-bit processor scalability, and a deterministic L2 memory subsystem enabling Linux® and real-time applications.

This document is intended as a reference guide for new users of PolarFire SoC FPGA. This document aims to ease the migration of software running on the Processing System (PS) of Xilinx's Zynq®-7000 SoC to Microprocessor Subsystem (MSS) of PolarFire SoC FPGA.

About this Document

The goal is to provide an overview of the decisions and steps involved in developing an application running on either bare metal or Linux OS. This document is useful to users attempting to migrate their Xilinx Zynq-7000 projects. This document includes sections that provide the architectural comparison and flow steps including booting options, and details on peripherals migration. Links to useful documents and reference applications are provided as and when required for users to explore in greater detail.

Intended Audience and Prerequisites

This document is for software developers and system architects with the following prerequisites:

- Familiarity with the Zynq-7000 SoC and PolarFire SoC architectures.
- Familiarity with the booting flow of Zynq-7000 SoC device and PolarFire SoC FPGA.
- Experience in application development for a Zynq-7000 SoC device and PolarFire SoC device.
- Experience with embedded software development.

Scope of the Document

This document helps users to understand the following migration aspects:

- Architectural differences between Zynq-7000 SoC and PolarFire SoC.
- High-level picture of software migration from Zynq-7000 SoC to PolarFire SoC.
- Peripheral configuration differences between Zynq-7000 SoC and PolarFire SoC.
- Boot flow differences between Zynq-7000 SoC and PolarFire SoC.

Important: The RTOS option is available on PolarFire SoC but it is not covered in this document.

Table of Contents

1. Device Architecture

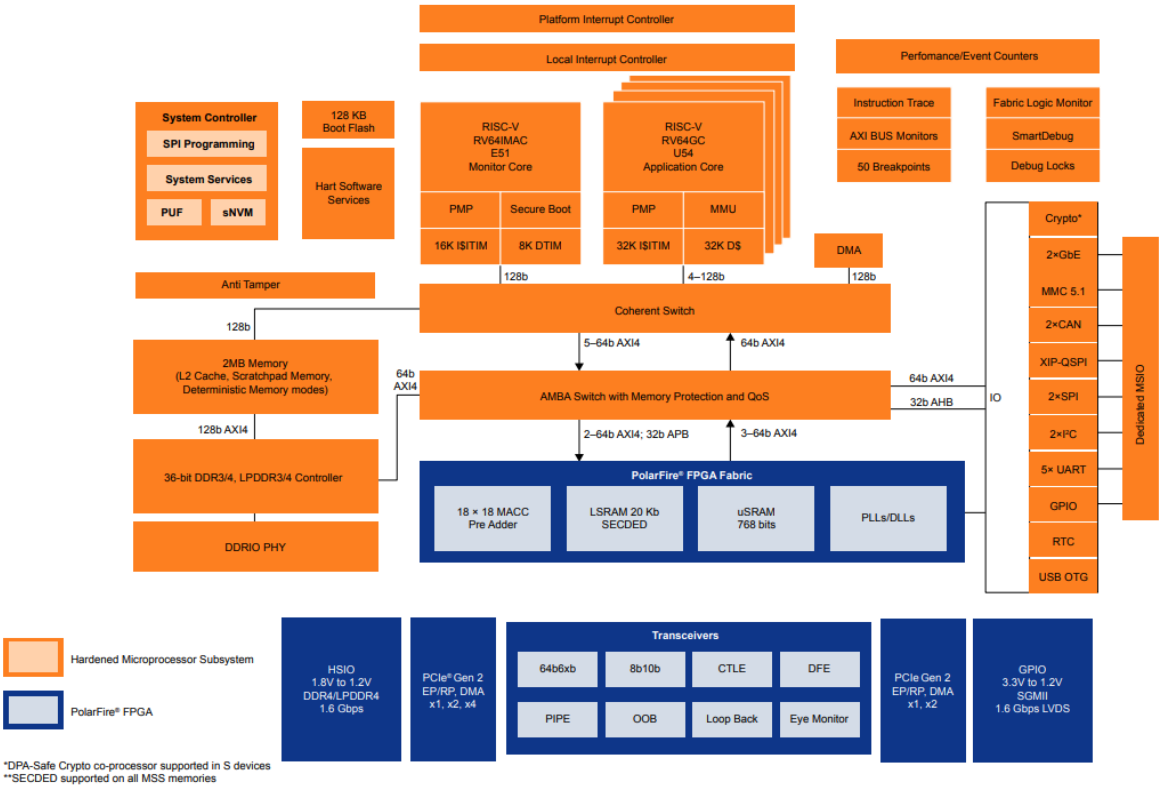
The PolarFire SoC family is built on the award-winning, mid-range, low power PolarFire FPGA architecture. PolarFire SoC devices deliver up to 50% lower power than alternative FPGA solutions, span from 25K to 460K Logic Elements (LEs) and feature 12.7G transceivers in all densities.

Featuring a five-core Linux-capable processor subsystem based on the RISC-V Instruction Set Architecture (ISA), PolarFire SoC brings to market an innovative, mid-range, embedded computing platform that inherits all the benefits of the PolarFire FPGA product family. The RISC-V CPU micro-architecture implementation is a simple, 5-stage single-issue in-order execution pipeline that is free from Meltdown and Spectre exploits found in common out-of-order machines.

All five CPU cores are coherent with the memory subsystem allowing a versatile mix of deterministic real-time systems and Linux in a single, multi-core CPU cluster. With built-in Secure Boot, innovative Linux and Real-Time modes, a large flexible L2 memory subsystem, and a rich set of embedded peripherals, PolarFire SoC offers new choices in secure, power-efficient, embedded computing platforms for designers. This document describes the features of PolarFire SoC extended commercial (0 °C to 100 °C T_j) and industrial (–40 °C to 100 °C T_j) device offerings.

The following figure shows the PolarFire SoC architecture.

Figure 1-1. PolarFire SoC Device Architecture



For more information about the architecture and functional blocks of PolarFire SoC MSS, see PolarFire SoC FPGA Technical Reference Manual. For more information about PolarFire SoC embedded software, see github.com/polarfire-soc.

1.1 Hart Software Services

Hart Software Services (HSS) is a collection of software services that run on the E51 monitor core. HSS uses bare metal drivers and the XML configuration file generated by the PolarFire SoC MSS Configurator to configure and initialize the PolarFire SoC MSS at power-up.

In general, HSS performs the following functions:

- Programming memory using USB mass storage or YMODEM transfer.
- Copying a program (Linux or Bare Metal) from a non-volatile storage (for example, eMMC or SD card) to the Loosely Integrated Memory (LIM), L2-Scratchpad, or DDR memory.
- Creating a payload containing multiple applications to be booted and run.
- Facilitating Inter-processor communication in MSS.

1.1.1 HSS as a ZSBL

In PolarFire SoC, HSS is the Zero Stage Boot Loader (ZSBL) which runs in Machine mode on the E51 monitor core. The HSS loads U-Boot which is the First Stage Boot Loader (FSBL). U-Boot runs in Supervisor mode for loading Linux.

2. Architectural Comparison

Users must understand the architectural differences before starting the migration process. The Zynq-7000 PS is based on dual-core ARM® Cortex™-A9 processor, whereas the PolarFire SoC MSS is based on 5x core 64-bit RV64 RISC-V processor subsystem from SiFive. As the PolarFire SoC family includes a multi-core MSS, users must determine the type of application execution they require, Asymmetric Multi-Processing (AMP) or Symmetric Multi-Processing (SMP) before the migration and follow the same type of execution after migration.

The following table maps the Zynq-7000 PS functional blocks to PolaFire SoC MSS.

Table 2-1. Architectural Comparison

Zynq®-7000 SoC	PolarFire® SoC
Processing System	Microprocessor Sub-System
DDR Memory Controller (DDR2, DDR3, and LPDDR2)	MSS DDR Controller (LPDDR4, LPDDR3, DDR3, and DDR4)
DMA Controller	DMA Controller
256 KB On-chip RAM	2 MB on-chip configurable RAM available for L2 cache/LIM/ Scratchpad On chip 128 KB ROM (eNVM)
Quad SPI Controller	Quad SPI Controller
2x 10/100/1G Ethernet	2x 10/100/1G Ethernet
2x USB On-The-Go (OTG)	1x USB On-The-Go (OTG)
SD/SDIO Controller (sdhci)	eMMC SD/SDIO (sd4hc)
2x UART	5xMMUART
2x 12C	2x 12C
2x CAN	2x CAN
2x SPI Controller	2x SPI Controller
Security (AES, SHA, RSA)	Security (AES, SHA, RSA)[1]
GPIO	GPIO (x3)

Note:

1. Security features are available in the “S” device family. For more information about the PolarFire SoC device ordering, see PolarFire SoC Product Overview.

On comparison, the PolarFire SoC architecture combines the high-performance 64-bit RISC-V Linux-capable MSS with the PolarFire FPGA fabric. This powerful combination brings in low-power, thermal efficiency, and defence grade security to embedded systems.

2.1 Boot Process Comparison

The following table lists the booting process difference between Zynq-7000 SoC and PolarFire SoC.

Table 2-2. Boot Process Difference

Zynq®-7000 SoC Boot Process	PolarFire® SoC Boot Process
The booting process is executed in the In-Application Programming (IAP) mode, in which, the Processing System (CPU) boots first and then the FPGA is configured.	The FPGA fabric is configured first along with eNVM to boot the MSS. The FPGA releases the MSS out of reset and the E51 monitor core executes HSS from its L2-Sratchpad memory.

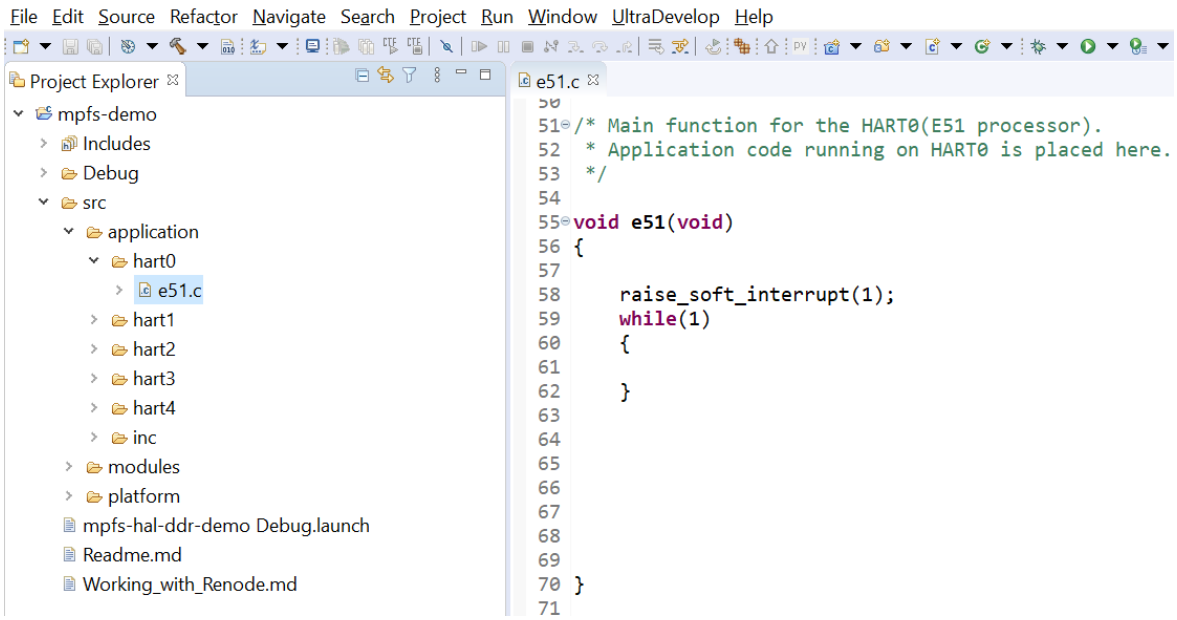
Important: For more information about the PolarFire SoC boot process, see PolarFire and PolarFire SoC FPGA Power-up and Resets User Guide.

2.1.1 Bare-Metal Boot Process

Bare-metal applications for the PolarFire SoC are developed using Microchip's SoftConsole. The SoftConsole IDE provides output files in the `.hex` format, which can be used in the Libero® SoC design flow to include in the FPGA programming bitstream file.

The following figure shows a SoftConsole bare-metal application with five harts (processor cores) including the E51 monitor core.

Figure 2-1. PolarFire SoC Bare-Metal Project



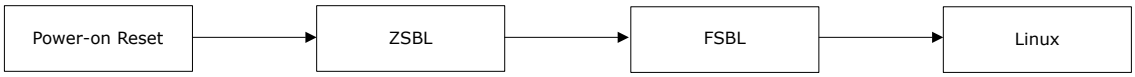
For a simple bare-metal application demo to execute on different harts, see the mpfs-hal-simple-demo in `driver-examples/mss/mpfs-hal`.

Important: In PolarFire SoC, during Asymmetric Multi-Processing (AMP) an application core cannot reset another application core. For example, when U54_1 application core is executing Linux and U54_2 application core is executing a bare-metal application. In this case, U54_1 cannot reset U54_2.

2.1.2 Linux Boot Process

This section describes the boot sequence for running Linux on all U54 applications cores. The following figure shows the Linux boot process.

Figure 2-2. Linux Boot Flow



The Linux boot process consists of the following stages:

1. The Zero Stage Boot Loader (ZSBL), is loaded from the on-chip BootFlash (also known as eNVM) and executed from the L2-Scratchpad.
2. The ZSBL loads the First Stage Boot Loader (FSBL) from a boot device to external RAM or on-chip cache. The boot device can be eNVM, embedded Memory Microcontroller (eMMC) or external SPI Flash.
3. The FSBL loads the Linux operating system from boot device to external RAM.
4. Finally, Linux is executed from the external RAM.

For more information about Linux boot flow and the resources used to build Linux for PolarFire SoC, see PolarFire SoC Software Tool Flow and PolarFire SoC Yocto BSP.

2.2 Asymmetric Multi-Processing

Asymmetric Multiprocessing (AMP) is a type of multi-core software architecture that allows multiple operating systems or software contexts to run simultaneously and independently of each other. In an AMP system, it is possible to allocate hardware resources to a specific software context. These hardware resources include processor cores, peripherals, and physical memory regions.

PolarFire SoC MSS includes a CPU Core Complex that contains one E51 monitor core and four U54 application cores. Each processor core is identified as a hardware thread (also known as a hart). The PolarFire SoC MSS can be configured to run up to two independent software contexts. Each software context can have its own operating system, memory regions, and hardware resources assigned. For more information about configuring the MSS, see PolarFire SoC MSS Configurator User Guide.

PolarFire SoC MSS supports the following AMP software architecture:

- Single E51 monitor core dedicated for running the Hart Software Services (HSS).
- Four U54 application cores distributed between two independent software contexts.

For information about the PolarFire SoC AMP boot flow, see PolarFire SoC Asymmetric Multiprocessing.

2.3 Symmetric Multi-Processing

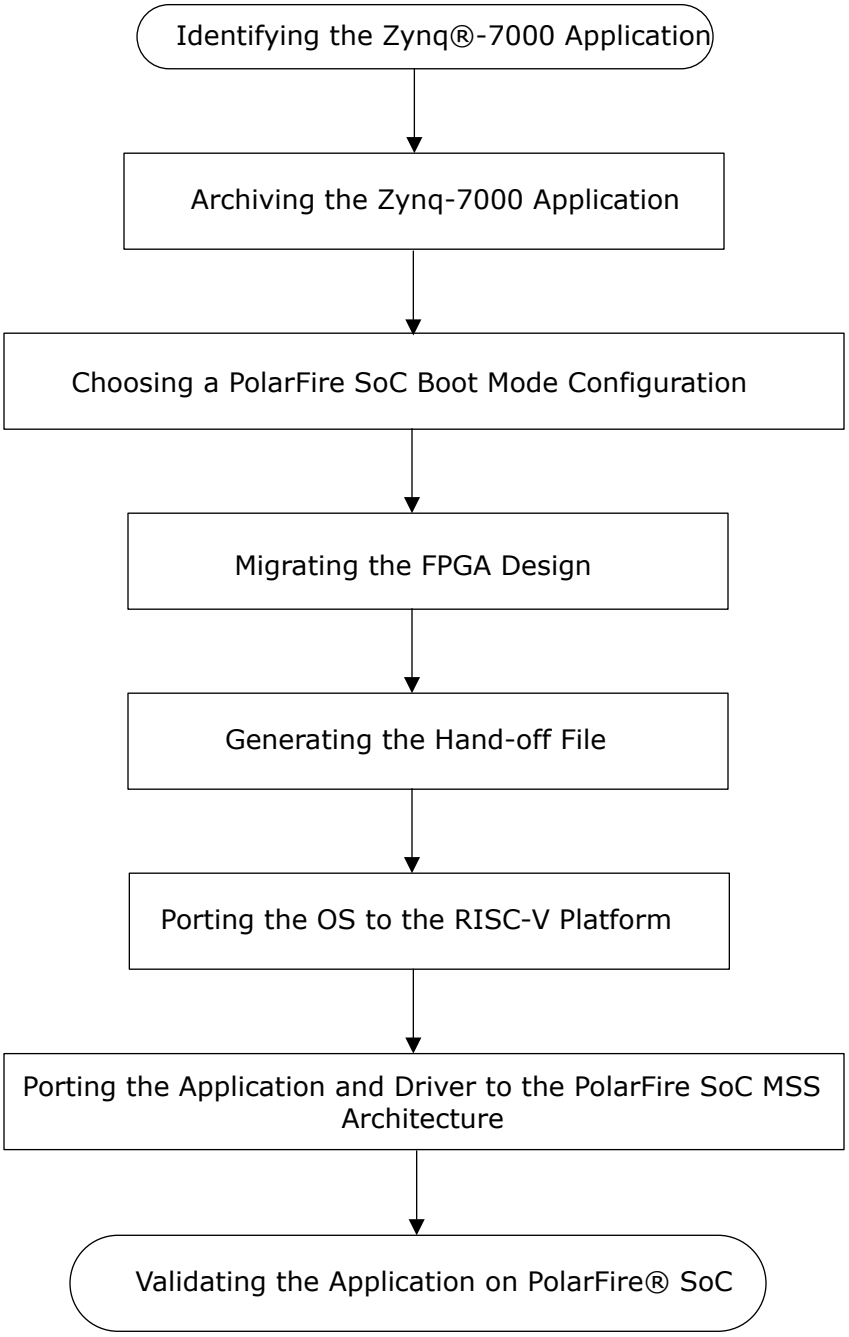
Symmetrical Multi-Processing (SMP) is a type of multi-core software architecture, that enables running applications using a single operating system, in which all processor cores share the same cache memory and peripherals. The operating system handles the load balancing for effective utilization of resources like cache memory, peripherals, and interrupts.

In PolarFire SoC, when the MSS is in the SMP mode, a single operating system runs on all four U54 application cores with multiple applications, and each application can span into multiple threads.

3. Migration Process

The following figure and sections show high-level steps of the migration process.

Figure 3-1. Migration Process



3.1 Archiving the Zynq-7000 SoC Project

Archive the working Zynq-7000 project which includes both Programmable Logic (PL) and Processing System (PS).

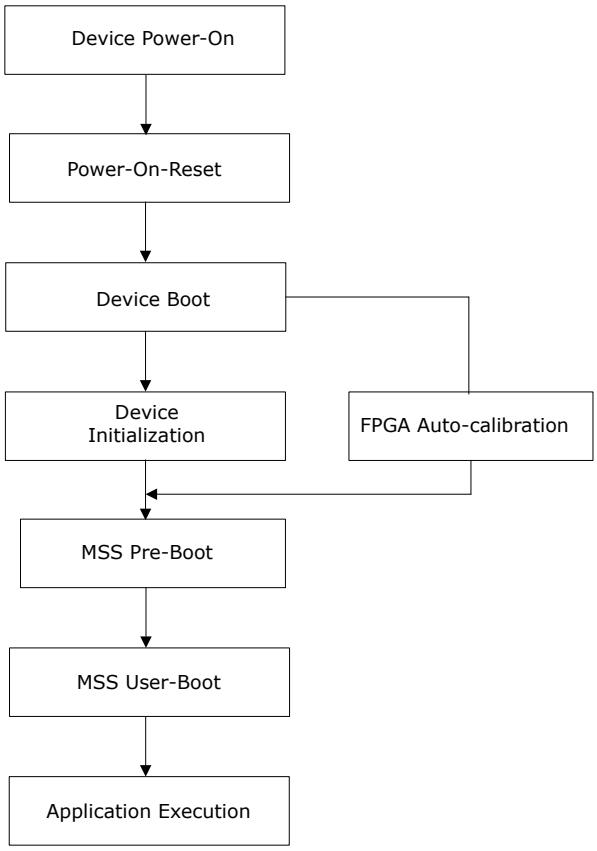
3.2 Choosing a Boot Mode Configuration

In Zynq-7000 SoC, the processor always boots first and then the PS initializes the PL portion of the device with a bitstream. In the PolarFire SoC boot process, the FPGA is always configured first with a bitstream by the system controller and then the MSS boots up.

In PolarFire SoC, the boot-up sequence starts when the device is powered-up or reset. It ends when the processor is ready to execute an application program. This booting sequence runs through several stages before it begins the execution of programs. A set of operations are performed during the boot-up process, that includes Power-On Reset of the hardware, peripheral initialization, memory initialization, and loading the user-defined application from non-volatile memory to the volatile memory for execution.

The following figure shows the PolarFire SoC boot sequence.

Figure 3-2. PolarFire SoC Boot Sequence



As shown in the preceding figure, the following boot sequences are executed after the FPGA initialization.

MSS Pre-Boot Sequence

After the initializing and configuring the FPGA, MSS Pre-boot sequence is executed. PolarFire SoC supports different boot modes. Before migration, the user must select a boot mode configuration.

The following events occur during the MSS pre-boot stage:

1. Power-up of the MSS embedded Non-Volatile Memory (eNVM)
2. Initialization of ECC with the MSS Core Complex L2 cache
3. Authentication of User boot code (if User Secure boot option is enabled)
4. Handover of operational MSS to User Boot code

The MSS supports the following boot modes, which can be configured.

Table 3-1. MSS Boot Modes

Boot Mode	Description
Idle boot (0)	MSS boots from boot ROM and executes a while a loop if MSS is not configured
Non-secure boot (1)	MSS boots directly from eNVM
User secure boot (2)	MSS boots from sNVM
Factory secure boot (3)	MSS Core Complex boots using the factory secure boot protocol

The boot option is selected as part of the Libero design flow. Changing the mode can only be achieved through the generation of a new FPGA programming file. Changing the boot mode can also be achieved using the SoftConsole IDE.

MSS User Boot Sequence

MSS user boot takes place when the control is given from the system controller to the MSS. After MSS pre-boot, System Controller releases the MSS out of reset and can be booted up with one of the following:

- SMP (bare-metal, RTOS, or Linux)
- AMP (bare-metal, RTOS, or Linux)

For more information on boot modes and secure boot, see PolarFire SoC Boot Modes Fundamentals.

3.3 Migrating the FPGA design

Migrate the FPGA design with the latest Libero SoC software by choosing the required hardware configuration. The following references are useful for migrating the FPGA design.

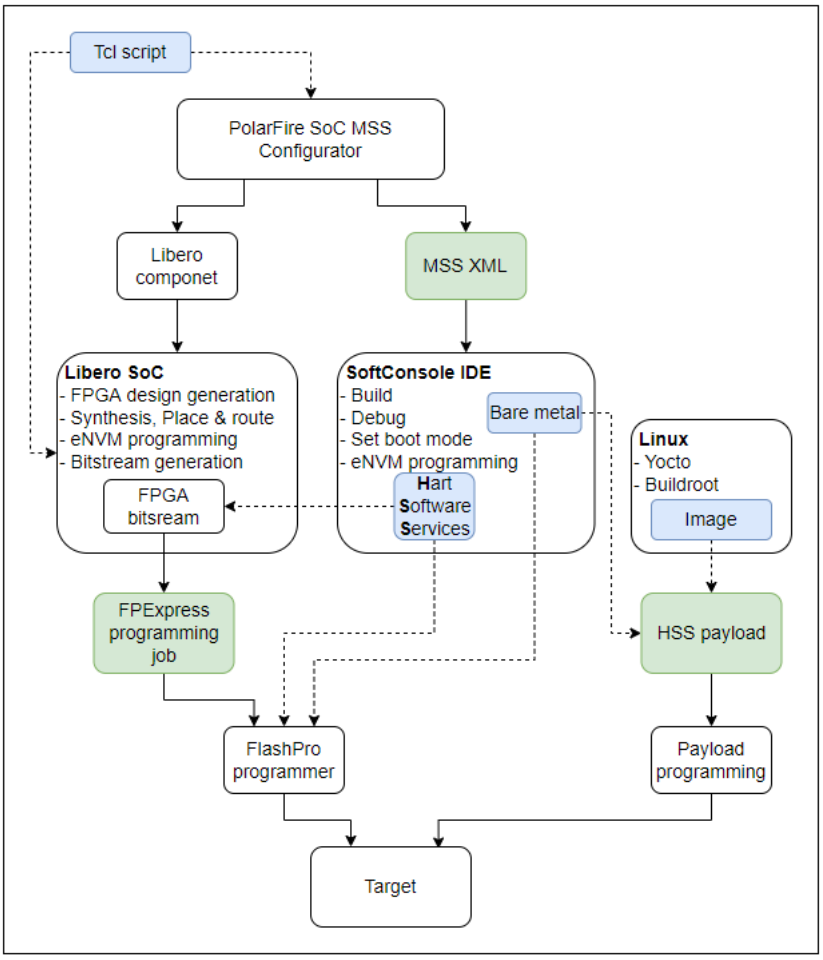
- PolarFire SoC Icicle Kit Reference Design Generation Tcl Scripts for Libero® SoC v2022.1
- Design Migration Overview from Kintex®-7 to PolarFire®

3.4 Generating the Hand-Off File for Boot Mode 1

Once an MSS configuration has been generated using the MSS Configurator, an `.xml` file, that includes all the MSS hardware configuration details, is generated. This `.xml` file is used as an input to the HSS or bare-metal SoftConsole project for generating `.hex` file. This `.hex` file is provided as a input to the eNVM configuration to program in boot mode 1.

The following figure shows the full development flow for both the FPGA and software in PolarFire SoC. Blue boxes are source files and green are output files that can be generated and are provided via GitHub for the user.

Figure 3-3. Full Development Flow



For more information about the full development flow, see PolarFire SoC Software Tool Flow.

3.5 Porting the OS (Linux) to PolarFire SoC MSS

The PolarFire SoC MSS includes a 5x core 64-bit RISC-V CPU, L2 cache, a rich set of peripherals, and a hardened DDR Controller, DMA Controller, and more features. To port the OS to PolarFire SoC MSS, it is necessary to get familiarized with the RISC-V CPU architecture and the overall system memory map. This helps in modifying the data types, register map, and the memory addressing scheme in the source code. For more information about the RISC-V architecture, see the RISC-V Instruction Set Manual. For more information about the overall MSS architecture and its overall memory map, see PolarFire SoC FPGA MSS Technical Reference Manual.

3.6 Porting the Application to PolarFire SoC MSS

Mostly, all the application software running on the Zynq-7000 SoC may be portable with certain dependency changes to PolarFire SoC because the Zynq-7000 SoC is an ARM Cortex-A9 variant, whereas PolarFire SoC is a RISC-V variant. Linux is one of the most popular operating system choices for both Zynq-7000 SoC and PolarFire SoC, because Linux is used for running application-class embedded processing applications. PolarFire SoC supports the Yocto and buildroot build environment and other commercial real-time operating systems such as VxWorks and Zypher. See the Zynq-7000 device tree and the PolarFire SoC device tree entries.

Porting the application involves modifying the peripheral configurations to PolarFire SoC MSS. The portability of the peripherals depends on the operating system. The operating system provides an abstraction layer between the software and application running on the processor and its peripherals.

Migrating an application involves the following high-level steps:

1. Modifying all the user peripherals as per the configurations.
2. Modifying the device tree as per the memory maps, modes, speed, clock, interrupt numbers, and other configurations related to peripherals.
3. Compiling the application with an architecture-specific toolchain.
4. Executing the application on PolarFire SoC.

Users must integrate the required user peripheral by running `make mpfs_defconfig` and `make menuconfig` to create a `defconfig` based on the PolarFire SoC set up. The operating system uses a device tree to automatically integrate necessary drivers in PolarFire SoC. For migrating to PolarFire SoC, create a device tree that integrates the driver tree infrastructure within the operating system.

This document covers the migration details of the following peripherals:

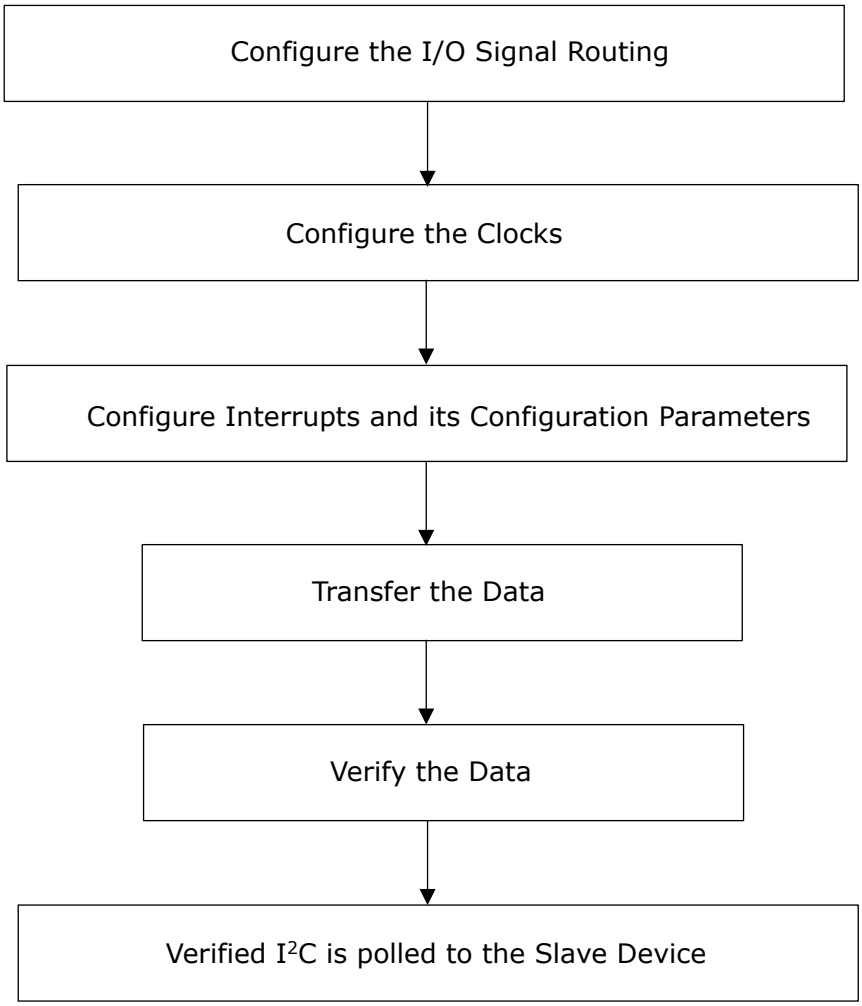
3.6.1 I²C

PolarFire SoC FPGAs contain two identical I²C peripherals in the microprocessor subsystem (I2C_0 and I2C_1), that provide a mechanism for serial communication between the PolarFire SoC FPGA and the external I²C compliant devices.

When porting I²C from Zynq-7000 SoC to PolarFire SoC, change the base address and alternate base address to the new addresses related to PolarFire SoC. The base addresses and register descriptions of I²C are listed in Polar Fire SoC Device Register Map.

The following figure shows the high-level steps to be completed for porting I²C to PolarFire SoC.

Figure 3-4. I2C Migration Process



When porting I2C controller from a Zynq-7000 SoC device to PolarFire SoC, change the base addresses of I2C_0 and I2C_1 to the corresponding PolarFire SoC addresses. The base addresses and register descriptions of I2C_0 and I2C_1 are listed in Polar Fire SoC Device Register Map.

The following tables compares the I2C register set of Zynq-7000 SoC and PolarFire SoC.

Table 3-2. I2C Register Comparison

S. No	Register	Zynq®-7000 SoC	Offset in Zynq-7000 SoC	PolarFire® SoC	Offset in PolarFire SoC
1	Configuration	Control_Reg0	0x000	I2C0_CTRL	0x000
2	Data	I2C_address_reg0 I2C_data_reg0 Transfer_size_register0 Slave_mon_pause_reg0 Time_out_reg0 Staus_reg0	0x08 0x0c 0x14 0x18 0x1c 0x04	I2C0_SLAVE0ADR I2C0_DATA I2C0_STATUS I2C0_SMBUS I2C0_FREQ I2C0_GLITCHREG I2C0_SLAVE1ADR	0x00C 0x008 0x004 0x010 0x014 0x018 0x01C
3	Interrupt processing	Interrupt_status_reg0 Interrupt_mask_reg0 Interrupt_enable_reg0 Interrupt_disable_reg0	0x10 0x20 0x24 0x28	SMBUS_RESET SMBALERT_INT_ENB SMBSUS_INT_ENB SMBUS_ENB SMBALERT_NI_STATUS SMBALERT_NO_CTRL SMBSUS_NI_STATUS SMBSUS_NO_CTRL CORE_I2C_SMBUS	(0x10) (0x0) (0x1) (0x2) (0x3) (0x4) (0x5) (0x6) (0x7)

3.6.1.1 Features

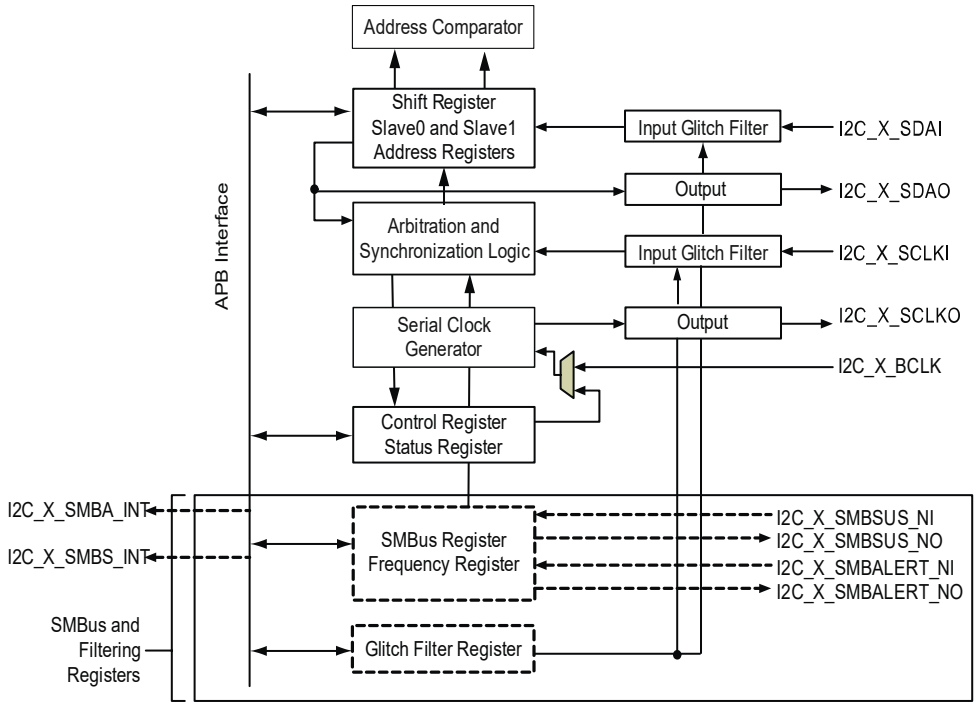
In PolarFire SoC, I2C peripherals support the following protocols.

- I2C protocol as per v2.1 specification
- SMBus protocol as per v2.0 specification
- PMBus protocol as per v1.1 specification

3.6.1.2 Functional Description

The following figure shows the I2C block of PolarFire SoC MSS.

Figure 3-5. I²C Block Diagram



In PolarFire SoC, I2C peripherals consist mainly of the following components.

- Input Glitch Filter
- Arbitration and Synchronization Logic
- Address Comparator
- Serial Clock Generator

3.6.1.3 Routing of I²C Signals

In PolarFire SoC, I²C signals are routed through MSSIO pins.

3.6.1.4 I²C Bare-Metal Driver

Use the `mss-i2c` driver in `driver-examples/mss` from this GitHub repository, and modify as required.

3.6.1.5 I²C Linux Driver

Use the I²C Linux driver and verify the application code dependencies, if any, with a valid API.

3.6.1.6 I²C Device Tree Binding

The following table lists the I²C device tree binding information.

Table 3-3. I²C Device Tree Binding Comparison

Information	Zynq®-7000 SoC		PolarFire® SoC	
Type	i2c0	i2c1	i2c0	i2c1
Compatible	cdns, i2c-r1p10	cdns, i2c-r1p10	microchip, mpfs-i2c	microchip, mpfs-i2c
Input clock specifier	&clkc 38	&clkc 39	&clkcfg 16	&clkcfg 17
Operating frequency	Configurable	Configurable	100000 (Configurable)	100000 (Configurable)
Interrupt specifier	GIC_SPI 25 IRQ_TYPE_LEVEL_HIGH	—	PLIC_INT_I2C0_MAIN	PLIC_INT_I2C1_MAIN

.....continued				
Information	**Zynq®-7000 SoC**		**PolarFire® SoC**	
Physical base address	0xE0004000	0xE0005000	0x0 0x2010b000	0x0 0x2010b000
Range	0x1000	0x1000	0x1000	0x1000

3.6.2 MMUART

Multi-Mode Universal Asynchronous/Synchronous Receiver/Transmitter (MMUART) performs serial-to-parallel conversion on data originating from modems or other serial devices, and performs parallel-to-serial conversion on data from the MSS Core Complex processor or fabric master to these devices. PolarFire SoC FPGAs contain five identical MMUART peripherals in the microprocessor subsystem (MMUART_0, MMUART_1, MMUART_2, MMUART_3, and MMUART_4).

When porting UART controller from a Zynq-7000 SoC to PolarFire SoC, change the base address and alternate base address to the new addresses related to the PolarFire SoC MMUART0, MMUART1, MMUART2, MMUART3, and MMUART4. The base addresses and register descriptions of MMUART are listed in Polar Fire SoC Device Register Map.

3.6.2.1 Features

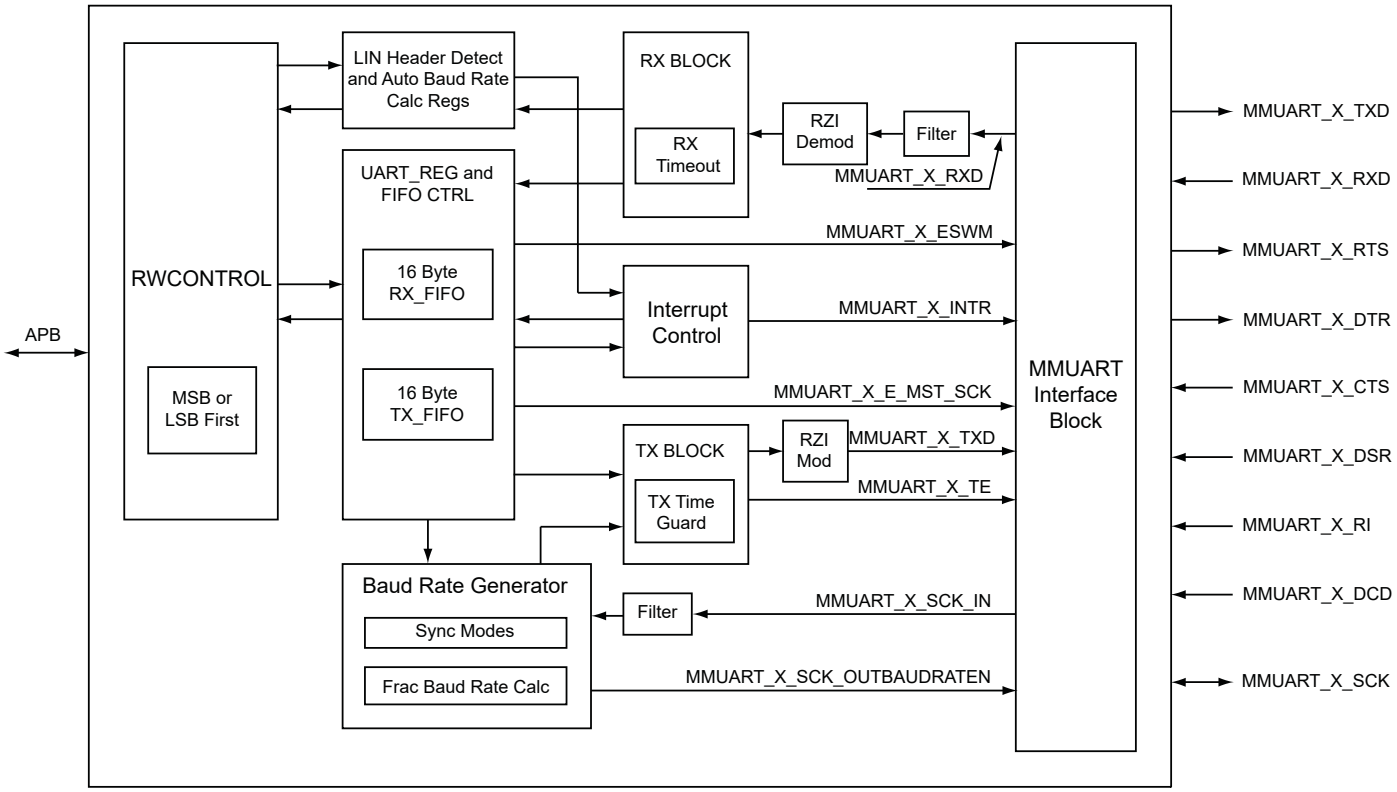
In PolarFire SoC, MMUART supports the following features.

- Asynchronous and synchronous operations
- Full programmable serial interface characteristics
  - Data width is programmable to 5, 6, 7, or 8 bits
  - Even, odd, or no-parity bit generation/detection
  - 1, 1½, and 2 stop bit generation
- 9-bit address flag capability used for multi-drop addressing topologies
- Separate transmit (Tx) and receive (Rx) FIFOs to reduce processor interrupt service loading
- Single-wire Half-Duplex mode in which Tx pad can be used for bidirectional data transfer
- Local Interconnect Network (LIN) header detection and auto-baud rate calculation
- Communication with ISO 7816 smart cards
- Fractional baud rate capability
- Return to Zero Inverted (RZI) mod/demod blocks that allow Infrared Data Association (IrDA) and Serial Infrared (SIR) communications
- MSb or the LSb is the first bit while sending or receiving data

3.6.2.2 Functional Description

The following figure shows the PolarFire SoC MMUART block diagram.

Figure 3-6. MMUART Block Diagram



For more information about MMUART functional description, see PolarFire SoC FPGA MSS Technical Reference Manual.

3.6.2.3 MMUART Operational Modes

PolarFire SoC MMUART (x5) supports the following operational modes:

- Normal mode
- Automatic echo mode
- Local loopback mode
- Remote loopback mode

3.6.2.4 Routing of MMUART Signals

In PolarFire SoC, MMUART signals are routed through MSSIO or the FPGA Fabric I/O pins.

3.6.2.5 MMUART Bare-Metal Driver

Use the `mss-mmuart` driver in `driver-examples/mss` from this GitHub repository, and modify as required.

3.6.2.6 MMUART Linux Driver

Use the MMUART Linux driver and verify the application code dependencies if any with a valid API.

3.6.2.7 MMUART Device Tree Binding

The following table lists the MMUART device tree binding information.

Table 3-4. PolarFire SoC MMUART Device Tree

Information	MMUART Device Tree Binding Details				
Type	mmuart0	mmuart1	mmuart2	mmuart3	mmuart4
Compatible	ns16550a	ns16550a	ns16550a	ns16550a	ns16550a
Input clocks	&clkc 9	&clkc 10	&clkc 11	&clkc 12	&clkc 13
Baud rate	115200 (default)	115200 (default)	115200 (default)	115200 (default)	115200 (default)

.....continued					
Information	**MMUART Device Tree Binding Details**				
Physical base address Size of register	0x20000000 0x400	0x20100000 0x400	0x20102000 0x400	0x20104000 0x400	0x20106000 0x400
Interrupt Number	PLIC_INT_MMUART0 90	PLIC_INT_MMUART0 91	PLIC_INT_MMUART0 92	PLIC_INT_MMUART0 93	PLIC_INT_MMUART0 94

Table 3-5. Zynq-7000 SoC UART Device Tree

Information	MMUART Device Tree Binding Details	
	uart0	uart1
Compatible	xlnx, xuartps	xlnx, xuartps
Input clocks	&clkc 23, &clkc 40	&clkc 24, &clkc 41
Baud rate	115200 (default)	115200 (default)
Physical base address Range	0xE0000000 0x1000	0xE0001000 0x1000
Interrupt number	27	50

3.6.3 CAN Controller

PolarFire SoC FPGAs contain two identical integrated Control Area Network (CAN) peripherals. The CAN controller is an Advanced Peripheral Bus (APB) slave on the MSS AHB bus matrix. A master such as the MSS Core Complex or a master in the FPGA fabric configures the CAN controller through the APB slave.

When porting CAN controller from a Zynq-7000 SoC to PolarFire SoC, change the base address and alternate base address to the new addresses related to PolarFire SoC CAN_A and CAN_B. The base addresses and register descriptions of CAN Controllers are listed in Polar Fire SoC Device Register Map.

The following tables compares the CAN register set of Zynq-7000 SoC and PolarFire SoC.

Table 3-6. CAN Register Comparison

Zynq®-7000 SoC			PolarFire® SoC		
Register Type	**Register Name**	**Offset**	**Register Type**	**Register Name**	**Offset**
Configuration and Control	SRR	0x00	configuration	CAN_CONFIG	0x018
	MSR	0x04	Command for different modes of operation	CAN_COMMAND	0x014
	BRPR	0x08	Interrupt Enable	INT_ENABLE	0x004
	BTR	0x0c	Interrupt status register	INT_STATUS	0x000
	TCR	0x28	Receive (RX) message buffer status	RX_BUF_STATUS	0x008
	—	—	Transmit (TX) message buffer status	TX_BUF_STATUS	0x00C

.....continued					
Zynq®-7000 SoC			**PolarFire® SoC**		
Register Type	**Register Name**	**Offset**	**Register Type**	**Register Name**	**Offset**
Interrupt processing	ISR	0x1c	Error status indicator register	ERROR_STATUS	0x010
	IER	0x20	Error capture register	ECR	0x01c
	ICR	0x24	—	—	—
	WIR	0x2c	—	—	—
			—	—	—
Status	ECR	—	—	—	—
	ESR	—	—	—	—
	SR	—	—	—	—
Transmit FIFO	TXFIFO_ID	0x30	—	—	—
	TXFIFO_DLC	0x34	—	—	—
	TXFIFO_DATA1	0x38	—	—	—
	TXFIFO_DATA2	0x3c	—	—	—
Transmit high-priority buffer	TXHPB_ID	0x40	—	—	—
	TXHPB_DLC	0x44	—	—	—
	TXHPB_DATA1	0x48	—	—	—
	TXHPB_DATA2	0x4c	—	—	—
Receive FIFO	RXFIFO_ID	0x50	—	—	—
	RXFIFO_DLC	0x54	—	—	—
	RXFIFO_DATA1 0x58	0x58	—	—	—
	RXFIFO_DATA2	0x5c	—	—	—
Acceptance filter	AFR	0x60	—	—	—
	AFMR1	0x64	—	—	—
	AFIR1	0x68	—	—	—
	AFMR2	0x6C	—	—	—
	AFIR2	0x70	—	—	—
	AFIR3	0x74	—	—	—
	AFMR3	0x78	—	—	—
	AFMR4	0x7C	—	—	—
	AFIR4	0x80	—	—	—

3.6.3.1 Features

CAN controller supports the following features:

- Full CAN 2.0B compliant
- Conforms to ISO 11898-1
- Maximum baud rate of 1 Mbps with 8 MHz CAN clock

- APB 3.0 compliant
- APB interface has clock-domain-crossing to CAN logic, allowing APB to operate at any frequency
- 32 receive (Rx) buffers
- Each buffer has its own message filter
- Message filter covers: ID, IDE, Remote Transmission Request (RTR), data byte 1, and data byte 2
- Message buffers can be linked together to build a bigger message array
- Automatic RTR response handler with optional generation of RTR interrupt
- 32 transmit (Tx) message holding registers with programmable priority arbitration
- Message abort command
- Single-Shot Transmission (SST); no automatic retransmission upon error or arbitration loss
- AMBA 3 APB Interface
- Full synchronous zero wait-states interface
- Status and configuration interface
- Local interrupt controller covering message and CAN error sources
- Optimized for low gate-count implementation
- Single port, synchronous memory based
- 100% synchronous design

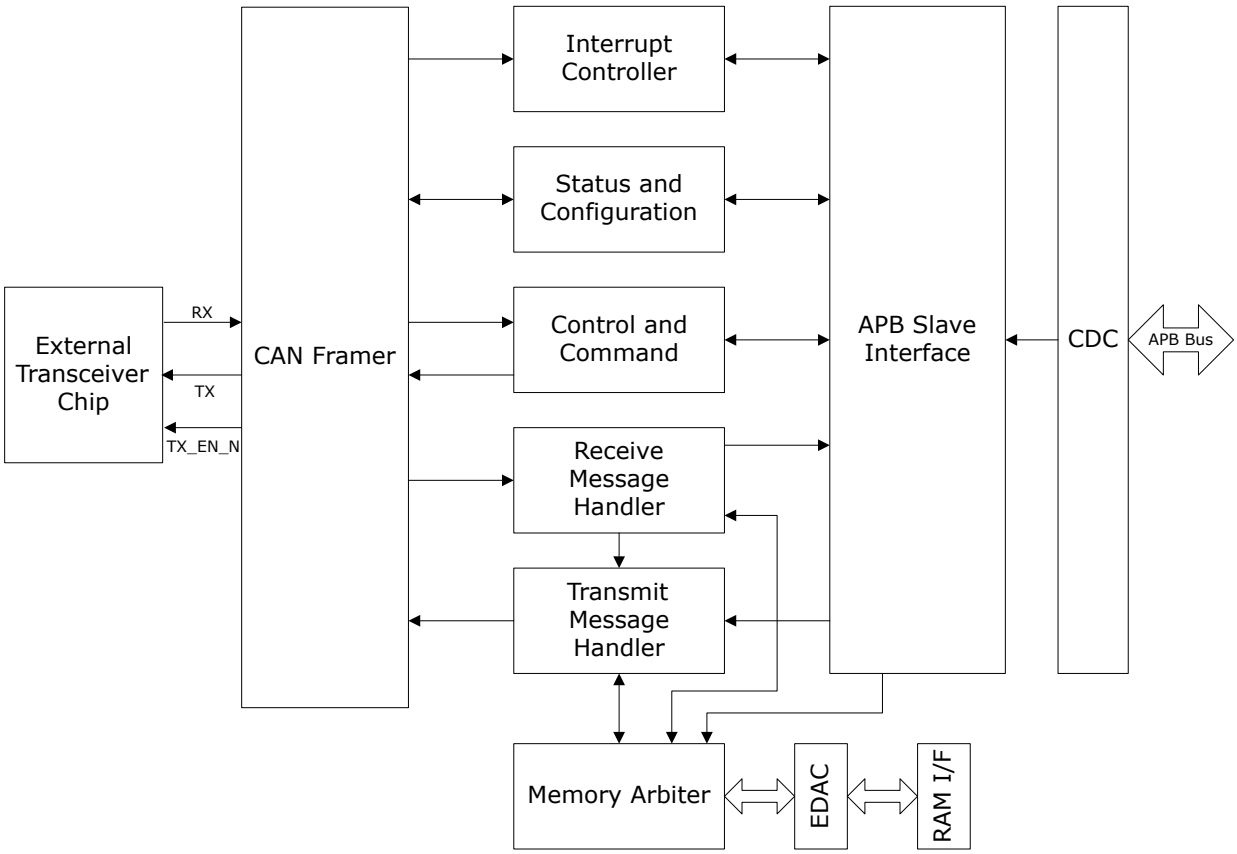
In PolarFire SoC, CAN supports the following test and debug features.

- Listen Only mode
- Internal Loopback mode
- External Loopback mode
- SRAM Test mode
- Error Capture register
- Provides option to either: Show current bit position within CAN message
- Provides option to either: Show bit position and type of last captured CAN error

3.6.3.2 Functional Description

The following figure shows the PolarFire SoC CAN Controller block diagram.

Figure 3-7. CAN Controller Block Diagram



The CAN controller in the PolarFire SoC FPGAs support the concept of mailboxes. The CAN controller contains 32 receive buffers. Each buffer has its own message filter and 32 transmit buffers with prioritized arbitration scheme. For optimal support of Higher-Layer Protocols (HLP) such as DeviceNet, the message filter also covers the first two data bytes of the message payload.

Transmit and receive message buffers are single error corrected, double error detected (SECDED) through the Error Detection and Correction (EDAC) controller. To remove the requirement of APB clock in multiples of 8 MHz, a separate CAN clock is provided and a Clock Domain Crossing (CDC) logic is added from the APB bus. The CDC logic uses toggle synchronizers and there is no restriction on the APB clock relative to the CAN clock.

3.6.3.3 CAN Operational Modes

Sleep mode and Snoop Mode (diagnostics) are not supported in PolarFire SoC.

3.6.3.4 Routing of CAN Signals

In PolarFire SoC, CAN signals can be routed to MSSIO pins or Fabric I/Os.

3.6.3.5 CAN Bare-Metal Driver

Use the `mss-can` driver in `driver-examples/mss` from this GitHub repository, and modify as required.

3.6.3.6 CAN Linux Driver

Use the CAN Linux driver and verify the application code dependencies if any with a valid API.

3.6.3.7 CAN Device Tree Binding

The following table lists the CAN device tree binding information.

Information	Zynq®-7000 SoC		PolarFire® SoC	
Type	can0	can1	can0	can1

.....continued				
Information	**Zynq®-7000 SoC**		**PolarFire® SoC**	
Compatible	xlnx, zynq-can-1.0	xlnx, zynq-can-1.0	mpfs-can-uio	mpfs-can-uio
Input clocks	can_clk, pclk	can_clk, pclk	CLK_CAN0	CLK_CAN1
Interrupt number	28	29	56	57
Interrupt parent	&intc	&intc	&plic	&plic
Physical base address Range	0xe0008000 0x1000	0xe0009000 0x1000	0x2010c000 0x1000	0x2010d000 0x1000
CAN Rx fifo depth	0x40	0x40	—	—
CAN Rx fifo depth	0x40	0x40	—	—

3.6.4 Quad SPI with XIP

In PolarFire SoC, the QSPI controller is an AHB slave that provides a serial interface compliant with the Motorola SPI format. QSPI with Execute In Place (XIP) support allows a processor to directly boot rather than moving the SPI content to SRAM before execution.

When porting Quad SPI (QSPI) from Zynq-7000 SoC to PolarFire SoC, change the base address and alternate base address to the new addresses related to PolarFire SoC. The base addresses and register descriptions of QSPI are listed in Polar Fire SoC Device Register Map.

The following table lists the PolarFire SoC Quad SPI register map with offsets.

S. No	CAN Controller Registers	Register Name Description	Offset
1	QSPIXIP: CONTROL	Control register	0x000
2	QSPIXIP: FRAMES	This register is only functional in master mode It is used to start a master transfer.	0x004
3	QSPIXIP: Reserved	Reserved register	0x008
4	QSPIXIP: Interrupt Enable Register	This is interrupt enable register for external register.	0x00C
5	QSPIXIP: Status Register	Indicates the status of the SPI core.	0x010
6	QSPIXIP: Direct Access	This register allows direct access to the QSPI interface pins to support access to non-standard SPI devices via direct CPU control.	0x00C
7	QSPIXIP: Upper Address	This register set the upper 8-bits of the address [31:24] when four byte addressing is being used in XIP mode.	0x010

XIP Mode

When in XIP mode, only writes can be performed to the registers. Read operations to the registers fetch SPI contents.

3.6.4.1 Features

In PolarFire SoC, Quad SPI supports the following features:

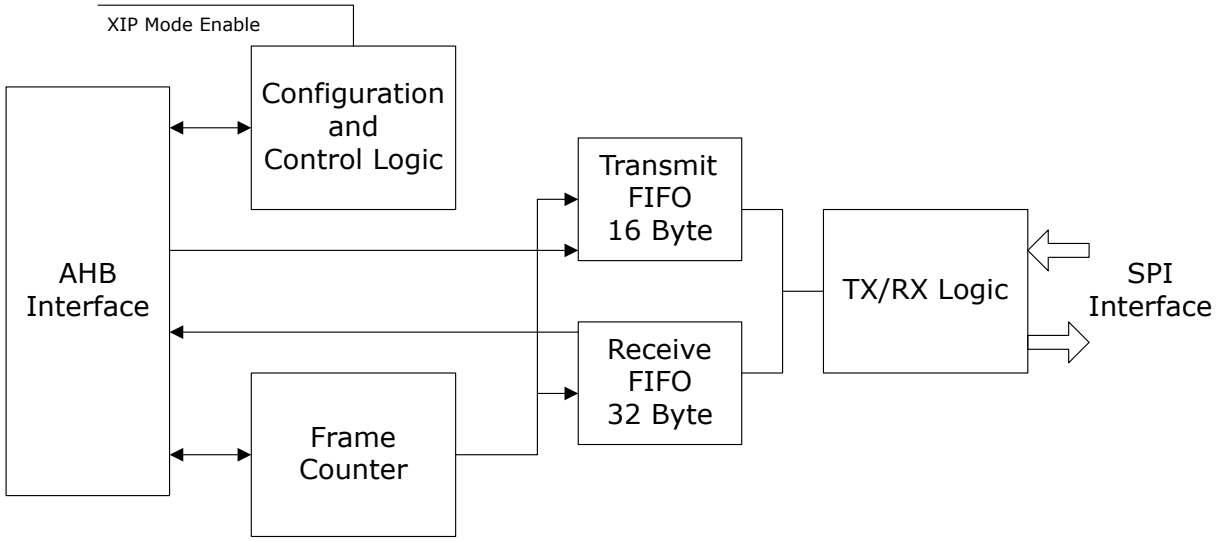
- Master only operation with SPI data-rate
  - Programmable SPI clock—HCLK/2, HCLK/4, or HCLK/6
  - Maximum data-rate is HCLK/2
- FIFOs
  - Transmit and Receive FIFO
  - 16-byte transmit FIFO depth
  - 32-byte receive FIFO depth

  - AHB interface transfers up to four bytes at a time
- SPI Protocol
  - Master operation
  - Motorola SPI supported
  - Slave Select operation in idle cycles configurable
  - Extended SPI operation (1, 2, and 4-bit)
  - QSPI operation (4-bit operation)
  - BSPI operation (2-bit operation)
  - Execute in place (XIP)
  - Three or four-byte SPI address.
- Frame Size
  - 8-bit frames directly
  - Back-to-back frame operation supports greater than 8-bit frames
  - Up to 4 GB Transfer (2 × 32 bytes)
- Processor overhead reduction
  - SPI Flash command/data packets with automatic data generation and discard function
- Direct Mode
  - Allows a CPU to directly control the SPI interface pins.

3.6.4.2 Functional Description

The following figure shows the PolarFire SoC Quad SPI block diagram.

Figure 3-8. QSPI Controller Block Diagram



The QSPI controller supports only Master mode operation. The Master mode operation runs directly off the controller clock (HCLK) and supports SPI transfer rates at the HCLK/2 frequency and slower.

The SPI peripherals consist mainly of the following components.

- Transmit and receive FIFOs
- Configuration and control logic

For more information about the QSPI functional description, see PolarFire SoC FPGA MSS Technical reference Manual.

3.6.4.3 Routing of QSPI Signals

The QSPI data signals are routed to MSS I/O or fabric I/O pins.

3.6.4.4 QSPI Bare-Metal Drivers

Use the `mss-qspi` driver in `driver-examples/mss` from this GitHub repository, and modify as required.

3.6.4.5 QSPI Linux Driver

Use the QSPI Linux driver and verify the application code dependencies if any with a valid API.

3.6.4.6 QSPI Device Tree Binding

The following table lists the QSPI device tree binding information.

Information	Zynq®-7000 SoC	PolarFire® SoC
Compatible	xlnx, zynq-qspi-1.0	“microchip, mpfs-qspi”
Input clocks	“ref_clk”, “Polk”;	CLK_QSPI
Interrupt number	19	PLIC_INT_QSPI (85)
Interrupt parent	&intc	&plic
Physical base address Range	0xe000d000 0x1000	0x21000000 0x1000

3.6.5 SPI Controller

When porting SPI Controller from Zynq-7000 SoC to PolarFire SoC, change the base address and alternate base address to the new addresses related to PolarFire SoC. The base addresses and register descriptions of SPI are listed in Polar Fire SoC Device Register Map.

The following table lists the PolarFire SoC SPI register map with offsets.

Table 3-7. PolarFire SoC SPI Register Map

S. No	Register Name Description	Register Name	Offset
1	Control register	SPI: CONTROL	0x000
2	Transmit and receive data frame size.	SPI: FRAMESIZE	0x004
3	Output clock generator (Master mode)	SPI: CLK_GEN	0x018
4	Transmit data register	SPI: TX_DATA	0x014
5	Status register	SPI: STATUS	0x008
6	Receive data register	SPI: RX_DATA	0x010
7	Interrupt clear register	SPI: INT_CLEAR	0x00C
8	Specifies slave selected (Master mode)	SPI: SLAVE_SELECT	0x01C
9	Masked interrupt status	SPI: INTMASK	0x020
10	Raw interrupt status	SPI: INTRAW	0x024
11	Control bits for enhanced modes	SPI: CONTROL2	0x028
12	Command register	SPI: COMMAND	0x02C
13	PACKET SIZE	SPI: PKTSIZE	0x030
14	Command size	SPI: CMD_SIZE	0x034
15	Slave hardware status	SPI: HWSTATUS	0x038
16	Status register	SPI: STAT8	0x03C

SPI Operational Modes

In PolarFire SoC, SPI supports Master and Slave modes of operation.

3.6.5.1 Features

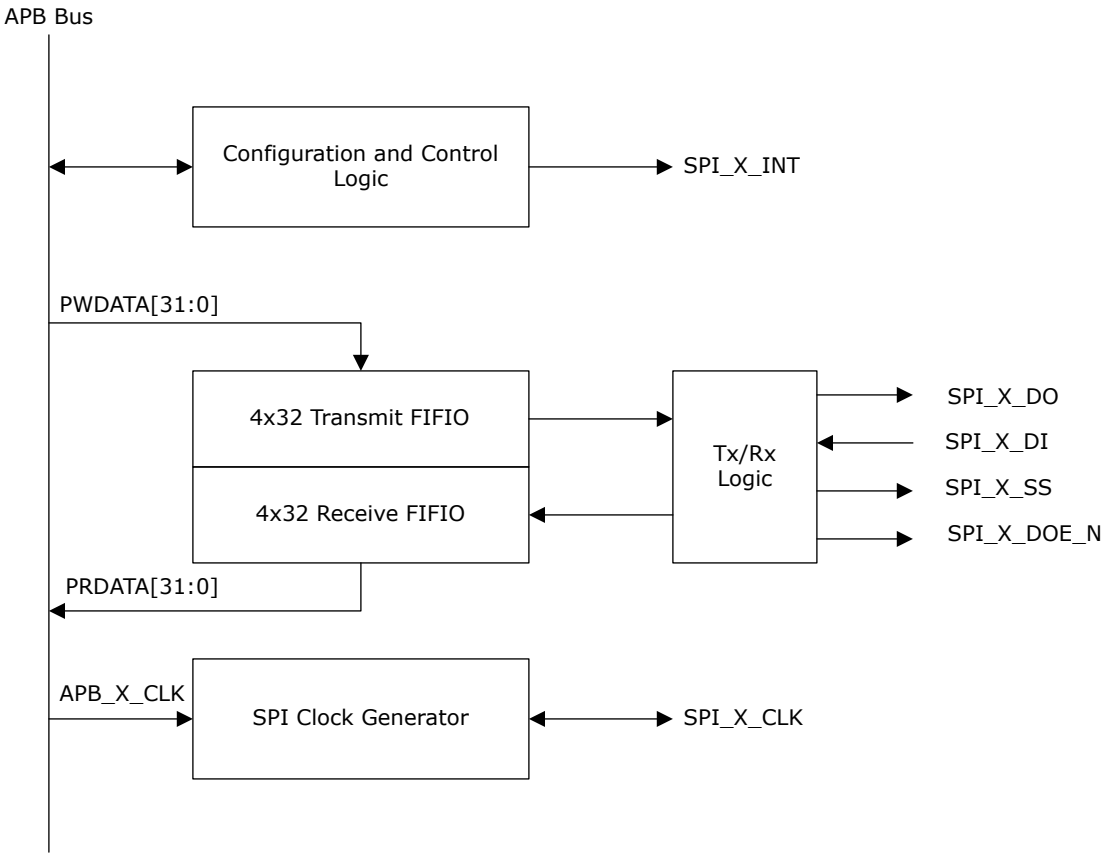
In PolarFire SoC, SPI peripherals support the following features.

- Master and Slave modes
- Configurable Slave Select operation
- Configurable clock polarity
- Separate transmit (Tx) and receive (Rx) FIFOs to reduce interrupt service loading

3.6.5.2 Functional Description

The following figure shows the PolarFire SoC SPI controller block diagram.

Figure 3-9. SPI Controller Block Diagram



The SPI controller supports Master and Slave modes of an operation.

- In Master mode, the SPI generates SPI_X_CLK, selects a slave using SPI_X_SS, transmits the data on SPI_X_DO, and receives the data on SPI_X_DI.
- In Slave mode, the SPI is selected by SPI_X_SS. The SPI receives a clock on SPI_X_CLK and incoming data on SPI_X_DI.

The SPI peripherals consist mainly of the following components (see Figure 3-9).

- Transmit and receive FIFOs
- Configuration and control logic
- SPI clock generator

3.6.5.3 Routing of SPI Signals

The SPI signals are routed to MSS I/O or fabric I/O pins.

3.6.5.4 SPI Bare-Metal Driver

Use the `mss-spi` driver in `driver-examples/mss` from this GitHub repository, and modify as required.

3.6.5.5 SPI Linux Driver

Use the SPI Linux driver and verify the application code dependencies if any with a valid API.

3.6.5.6 SPI Device Tree Binding

The following table lists the SPI device tree binding information.

Information	Zynq®-7000 SoC		PolarFire® SoC	
Type	spi0	spi1	spi0	spi1
Compatible	xlnx, zynq-spi-r1p6	xlnx, zynq-spi-r1p6	microchip, mpfs-spi	microchip, mpfs-spi
Interrupt number	26	49	PLIC_INT_SPI0 (54)	PLIC_INT_SPI1 (55)
Physical base address	0Xe0006000	0xe0007000	0x20108000	0x20109000
Input clocks	clkc 25, clkc 34	clkc 26, clkc 35	CLK_SPI0	CLK_SPI1

3.6.6 Gigabit Ethernet MAC

The PolarFire SoC MSS contains two hardened Gigabit Ethernet MAC IP blocks—GEM_0 and GEM_1— to enable Ethernet solutions over copper or optical cabling.
GEM_0 and GEM_1 are functionally identical, hence, GEM_0 and GEM_1 are referred as GEM throughout the document.

GEM supports 10 Mb/s, 100 Mb/s, and 1000 Mb/s (1 Gb/s) speeds. GEM provides a complete range of solutions for implementing IEEE® 802.3 standard-compliant Ethernet interfaces for chip-to-chip, board-to-board, and backplane interconnects.

When porting Gigabit Ethernet MAC (GEM) from Zynq-7000 SoC to PolarFire SoC, change the base address and alternate base address specific to PolarFire SoC. The base addresses and register descriptions of GEM are listed in Polar Fire SoC Device Register Map.

3.6.6.1 Features

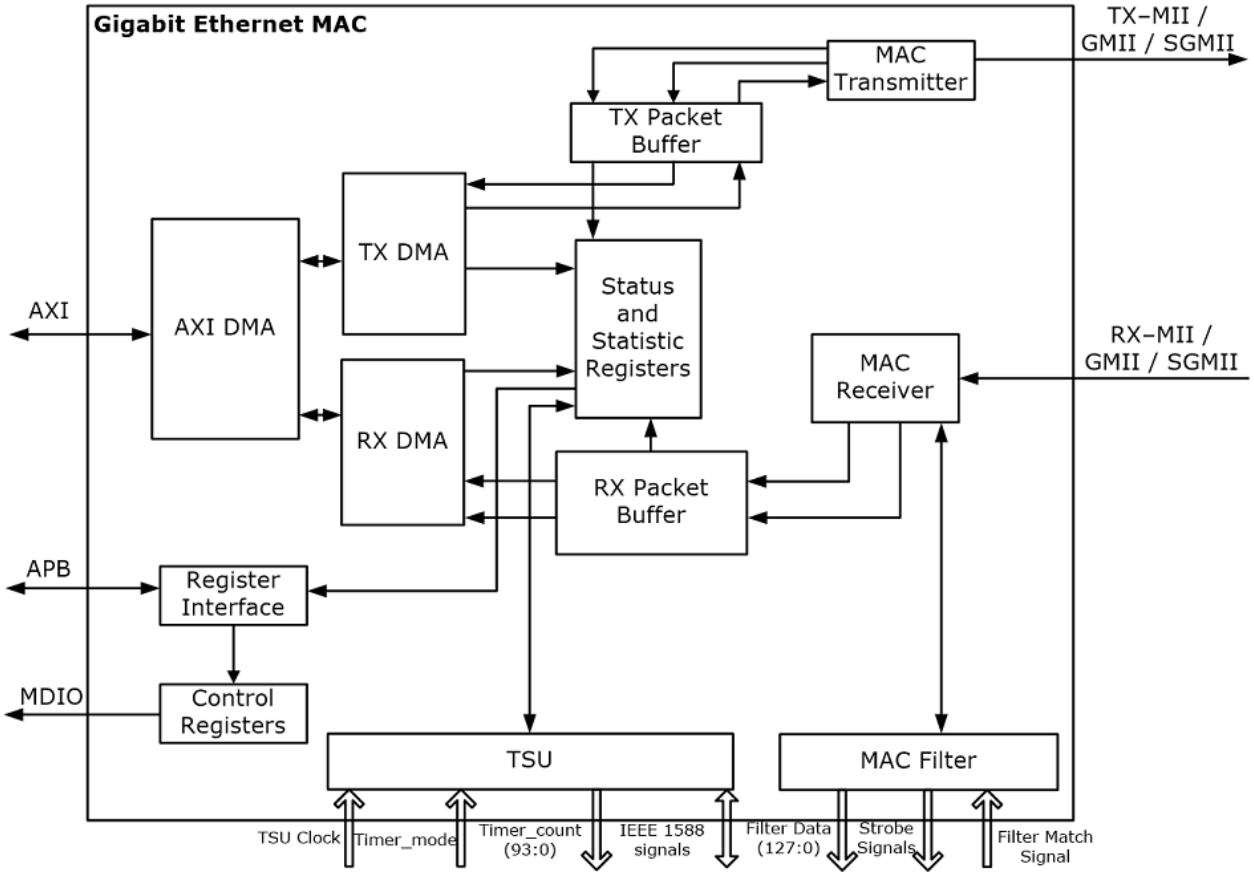
GEM supports the following features.

- IEEE 802.3 compliant
- IEEE 802.1Q TSN features:
  - IEEE 802.1AS
  - IEEE 802.1Qav
  - IEEE 802.1Qbv
  - IEEE 802.1CB frame redundancy and elimination
  - IEEE 802.1Qci receive (ingress) traffic policing
  - IEEE 802.3br frame preemption (or interspersing express traffic)
  - IEEE 802.1Qbb priority-based flow control
  - IEEE 802.1Q VLAN tagging with recognition of incoming VLAN and priority tagged frames
- DMA support
- TCP/IP offloading capability
- Integrated 1000 BASE-X PCS for SGMII-based applications
- Programmable jumbo frames up to 10,240 bytes
- Frame Filtering
- Full and half duplex modes at 10/100M and full duplex at 1 Gbps interface speeds for MII, GMII, and SGMII.
- Wake-on LAN support

3.6.6.2 Functional Description

The following figure shows the PolarFire SoC GEM block diagram.

Figure 3-10. Functional Block Diagram



GEM includes the following functional blocks:

- Integrated 1000BASE-X Physical Coding Sublayer (PCS) for encoding and decoding the data and for Auto Negotiation (AN)
- Time Stamping Unit (TSU) for timer operations
- TSN block to support Timing Sensitive Networking (TSN) features
- High-speed AXI DMA block to transfer data to and from the processor
- Filter block filters out the received frames

For more information about GEM functional description, see PolarFire SoC FPGA MSS Technical Reference Manual.

3.6.6.3 GEM Register Comparison

The following table lists GEM registers of Zynq-7000 SoC and PolarFire SoC.

Table 3-8. GEM Register Comparison

PolarFire® SoC	Zynq®-7000 SoC	Offset in Zynq-7000 SoC	Offset in PolarFire SoC
NETWORK_CONTROL	net_ctrl	0x00	0x00
NETWORK_CONFIG	net_cfg	0x04	0x0004
NETWORK_STATUS	net_status	0x08	0x0008
tx_pause_quantum	tx_pauseq	0x3C	0x003C
tx_pfc_pause	tx_pfc_pause	0xC4	0x00C4
pause_time	rx_pauseq	0x38	0x0038
stretch_ratio	ipg_stretch	0xBC	0x00BC

.....continued			
PolarFire® SoC	**Zynq®-7000 SoC**	**Offset in Zynq-7000 SoC**	**Offset in PolarFire SoC**
stacked_vlan	stacked_vlan	0xC0	0x00C0

The following table lists the GEM DMA unit registers of Zynq-7000 SoC and PolarFire SoC.

Table 3-9. GEM DMA Register Comparison

PolarFire® SoC	Zynq®-7000 SoC	Offset in Zynq-7000 SoC	Offset in PolarFire SoC
dma_config	dma_cfg	0x10	0x0010
transmit_status	tx_status	0x14	0x0014
transmit_q_ptr	tx_qbar	0x1C	0x001C
receive_q_ptr	rx_qbar	0x18	0x0018
receive_status	rx_status	0x20	0x0020

The following table lists the GEM interrupts registers of Zynq-7000 SoC and PolarFire SoC.

Table 3-10. GEM Interrupt Register Comparison

PolarFire® SoC	Zynq®-7000 SoC	Offset in Zynq-7000 SoC	Offset in PolarFire SoC
int_status	intr_status	0x24	0x0024
int_enable	intr_en	0x28	0x0028
nt_disable	intr_dis	0x2c	0x002C
int_mask	intr_mask	0x30	0x0030
wol_register	wake_on_lan	0xB8	0x00B8

The following table lists the GEM PHY maintenance register of Zynq-7000 SoC and PolarFire SoC.

Table 3-11. GEM PHY Register Comparison

PolarFire® SoC	Zynq®-7000 SoC	Offset in Zynq-7000 SoC	Offset in PolarFire SoC
phy_management	phy_maint	0x34	0x0034

The following table lists the GEM MAC address filtering ID match registers of Zynq-7000 SoC and PolarFire SoC.

Table 3-12. GEM MAC Address Filtering and ID Match Registers

PolarFire® SoC	Zynq®-7000 SoC	Offset in Zynq-7000 SoC	Offset in PolarFire SoC
hash_bottom	hash_bot	0x80	0x0080
hash_top	hash_top	0x84	0x0084
spec_add1_bottom	spec_addr1_bot	0x88	0x0088
spec_add1_top	spec_addr1_top	0x8C	0x008C
spec_add2_bottom	spec_addr2_bot	0x90	0x0090
spec_add2_top	spec_add2_top	0x94	0x0094
spec_add3_bottom	spec_add3_bot	0x98	0x0098
spec_add3_top	spec_add3_top	0x9C	0x009C
spec_add4_bottom	spec_add4_bot	0xA0	0x00A0

.....continued			
PolarFire® SoC	**Zynq®-7000 SoC**	**Offset in Zynq-7000 SoC**	**Offset in PolarFire SoC**
spec_add4_top	spec_add4_top	0xA4	0x00A4
mask_add1_bottom	mask_add1_bot	0xC8	0x00C4
mask_add1_top	mask_add1_top	0xCC	0x00C8
spec_type1	type_id_match1	0xA8	0x00A8
spec_type2	type_id_match2	0XAC	0x00AC
spec_type3	type_id_match3	0XB0	0X00B0
spec_type4	type_id_match4	0xB4	0x00B4

The following table lists the GEM Module ID register.

Table 3-13. GEM Module ID Register Comparison

PolarFire SoC	Zynq-7000 SoC	Offset in Zynq-7000 SoC	Offset in PolarFire SoC
revision_reg (gem_revision_reg_value)	module_id	0xFC	0x00FC

3.6.6.4 Routing of GEM Signals

In PolarFire SoC, GEM signals can be routed to MSS I/O pins for SGMII and to fabric I/O pins for GMII.

3.6.6.5 GEM Bare-Metal Driver

Use the `mss-ethernet-mac` driver in `driver-examples/mss` from this GitHub repository, and modify as required.

3.6.6.6 GEM Linux Driver

Use the GEM Linux driver and verify the application code dependencies if any with a valid API.

3.6.6.7 GEM Device Tree Binding

The following table lists the GEM device tree binding information.

Table 3-14. GEM Device Tree Binding

Information	Zynq®-7000 SoC		PolarFire® SoC	
Compatible	xlnx,zynq-gem	xlnx,zynq-gem	cdns,macb	cdns,macb
Interrupt Number	22	45	PLIC_INT_MAC0_INT- 64 PLIC_INT_MAC0_QUEUE1 -65 PLIC_INT_MAC0_QUEUE2-66 PLIC_INT_MAC0_QUEUE3-67 PLIC_INT_MAC0_EMAC-68 PLIC_INT_MAC0_MMSL-69	PLIC_INT_MAC1_INT-70 PLIC_INT_MAC1_QUEUE1-71 PLIC_INT_MAC1_QUEUE2-72 PLIC_INT_MAC1_QUEUE3-73 PLIC_INT_MAC1_EMAC-74 PLIC_INT_MAC1_MMSL-75
Physical base address Range	0xe000b000 0x1000	0xe000c000 0x1000	0x20110000 0x2000	0x20112000 0x2000
Input clocks	clkc 30, clkc13	clkc 31, clkc14	CLK_MAC0 5	CLK_MAC1 6
PHY type	marvell,88e1116r	marvell,88e1116r	Microchip's VSC8662XIC	Microchip's VSC8662XIC

3.6.7 General Purpose I/O

PolarFire SoC FPGAs contain three identical general-purpose input/output (GPIO) blocks in the MSS (GPIO_0, GPIO_1, and GPIO_2). Each GPIO block is an APB slave that provides access to 32 GPIOs. MSS Masters and fabric Masters can access the MSS GPIO block through the AMBA interconnect.

3.6.7.1 Features

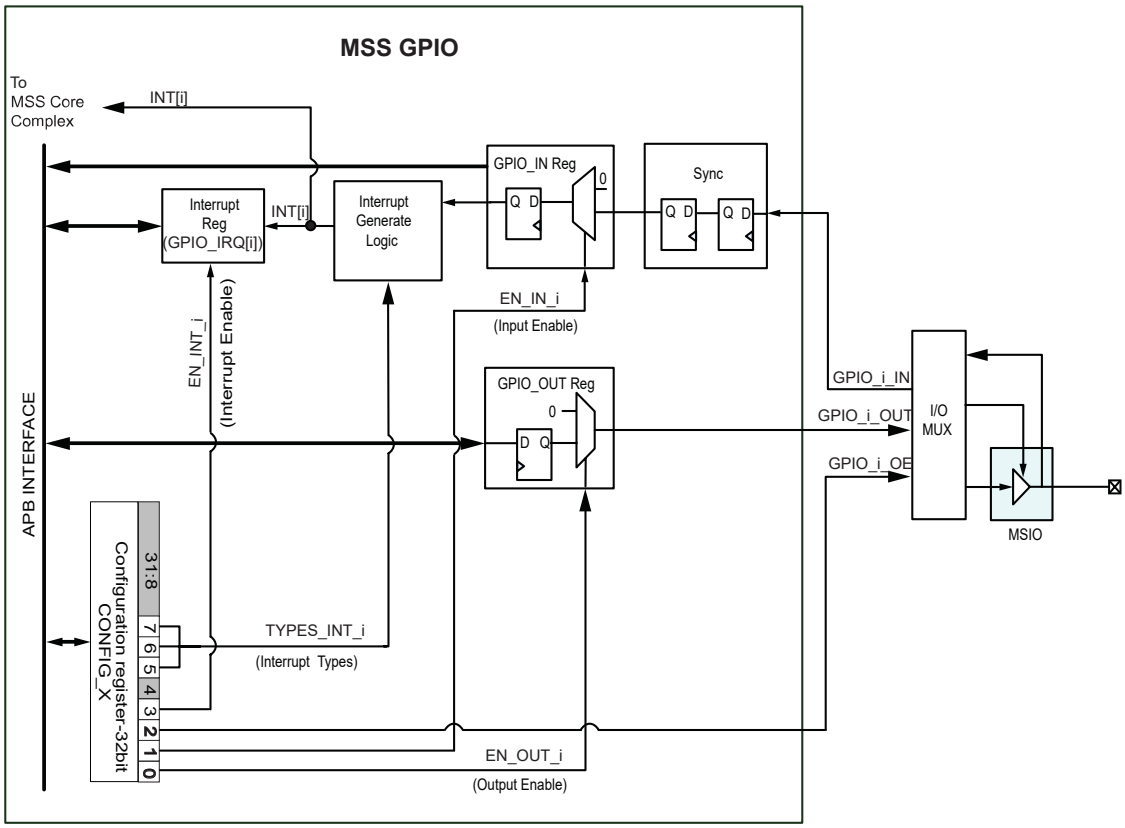
MSS GPIO supports the following features.

- GPIO_0 drives up to 14 MSIOs
- GPIO_1 drives up to 24 MSIOs
- GPIO_2 drives up to 32 device IOs via the FPGA fabric.
- 32 individually configurable GPIOs
- Each GPIO is dynamically programmable as an input, output, or bidirectional I/O.
- Each GPIO can be configured as an interrupt source to the MSS processor in Input mode
- The Reset state of the GPIOs is configurable
- The GPIOs can be selectively reset by either the Hard Reset (Power-on Reset, User Reset from the fabric) or the Soft Reset from the SYSREG block

3.6.7.2 Functional Description

The following figure shows the MSS GPIO block diagram.

Figure 3-11. MSS GPIO Block Diagram



The MSS GPIO block contains the following registers.

- 32-bit input register (GPIO_IN), which holds the input values
- 32-bit output register (GPIO_OUT), which holds the output values
- 32-bit interrupt register (GPIO_INTR), which holds the interrupt state
- 32 configuration registers (GPIO_X_CONFIG), one register for each GPIO

For more information about the GPIO functional description, see PolarFire SoC FPGA MSS Technical Reference Manual.

3.6.7.3 Routing of GPIO Signals

In PolarFire SoC, GPIO signals can be routed to MSS I/O pins or to fabric I/O pins.

3.6.7.4 GPIO Bare-Metal Driver

Use the `mss-gpio` driver in `driver-examples/mss` from this GitHub repository, and modify as required.

3.6.7.5 GPIO Linux Driver

Use the GPIO Linux driver and verify the application code dependencies if any with a valid API.

3.6.7.6 GPIO Device Tree Binding

The following table lists the GPIO device tree binding information.

Table 3-15. GPIO Device Tree Binding

Information	Zynq®-7000 SoC	PolarFire® SoC		
Type	gpio0	gpio0	gpio1	gpio2
Compatible	xlnx, zynq-gpio-1.0	microchip, mpfs-gpio	microchip, mpfs-gpio	microchip, mpfs-gpio
Interrupt Number	20	&plic	&plic	&plic
Physical base address Range	0xe000a000 0x1000	0x20120000 0x1000	0x20121000 0x1000	0x20122000 0x1000
Input clocks	&clkc	CLK_GPIO0	CLK_GPIO1	CLK_GPIO2

3.7 Validating the Application

Once the PolarFire SoC MSS boots up, validate a bare-metal application and bring up Linux. Validate the following bare-metal and Linux sample applications.

- Bare-Metal application
- Linux application

4. Revision History

Revision	Date	Description
A	09/2022	Initial Revision

Microchip FPGA Support

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

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

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

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

Microchip Information

The Microchip Website

Microchip provides online support via our website at www.microchip.com/. This website is used to make files and information easily available to customers. Some of the content available includes:

- **Product Support** – Data sheets and errata, application notes and sample programs, design resources, user's guides and hardware support documents, latest software releases and archived software
- **General Technical Support** – Frequently Asked Questions (FAQs), technical support requests, online discussion groups, Microchip design partner program member listing
- **Business of Microchip** – Product selector and ordering guides, latest Microchip press releases, listing of seminars and events, listings of Microchip sales offices, distributors and factory representatives

Product Change Notification Service

Microchip's product change notification service helps keep customers current on Microchip products. Subscribers will receive email notification whenever there are changes, updates, revisions or errata related to a specified product family or development tool of interest.

To register, go to www.microchip.com/pcn and follow the registration instructions.

Customer Support

Users of Microchip products can receive assistance through several channels:

- Distributor or Representative
- Local Sales Office
- Embedded Solutions Engineer (ESE)
- Technical Support

Customers should contact their distributor, representative or ESE for support. Local sales offices are also available to help customers. A listing of sales offices and locations is included in this document.

Technical support is available through the website at: www.microchip.com/support

Microchip Devices Code Protection Feature

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

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

Legal Notice

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

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

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

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

Trademarks

The Microchip name and logo, the Microchip logo, Adaptec, AVR, AVR logo, AVR Freaks, BesTime, BitCloud, CryptoMemory, CryptoRF, dsPIC, flexPWR, HELDO, IGLOO, JukeBlox, KeeLoq, Kleer, LANCheck, LinkMD, maXStylus, maXTouch, MediaLB, megaAVR, Microsemi, Microsemi logo, MOST, MOST logo, MPLAB, OptoLyzer, PIC, picoPower, PICSTART, PIC32 logo, PolarFire, Prochip Designer, QTouch, SAM-BA, SenGenuity, SpyNIC, SST, SST Logo, SuperFlash, Symmetricom, SyncServer, Tachyon, TimeSource, tinyAVR, UNI/O, Vectron, and XMEGA are registered trademarks of Microchip Technology Incorporated in the U.S.A. and other countries.

AgileSwitch, APT, ClockWorks, The Embedded Control Solutions Company, EtherSynch, Flashtec, Hyper Speed Control, HyperLight Load, Libero, motorBench, mTouch, Powermite 3, Precision Edge, ProASIC, ProASIC Plus, ProASIC Plus logo, Quiet- Wire, SmartFusion, SyncWorld, Temux, TimeCesium, TimeHub, TimePictra, TimeProvider, TrueTime, and ZL are registered trademarks of Microchip Technology Incorporated in the U.S.A.

Adjacent Key Suppression, AKS, Analog-for-the-Digital Age, Any Capacitor, AnyIn, AnyOut, Augmented Switching, BlueSky, BodyCom, Clockstudio, CodeGuard, CryptoAuthentication, CryptoAutomotive, CryptoCompanion, CryptoController, dsPICDEM, dsPICDEM.net, Dynamic Average Matching, DAM, ECAN, Espresso T1S, EtherGREEN, GridTime, IdealBridge, In-Circuit Serial Programming, ICSP, INICnet, Intelligent Paralleling, IntelliMOS, Inter-Chip Connectivity, JitterBlocker, Knob-on-Display, KoD, maxCrypto, maxView, memBrain, Mindi, MiWi, MPASM, MPF, MPLAB Certified logo, MPLIB, MPLINK, MultiTRAK, NetDetach, Omniscient Code Generation, PICDEM, PICDEM.net, PICkit, PICtail, PowerSmart, PureSilicon, QMatrix, REAL ICE, Ripple Blocker, RTAX, RTG4, SAM-

ISBN: 978-1-6683-1220-9

Quality Management System

For information regarding Microchip's Quality Management Systems, please visit www.microchip.com/quality.

Worldwide Sales and Service

AMERICAS	ASIA/PACIFIC	ASIA/PACIFIC	EUROPE
Corporate Office 2355 West Chandler Blvd. Chandler, AZ 85224-6199 Tel: 480-792-7200 Fax: 480-792-7277 Technical Support: www.microchip.com/support Web Address: www.microchip.com **Atlanta** Duluth, GA Tel: 678-957-9614 Fax: 678-957-1455 **Austin, TX** Tel: 512-257-3370 **Boston** Westborough, MA Tel: 774-760-0087 Fax: 774-760-0088 **Chicago** Itasca, IL Tel: 630-285-0071 Fax: 630-285-0075 **Dallas** Addison, TX Tel: 972-818-7423 Fax: 972-818-2924 **Detroit** Novi, MI Tel: 248-848-4000 **Houston, TX** Tel: 281-894-5983 **Indianapolis** Noblesville, IN Tel: 317-773-8323 Fax: 317-773-5453 Tel: 317-536-2380 **Los Angeles** Mission Viejo, CA Tel: 949-462-9523 Fax: 949-462-9608 Tel: 951-273-7800 **Raleigh, NC** Tel: 919-844-7510 **New York, NY** Tel: 631-435-6000 **San Jose, CA** Tel: 408-735-9110 Tel: 408-436-4270 **Canada - Toronto** Tel: 905-695-1980 Fax: 905-695-2078	**Australia - Sydney** Tel: 61-2-9868-6733 **China - Beijing** Tel: 86-10-8569-7000 **China - Chengdu** Tel: 86-28-8665-5511 **China - Chongqing** Tel: 86-23-8980-9588 **China - Dongguan** Tel: 86-769-8702-9880 **China - Guangzhou** Tel: 86-20-8755-8029 **China - Hangzhou** Tel: 86-571-8792-8115 **China - Hong Kong SAR** Tel: 852-2943-5100 **China - Nanjing** Tel: 86-25-8473-2460 **China - Qingdao** Tel: 86-532-8502-7355 **China - Shanghai** Tel: 86-21-3326-8000 **China - Shenyang** Tel: 86-24-2334-2829 **China - Shenzhen** Tel: 86-755-8864-2200 **China - Suzhou** Tel: 86-186-6233-1526 **China - Wuhan** Tel: 86-27-5980-5300 **China - Xian** Tel: 86-29-8833-7252 **China - Xiamen** Tel: 86-592-2388138 **China - Zhuhai** Tel: 86-756-3210040	**India - Bangalore** Tel: 91-80-3090-4444 **India - New Delhi** Tel: 91-11-4160-8631 **India - Pune** Tel: 91-20-4121-0141 **Japan - Osaka** Tel: 81-6-6152-7160 **Japan - Tokyo** Tel: 81-3-6880- 3770 **Korea - Daegu** Tel: 82-53-744-4301 **Korea - Seoul** Tel: 82-2-554-7200 **Malaysia - Kuala Lumpur** Tel: 60-3-7651-7906 **Malaysia - Penang** Tel: 60-4-227-8870 **Philippines - Manila** Tel: 63-2-634-9065 **Singapore** Tel: 65-6334-8870 **Taiwan - Hsin Chu** Tel: 886-3-577-8366 **Taiwan - Kaohsiung** Tel: 886-7-213-7830 **Taiwan - Taipei** Tel: 886-2-2508-8600 **Thailand - Bangkok** Tel: 66-2-694-1351 **Vietnam - Ho Chi Minh** Tel: 84-28-5448-2100	**Austria - Wels** Tel: 43-7242-2244-39 Fax: 43-7242-2244-393 **Denmark - Copenhagen** Tel: 45-4485-5910 Fax: 45-4485-2829 **Finland - Espoo** Tel: 358-9-4520-820 **France - Paris** Tel: 33-1-69-53-63-20 Fax: 33-1-69-30-90-79 **Germany - Garching** Tel: 49-8931-9700 **Germany - Haan** Tel: 49-2129-3766400 **Germany - Heilbronn** Tel: 49-7131-72400 **Germany - Karlsruhe** Tel: 49-721-625370 **Germany - Munich** Tel: 49-89-627-144-0 Fax: 49-89-627-144-44 **Germany - Rosenheim** Tel: 49-8031-354-560 **Israel - Ra'anana** Tel: 972-9-744-7705 **Italy - Milan** Tel: 39-0331-742611 Fax: 39-0331-466781 **Italy - Padova** Tel: 39-049-7625286 **Netherlands - Drunen** Tel: 31-416-690399 Fax: 31-416-690340 **Norway - Trondheim** Tel: 47-72884388 **Poland - Warsaw** Tel: 48-22-3325737 **Romania - Bucharest** Tel: 40-21-407-87-50 **Spain - Madrid** Tel: 34-91-708-08-90 Fax: 34-91-708-08-91 **Sweden - Gothenberg** Tel: 46-31-704-60-40 **Sweden - Stockholm** Tel: 46-8-5090-4654 **UK - Wokingham** Tel: 44-118-921-5800 Fax: 44-118-921-5820