PolarFire SoC MSS Configurator User Guide

Libero SoC v2025.1

Introduction (Ask a Question)

The PolarFire® SoC Microcontroller Subsystem (MSS) Configurator provides a graphical user interface that allows embedded software engineers to define the MSS start-up state quickly. It exports an XML file that is used by the embedded software flow that converts the XML into initialization constructs. Additionally, the tool outputs a CXZ file for inclusion into your Libero® design flow. The CXZ file contains information about metadata and port needed by the FPGA designer to complete the MSS and FPGA fabric connectivity.

MSS configurator is available as a stand-alone application and as part of the Libero SoC design tool suite. The information in this user guide applies to both.

Important: Older versions of this documentation uses the terms "Master" and "Slave." The equivalent Microchip terminology used in this document is "Initiator" and "Target" respectively.

Table of Contents

1. Installing the PolarFire SoC MSS Configurator (Ask a Question)

The PolarFire SoC MSS Configurator bundled with Libero is available at the following location in the Libero installation section:

- `Windows: <$Installation_Directory>\Microsemi\Libero_SoC_vX.X\Designer\bin64\pfsoc_mss.exe`
- `Linux: <$Installation_Directory>\Microsemi\Libero_SoC_vX.X\bin64\pfsoc_mss`

The PolarFire SoC MSS Configurator can also be installed as a stand-alone application.

For more information about how to install Libero, see Libero SoC v12.0 and later.

1.1. Input and Output Files (Ask a Question)

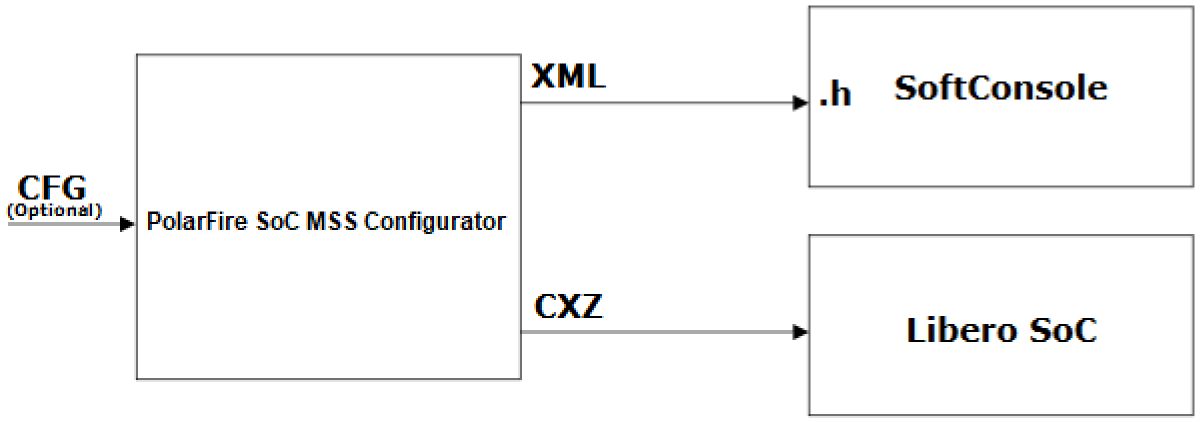
The following sections describe the PolarFire SoC MSS Configurator input and output files.

1.1.1. Output Files (Ask a Question)

The PolarFire SoC MSS Configurator generates the output file formats as shown in the following figure.

- **XML Configuration File** — Contains the MSS memory map, clock, DDR memory controller, and peripheral configuration. The XML file is used to generate hardware files required for building the firmware project.
- **CXZ File** — Encapsulates the hardware design of the MSS block and can be imported into Libero SoC project.

Figure 1-1. PolarFire SoC MSS Configurator Block Diagram



1.1.2. Input files (Ask a Question)

The PolarFire SoC MSS Configurator can be invoked without any input files. A configuration file `(.cfg)` from an earlier MSS configurator session can be optionally provided to the PolarFire SoC MSS Configurator.

Note: A `.cfg` file and a report file can also be generated from the PolarFire SoC MSS Configurator.

2. Running the PolarFire SoC MSS Configurator (Ask a Question)

You can run the PolarFire SoC MSS Configurator in Batch mode or Interactive mode. This chapter describes the following sections:

- Batch Mode
- Interactive Mode
- Using the PolarFire SoC MSS Configurator GUI

2.1. Batch Mode (Ask a Question)

The PolarFire SoC MSS Configurator application can be executed in the Batch mode for scripted execution as follows:

- Windows®:

```
<Libero SoC or Standalone MSS Configurator installation area>\bin64\pfsoc_mss.exe
-GENERATE -CONFIGURATION_FILE:<absolute or relative path for configuration file name
(.cfg)> -OUTPUT_DIR:<absolute or relative path for output directory> -EXPORT_HDL:<true/
false> -LOGFILE:<absolute or relative path for logfile file name>
```

  -EXPORT_HDL and -LOGFILE are optional arguments.

- Linux®:

```
<Libero SoC or Standalone MSS Configurator installation area>/bin64/pfsoc_mss.exe
-GENERATE -CONFIGURATION_FILE:<absolute or relative path for configuration file name
(.cfg)> -OUTPUT_DIR:<absolute or relative path for output directory> -EXPORT_HDL:<true/
false> -LOGFILE:<absolute or relative path for logfile file name>
```

  -EXPORT_HDL and -LOGFILE are optional arguments.

2.2. Interactive Mode (Ask a Question)

The Interactive (GUI) mode in the PolarFire SoC MSS Configurator provides the following high-level options.

Table 2-1. Configurator-Project Menu Options

Option	Description
New	Starts configuring a new MSS subsystem.
Open	Opens a configuration (`.cfg`) file.
Save/Save As	Saves the current configuration of the MSS subsystem to a configuration (`.cfg`) file.
Generate	Generates MSS configuration (`.xml`) and component (`.cxz`) files after configuring the MSS subsystem.
Close	Closes the current configuration (`.cfg`) file.

PolarFire SoC MSS Configurator can also be invoked in GUI mode for a specific configuration file as below:

- Windows:

```
<Libero SoC or Standalone MSS Configurator installation area>\bin64\pfsoc_mss.exe
-CONFIGURATION_FILE:<absolute or relative path for configuration file name (.cfg)>
-OUTPUT_DIR:<absolute or relative path for output directory>
```

- Linux:

```
<Libero SoC or Standalone MSS Configurator installation area>/bin64/pfsoc_mss.exe
-CONFIGURATION_FILE:<absolute or relative path for configuration file name (.cfg)>
-OUTPUT_DIR:<absolute or relative path for output directory>
```

Important: `-OUTPUT_DIR` is an optional argument in GUI mode. If specified, the default output directory location in GUI will always point to the specified location.

2.3. Using the PolarFire SoC MSS Configurator GUI (Ask a Question)

The PolarFire SoC MSS Configurator GUI has the following tabs.

- Peripherals
- DDR Memory
- L2 Cache
- Crypto
- MSS to/from Fabric Interface Controllers
- Clocks
- MSS I/O Attributes
- Memory Partition and Protection
- Misc

2.3.1. Clocks (Ask a Question)

Use the **Clocks** tab to configure the MSS PLL clock frequency and clock sources. For more information, see the PolarFire Family Clocking Resources User Guide.

There are three PLLs inside MSS, which generate the necessary clocks:

- MSS PLL
- DDR PLL
- SGMII PLL

Each PLL generates four output clocks from one input reference clock.

The actual output that PLL Solver generates is shown in the GUI next to the **Actual:** label. If the requirement is met, the color of the label is blue. If it is not met (from you), it is red.

In MSS PLL, you must specify the four output clock requirements.

- Output 0 (CPU Clock)
- Output 1 (Crypto Clock)
- Output 2 (eMMC Clock)
- Output 3 (CAN Clock)

The following conditions pose restrictions on the PLL solver. Solver attempts to solve the fixed requirements first and then the ones without restriction.

1. When eMMC is enabled, the output clock requirement is fixed at 200 MHz.
2. When the CAN peripheral is enabled, the output clock requirement must be a multiple of 8 (maximum 80 MHz).
3. Crypto can be at most 200 MHz for STD speed grade and 213 MHz for -1 speed grade.
4. CPU has the maximum frequency of 625 MHz for STD speed grade and 667 MHz for -1 speed grade.
   Note: When the actual frequency found by the PLL solver exceeds the maximum required clock frequency for MSS PLL and Crypto, MSS generation fails with the following error message.

```
ERROR: Invalid solution found for Output0 of MSS PLL. 800 MHz is greater than the
supported value of 667 MHz.
```

In DDR PLL, enter the required output clock frequency in the DDR tab in Line Edit "Memory clock frequency." All four output clocks are generated by the PLL that has the same frequency.

In SGMII PLL, the output frequency requirement is fixed at 625 MHz for all four outputs. You do not have to enter this. Additionally, the output clocks are phase-shifted by 90° (output clock 0 has a 0° phase shift, output clock 1 has a 90° phase shift, and so on).

The following table lists the option provided in the MSS **Clocks** selection tab.

Table 2-2. MSS Clock Selection Tab

Option	Description
eMMC/SD/SDIO clock source	eMMC/SD/SDIO can be clocked either through MSS PLL or Fabric I/O.
CAN clock source	CAN can be clocked either through MSS PLL or Fabric I/O.
MSS PLL reference clock source	MSS can be clocked from dedicated I/O from Bank 5 (REFCLK) or North West PLL output.
MSS PLL required clock frequency	• For STD speed, you can set the frequency value of up to 625 MHz. • For -1 speed, you can set the frequency value of up to 667 MHz. All MSS clock frequencies are derived from this setting.
MSS CPU cores clock frequency divider	The MSS CPU clock frequency is based on the MSS PLL clock frequency and is set using the divider values /1, /2, /4, or /8. The frequency must be greater than or equal to the MSS AXI clock, and can have a maximum value of 667 MHz for -1 speed, and 625 MHz for STD speed.
MSS AXI clock frequency divider	The MSS AXI clock frequency is based on the MSS CPU clock frequency and is set using the divider values of /1, /2, /4, or /8. The frequency must be greater than or equal to MSS AHB/APB clock, and can have a maximum value of 333.50 MHz for -1 speed, and 312.50 MHz for STD speed.
MSS AHB/APB clock frequency divider	The MSS AHB/APB clock frequency is based on the MSS CPU clock frequency and is set using the divider values /2, /4, or /8. The maximum supported frequency is 166.75 MHz for -1 speed, and 156.25 MHz for STD speed.
DDR reference clock source	You can select the North West (NW) PLL ports or I/Os from Bank 5.
RTC/MAC SGMII reference clock input source	You can select the NW PLL ports or I/Os from Bank 5.
Dedicated I/O from Bank5 (REFCLK) frequency	This is a dedicated I/O from Bank 5 to the MSS PLL. It can either be 100 MHz or 125 MHz.
NW PLL (REF_0_PLL_NW) frequency (MHz)/NW PLL (REF_1_PLL_NW) frequency (MHz)	This option is available when any peripheral is clocked from NW PLL output. It can be any value between 50 MHz and 125 MHz.
Crypto clock frequency from MSS (MHz)	You can set the reference clock frequency for Crypto between 1 MHz and 200 MHz for STD speed grade, and between 1 MHz and 213 MHz for -1 speed grade.
MSS CAN clock frequency (MHz)	The MSS CPU clock frequency is based on the MSS PLL clock frequency. The supported frequencies in MHz are 8, 16, 24, 32, 40, 48, 56, 64, 72, and 80.

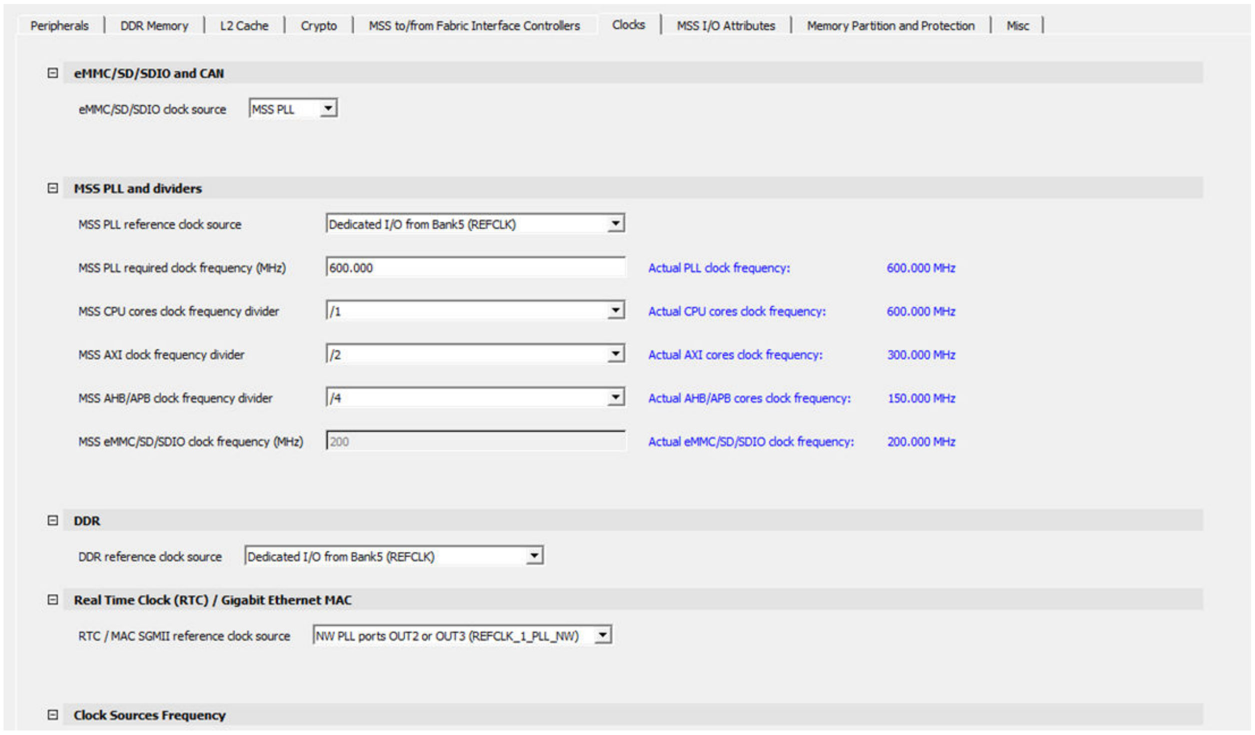
Note: The **DDR Reference Clock Input Source** option appears only when the DDR memory type is selected from the **DDR Memory** tab.

The following figure shows the **Clocks** tab in the PolarFire SoC MSS Configurator. In this example, the following configurations are used:

- Dedicated I/Os from Bank 5 (REFCLK) are selected as the reference clock input source for the MSS. The MSS PLL clock frequency is set to 600 MHz.

- Dedicated I/Os from Bank 5 (REFCLK) are used to source the reference clock input frequency for the DDR subsystem.
- The DDR clock source and MSS clock source are set to 125 MHz.

Figure 2-1. Clocks Tab


Peripherals
DDR Memory
L2 Cache
Crypto
MSS to/from Fabric Interface Controllers
Clocks
MSS I/O Attributes
Memory Partition and Protection
Misc

eMMC/SD/SDIO and CAN

eMMC/SD/SDIO clock source: MSS PLL

MSS PLL and dividers

MSS PLL reference clock source	Dedicated I/O from Bank5 (REFCLK)		
MSS PLL required clock frequency (MHz)	600.000	Actual PLL clock frequency:	600.000 MHz
MSS CPU cores clock frequency divider	/1	Actual CPU cores clock frequency:	600.000 MHz
MSS AXI clock frequency divider	/2	Actual AXI cores clock frequency:	300.000 MHz
MSS AHB/APB clock frequency divider	/4	Actual AHB/APB cores clock frequency:	150.000 MHz
MSS eMMC/SD/SDIO clock frequency (MHz)	200	Actual eMMC/SD/SDIO clock frequency:	200.000 MHz

DDR

DDR reference clock source: Dedicated I/O from Bank5 (REFCLK)

Real Time Clock (RTC) / Gigabit Ethernet MAC

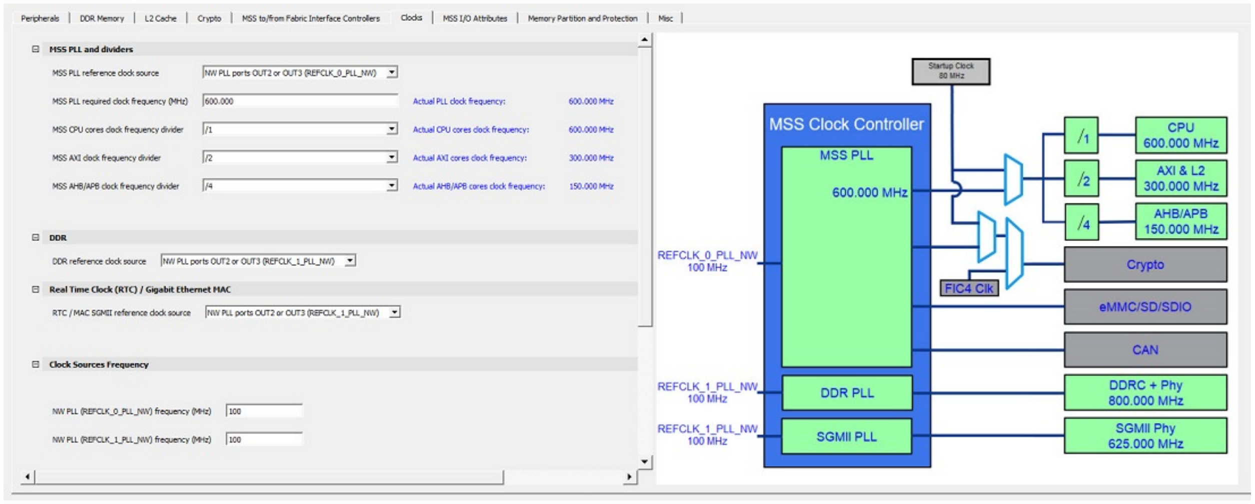
RTC / MAC SGMII reference clock source: NW PLL ports OUT2 or OUT3 (REFCLK_1_PLL_NW)

Clock Sources Frequency

The following figure shows the **Clocks** tab with PLL in the PolarFire SoC MSS Configurator. In this example, the following configuration is used:

- NW PLL ports OUT2 or OUT3 (REFCLK_0_PLL_NW) are selected as the reference clock input source for the MSS. The MSS PLL clock frequency is set to 600.000 MHz.
- NW PLL ports OUT2 or OUT3 (REFCLK_1_PLL_NW) are used to source the reference clock input frequency for the DDR subsystem and Real Time Clock/Gigabit Ethernet MAC.
- The DDR Clock Source and MSS Clock Source are set to 100 MHz.

Figure 2-2. Clocks Tab with PLL



MSS PLL and dividers

MSS PLL reference clock source: NW PLL ports OUT2 or OUT3 (REFCLK_0_PLL_NW)

MSS PLL required clock frequency (MHz): 600.000 — Actual PLL clock frequency: 600.000 MHz

MSS CPU cores clock frequency divider: /1 — Actual CPU cores clock frequency: 600.000 MHz

MSS AXI clock frequency divider: /2 — Actual AXI cores clock frequency: 300.000 MHz

MSS AHB/APB clock frequency divider: /4 — Actual AHB/APB cores clock frequency: 150.000 MHz

DDR

DDR reference clock source: NW PLL ports OUT2 or OUT3 (REFCLK_1_PLL_NW)

Real Time Clock (RTC) / Gigabit Ethernet MAC

RTC / MAC SGMII reference clock source: NW PLL ports OUT2 or OUT3 (REFCLK_1_PLL_NW)

Clock Sources Frequency

NW PLL (REFCLK_0_PLL_NW) frequency (MHz): 100

NW PLL (REFCLK_1_PLL_NW) frequency (MHz): 100

For more information about configuring the MSS DDR subsystem, see DDR Memory.

2.3.2. MSS To/From Fabric Interface Controllers (Ask a Question)

Using the **MSS to/from Fabric Interface Controllers** tab, any combination of **FIC_0**, **FIC_1**, **FIC_2**, and **FIC_3** can be enabled and configured to support initiator and target interfaces. For more information, see the PolarFire SoC FPGA MSS Technical Reference Manual.

FIC_0, **FIC_1**, and **FIC_2** support AXI4 interfaces, while **FIC_3** supports APB.

For **FIC_0** and **FIC_1** interfaces, both initiator and target interfaces can be enabled at the same time. **FIC_2** interface can support only target interface, and **FIC_3** interface can only support initiator interface (MSS is initiator).

FIC_0 and **FIC_1** have both initiator and target interfaces to and from the FPGA fabric, while **FIC_2** and **FIC_3** support target or initiator interfaces, respectively.

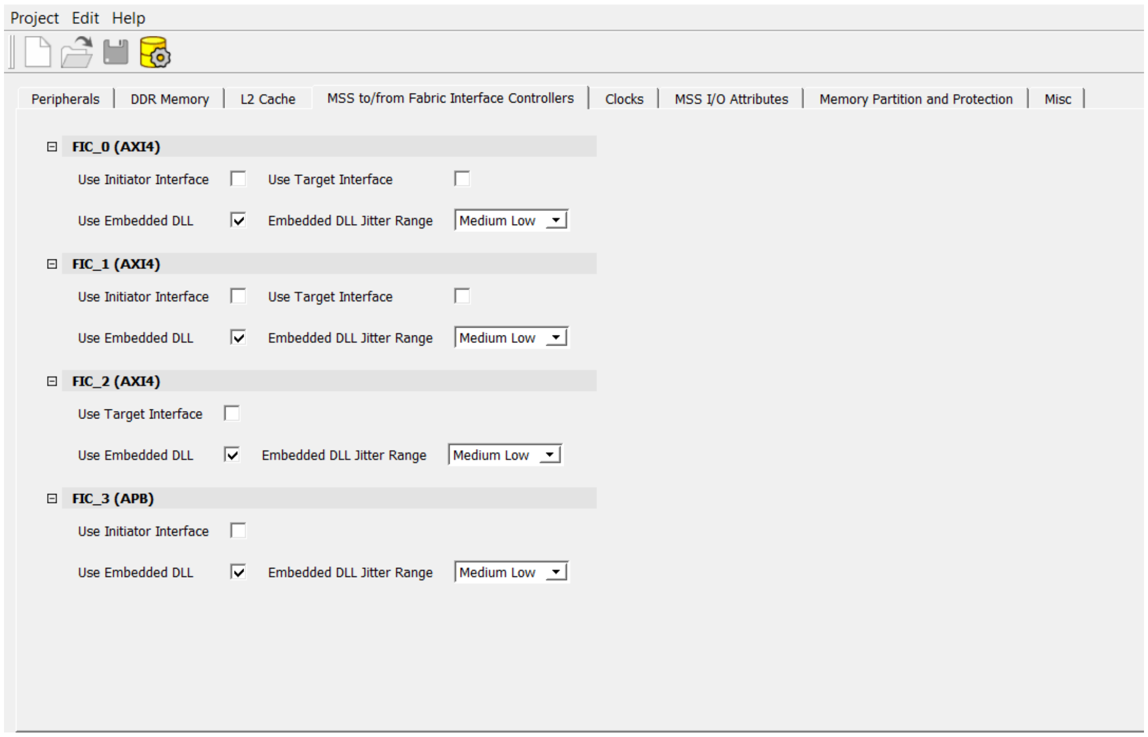
The **Jitter Range** for the Embedded DLLs can be selected in the **Fabric Interface Controller** tab. The **Embedded DLL Jitter Range** drop-down has the following options:

- Low
- Medium Low
- Medium High
- High

Important: The default selection is Medium Low.

The following figure shows all FIC options available and enabled. By default, the DLLs of all the FICs are enabled.

Figure 2-3. MSS to/from Fabric Interface Controllers Tab



FIC_0 (AXI4)
Use Initiator Interface ☐ Use Target Interface ☐
Use Embedded DLL ✓ Embedded DLL Jitter Range Medium Low

FIC_1 (AXI4)
Use Initiator Interface ☐ Use Target Interface ☐
Use Embedded DLL ✓ Embedded DLL Jitter Range Medium Low

FIC_2 (AXI4)
Use Target Interface ☐
Use Embedded DLL ✓ Embedded DLL Jitter Range Medium Low

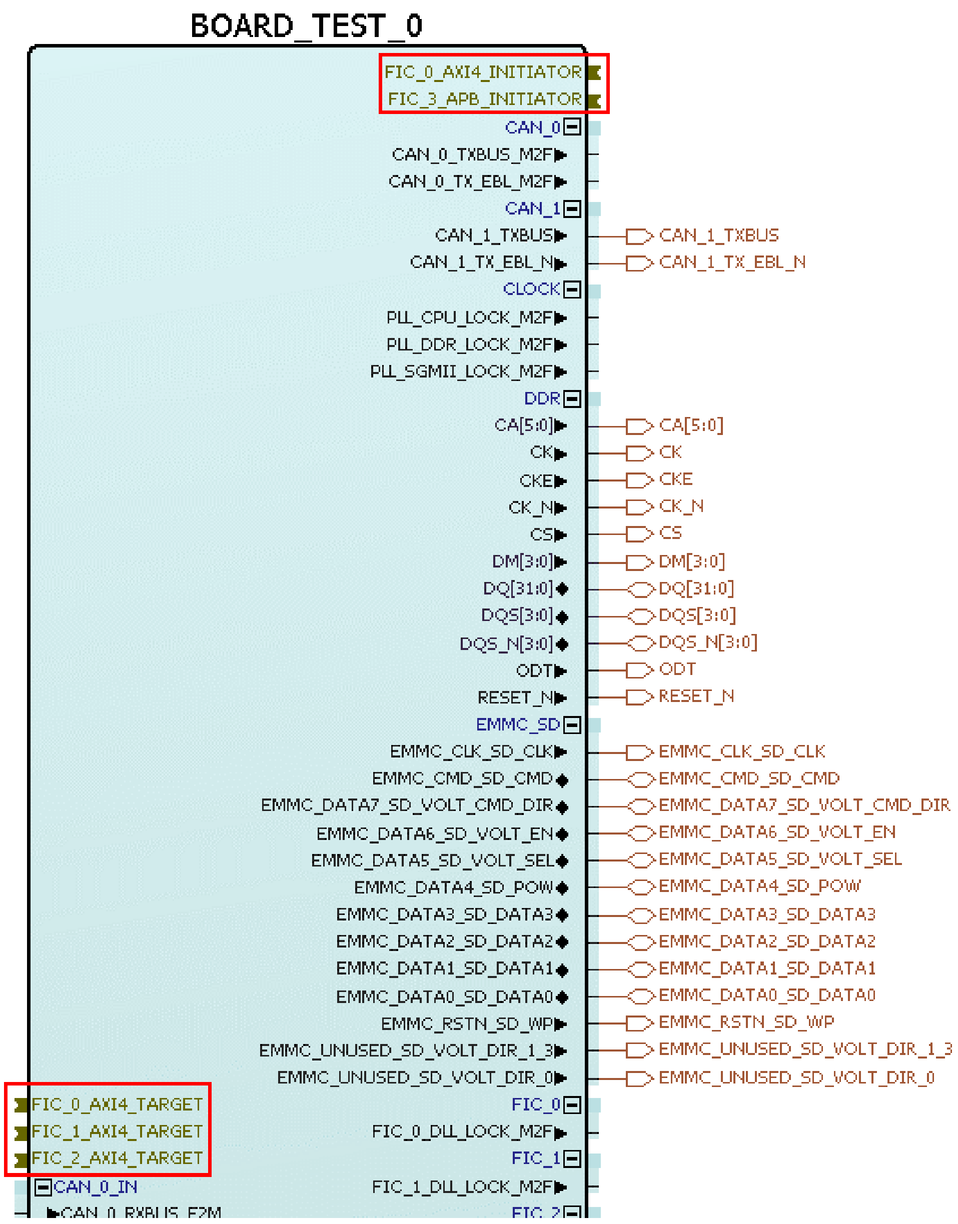
FIC_3 (APB)
Use Initiator Interface ☐
Use Embedded DLL ✓ Embedded DLL Jitter Range Medium Low

Important: The FIC interface can operate up to 250 MHz. The FIC clock is independent of the MSS clock. If the frequency of the FIC block is greater than or equal to 125 MHz, the embedded DLL must be enabled to remove the clock insertion delay. If the frequency of the FIC block is less than 125 MHz, the embedded DLL must be bypassed.

When an initiator interface is enabled for a FIC, that initiator interface must be connected to a target in the fabric. When a target interface is enabled for an FIC, that target interface must be connected to an initiator in the fabric.

There is a clock domain crossing logic in the FIC block to address the asynchronous MSS and Fabric clocks and therefore, user logic is not required to implement clock domain crossing synchronization for this interface.

Figure 2-4. FIC Interfaces Enabled



Note: The MSS SmartDesign component is visible only after importing the MSS CXZ file.

2.3.3. Peripherals (Ask a Question)

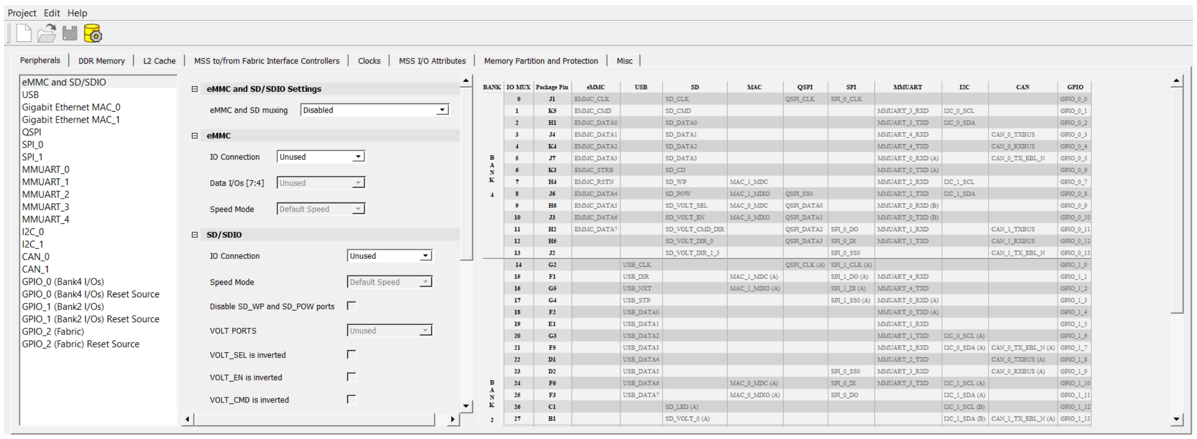
Select the following I/Os using the **Peripherals** tab:

- GPIOs from Bank 2 and Bank 4, which are dedicated to the MSS.
- Fabric I/Os, if the dedicated I/Os from Bank 2 and Bank 4 are not available.
- GPIOs from Bank 5 are dedicated to SGMII but can be routed to GMII or MII fabric I/Os. GPIO from Bank 5 are displayed only when Gigabit Ethernet MAC_0 or Gigabit Ethernet MAC_1 is selected.

Important: I/Os from the DDR bank are dedicated to the DDR Controller in the MSS.

For more information, see the PolarFire SoC FPGA MSS Technical Reference Manual. The following figure shows the **Peripherals** tab on the PolarFire SoC MSS Configurator.

Figure 2-5. Peripherals Tab


Project Edit Help
Peripherals | DDR Memory | L2 Cache | MSS to/from Fabric Interface Controllers | Clocks | MSS I/O Attributes | Memory Partition and Protection | Misc

eMMC and SD/SDIO
USB
Gigabit Ethernet MAC_0
Gigabit Ethernet MAC_1
QSPI
SPI_0
SPI_1
MMUART_0
MMUART_1
MMUART_2
MMUART_3
MMUART_4
I2C_0
I2C_1
CAN_0
CAN_1
GPIO_0 (Bank4 I/Os)
GPIO_0 (Bank4 I/Os) Reset Source
GPIO_1 (Bank2 I/Os)
GPIO_1 (Bank2 I/Os) Reset Source
GPIO_2 (Fabric)
GPIO_2 (Fabric) Reset Source

eMMC and SD/SDIO Settings
- eMMC and SD muxing: Disabled

eMMC
- IO Connection: Unused
- Data I/Os [7:4]: Unused
- Speed Mode: Default Speed

SD/SDIO
- IO Connection: Unused
- Speed Mode: Default Speed
- Disable SD_WP and SD_POW ports: ☐
- VOLT PORTS: Unused
- VOLT_SEL is inverted: ☐
- VOLT_EN is inverted: ☐
- VOLT_CMD is inverted: ☐

BANK	IO MUX	Package Pin	eMMC	USB	SD	MAC	QSPI	SPI	MMUART	I2C	CAN	GPIO
BANK 4	0	J1	EMMC_CLK		SD_CLK		QSPI_CLK	SPI_0_CLK				GPIO_0_0
	1	K5	EMMC_CMD		SD_CMD				MMUART_3_RXD	I2C_0_SCL		GPIO_0_1
	2	H1	EMMC_DATA0		SD_DATA0				MMUART_3_TXD	I2C_0_SDA		GPIO_0_2
	3	J4	EMMC_DATA1		SD_DATA1				MMUART_4_RXD		CAN_0_TXBUS	GPIO_0_3
	4	K4	EMMC_DATA2		SD_DATA2				MMUART_4_TXD		CAN_0_RXBUS	GPIO_0_4
	5	J7	EMMC_DATA3		SD_DATA3				MMUART_0_RXD (A)		CAN_0_TX_EBL_N	GPIO_0_5
	6	K3	EMMC_STRB		SD_CD				MMUART_0_TXD (A)			GPIO_0_6
	7	H4	EMMC_RSTN		SD_WP	MAC_1_MDC			MMUART_2_RXD	I2C_1_SCL		GPIO_0_7
	8	J6	EMMC_DATA4		SD_POW	MAC_1_MDIO	QSPI_SS0		MMUART_2_TXD	I2C_1_SDA		GPIO_0_8
	9	H6	EMMC_DATA5		SD_VOLT_SEL	MAC_0_MDC	QSPI_DATA0		MMUART_0_RXD (B)			GPIO_0_9
	10	J3	EMMC_DATA6		SD_VOLT_EN	MAC_0_MDIO	QSPI_DATA1		MMUART_0_TXD (B)			GPIO_0_10
	11	H2	EMMC_DATA7		SD_VOLT_CMD_DIR		QSPI_DATA2	SPI_0_DO	MMUART_1_RXD		CAN_1_TXBUS	GPIO_0_11
	12	H5			SD_VOLT_DIR_0		QSPI_DATA3	SPI_0_DI	MMUART_1_TXD		CAN_1_RXBUS	GPIO_0_12
	13	J2			SD_VOLT_DIR_1_3			SPI_0_SS0			CAN_1_TX_EBL_N	GPIO_0_13
BANK 2	14	G2		USB_CLK			QSPI_CLK (A)	SPI_1_CLK (A)				GPIO_1_0
	15	F1		USB_DIR		MAC_1_MDC (A)		SPI_1_DO (A)	MMUART_4_RXD			GPIO_1_1
	16	G5		USB_NXT		MAC_1_MDIO (A)		SPI_1_DI (A)	MMUART_4_TXD			GPIO_1_2
	17	G4		USB_STP				SPI_1_SS0 (A)	MMUART_0_RXD (A)			GPIO_1_3
	18	F2		USB_DATA0					MMUART_0_TXD (A)			GPIO_1_4
	19	E1		USB_DATA1					MMUART_1_RXD			GPIO_1_5
	20	G3		USB_DATA2					MMUART_1_TXD	I2C_0_SCL (A)		GPIO_1_6
	21	F5		USB_DATA3					MMUART_2_RXD	I2C_0_SDA (A)	CAN_0_TX_EBL_N (A)	GPIO_1_7
	22	D1		USB_DATA4					MMUART_2_TXD		CAN_0_TXBUS (A)	GPIO_1_8
	23	D2		USB_DATA5				SPI_0_SS0	MMUART_3_RXD		CAN_0_RXBUS (A)	GPIO_1_9
	24	F6		USB_DATA6		MAC_0_MDC (A)		SPI_0_DI	MMUART_3_TXD	I2C_1_SCL (A)		GPIO_1_10
	25	F3		USB_DATA7		MAC_0_MDIO (A)		SPI_0_DO		I2C_1_SDA (A)		GPIO_1_11
	26	C1			SD_LED (A)					I2C_1_SCL (B)		GPIO_1_12
	27	B1			SD_VOLT_0 (A)					I2C_1_SDA (B)	CAN_1_TX_EBL_N (A)	GPIO_1_13

By default, all peripherals are marked as **Unused**. To include peripherals that are required in the design, select the peripheral from the left-hand side of the window and use the corresponding drop-down to assign MSS I/Os or fabric I/Os.

The I/Os associated with the following peripherals are dedicated and cannot be assigned to fabric I/Os:

- USB peripherals are dedicated in Bank 2.
- eMMC peripherals are dedicated in Bank 4.
- Ports SD_POW and SD_WP can be disabled when not in use and can be used for other interfaces.
- SD/SDIO peripherals are dedicated in Bank 4.

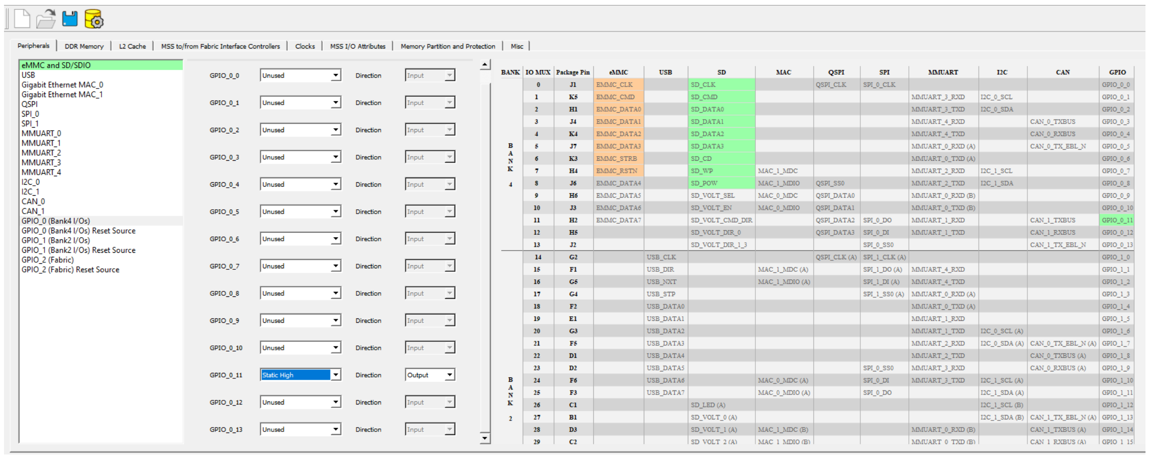
The GPIOs in Bank 2 and Bank 4 have the following options:

- Unused
- MSS I/Os Bank2/4
- Static High
- Static Low

Important:
- **Static High** and **Static Low** can only be set when **eMMC and SD muxing** is enabled.
- **GPIO_2 (Fabric)** does not support **Static High** and **Static Low** options.

According to the options selected, the affected GPIOs are highlighted in green, as shown in the following figure.

Figure 2-6. GPIO Options


Peripherals
DDR Memory
L2 Cache
MSS to/from Fabric Interface Controllers
Clocks
MSS I/O Attributes
Memory Partition and Protection
Misc

eMMC and SD/SDIO
USB
Gigabit Ethernet MAC_0
Gigabit Ethernet MAC_1
QSPI
SPI_0
SPI_1
MMUART_0
MMUART_1
MMUART_2
MMUART_3
MMUART_4
I2C_0
I2C_1
CAN_0
CAN_1
GPIO_0 (Bank4 I/Os)
GPIO_0 (Bank4 I/Os) Reset Source
GPIO_1 (Bank2 I/Os)
GPIO_1 (Bank2 I/Os) Reset Source
GPIO_2 (Fabric)
GPIO_2 (Fabric) Reset Source

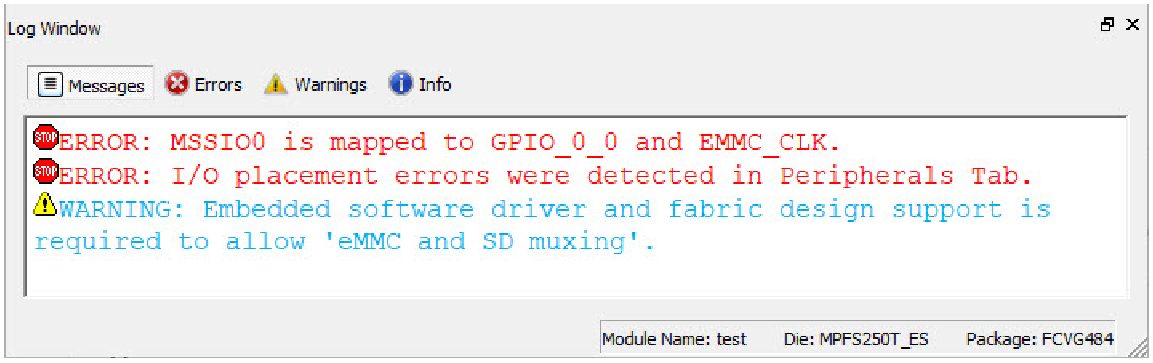
GPIO_0_0	Unused	Direction	Input
GPIO_0_1	Unused	Direction	Input
GPIO_0_2	Unused	Direction	Input
GPIO_0_3	Unused	Direction	Input
GPIO_0_4	Unused	Direction	Input
GPIO_0_5	Unused	Direction	Input
GPIO_0_6	Unused	Direction	Input
GPIO_0_7	Unused	Direction	Input
GPIO_0_8	Unused	Direction	Input
GPIO_0_9	Unused	Direction	Input
GPIO_0_10	Unused	Direction	Input
GPIO_0_11	Static High	Direction	Output
GPIO_0_12	Unused	Direction	Input
GPIO_0_13	Unused	Direction	Input

BANK	IO MUX	Package Pin	eMMC	USB	SD	MAC	QSPI	SPI	MMUART	I2C	CAN	GPIO
BANK 4	0	J1	EMMC_CLK		SD_CLK		QSPI_CLK	SPI_0_CLK				GPIO_0_0
	1	K5	EMMC_CMD		SD_CMD				MMUART_3_RXD	I2C_0_SCL		GPIO_0_1
	2	H1	EMMC_DATA0		SD_DATA0				MMUART_3_TXD	I2C_0_SDA		GPIO_0_2
	3	J4	EMMC_DATA1		SD_DATA1				MMUART_4_RXD		CAN_0_TXBUS	GPIO_0_3
	4	K4	EMMC_DATA2		SD_DATA2				MMUART_4_TXD		CAN_0_RXBUS	GPIO_0_4
	5	J7	EMMC_DATA3		SD_DATA3				MMUART_0_RXD (A)		CAN_0_TX_EBL_N	GPIO_0_5
	6	K3	EMMC_STRB		SD_CD				MMUART_0_TXD (A)			GPIO_0_6
	7	H4	EMMC_RSTN		SD_WP	MAC_1_MDC			MMUART_2_RXD	I2C_1_SCL		GPIO_0_7
	8	J6	EMMC_DATA4		SD_POW	MAC_1_MDIO	QSPI_SS0		MMUART_2_TXD	I2C_1_SDA		GPIO_0_8
	9	H6	EMMC_DATA5		SD_VOLT_SEL	MAC_0_MDC	QSPI_DATA0		MMUART_0_RXD (B)			GPIO_0_9
	10	J3	EMMC_DATA6		SD_VOLT_EN	MAC_0_MDIO	QSPI_DATA1		MMUART_0_TXD (B)			GPIO_0_10
	11	H2	EMMC_DATA7		SD_VOLT_CMD_DIR		QSPI_DATA2	SPI_0_DO	MMUART_1_RXD		CAN_1_TXBUS	GPIO_0_11
	12	H5			SD_VOLT_DIR_0		QSPI_DATA3	SPI_0_DI	MMUART_1_TXD		CAN_1_RXBUS	GPIO_0_12
	13	J2			SD_VOLT_DIR_1_3			SPI_0_SS0			CAN_1_TX_EBL_N	GPIO_0_13
BANK 2	14	G2		USB_CLK			QSPI_CLK (A)	SPI_1_CLK (A)				GPIO_1_0
	15	F1		USB_DIR		MAC_1_MDC (A)		SPI_1_DO (A)	MMUART_4_RXD			GPIO_1_1
	16	G5		USB_NXT		MAC_1_MDIO (A)		SPI_1_DI (A)	MMUART_4_TXD			GPIO_1_2
	17	G4		USB_STP				SPI_1_SS0 (A)	MMUART_0_RXD (A)			GPIO_1_3
	18	F2		USB_DATA0					MMUART_0_TXD (A)			GPIO_1_4
	19	E1		USB_DATA1					MMUART_1_RXD			GPIO_1_5
	20	G3		USB_DATA2					MMUART_1_TXD	I2C_0_SCL (A)		GPIO_1_6
	21	F5		USB_DATA3					MMUART_2_RXD	I2C_0_SDA (A)	CAN_0_TX_EBL_N (A)	GPIO_1_7
	22	D1		USB_DATA4					MMUART_2_TXD		CAN_0_TXBUS (A)	GPIO_1_8
	23	D2		USB_DATA5				SPI_0_SS0	MMUART_3_RXD		CAN_0_RXBUS (A)	GPIO_1_9
	24	F6		USB_DATA6		MAC_0_MDC (A)		SPI_0_DI	MMUART_3_TXD	I2C_1_SCL (A)		GPIO_1_10
	25	F3		USB_DATA7		MAC_0_MDIO (A)		SPI_0_DO		I2C_1_SDA (A)		GPIO_1_11
	26	C1			SD_LED (A)					I2C_1_SCL (B)		GPIO_1_12
	27	B1			SD_VOLT_0 (A)					I2C_1_SDA (B)	CAN_1_TX_EBL_N (A)	GPIO_1_13
	28	D3			SD_VOLT_1 (A)	MAC_1_MDC (B)			MMUART_0_RXD (B)		CAN_1_TXBUS (A)	GPIO_1_14
	29	C2			SD_VOLT_2 (A)	MAC_1_MDIO (B)			MMUART_0_TXD (B)		CAN_1_RXBUS (A)	GPIO_1_15

Note: If the I/Os for a peripheral are selected in a bank, you cannot select the same I/Os for another peripheral from the same bank. If you try, the tool generates the following error message in the log window:

```
I/O placement errors were detected in Peripherals Tab.
```

Figure 2-7. I/O Placement Error Message in Log Window



```
ERROR: MSSIO0 is mapped to GPIO_0_0 and EMMC_CLK.
ERROR: I/O placement errors were detected in Peripherals Tab.
WARNING: Embedded software driver and fabric design support is
required to allow 'eMMC and SD muxing'.
```

Module Name: test   Die: MPFS250T_ES   Package: FCVG484

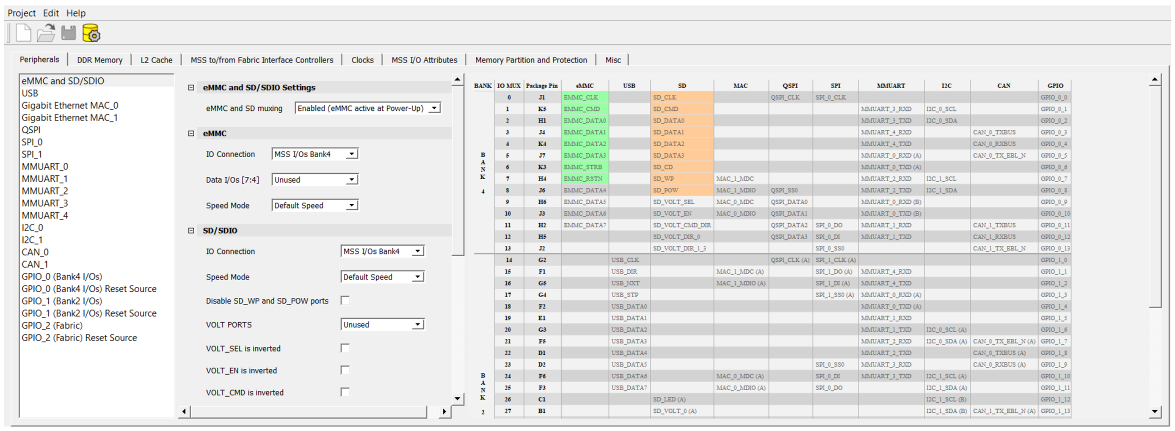
You can choose to enable either eMMC or SD at power-up using the eMMC and SD muxing option. The highlighted green-colored ports denote which I/O setting is active, whereas the orange-colored ports indicate which I/O setting is inactive.

The following warning message is generated in the log window, when the eMMC or SD setting is enabled.

```
Embedded software driver and fabric design support is required to allow 'eMMC and SD muxing'.
```

The eMMC and SD cannot be used simultaneously, as shown in the following figure.

Figure 2-8. Overlapping I/O Warning



Project Edit Help

Peripherals | DDR Memory | L2 Cache | MSS to/from Fabric Interface Controllers | Clocks | MSS I/O Attributes | Memory Partition and Protection | Misc

eMMC and SD/SDIO
USB
Gigabit Ethernet MAC_0
Gigabit Ethernet MAC_1
QSPI
SPI_0
SPI_1
MMUART_0
MMUART_1
MMUART_2
MMUART_3
MMUART_4
I2C_0
I2C_1
CAN_0
CAN_1
GPIO_0 (Bank4 I/Os)
GPIO_0 (Bank4 I/Os) Reset Source
GPIO_1 (Bank2 I/Os)
GPIO_1 (Bank2 I/Os) Reset Source
GPIO_2 (Fabric)
GPIO_2 (Fabric) Reset Source

eMMC and SD/SDIO Settings

eMMC and SD muxing	Enabled (eMMC active at Power-Up)

eMMC

IO Connection	MSS I/Os Bank4
Data I/Os [7:4]	Unused
Speed Mode	Default Speed

SD/SDIO

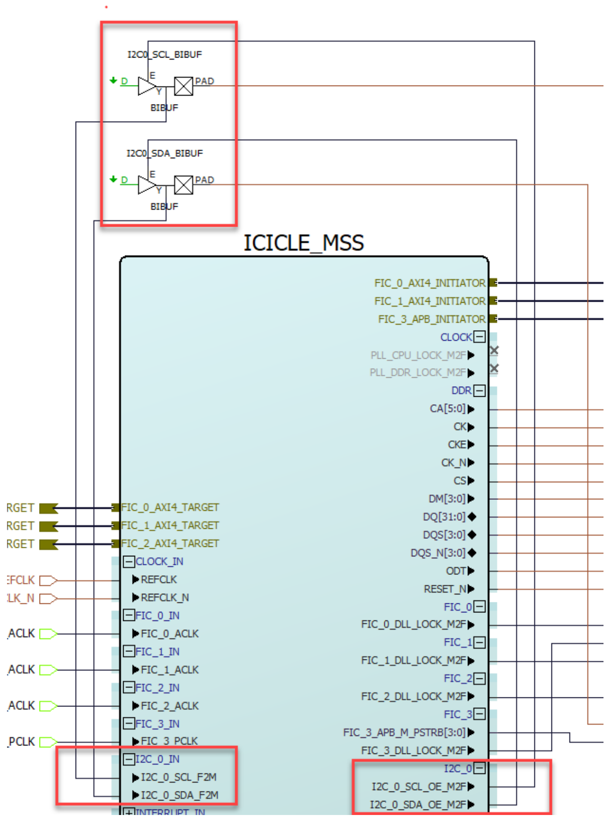
IO Connection	MSS I/Os Bank4
Speed Mode	Default Speed
Disable SD_WP and SD_POW ports	☐
VOLT PORTS	Unused
VOLT_SEL is inverted	☐
VOLT_EN is inverted	☐
VOLT_CMD is inverted	☐

BANK	IO MUX	Package Pin	eMMC	USB	SD	MAC	QSPI	SPI	MMUART	I2C	CAN	GPIO
BANK 4	0	J1	EMMC_CLK		SD_CLK		QSPI_CLK	SPI_0_CLK				GPIO_0_0
	1	K5	EMMC_CMD		SD_CMD				MMUART_3_RXD	I2C_0_SCL		GPIO_0_1
	2	H1	EMMC_DATA0		SD_DATA0				MMUART_3_TXD	I2C_0_SDA		GPIO_0_2
	3	J4	EMMC_DATA1		SD_DATA1				MMUART_4_RXD		CAN_0_TXBUS	GPIO_0_3
	4	K4	EMMC_DATA2		SD_DATA2				MMUART_4_TXD		CAN_0_RXBUS	GPIO_0_4
	5	J7	EMMC_DATA3		SD_DATA3				MMUART_0_RXD (A)		CAN_0_TX_EBL_N	GPIO_0_5
	6	K3	EMMC_STRB		SD_CD				MMUART_0_TXD (A)			GPIO_0_6
	7	H4	EMMC_RSTN		SD_WP	MAC_1_MDC			MMUART_2_RXD	I2C_1_SCL		GPIO_0_7
	8	J6	EMMC_DATA4		SD_POW	MAC_1_MDIO	QSPI_SS0		MMUART_2_TXD	I2C_1_SDA		GPIO_0_8
	9	H6	EMMC_DATA5		SD_VOLT_SEL	MAC_0_MDC	QSPI_DATA0		MMUART_0_RXD (B)			GPIO_0_9
	10	J3	EMMC_DATA6		SD_VOLT_EN	MAC_0_MDIO	QSPI_DATA1		MMUART_0_TXD (B)			GPIO_0_10
	11	H2	EMMC_DATA7		SD_VOLT_CMD_DIR		QSPI_DATA2	SPI_0_DO	MMUART_1_RXD		CAN_1_TXBUS	GPIO_0_11
	12	H5			SD_VOLT_DIR_0		QSPI_DATA3	SPI_0_DI	MMUART_1_TXD		CAN_1_RXBUS	GPIO_0_12
	13	J2			SD_VOLT_DIR_1_3			SPI_0_SS0			CAN_1_TX_EBL_N	GPIO_0_13
BANK 2	14	G2		USB_CLK			QSPI_CLK (A)	SPI_1_CLK (A)				GPIO_1_0
	15	F1		USB_DIR		MAC_1_MDC (A)		SPI_1_DO (A)	MMUART_4_RXD			GPIO_1_1
	16	G5		USB_NXT		MAC_1_MDIO (A)		SPI_1_DI (A)	MMUART_4_TXD			GPIO_1_2
	17	G4		USB_STP				SPI_1_SS0 (A)	MMUART_0_RXD (A)			GPIO_1_3
	18	F2		USB_DATA0					MMUART_0_TXD (A)			GPIO_1_4
	19	E1		USB_DATA1					MMUART_1_RXD			GPIO_1_5
	20	G3		USB_DATA2					MMUART_1_TXD	I2C_0_SCL (A)		GPIO_1_6
	21	F5		USB_DATA3					MMUART_2_RXD	I2C_0_SDA (A)	CAN_0_TX_EBL_N (A)	GPIO_1_7
	22	D1		USB_DATA4					MMUART_2_TXD		CAN_0_TXBUS (A)	GPIO_1_8
	23	D2		USB_DATA5				SPI_0_SS0	MMUART_3_RXD		CAN_0_RXBUS (A)	GPIO_1_9
	24	F6		USB_DATA6		MAC_0_MDC (A)		SPI_0_DI	MMUART_3_TXD	I2C_1_SCL (A)		GPIO_1_10
	25	F3		USB_DATA7		MAC_0_MDIO (A)		SPI_0_DO		I2C_1_SDA (A)		GPIO_1_11
	26	C1			SD_LED (A)					I2C_1_SCL (B)		GPIO_1_12
	27	B1			SD_VOLT_0 (A)					I2C_1_SDA (B)	CAN_1_TX_EBL_N (A)	GPIO_1_13

2.3.3.1. I2C Port Configuration for Fabric I/O (Ask a Question)

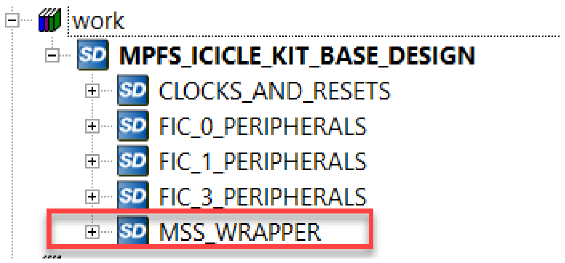
Fabric connections for the MSS I2C peripherals do not generate an output port, instead only an Output Enable (OE) port is available on the MSS component. You must instantiate an BIBUF with the D connected to 'GND' and the E and Y pins connected to OE and IN, respectively. The following figure shows these connections.

Figure 2-9. Fabric Connections



To see the BIBUF configuration required to use the I2C peripheral with fabric connections, see the PolarFire SoC Icicle Kit Reference Design from **MSS_WRAPPER subsystem** > **SmartDesign** .

Figure 2-10. MPFS Icicle Kit Base Design Hierarchy



```
work
  MPFS_ICICLE_KIT_BASE_DESIGN
    CLOCKS_AND_RESETS
    FIC_0_PERIPHERALS
    FIC_1_PERIPHERALS
    FIC_3_PERIPHERALS
    MSS_WRAPPER
```

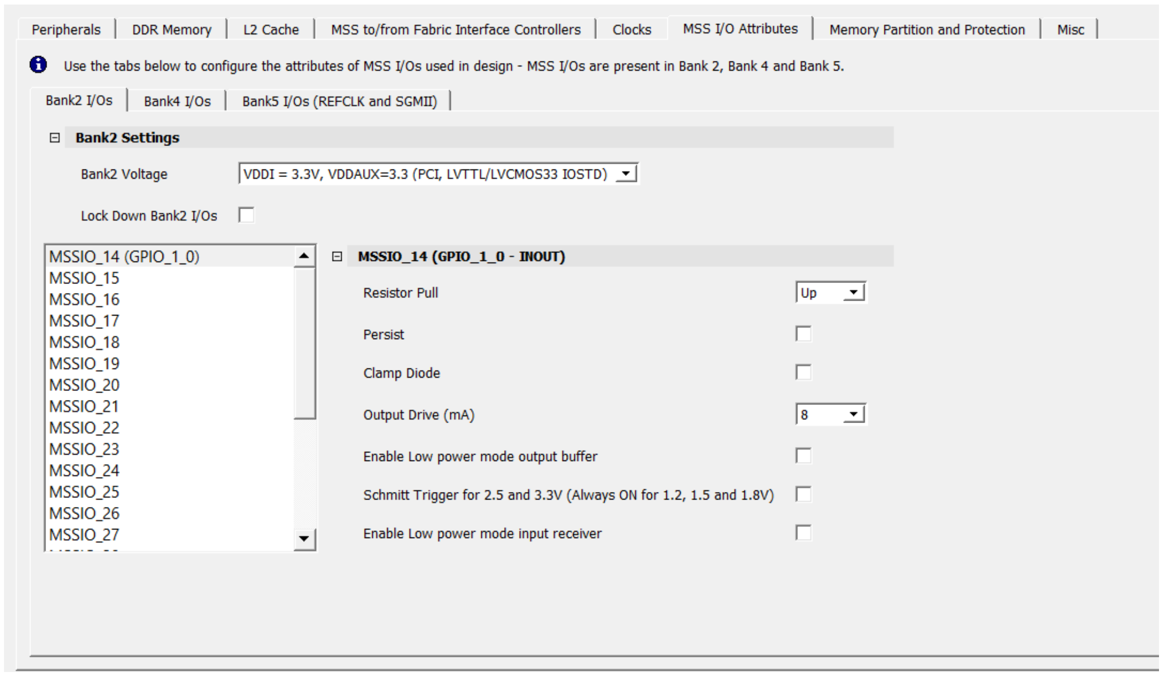
2.3.4. MSS I/O Attributes (Ask a Question)

MSS I/Os are present in Bank2, Bank4, and Bank5. The following sections describe the supported attributes for each bank.

2.3.4.1. Bank2 I/Os (Ask a Question)

The MSS I/Os are available across Bank2. The **Bank2 I/Os** tab allows you to select the electrical characteristics of the MSS I/Os. Each MSS I/O along with the settings must be enabled one-by-one.

Figure 2-11. Bank2 I/Os Tab



Peripherals
DDR Memory
L2 Cache
MSS to/from Fabric Interface Controllers
Clocks
MSS I/O Attributes
Memory Partition and Protection
Misc

Use the tabs below to configure the attributes of MSS I/Os used in design - MSS I/Os are present in Bank 2, Bank 4 and Bank 5.

Bank2 I/Os
Bank4 I/Os
Bank5 I/Os (REFCLK and SGMII)

Bank2 Settings

Bank2 Voltage	VDDI = 3.3V, VDDAUX=3.3 (PCI, LVTTL/LVCMOS33 IOSTD)
Lock Down Bank2 I/Os	☐

MSSIO_14 (GPIO_1_0)
MSSIO_15
MSSIO_16
MSSIO_17
MSSIO_18
MSSIO_19
MSSIO_20
MSSIO_21
MSSIO_22
MSSIO_23
MSSIO_24
MSSIO_25
MSSIO_26
MSSIO_27

MSSIO_14 (GPIO_1_0 - INOUT)

Resistor Pull	Up
Persist	☐
Clamp Diode	☐
Output Drive (mA)	8
Enable Low power mode output buffer	☐
Schmitt Trigger for 2.5 and 3.3V (Always ON for 1.2, 1.5 and 1.8V)	☐
Enable Low power mode input receiver	☐

Note: Enable the Lock Down Bank2 I/Os option to lock all the Bank2 MSSIOs.

The following table lists the I/O standards and the output drive.

Important: This is applicable to Bank2 and Bank4 MSSIOs only.

Table 2-3. I/O Standard and Output Drive

Protocol	Speed Mode	LVCMOS Standard	Output Current Drive	Supported Out Drive Values
USB 2.0	High	3.3V	8 mA-20 mA	8, 12, 16, and 20 mA
		2.5V	6 mA-16 mA	6, 8, 12, and 16 mA
		1.8V	6 mA-12 mA	6, 8, 10, and 12 mA
eMMC	Default speed	3.3V	3.3V => 2-20 mA	2, 4, 8, 12, 16, and 20 mA
		1.8V	1.8V => 2-10 mA	2, 4, 6, 8, and 10 mA
		1.2V	1.2V => 2-8 mA	2, 4, 6, and 8 mA
	High speed	3.3V	3.3V => 4-20 mA	4, 8, 12, 16, and 20 mA
		1.8V	1.8V => 4-10 mA	4, 6, 8, and 10 mA
		1.2V	1.2V => 4-8 mA	4, 6, and 8 mA
	High speed DDR	3.3V	3.3V => 2-20 mA	2, 4, 8, 12, 16, and 20 mA
		1.8V	1.8V => 2-10 mA	2, 4, 6, 8, and 10 mA
		1.2V	1.2V => 2-8 mA	2, 4, 6, and 8 mA
	HS200	1.8V	1.8V => 6-10 mA	6, 8, and 10 mA
		1.2V	1.2V => 4-8 mA	4, 6, and 8 mA
	HS400	1.8V	1.8V => 4-10 mA	4, 6, 8, and 10 mA
		1.2V	1.2V => 4-8 mA	4, 6, and 8 mA
	HS400-ES	1.8V	1.8V => 4-10 mA	4, 6, 8, and 10 mA
		1.2V	1.2V => 4-8 mA	4, 6, and 8 mA
SDIO	Low Speed	3.3V	2 mA-20 mA	2, 4, 8, 12, 16, and 20 mA
	Full Speed	3.3V	2 mA-20 mA	2, 4, 8, 12, 16, and 20 mA
SD	Default speed	3.3V	2 mA-20 mA	2, 4, 8, 12, 16, and 20 mA
	High speed	3.3V	4 mA-20 mA	4, 8, 12, 16, and 20 mA
	SDR12	1.8V	2 mA-10 mA	2, 4, 6, 8, and 10 mA
	SDR25	1.8V	4 mA-10 mA	4, 6, 8, and 10 mA
	SDR50	1.8V	4 mA-10 mA	4, 6, 8, and 10 mA
	DDR50	1.8V	4 mA-10 mA	4, 6, 8, and 10 mA
	SDR104	1.8V	6 mA-10 mA	6, 8, and 10 mA
CAN	—	3.3V	2 mA-20 mA	2, 4, 8, 12, 16, and 20 mA
QSPI		3.3V	3.3V => 8-20 mA	8, 12, 16, and 20 mA
		2.5V	2.5V => 8-16 mA	8, 12, and 16 mA
		1.8V	1.8V => 8-12 mA	8, 10, and 12 mA
		1.5V	1.5V => 8-10 mA	8 mA
		1.2V	1.2V => 6-8 mA	6 and 8 mA
SPI	Initiator	3.3V	3.3V => 8-20 mA	8, 12, 16, and 20 mA
		2.5V	2.5V => 8-16 mA	8, 12, and 16 mA
		1.8V	1.8V => 8-12 mA	8, 10, and 12 mA
		1.5V	1.5V => 8-10 mA	8 mA
		1.2V	1.2V => 6-8 mA	6 and 8 mA
	Target	3.3V	3.3V => 8-20 mA	8, 12, 16, and 20 mA
		2.5V	2.5V => 8-16 mA	8, 12, and 16 mA
		1.8V	1.8V => 8-12 mA	8, 10, and 12 mA
		1.5V	1.5V => 8 - 10 mA	8 mA
		1.2V	1.2V => 6-8 mA	6 and 8 mA

Table 2-3. I/O Standard and Output Drive (continued)

Protocol	Speed Mode	LVCMOS Standard	Output Current Drive	Supported Out Drive Values
MMUART	—	3.3V	3.3V => 2-20 mA	2, 4, 8, 12, 16, and 20 mA
		2.5V	2.5V => 4-16 mA	4, 6, 8, 12, and 16 mA
		1.8V	1.8V => 4-12 mA	4, 6, 8, 10, and 12 mA
I2C	Standard	3.3V	3.3V => 2-20 mA	2, 4, 8, 12, 16, and 20 mA
		1.8V	1.8V => 2 - 10 mA	2, 4, 6, 8, and 10 mA
	Fast	3.3V	3.3V => 2-20 mA	2, 4, 12, 16, and 20 mA
		1.8V	1.8V => 2-10 mA	2, 4, 6, 8, and 10 mA
Ethernet MAC(MDIO)	PHY Management Interface	3.3V	3.3V => 8-20 mA	8, 12, 16, and 20 mA
		2.5V	2.5V => 8-16 mA	8, 12, and 16 mA
		1.8V	1.8V => 8-12 mA	8, 10 and 12 mA
		1.5V	1.5V => 8 - 10 mA	8 mA
		1.2V	1.2V => 6-8 mA	6 and 8 mA
GPIO	—	3.3V	3.3V => 2-20 mA	2, 4, 8, 12, 16, and 20 mA
		2.5V	2.5V => 2-16 mA	2, 4, 6, 8, 12, and 16 mA
		1.8V	1.8V => 2-12 mA	2, 4, 6, 8, 10, and 12 mA
		1.5V	1.5V => 2-8 mA	2, 4, 6, and 8 mA
		1.2V	1.2V => 2-8 mA	2, 4, 6, and 8 mA

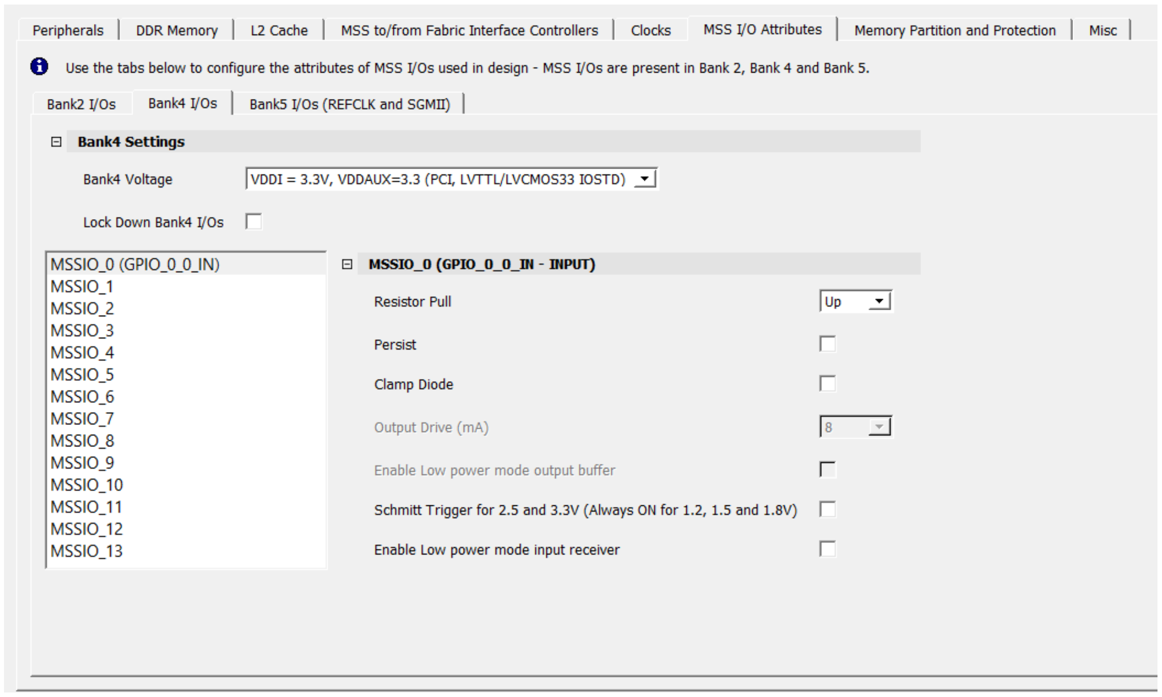
When a peripheral supports multiple speed modes, the following must be followed:

- I/O standard check must be specific for the peripheral and the speed mode is selected.
- Support for out drive values depends on the peripheral type, the speed mode, and the I/O standard.

2.3.4.2. Bank4 I/Os (Ask a Question)

The MSS I/Os are available across Bank 4. The **Bank4 I/Os** tab allows you to select the electrical characteristics of the MSS I/Os. Each MSS I/O along with the settings must be enabled one by one.

Figure 2-12. Bank4 I/Os Tab



Use the tabs below to configure the attributes of MSS I/Os used in design - MSS I/Os are present in Bank 2, Bank 4 and Bank 5.

Bank4 Settings

Bank4 Voltage: VDDI = 3.3V, VDDAUX=3.3 (PCI, LVTTL/LVCMOS33 IOSTD)

Lock Down Bank4 I/Os: ☐

MSSIO_0 (GPIO_0_0_IN - INPUT)

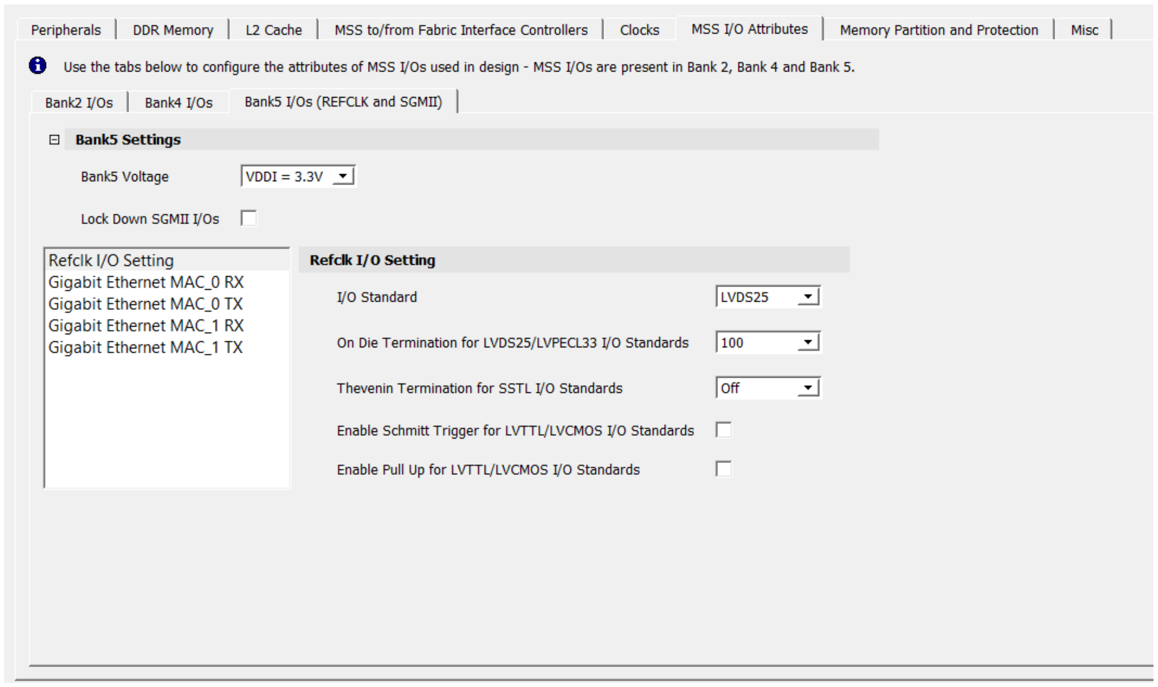
Resistor Pull	Up
Persist	☐
Clamp Diode	☐
Output Drive (mA)	8
Enable Low power mode output buffer	☐
Schmitt Trigger for 2.5 and 3.3V (Always ON for 1.2, 1.5 and 1.8V)	☐
Enable Low power mode input receiver	☐

Important: Enable the Lock Down Bank4 I/Os option to lock all the Bank4 MSSIOs.

2.3.4.3. Bank5 I/Os (REFCLK and SGMII) (Ask a Question)

Using the Bank5 I/Os (REFCLK and SGMII) tab, you can select the electrical characteristics of the Bank5 I/Os, as shown in the following figure. The tool generates a warning in the log window for unsupported selections.

Figure 2-13. Bank5 I/Os (REFCLK and SGMII) Tab with RefClk I/O Setting Option Selected


Peripherals
DDR Memory
L2 Cache
MSS to/from Fabric Interface Controllers
Clocks
MSS I/O Attributes
Memory Partition and Protection
Misc
Bank2 I/Os
Bank4 I/Os
Bank5 I/Os (REFCLK and SGMII)

Use the tabs below to configure the attributes of MSS I/Os used in design - MSS I/Os are present in Bank 2, Bank 4 and Bank 5.

Bank5 Settings

Bank5 Voltage: VDDI = 3.3V

Lock Down SGMII I/Os: ☐

Refclk I/O Setting
Gigabit Ethernet MAC_0 RX
Gigabit Ethernet MAC_0 TX
Gigabit Ethernet MAC_1 RX
Gigabit Ethernet MAC_1 TX

Refclk I/O Setting

I/O Standard	LVDS25
On Die Termination for LVDS25/LVPECL33 I/O Standards	100
Thevenin Termination for SSTL I/O Standards	Off
Enable Schmitt Trigger for LVTTL/LVCMOS I/O Standards	☐
Enable Pull Up for LVTTL/LVCMOS I/O Standards	☐

Important: Enable the Lock Down SGMII I/Os option to lock all the SGMII I/Os.

PolarFire SoC supports two full-duplex SGMII channels (Channel0 and Channel1). Each channel has one RX and one TX. There are two input and two output I/Os that must be configured for SGMII, and all I/Os are differential.

SGMII Inputs

MAC_0 (Channel0) RX and MAC_1 (Channel1) RX are inputs and have the following options:

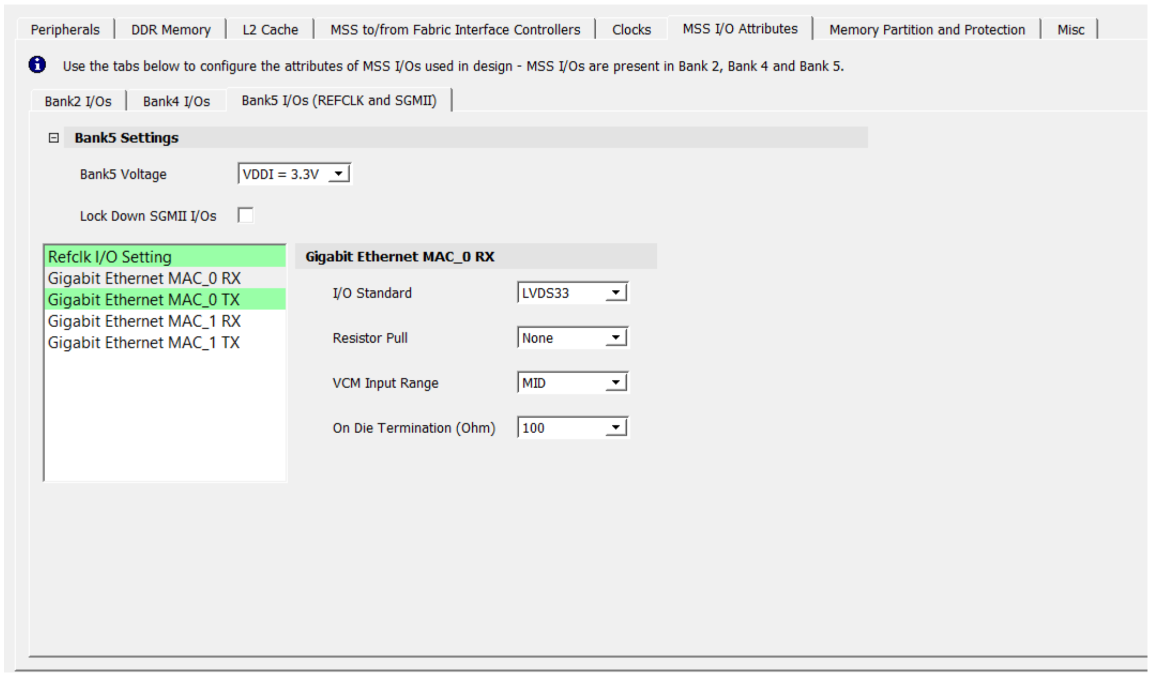
- I/O Standard

  Important: The IOSTD must match the bank voltage in the **Refclk I/O Setting** section. For example, if the bank 5 voltage VDDI is 3.3 V, you cannot set SGMII I/Os to any IOSTD, which is 2.5 V such as LVDS25, RSDS25, MINILVDS25, SUBLVDS25, PPDS25, and LCMDS25.

- Resistor Pull
- VCM Range
- On Die Termination (Ω)

The following figure shows the SGMII RX.

Figure 2-14. Bank5 I/Os (REFCLK and SGMII) Tab with Gigabit Ethernet MAC_0 RX Option Selected


Peripherals
DDR Memory
L2 Cache
MSS to/from Fabric Interface Controllers
Clocks
MSS I/O Attributes
Memory Partition and Protection
Misc
Bank2 I/Os
Bank4 I/Os
Bank5 I/Os (REFCLK and SGMII)
Refclk I/O Setting
Gigabit Ethernet MAC_0 RX
Gigabit Ethernet MAC_0 TX
Gigabit Ethernet MAC_1 RX
Gigabit Ethernet MAC_1 TX

Use the tabs below to configure the attributes of MSS I/Os used in design - MSS I/Os are present in Bank 2, Bank 4 and Bank 5.

Bank5 Settings

Bank5 Voltage: VDDI = 3.3V

Lock Down SGMII I/Os: ☐

Gigabit Ethernet MAC_0 RX

I/O Standard: LVDS33

Resistor Pull: None

VCM Input Range: MID

On Die Termination (Ohm): 100

SGMII RX Register Settings

The following table lists the Channel 0/Channel 1 RX register settings.

Table 2-4. Channel 0/1 RX Register Settings

GUI Labels/Parameter Name	Options
I/O Standard	LVDS33, LVDS25, RSDS33, RSDS25, MINILVDS33, MINILVDS25, SUBLVDS33, SUBLVDS25, PPDS33, PPDS25, LCMDS33, LCMDS25
Resistor Pull	None, Up, Down
VCM Input Range	MID, LOW
On Die Termination (Ohm)	OFF, 100

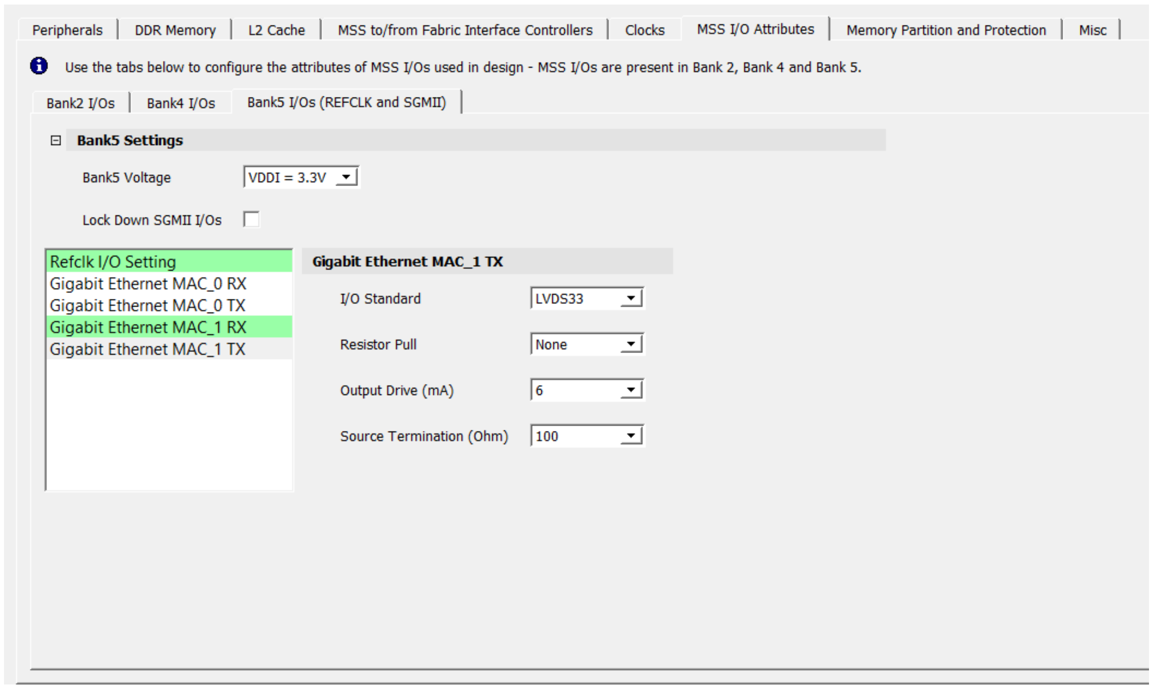
SGMII Outputs

MAC_0 (Channel0) TX and MAC_1 (Channel1) TX are outputs and have the following options:

- I/O Standard
- Resistor Pull
- Output Drive
- Source Termination (Ohm)

The following figure shows the SGMII TX.

Figure 2-15. Bank5 I/Os(REFCLK and SGMII) Tab with Gigabit Ethernet MAC_1 TX Option Selected


Peripherals
DDR Memory
L2 Cache
MSS to/from Fabric Interface Controllers
Clocks
MSS I/O Attributes
Memory Partition and Protection
Misc
Bank2 I/Os
Bank4 I/Os
Bank5 I/Os (REFCLK and SGMII)
Refclk I/O Setting
Gigabit Ethernet MAC_0 RX
Gigabit Ethernet MAC_0 TX
Gigabit Ethernet MAC_1 RX
Gigabit Ethernet MAC_1 TX

Use the tabs below to configure the attributes of MSS I/Os used in design - MSS I/Os are present in Bank 2, Bank 4 and Bank 5.

Bank5 Settings

Bank5 Voltage: VDDI = 3.3V

Lock Down SGMII I/Os: ☐

Gigabit Ethernet MAC_1 TX

I/O Standard: LVDS33

Resistor Pull: None

Output Drive (mA): 6

Source Termination (Ohm): 100

SGMII TX Register Settings

The following table lists the Channel 0/Channel 1 TX register settings.

Table 2-5. Channel 0 /1 TX Register Settings

GUI Labels/Parameter Name	Options
I/O Standard	LVDS33, LVDS25, RSDS33, RSDS25, MINILVDS33, MINILVDS25, SUBLVDS33, SUBLVDS25, PPDS33, PPDS25, LCMDS33, LCMDS25
Resistor Pull	None, Up, Down
Output Drive (mA)	1.5, 2, 3, 3.5, 4, 6
Source Termination (Ohm)	OFF, 100

SGMII Output I/O Standard Settings

Based on the selected I/O standards, you must enforce the following DRC checks:

Table 2-6. SGMII Output DRC Check

I/O_TYPE	Direction	Legal Output DRIVE Settings (mA)
LVDS33	Output	6, 4, 3.5, 3
LVDS25	Output	6, 4, 3.5, 3
RSDS33	Output	4, 3, 2, 1.5
RSDS25	Output	4, 3, 2, 1.5
MINILVDS33	Output	6, 4, 3.5, 3
MINILVDS25	Output	6, 4, 3.5, 3
SUBLVDS33	Output	3, 2, 1.5, 1
SUBLVDS25	Output	3, 2, 1.5, 1
PPDS33	Output	4, 3, 2, 1.5
PPDS25	Output	4, 3, 2, 1.5
LCMDS33	Output	6, 4, 3.5, 3

Table 2-6. SGMII Output DRC Check (continued)

I/O_TYPE	Direction	Legal Output DRIVE Settings (mA)
LCMDS25	Output	6, 4, 3.5, 3

I/O Standard and Supported Output Drive

Voltage selection for Bank 5 must match the I/O Standard selected for TX and RX in both channels.

Example 2-1.

Bank5 Voltage selection -> VDDI = 3.3V -> LVDS33 is legal but not LVDS25

Table 2-7. Legal Values/Settings for I/O_TYPE/Output Drive Combination

I/O_TYPE	Legal Output DRIVE Settings (mA)
LVDS33	6, 4, 3.5, 3
LVDS25	6, 4, 3.5, 3
RSDS33	4, 3, 2, 1.5
RSDS25	4, 3, 2, 1.5
MINILVDS33	6, 4, 3.5, 3
MINILVDS25	6, 4, 3.5, 3
SUBLVDS33	3, 2, 1.5, 1
SUBLVDS25	3, 2, 1.5, 1
PPDS33	4, 3, 2, 1.5
PPDS25	4, 3, 2, 1.5
LCMDS33	6, 4, 3.5, 3
LCMDS25	6, 4, 3.5, 3

2.3.4.4. Bank2 and Bank4 I/Os Related to SD and eMMC Muxing (Ask a Question)

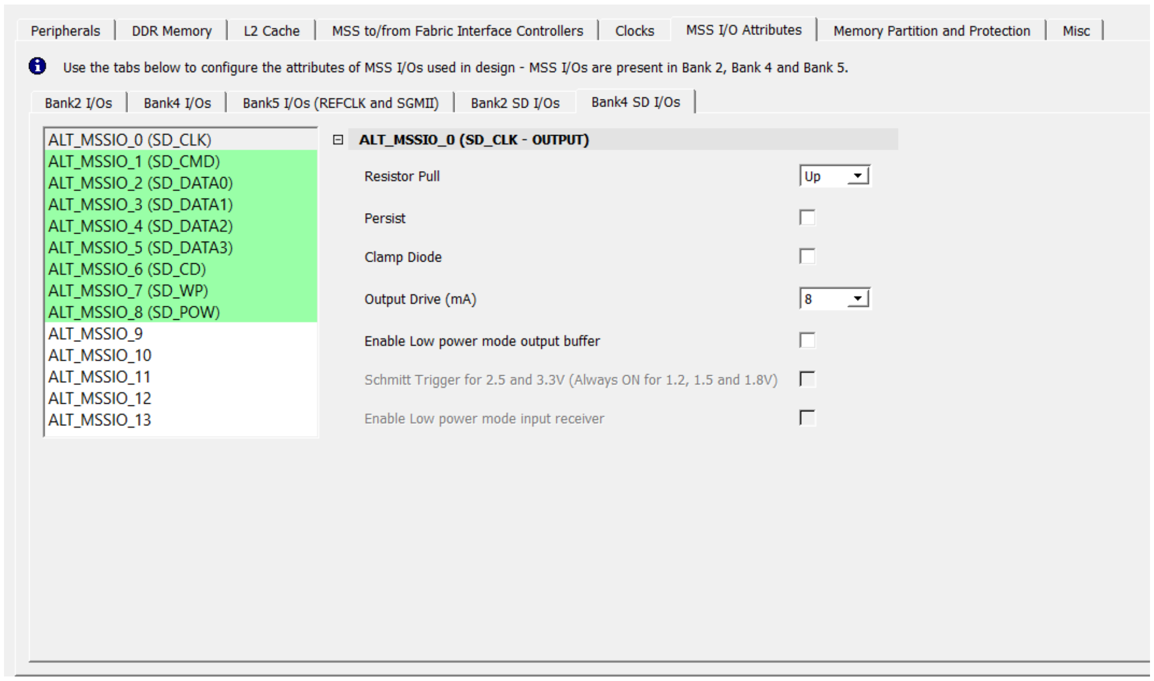
The eMMC and SD peripherals use the same MSS I/Os, so both peripherals cannot be active at the same time. However, the PolarFire SoC MSS Configurator allows you to configure the electrical characteristics of the MSS I/Os related to both peripherals when **eMMC and SD muxing** is enabled in the **Peripherals** tab, where one of the peripherals (eMMC or SD) is active at power-up and the other peripheral is not active at power-up. The MSS I/Os related to a peripheral that is active at power-up is listed in **Bank2 I/Os** tab and **Bank4 I/Os** tab. MSS I/Os related to a peripheral that is not active at power-up is shown as follows:

In the highlighted tabs, you might be able to select the electrical characteristics of eMMC or SD MSS I/Os only (electrical characteristics of MSS I/Os related to any other peripherals are grayed-out).

- If **eMMC and SD muxing** is selected as **Enabled (eMMC active at Power-Up)** in **Peripherals** tab, the highlighted tabs will be shown to the user.

Important: The eMMC MSS I/Os are listed under Bank2/Bank4 I/Os tabs, and SD MSS I/Os are listed in the highlighted tabs.

Figure 2-16. Bank2 and Bank4 SD I/Os Tabs


Peripherals
DDR Memory
L2 Cache
MSS to/from Fabric Interface Controllers
Clocks
MSS I/O Attributes
Memory Partition and Protection
Misc

Use the tabs below to configure the attributes of MSS I/Os used in design - MSS I/Os are present in Bank 2, Bank 4 and Bank 5.

ALT_MSSIO_0 (SD_CLK)
ALT_MSSIO_1 (SD_CMD)
ALT_MSSIO_2 (SD_DATA0)
ALT_MSSIO_3 (SD_DATA1)
ALT_MSSIO_4 (SD_DATA2)
ALT_MSSIO_5 (SD_DATA3)
ALT_MSSIO_6 (SD_CD)
ALT_MSSIO_7 (SD_WP)
ALT_MSSIO_8 (SD_POW)
ALT_MSSIO_9
ALT_MSSIO_10
ALT_MSSIO_11
ALT_MSSIO_12
ALT_MSSIO_13

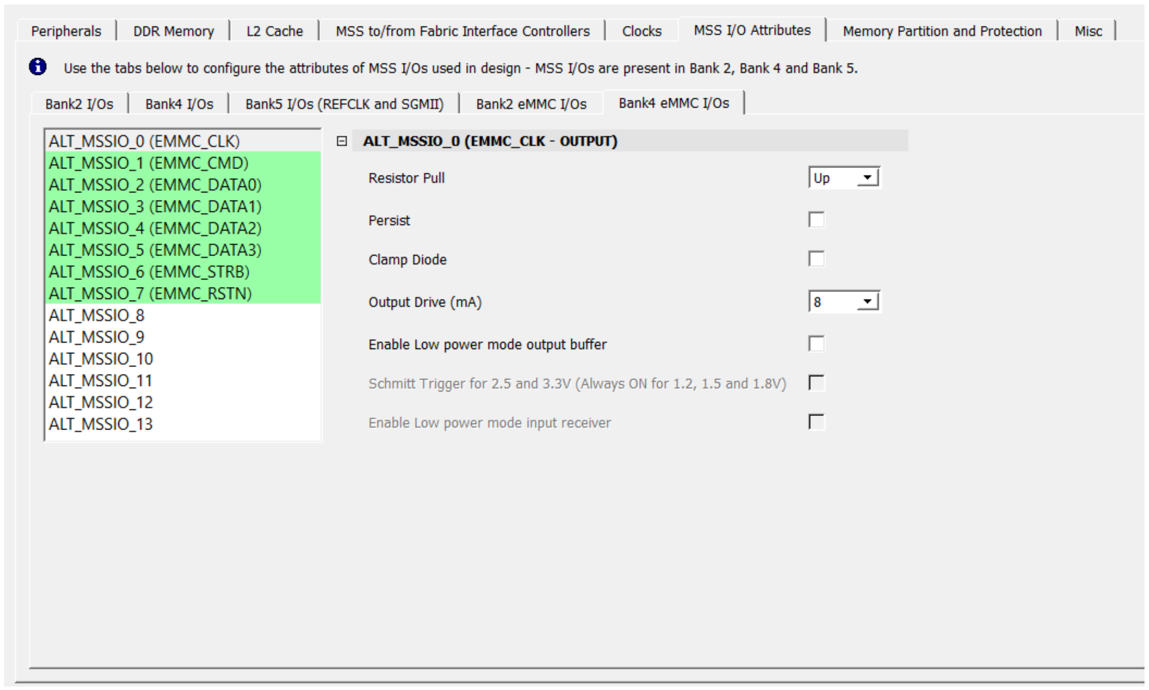
ALT_MSSIO_0 (SD_CLK - OUTPUT)

Resistor Pull	Up
Persist	☐
Clamp Diode	☐
Output Drive (mA)	8
Enable Low power mode output buffer	☐
Schmitt Trigger for 2.5 and 3.3V (Always ON for 1.2, 1.5 and 1.8V)	☐
Enable Low power mode input receiver	☐

- If **eMMC and SD muxing** is selected as **Enabled (SD active at Power-Up)** in **Peripherals** tab, the highlighted tabs will be shown to the user.

Important: The SD MSS I/Os are listed under Bank2/Bank4 I/Os tabs, and eMMC MSS I/Os are listed in the highlighted tabs.

Figure 2-17. Bank2 and Bank4 eMMC I/Os Tabs


Peripherals
DDR Memory
L2 Cache
MSS to/from Fabric Interface Controllers
Clocks
MSS I/O Attributes
Memory Partition and Protection
Misc

Use the tabs below to configure the attributes of MSS I/Os used in design - MSS I/Os are present in Bank 2, Bank 4 and Bank 5.

ALT_MSSIO_0 (EMMC_CLK)
ALT_MSSIO_1 (EMMC_CMD)
ALT_MSSIO_2 (EMMC_DATA0)
ALT_MSSIO_3 (EMMC_DATA1)
ALT_MSSIO_4 (EMMC_DATA2)
ALT_MSSIO_5 (EMMC_DATA3)
ALT_MSSIO_6 (EMMC_STRB)
ALT_MSSIO_7 (EMMC_RSTN)
ALT_MSSIO_8
ALT_MSSIO_9
ALT_MSSIO_10
ALT_MSSIO_11
ALT_MSSIO_12
ALT_MSSIO_13

ALT_MSSIO_0 (EMMC_CLK - OUTPUT)

Resistor Pull	Up
Persist	☐
Clamp Diode	☐
Output Drive (mA)	8
Enable Low power mode output buffer	☐
Schmitt Trigger for 2.5 and 3.3V (Always ON for 1.2, 1.5 and 1.8V)	☐
Enable Low power mode input receiver	☐

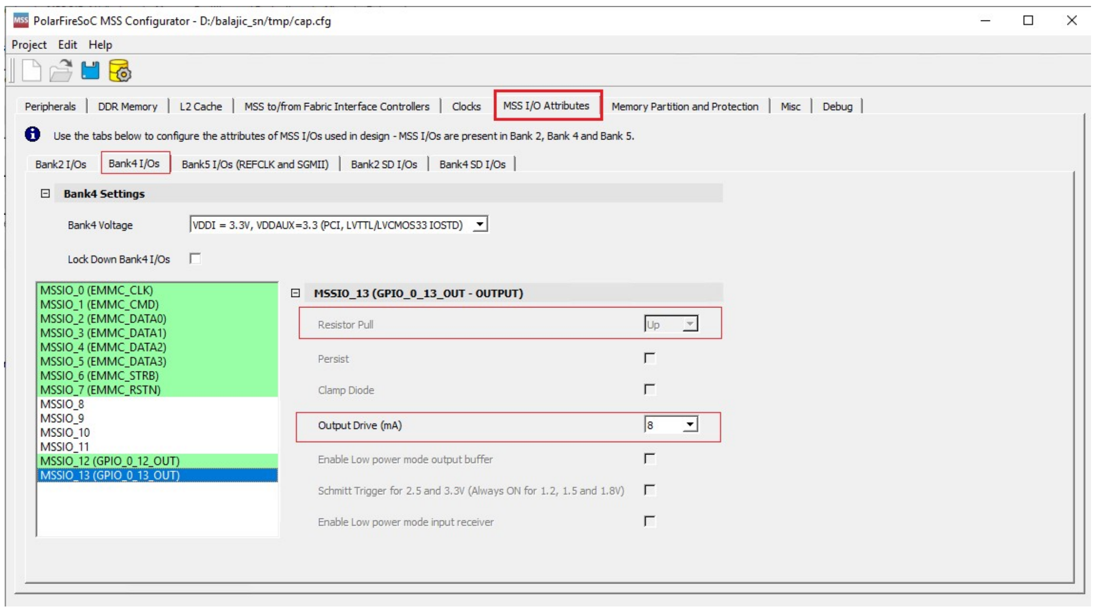
Important: When **eMMC and SD muxing** is enabled, the user must ensure that the required embedded software driver and fabric design support is available to dynamically switch between eMMC and SD peripherals.

When **eMMC and SD muxing** is enabled, MSSIO PADs can be configured as Pull-Up or Pull-Down by selecting GPIOs as **Static High** or **Static Low**. When **Static High** or **Static Low** is selected for a GPIO in Bank 4 or Bank 2, all the I/O attributes of the corresponding MSSIO are greyed out except **Output Drive**. **Resistor Pull** is set to **Up** when **Static High** is selected and to **Down** when **Static Low** is selected by default for the corresponding MSSIOs in **Bank2 I/Os** and **Bank4 I/Os** tab and vice versa in the **Bank2 eMMC(SD) I/Os** and **Bank4 eMMC(SD) I/Os** tab.

For example: when **Static High** is selected for `GPIO_0_13` in the **Peripherals** tab, the **MSSIO I/O Attributes** tab will be setup as follows:

- Bank4 I/Os settings are shown in the following figure.

Figure 2-18. Bank4 I/O Settings


PolarFireSoC MSS Configurator - D:/balajic_sn/tmp/cap.cfg
Project Edit Help
Peripherals | DDR Memory | L2 Cache | MSS to/from Fabric Interface Controllers | Clocks | MSS I/O Attributes | Memory Partition and Protection | Misc | Debug
Bank2 I/Os | Bank4 I/Os | Bank5 I/Os (REFCLK and SGMII) | Bank2 SD I/Os | Bank4 SD I/Os

Use the tabs below to configure the attributes of MSS I/Os used in design - MSS I/Os are present in Bank 2, Bank 4 and Bank 5.

Bank4 Settings

Bank4 Voltage	VDDI = 3.3V, VDDAUX=3.3 (PCI, LVTTL/LVCMOS33 IOSTD)
Lock Down Bank4 I/Os	☐

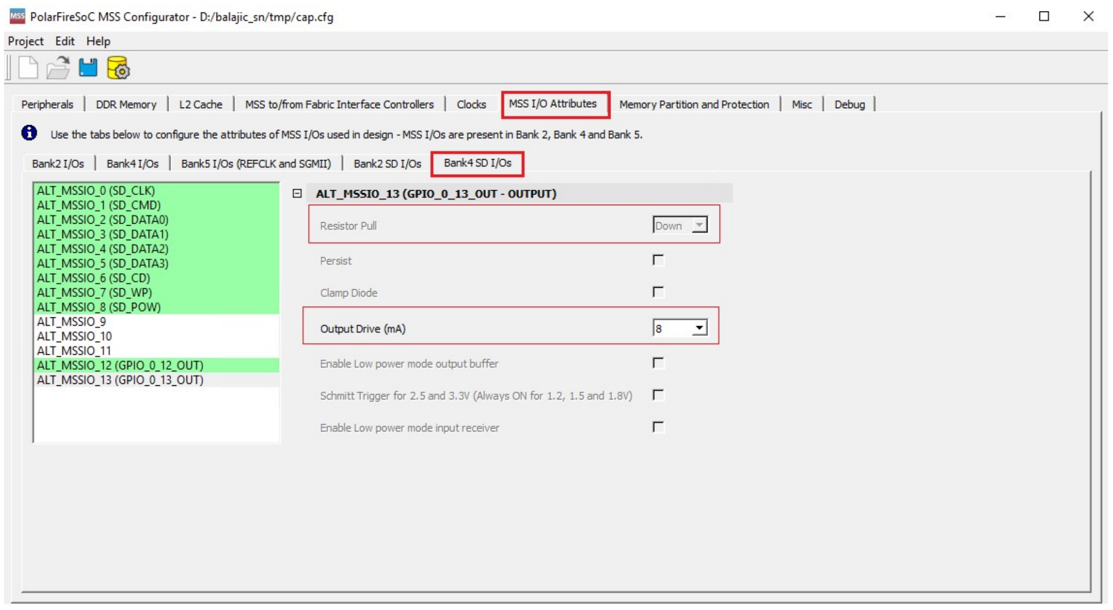
MSSIO_0 (EMMC_CLK)
MSSIO_1 (EMMC_CMD)
MSSIO_2 (EMMC_DATA0)
MSSIO_3 (EMMC_DATA1)
MSSIO_4 (EMMC_DATA2)
MSSIO_5 (EMMC_DATA3)
MSSIO_6 (EMMC_STRB)
MSSIO_7 (EMMC_RSTN)
MSSIO_8
MSSIO_9
MSSIO_10
MSSIO_11
MSSIO_12 (GPIO_0_12_OUT)
MSSIO_13 (GPIO_0_13_OUT)

MSSIO_13 (GPIO_0_13_OUT - OUTPUT)

Resistor Pull	Up
Persist	☐
Clamp Diode	☐
Output Drive (mA)	8
Enable Low power mode output buffer	☐
Schmitt Trigger for 2.5 and 3.3V (Always ON for 1.2, 1.5 and 1.8V)	☐
Enable Low power mode input receiver	☐

- Alternate Bank4 I/Os settings are shown in the following figure.

Figure 2-19. Alternate Bank4 I/Os Settings


PolarFireSoC MSS Configurator - D:/balajic_sn/tmp/cap.cfg
Project Edit Help
Peripherals | DDR Memory | L2 Cache | MSS to/from Fabric Interface Controllers | Clocks | MSS I/O Attributes | Memory Partition and Protection | Misc | Debug
Bank2 I/Os | Bank4 I/Os | Bank5 I/Os (REFCLK and SGMII) | Bank2 SD I/Os | Bank4 SD I/Os

Use the tabs below to configure the attributes of MSS I/Os used in design - MSS I/Os are present in Bank 2, Bank 4 and Bank 5.

ALT_MSSIO_0 (SD_CLK)
ALT_MSSIO_1 (SD_CMD)
ALT_MSSIO_2 (SD_DATA0)
ALT_MSSIO_3 (SD_DATA1)
ALT_MSSIO_4 (SD_DATA2)
ALT_MSSIO_5 (SD_DATA3)
ALT_MSSIO_6 (SD_CD)
ALT_MSSIO_7 (SD_WP)
ALT_MSSIO_8 (SD_POW)
ALT_MSSIO_9
ALT_MSSIO_10
ALT_MSSIO_11
ALT_MSSIO_12 (GPIO_0_12_OUT)
ALT_MSSIO_13 (GPIO_0_13_OUT)

ALT_MSSIO_13 (GPIO_0_13_OUT - OUTPUT)

Resistor Pull	Down
Persist	☐
Clamp Diode	☐
Output Drive (mA)	8
Enable Low power mode output buffer	☐
Schmitt Trigger for 2.5 and 3.3V (Always ON for 1.2, 1.5 and 1.8V)	☐
Enable Low power mode input receiver	☐

2.3.5. DDR Memory (Ask a Question)

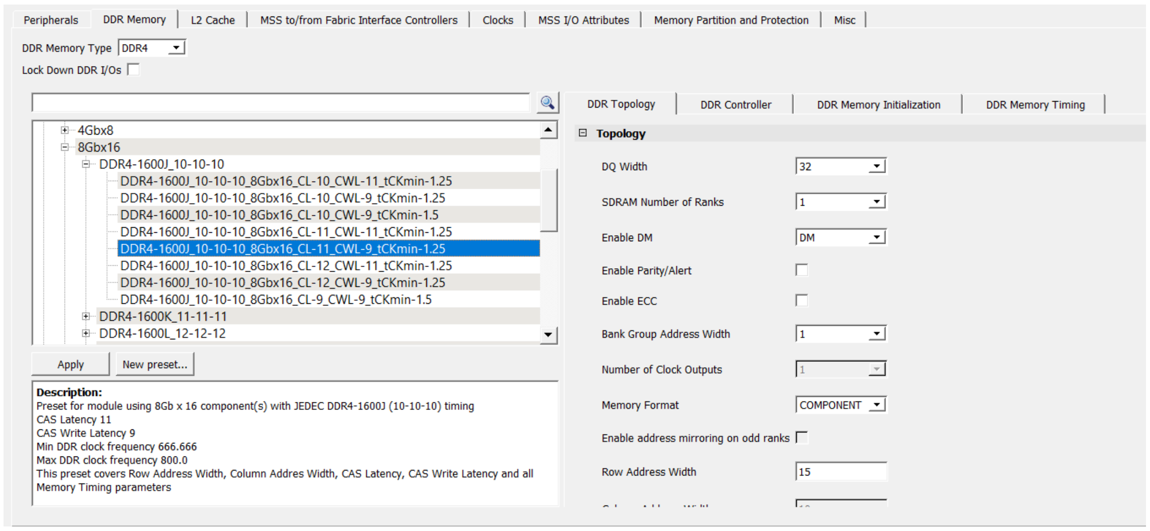
Select the required DDR type from the **DDR Memory Type** pull-down. The DDR configuration options are available on the **DDR Topology**, **DDR Controller**, **DDR Memory Initialization**, and **DDR Memory Timing** tabs (see the following figure).

For more information about the architecture and functional blocks of MSS DDR Controller, see PolarFire SoC FPGA MSS Technical Reference Manual. This section describes the options available in the **MSS Configurator** > **DDR Memory** tab for configuring MSS DDR Controller.

The **DDR Topology** tab, in the following figure, controls the physical aspects of the memory, such as data and address widths, enabling of ECC and DM, and setting the clock frequency.

- For DDR3 and DDR4, the COMPONENT, UDIMM, RDIMM, LRDIMM, and SODIMM memory formats are supported.
- For LPDDR3/4, only the COMPONENT memory format is supported.

Figure 2-20. DDR Memory Tab


Peripherals | DDR Memory | L2 Cache | MSS to/from Fabric Interface Controllers | Clocks | MSS I/O Attributes | Memory Partition and Protection | Misc
DDR Topology | DDR Controller | DDR Memory Initialization | DDR Memory Timing
Apply
New preset...

DDR Memory Type DDR4
Lock Down DDR I/Os ☐

4Gbx8
8Gbx16
DDR4-1600J_10-10-10
DDR4-1600J_10-10-10_8Gbx16_CL-10_CWL-11_tCKmin-1.25
DDR4-1600J_10-10-10_8Gbx16_CL-10_CWL-9_tCKmin-1.25
DDR4-1600J_10-10-10_8Gbx16_CL-10_CWL-9_tCKmin-1.5
DDR4-1600J_10-10-10_8Gbx16_CL-11_CWL-11_tCKmin-1.25
DDR4-1600J_10-10-10_8Gbx16_CL-11_CWL-9_tCKmin-1.25
DDR4-1600J_10-10-10_8Gbx16_CL-12_CWL-11_tCKmin-1.25
DDR4-1600J_10-10-10_8Gbx16_CL-12_CWL-9_tCKmin-1.25
DDR4-1600J_10-10-10_8Gbx16_CL-9_CWL-9_tCKmin-1.5
DDR4-1600K_11-11-11
DDR4-1600L_12-12-12

Description:
Preset for module using 8Gb x 16 component(s) with JEDEC DDR4-1600J (10-10-10) timing
CAS Latency 11
CAS Write Latency 9
Min DDR clock frequency 666.666
Max DDR clock frequency 800.0
This preset covers Row Address Width, Column Addres Width, CAS Latency, CAS Write Latency and all Memory Timing parameters

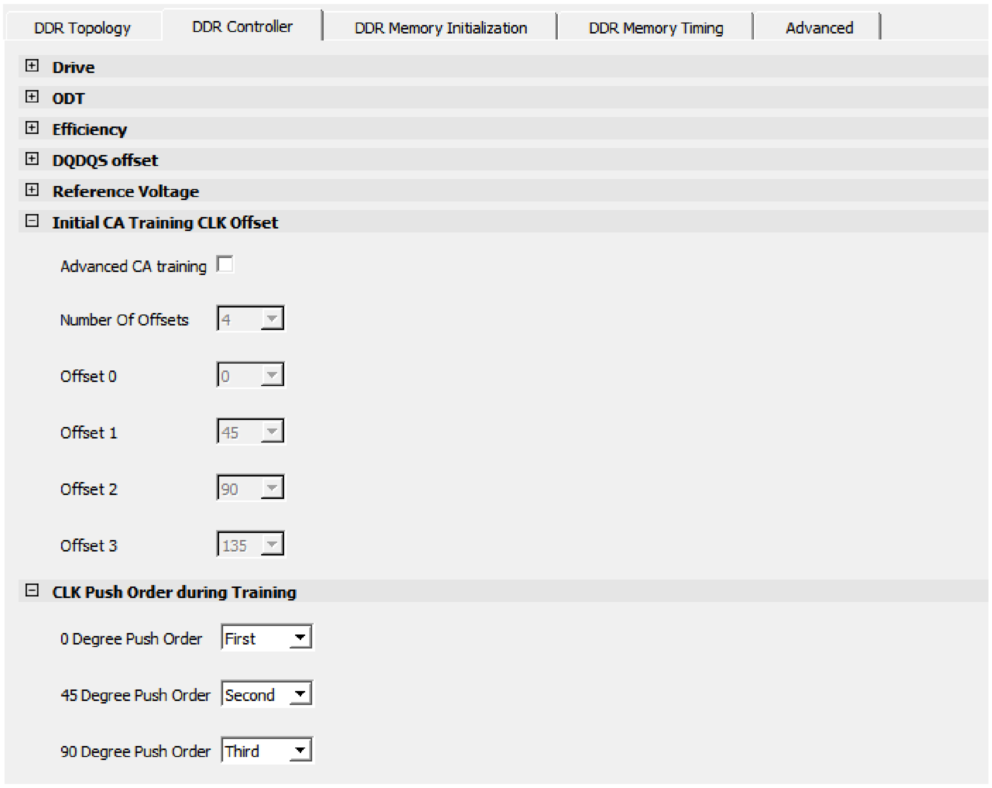
Topology

DQ Width	32
SDRAM Number of Ranks	1
Enable DM	DM
Enable Parity/Alert	☐
Enable ECC	☐
Bank Group Address Width	1
Number of Clock Outputs	1
Memory Format	COMPONENT
Enable address mirroring on odd ranks	☐
Row Address Width	15

Important:
- Enable the Lock Down DDR I/Os option to lock all the DDR I/Os.
- Configure the DDR parameters according to the data sheet from the DDR vendor.

The **DDR Controller** tab controls the DQS Drive, ODT, Precharge look-ahead, and Address Ordering.

Figure 2-21. DDR Controller Tab



Advanced CA training ☐

Number Of Offsets 4

Offset 0 0

Offset 1 45

Offset 2 90

Offset 3 135

0 Degree Push Order First

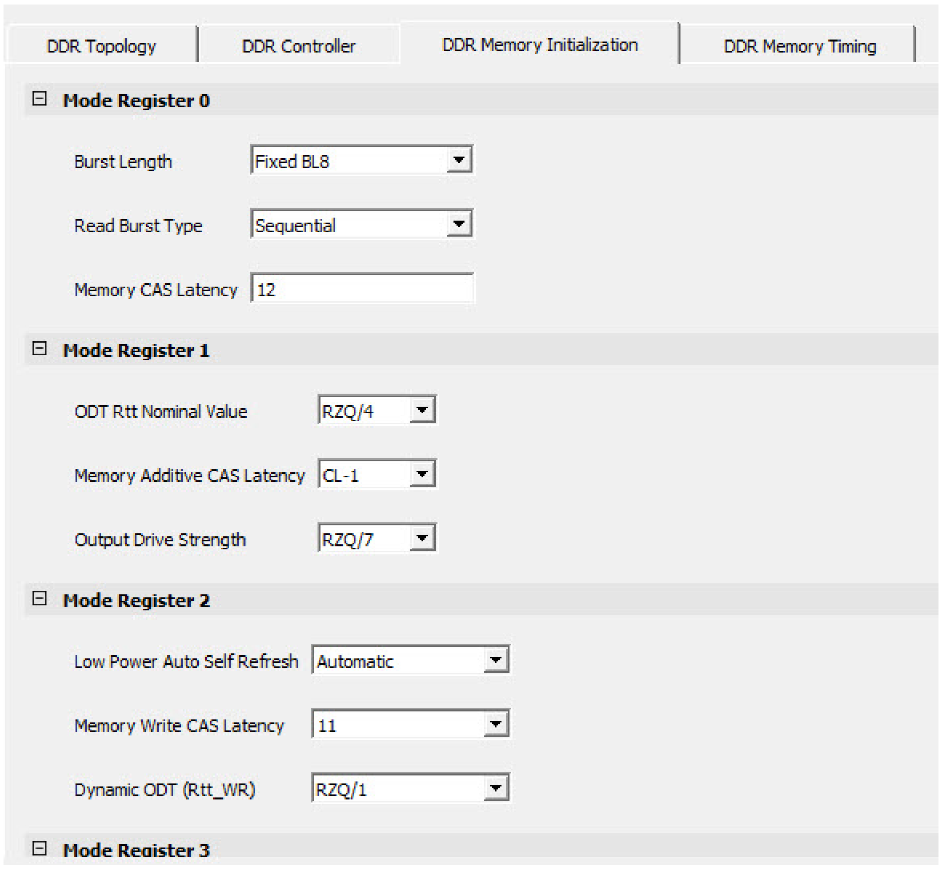
45 Degree Push Order Second

90 Degree Push Order Third

The **DDR Memory Initialization** tab controls the DDR mode register configuration according to the JEDEC specification. In the PolarFire SoC FPGA DDR architecture, these parameters are passed to the start-up code running on the E51 monitor core, which then performs the DDR initialization sequence and configures the mode registers.

The following figure shows the memory initialization configuration.

Figure 2-22. DDR Memory Initialization



Mode Register 0

Burst Length	Fixed BL8
Read Burst Type	Sequential
Memory CAS Latency	12

Mode Register 1

ODT Rtt Nominal Value	RZQ/4
Memory Additive CAS Latency	CL-1
Output Drive Strength	RZQ/7

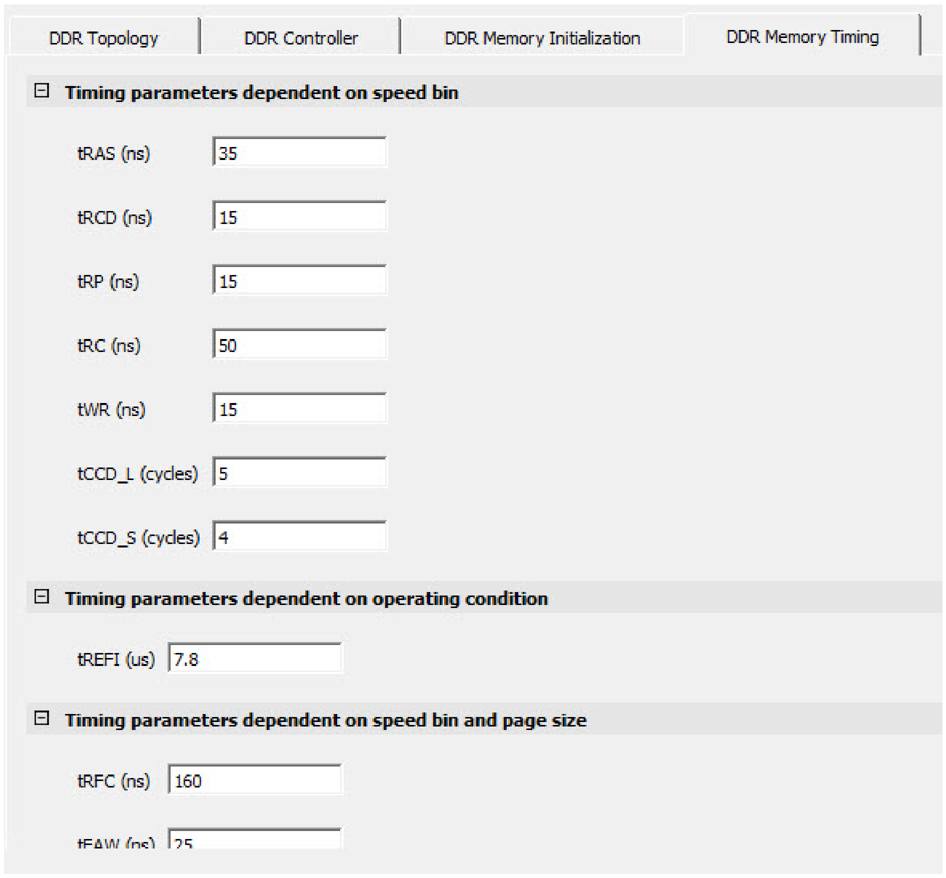
Mode Register 2

Low Power Auto Self Refresh	Automatic
Memory Write CAS Latency	11
Dynamic ODT (Rtt_WR)	RZQ/1

Mode Register 3

The **DDR Memory Timing** tab controls the timing parameters, which are translated to the appropriate configuration values for the DDR subsystem IP.

Figure 2-23. DDR Memory Timing Tab



Timing parameters dependent on speed bin

tRAS (ns)	35
tRCD (ns)	15
tRP (ns)	15
tRC (ns)	50
tWR (ns)	15
tCCD_L (cycles)	5
tCCD_S (cycles)	4

Timing parameters dependent on operating condition

tREFI (us)	7.8

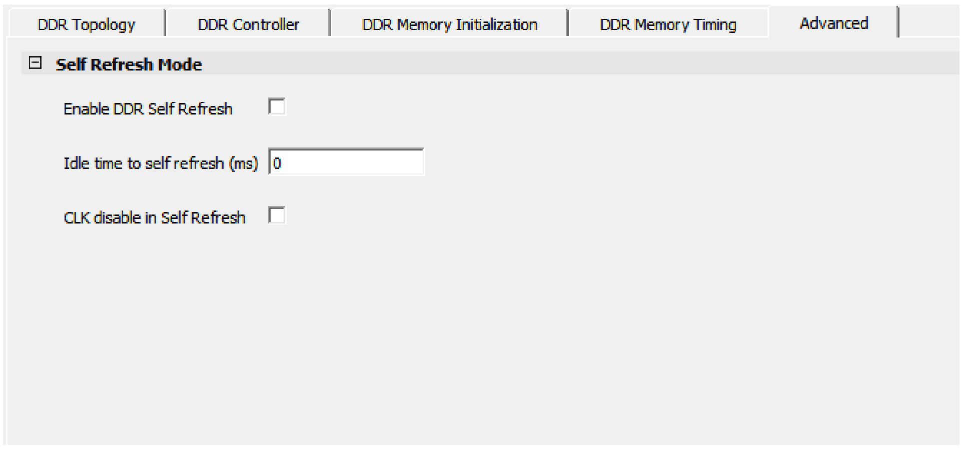
Timing parameters dependent on speed bin and page size

tRFC (ns)	160
tFAW (ns)	25

The **Advanced** tab controls the self refresh mode parameters (idle time to self refresh and clock disable in self refresh).

Important: This tab is only supported for LPDDR3 and LPDDR4.

Figure 2-24. Advanced Tab



Self Refresh Mode

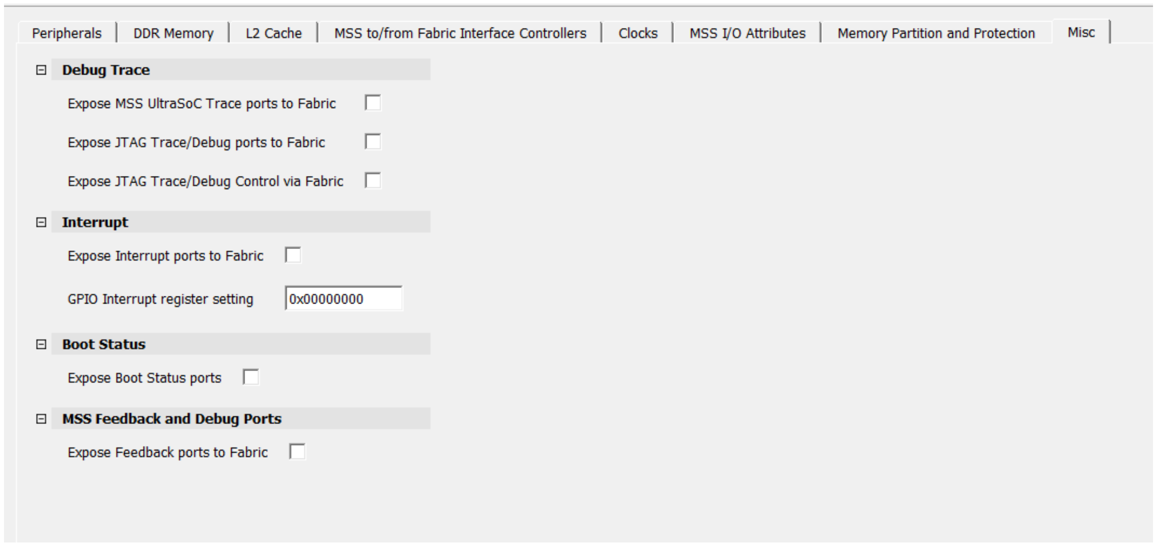
Enable DDR Self Refresh	☐
Idle time to self refresh (ms)	0
CLK disable in Self Refresh	☐

2.3.6. Misc (Ask a Question)

Use the **Misc** tab to enable the following options:

- Trace functionality
- JTAG (Debug) functionality
- Interrupts to/from MSS
- Configuring GPIO Interrupt Register
- Exposing Boot Status ports
- Exposing feedback ports to fabric

Figure 2-25. Misc Tab


Peripherals
DDR Memory
L2 Cache
MSS to/from Fabric Interface Controllers
Clocks
MSS I/O Attributes
Memory Partition and Protection
Misc

Debug Trace

Expose MSS UltraSoC Trace ports to Fabric ☐

Expose JTAG Trace/Debug ports to Fabric ☐

Expose JTAG Trace/Debug Control via Fabric ☐

Interrupt

Expose Interrupt ports to Fabric ☐

GPIO Interrupt register setting 0x00000000

Boot Status

Expose Boot Status ports ☐

MSS Feedback and Debug Ports

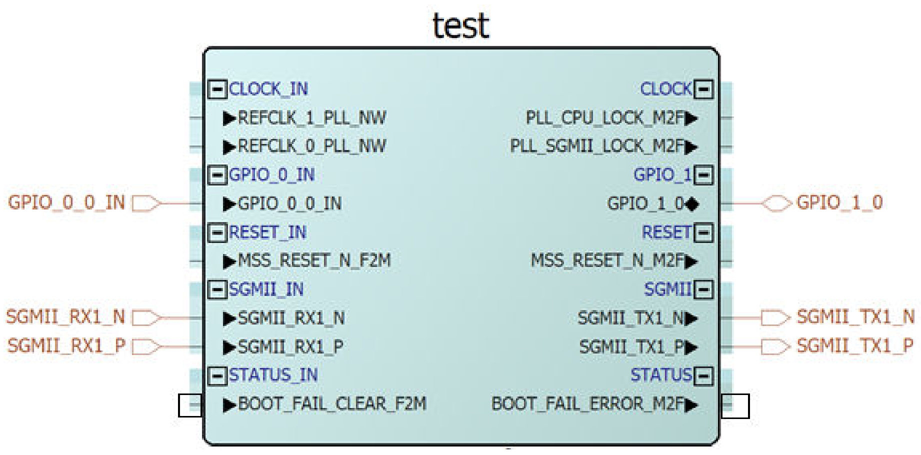
Expose Feedback ports to Fabric ☐

For more information, see PolarFire SoC FPGA MSS Technical Reference Manual.

By default, these options are marked as **Unused**. When any of the options are enabled, the corresponding ports are exposed on the MSS block (see the following figure).

Boot Status: When **Expose Boot Status ports** option is selected, the ports `BOOT_FAIL_CLEAR_F2M` and `BOOT_FAIL_ERROR_M2F` are exposed as shown in the following figure. Both the signals typically represent binary states through voltage levels and are generally synchronous with the MSS clock.

Figure 2-26. PFSOC_MSS_C0_0 Jtag Trace Enabled

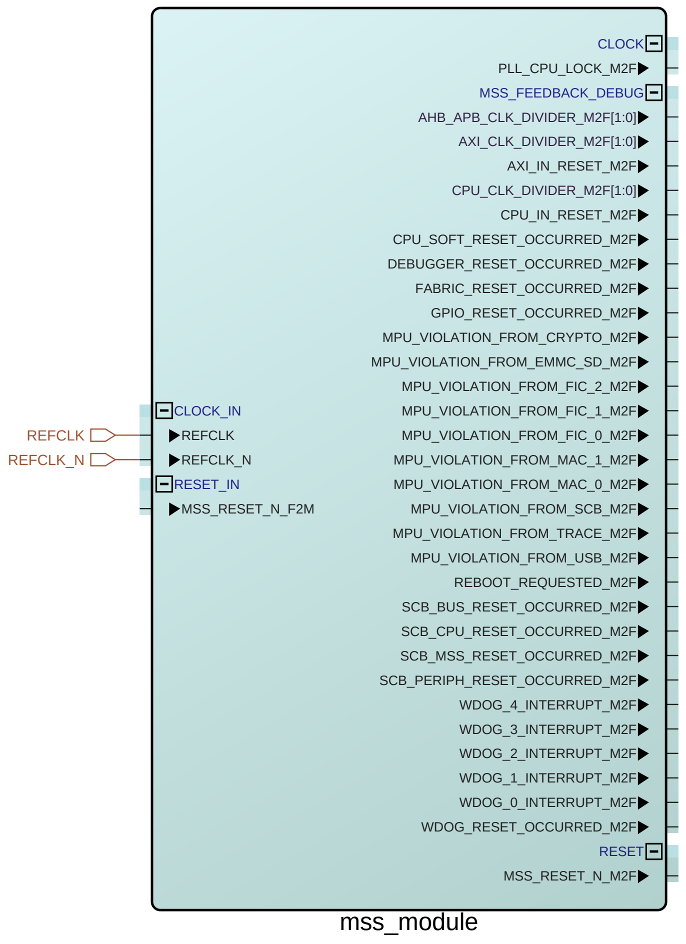


`Boot_fail_error_M2F` is a signal from the MSS to the FPGA fabric that indicates a boot failure has occurred. When the MSS detects the failure, it sets this signal high and informs the fabric to take action or log the issue.

`Boot_Fail_Clear_F2M` is a signal from the FPGA fabric to inform the MSS that the boot failure has been addressed. When the FPGA fabric handles the issue, it sets the signal high telling the MSS to clear the boot failure status.

MSS Feedback and Debug Ports: This option is only available for production devices. If selected, a group of `MSS_FEEDBACK_DEBUG` ports are exposed as shown in the following figure.

Figure 2-27. MSS Module With MSS_FEEDBACK_DEBUG Ports Exposed


CLOCK
PLL_CPU_LOCK_M2F
MSS_FEEDBACK_DEBUG
AHB_APB_CLK_DIVIDER_M2F[1:0]
AXI_CLK_DIVIDER_M2F[1:0]
AXI_IN_RESET_M2F
CPU_CLK_DIVIDER_M2F[1:0]
CPU_IN_RESET_M2F
CPU_SOFT_RESET_OCCURRED_M2F
DEBUGGER_RESET_OCCURRED_M2F
FABRIC_RESET_OCCURRED_M2F
GPIO_RESET_OCCURRED_M2F
MPU_VIOLATION_FROM_CRYPTO_M2F
MPU_VIOLATION_FROM_EMMC_SD_M2F
MPU_VIOLATION_FROM_FIC_2_M2F
MPU_VIOLATION_FROM_FIC_1_M2F
MPU_VIOLATION_FROM_FIC_0_M2F
MPU_VIOLATION_FROM_MAC_1_M2F
MPU_VIOLATION_FROM_MAC_0_M2F
MPU_VIOLATION_FROM_SCB_M2F
MPU_VIOLATION_FROM_TRACE_M2F
MPU_VIOLATION_FROM_USB_M2F
REBOOT_REQUESTED_M2F
SCB_BUS_RESET_OCCURRED_M2F
SCB_CPU_RESET_OCCURRED_M2F
SCB_MSS_RESET_OCCURRED_M2F
SCB_PERIPH_RESET_OCCURRED_M2F
WDOG_4_INTERRUPT_M2F
WDOG_3_INTERRUPT_M2F
WDOG_2_INTERRUPT_M2F
WDOG_1_INTERRUPT_M2F
WDOG_0_INTERRUPT_M2F
WDOG_RESET_OCCURRED_M2F
RESET
MSS_RESET_N_M2F
CLOCK_IN
REFCLK
REFCLK_N
RESET_IN
MSS_RESET_N_F2M
REFCLK
REFCLK_N
mss_module

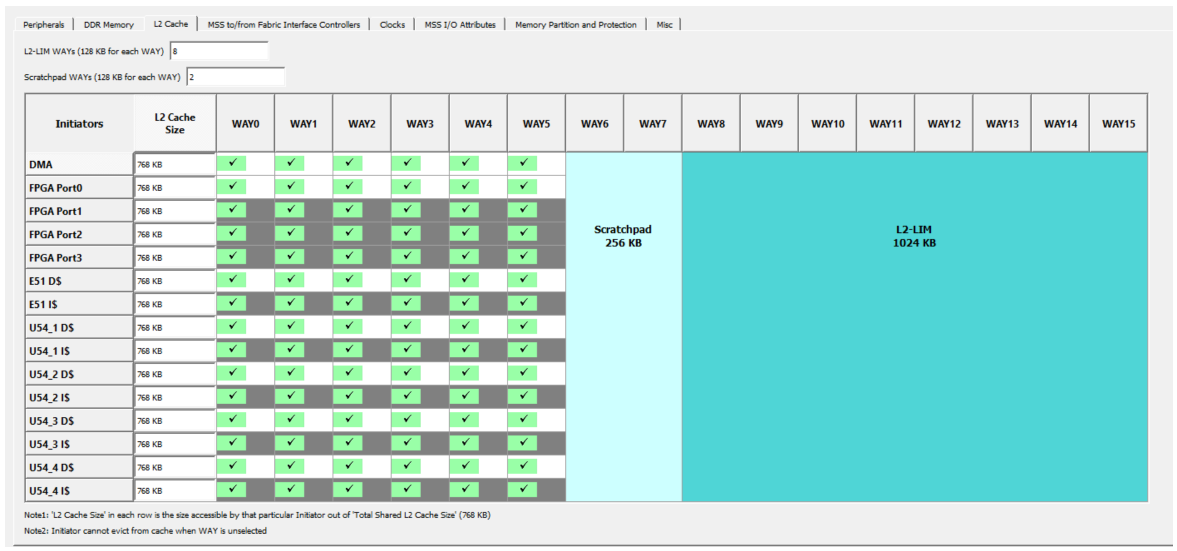
Important: In a Libero project, when System Controller Suspend Mode is enabled (**Project** > **Project Settings** > **Device settings**), the `PFSOC_SCSM` macro must be instantiated in the user design and the `REBOOT_REQUESTED_M2F` pin of MSS must be connected to the `SC_WAKE` pin of `PFSOC_SCSM` macro to wake up the system controller temporarily so it can reboot the MSS during normal operation.

2.3.7. L2 Cache (Ask a Question)

Level2 memory subsystem has three operating modes – Cache, Loosely-Integrated-Memory (LIM), and Scratchpad. These modes can be configured in MSS configurator using the options in the L2 Cache tab based on user needs. The goal is to make the configuration options easier to use and understand for the users.

You can allocate L2 memory for the processor or peripheral using the **L2 Cache** tab, as shown in the following figure.

Figure 2-28. L2 Cache


Peripherals
DDR Memory
L2 Cache
MSS to/from Fabric Interface Controllers
Clocks
MSS I/O Attributes
Memory Partition and Protection
Misc

L2-LIM WAYs (128 KB for each WAY) 8

Scratchpad WAYs (128 KB for each WAY) 2

Initiators	L2 Cache Size	WAY0	WAY1	WAY2	WAY3	WAY4	WAY5	WAY6	WAY7	WAY8	WAY9	WAY10	WAY11	WAY12	WAY13	WAY14	WAY15
DMA	768 KB	✓	✓	✓	✓	✓	✓	Scratchpad 256 KB		L2-LIM 1024 KB							
FPGA Port0	768 KB	✓	✓	✓	✓	✓	✓										
FPGA Port1	768 KB	✓	✓	✓	✓	✓	✓										
FPGA Port2	768 KB	✓	✓	✓	✓	✓	✓										
FPGA Port3	768 KB	✓	✓	✓	✓	✓	✓										
E51 D$	768 KB	✓	✓	✓	✓	✓	✓										
E51 I$	768 KB	✓	✓	✓	✓	✓	✓										
U54_1 D$	768 KB	✓	✓	✓	✓	✓	✓										
U54_1 I$	768 KB	✓	✓	✓	✓	✓	✓										
U54_2 D$	768 KB	✓	✓	✓	✓	✓	✓										
U54_2 I$	768 KB	✓	✓	✓	✓	✓	✓										
U54_3 D$	768 KB	✓	✓	✓	✓	✓	✓										
U54_3 I$	768 KB	✓	✓	✓	✓	✓	✓										
U54_4 D$	768 KB	✓	✓	✓	✓	✓	✓										
U54_4 I$	768 KB	✓	✓	✓	✓	✓	✓										

Note1: 'L2 Cache Size' in each row is the size accessible by that particular Initiator out of 'Total Shared L2 Cache Size' (768 KB)

Note2: Initiator cannot evict from cache when WAY is unselected

In the L2 Cache tab:

- There are 16 WAYs. WAY0 is always allocated for Cache.
- In the GUI, the default L2-LIM size will be set at 15 (WAY1 – WAY15). This means that 1 WAY (128 Kbytes) is configured as L2 cache and 1920 Kbytes is configured as LIM. User can increase or decrease the L2-LIM size to configure the LIM as Cache memory for various processors and peripherals.
- In the GUI, all the WAYs are enabled for Cache by default. The user can disable the selection to allocate it for Scratchpad.
- Cache size shows the amount of memory available and is shared among all processors and peripherals.

Important:
- All AXI4 front way ports must be identical.
- The core D and I Ways have to be identical and hence ways for port 1, 2, 3, and ways for core I are disabled for selection.

2.3.8. Crypto (Ask a Question)

The following table lists the crypto ownership modes.

Table 2-8. Crypto Ownership Modes

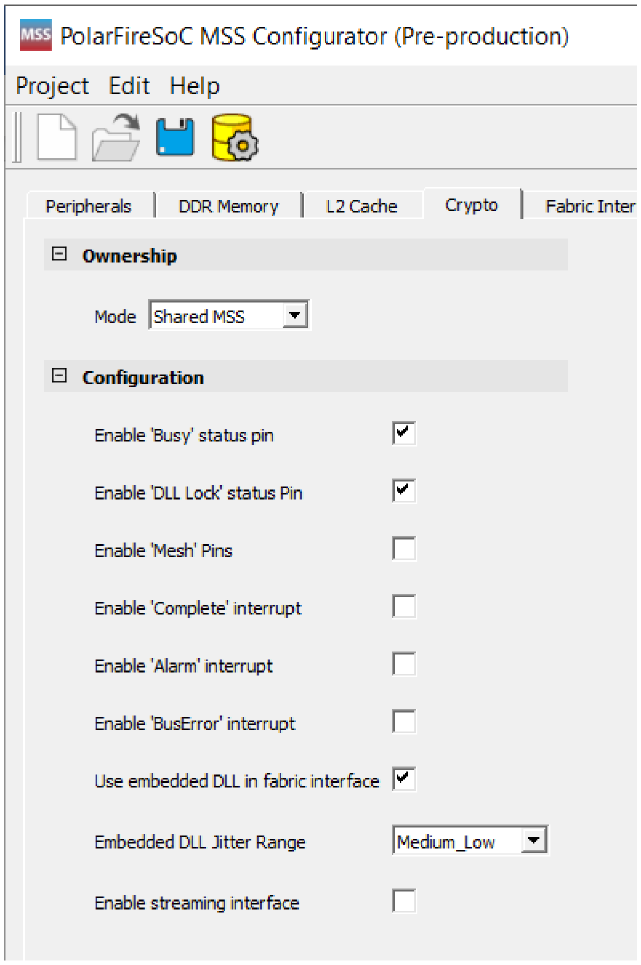
Owner	Description
MSS	The MSS owns the crypto.

Table 2-8. Crypto Ownership Modes (continued)

Owner	Description
Fabric	The Fabric owns the crypto and ports are exposed to the fabric.
Shared MSS	This is shared between the MSS and the Fabric. The MSS is the first owner. There are regular ports and other ports for handshaking to switch the ownership.
Shared Fabric	This is shared between the MSS and the Fabric. The Fabric is the first owner. There are regular ports and other ports for handshaking and to switch ownership.

Note: Streaming Interface is not available for Fabric mode.

Figure 2-29. Crypto Tab



Ownership

Mode: Shared MSS

Configuration

Enable 'Busy' status pin	✓
Enable 'DLL Lock' status Pin	✓
Enable 'Mesh' Pins	☐
Enable 'Complete' interrupt	☐
Enable 'Alarm' interrupt	☐
Enable 'BusError' interrupt	☐
Use embedded DLL in fabric interface	✓
Embedded DLL Jitter Range	Medium_Low
Enable streaming interface	☐

In the crypto modes:

- The status pins and interrupt pins are available as configuration options in all the ownership modes except the MSS ownership mode.
- There are three options that expose the DLL lock, Busy, and Mesh ports. The Busy and DLL Lock are ON by default, while the Mesh input pin connects to 0 when not used.
- The **Use embedded DLL in fabric interface** is always provided in all modes and is enabled by default when the **Enable streaming interface** option is selected.

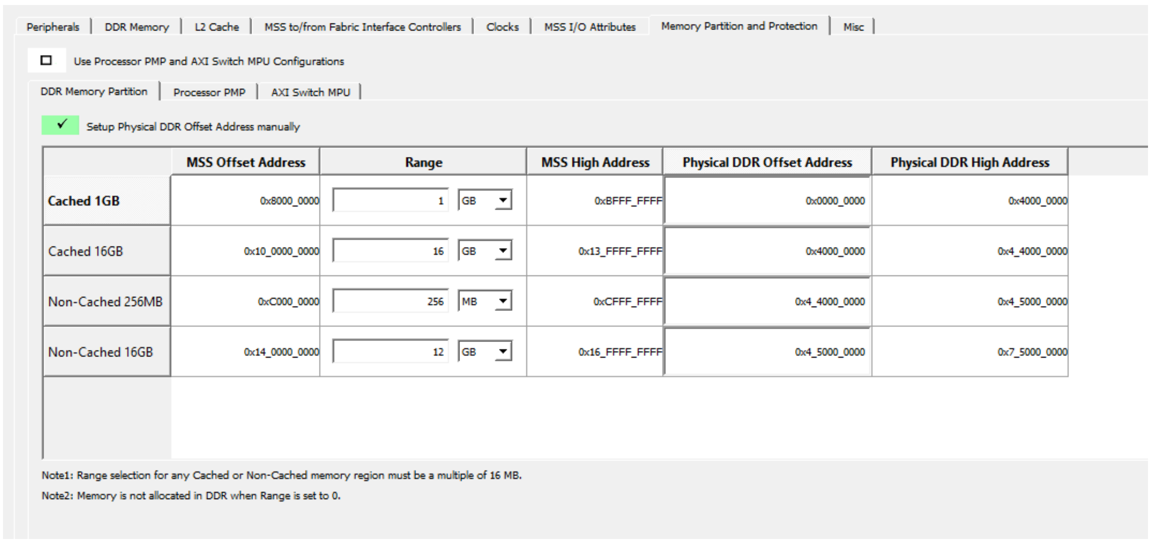
2.3.9. Memory Partition and Protection (Ask a Question)

The Physical Memory Protection (PMP) prevents a process (running on a RISC-V Processor) or an initiator (FPGA Fabric) from accessing memory that has not been allocated to it. RISC-V system has PMP unit, which provides control registers for each processor to allow physical memory access privileges (read, write, execute) to be specified for each physical memory region. Similarly, the AXI Switch has Memory Protection Unit (MPU) block which provides register control to setup memory access regions for FPGA initiators.

DDR Memory Partition

The DDR Memory Partition tab allows the DDR memory to be allocated to cached and non-cached regions depending on the amount of DDR memory physically connected.

Figure 2-30. DDR Memory Partition Tab



☐ Use Processor PMP and AXI Switch MPU Configurations

✓ Setup Physical DDR Offset Address manually

	MSS Offset Address	Range	MSS High Address	Physical DDR Offset Address	Physical DDR High Address
Cached 1GB	0x8000_0000	1 GB	0xBFFF_FFFF	0x0000_0000	0x4000_0000
Cached 16GB	0x10_0000_0000	16 GB	0x13_FFFF_FFFF	0x4000_0000	0x4_4000_0000
Non-Cached 256MB	0xC000_0000	256 MB	0xCFFF_FFFF	0x4_4000_0000	0x4_5000_0000
Non-Cached 16GB	0x14_0000_0000	12 GB	0x16_FFFF_FFFF	0x4_5000_0000	0x7_5000_0000

Note1: Range selection for any Cached or Non-Cached memory region must be a multiple of 16 MB.
Note2: Memory is not allocated in DDR when Range is set to 0.

In the DDR Memory Partition:

- The Offset Address (Base Address) for both cached and non-cached regions is fixed.
- High Address is the End Address based on the size.
- Users are expected to enter the Range. Based on the Range selection, High Address and Physical DDR Offset is updated.
  - When Range is set to zero, memory is not allocated in DDR.
  - When Range is set to a nonzero value, it must be a multiple of 16 MB.
- Physical DDR Offset is allocation of DDR memory (connected to the FPGA system) based on the nonzero Range value.

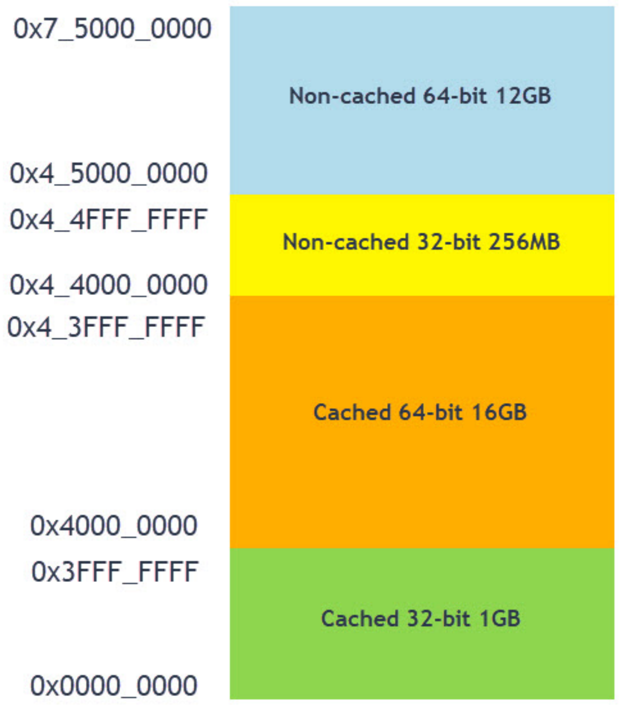
Important: When **Setup Physical DDR Address manually** option is enabled, the default Physical DDR Offset Address is set as 0x0000_0000 for all the regions and it is up to the user to change this address (when the size of the allocated memory region is greater than 0 bytes). When the size of allocated memory region is zero bytes, the offset and high address are reported as **N/A** and remains as a read-only field.

- Range is allocated sequentially starting with 0x0000_0000 in the following order:
  - Cached 32-bit
  - Cached 64-bit

- Non-Cached 32-bit
- Non-Cached 64-bit

The following figure shows the graphical representation of how the Physical DDR Offset allocation is done based on the preceding ranges:

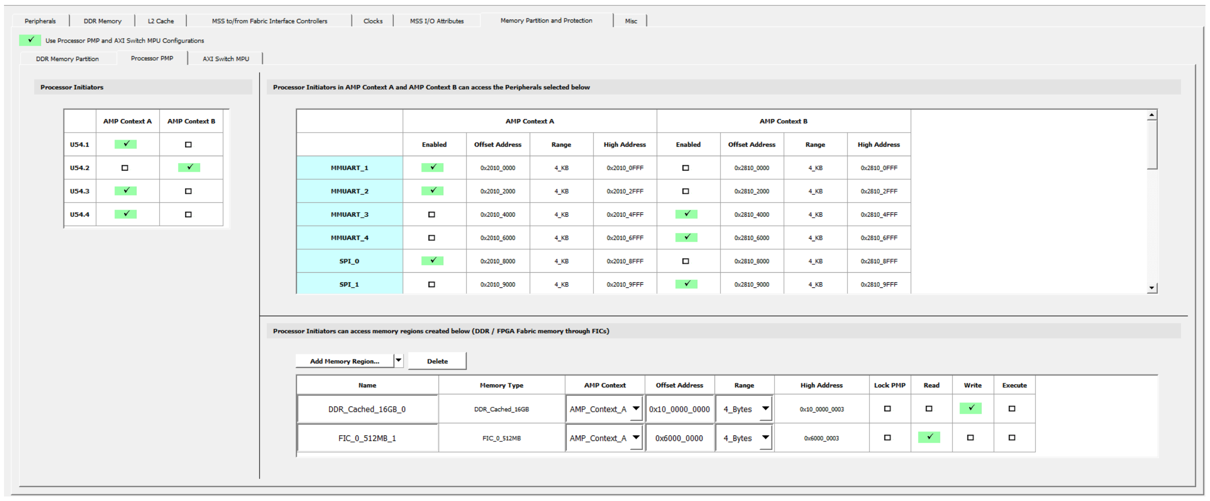
Figure 2-31. Physical DDR Offset — Graphical Representation



Processor PMP

The following figure shows the Processor PMP tab.

Figure 2-32. Processor PMP Tab



✓ Use Processor PMP and AXI Switch MPU Configurations

Processor Initiators

	AMP Context A	AMP Context B
U54.1	✓	☐
U54.2	☐	✓
U54.3	✓	☐
U54.4	✓	☐

Processor Initiators in AMP Context A and AMP Context B can access the Peripherals selected below

	AMP Context A				AMP Context B			
	Enabled	Offset Address	Range	High Address	Enabled	Offset Address	Range	High Address
MMUART_1	✓	0x2010_0000	4_KB	0x2010_0FFF	☐	0x2810_0000	4_KB	0x2810_0FFF
MMUART_2	✓	0x2010_2000	4_KB	0x2010_2FFF	☐	0x2810_2000	4_KB	0x2810_2FFF
MMUART_3	☐	0x2010_4000	4_KB	0x2010_4FFF	✓	0x2810_4000	4_KB	0x2810_4FFF
MMUART_4	☐	0x2010_6000	4_KB	0x2010_6FFF	✓	0x2810_6000	4_KB	0x2810_6FFF
SPI_0	✓	0x2010_8000	4_KB	0x2010_8FFF	☐	0x2810_8000	4_KB	0x2810_8FFF
SPI_1	☐	0x2010_9000	4_KB	0x2010_9FFF	✓	0x2810_9000	4_KB	0x2810_9FFF

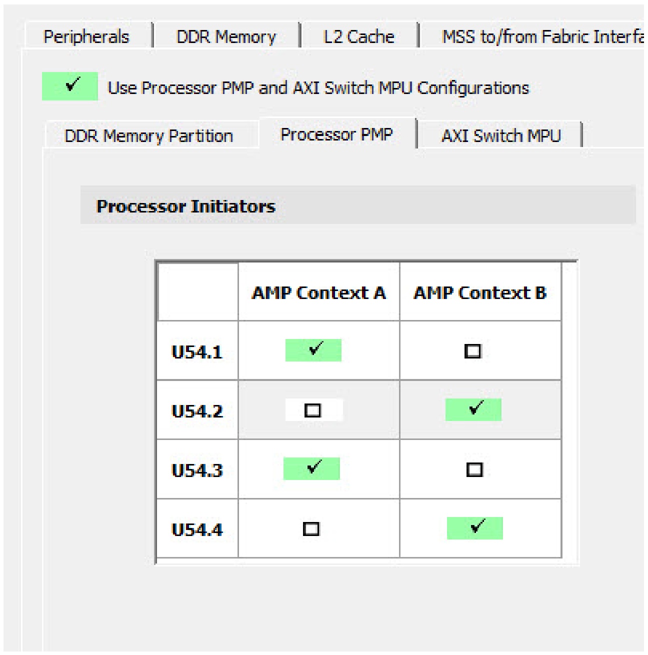
Processor Initiators can access memory regions created below (DDR / FPGA Fabric memory through FICs)

Name	Memory Type	AMP Context	Offset Address	Range	High Address	Lock PMP	Read	Write	Execute
DDR_Cached_16GB_0	DDR_Cached_16GB	AMP_Context_A	0x10_0000_0000	4_Bytes	0x10_0000_0003	☐	☐	✓	☐
FIC_0_512MB_1	FIC_0_512MB	AMP_Context_A	0x6000_0000	4_Bytes	0x6000_0003	☐	✓	☐	☐

Processor Initiators

Four CPU initiators can be enabled in one of the two Asymmetric Multi Processing (AMP) contexts. CPU initiators can access both CPU and Peripherals.

Figure 2-33. Processor Initiators



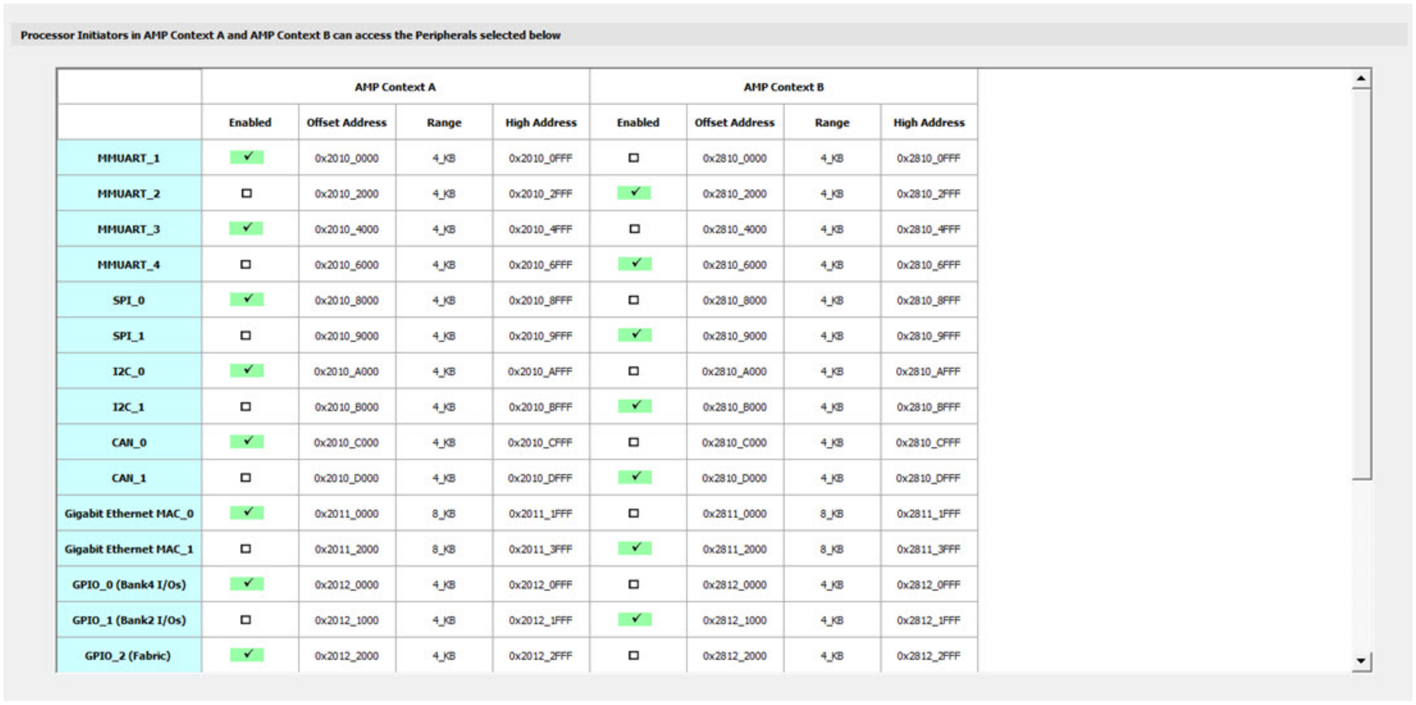
✓ Use Processor PMP and AXI Switch MPU Configurations

Processor Initiators

	AMP Context A	AMP Context B
U54.1	✓	☐
U54.2	☐	✓
U54.3	✓	☐
U54.4	☐	✓

Processor Initiators in Context A and Context B can access the following selected Peripherals
The AMP Context A is assigned to AHB0 bus interface and AMP Context B is assigned to AHB1 bus interface. Peripherals can be enabled in either Context A or in Context B. The range and base address for the peripherals are populated in Graphical User Interface and are not editable. The base address will be different for peripherals that are on dual AHB bus interfaces for Context A and Context B.

Figure 2-34. Processor Initiators in Context A and Context B Accessing the Peripherals

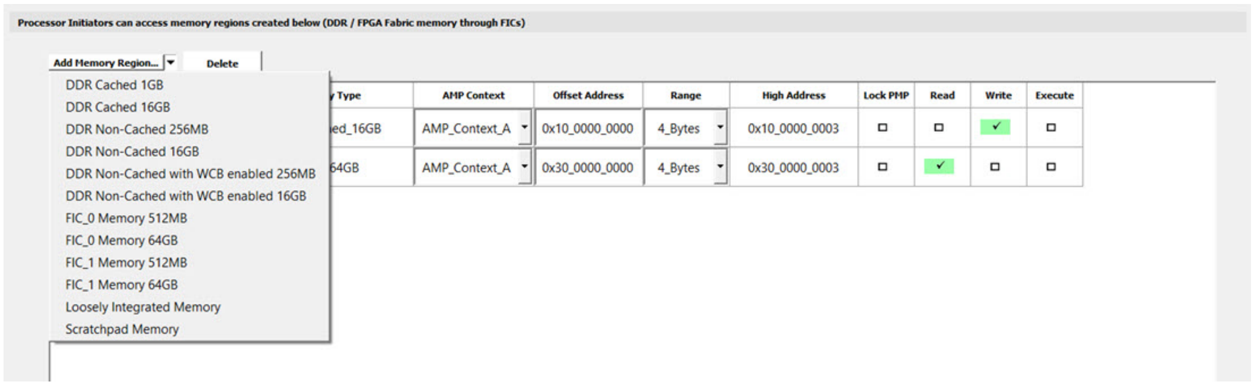


Processor Initiators in AMP Context A and AMP Context B can access the Peripherals selected below

	AMP Context A				AMP Context B			
	Enabled	Offset Address	Range	High Address	Enabled	Offset Address	Range	High Address
MMUART_1	✓	0x2010_0000	4_KB	0x2010_0FFF	☐	0x2810_0000	4_KB	0x2810_0FFF
MMUART_2	☐	0x2010_2000	4_KB	0x2010_2FFF	✓	0x2810_2000	4_KB	0x2810_2FFF
MMUART_3	✓	0x2010_4000	4_KB	0x2010_4FFF	☐	0x2810_4000	4_KB	0x2810_4FFF
MMUART_4	☐	0x2010_6000	4_KB	0x2010_6FFF	✓	0x2810_6000	4_KB	0x2810_6FFF
SPI_0	✓	0x2010_8000	4_KB	0x2010_8FFF	☐	0x2810_8000	4_KB	0x2810_8FFF
SPI_1	☐	0x2010_9000	4_KB	0x2010_9FFF	✓	0x2810_9000	4_KB	0x2810_9FFF
I2C_0	✓	0x2010_A000	4_KB	0x2010_AFFF	☐	0x2810_A000	4_KB	0x2810_AFFF
I2C_1	☐	0x2010_B000	4_KB	0x2010_BFFF	✓	0x2810_B000	4_KB	0x2810_BFFF
CAN_0	✓	0x2010_C000	4_KB	0x2010_CFFF	☐	0x2810_C000	4_KB	0x2810_CFFF
CAN_1	☐	0x2010_D000	4_KB	0x2010_DFFF	✓	0x2810_D000	4_KB	0x2810_DFFF
Gigabit Ethernet MAC_0	✓	0x2011_0000	8_KB	0x2011_1FFF	☐	0x2811_0000	8_KB	0x2811_1FFF
Gigabit Ethernet MAC_1	☐	0x2011_2000	8_KB	0x2011_3FFF	✓	0x2811_2000	8_KB	0x2811_3FFF
GPIO_0 (Bank4 I/Os)	✓	0x2012_0000	4_KB	0x2012_0FFF	☐	0x2812_0000	4_KB	0x2812_0FFF
GPIO_1 (Bank2 I/Os)	☐	0x2012_1000	4_KB	0x2012_1FFF	✓	0x2812_1000	4_KB	0x2812_1FFF
GPIO_2 (Fabric)	✓	0x2012_2000	4_KB	0x2012_2FFF	☐	0x2812_2000	4_KB	0x2812_2FFF

Processor Initiators can access memory regions created below (DDR/FPGA Fabric memory through FICs)
DDR memory appears at several address ranges depending on whether it is cached, non-cached, or access through a Write Combine Buffer (WCB). WCB improves performance (by combining multiple writes to the same address into a single write) for sequential accesses. Each AMP context needs to specify how much DDR memory of each type it needs. Some DDR memories may be shared between AMP Context to pass data. User can create protection for memory region accessed by processors by clicking on the **Add Memory Region...** button. User can delete any memory region by selecting the memory region and clicking **Delete** button. The following figure shows the memory regions available to Processor Initiators.

Figure 2-35. Memory Regions Available to Processor Initiators



Processor Initiators can access memory regions created below (DDR / FPGA Fabric memory through FICs)

y Type	AMP Context	Offset Address	Range	High Address	Lock PMP	Read	Write	Execute
ed_16GB	AMP_Context_A	0x10_0000_0000	4_Bytes	0x10_0000_0003	☐	☐	✓	☐
64GB	AMP_Context_A	0x30_0000_0000	4_Bytes	0x30_0000_0003	☐	✓	☐	☐

The following table lists different types of memory regions that user can create.

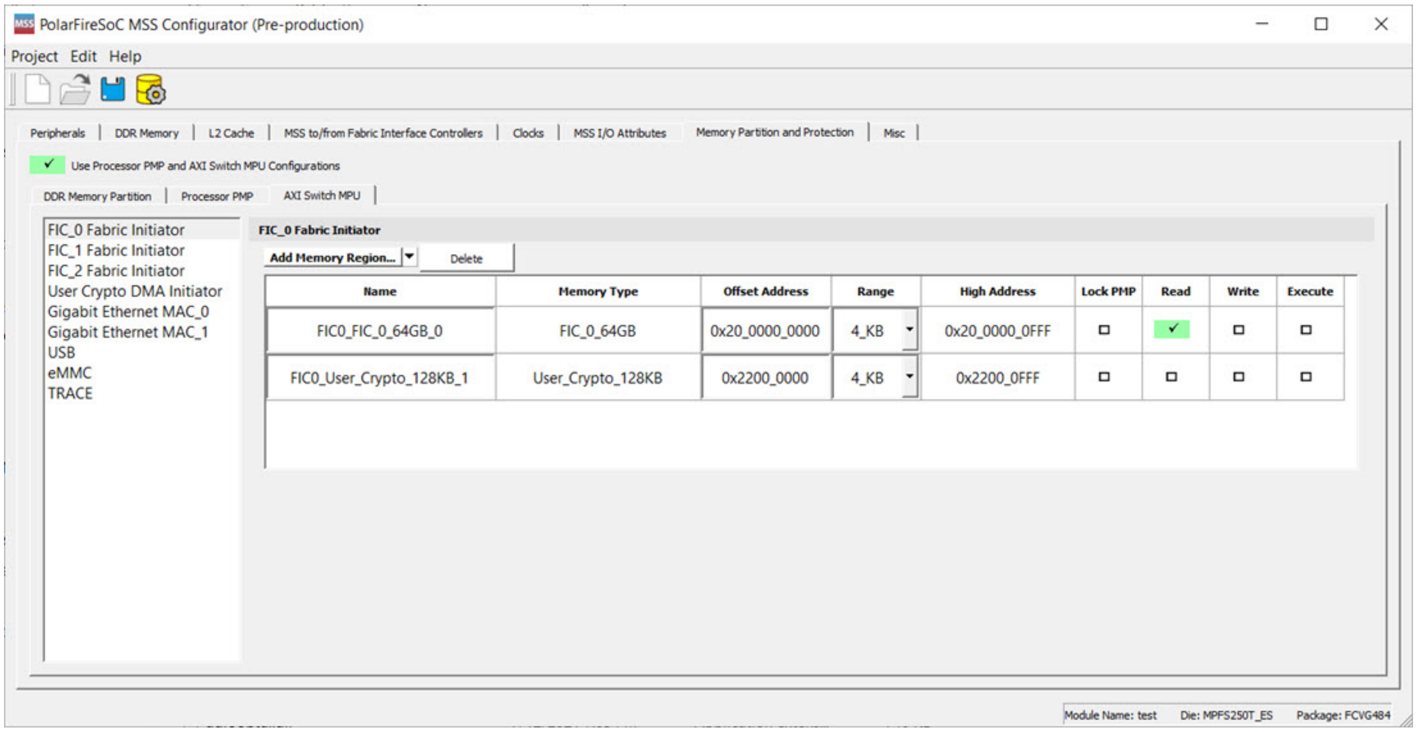
Table 2-9. Available Memory Region

Memory Type	Memory Size	Address Size	Address Range
DDR Cached	512 MB	32-bit	0x80000000 - 0xBFFFFFFF
	16 GB	64-bit	0x10_00000000 - 0x13_FFFFFFFF
DDR Non-Cached	256 MB	32-bit	0xC0000000 - 0xCFFFFFFF
	16 GB	64-bit	0x14_00000000 - 0x17_FFFFFFFF
DDR Non-Cached with WCB enabled	256 MB	32-bit	0xC0000000 - 0xCFFFFFFF
	16 GB	64-bit	0x14_00000000 - 0x17_FFFFFFFF
FIC_0 Memory	512 MB	32-bit	0x60000000 - 0x7FFFFFFF
	64 GB	64-bit	0x20_00000000 - 0x2F_FFFFFFFF
FIC_1 Memory	512 MB	32-bit	0xE0000000 - 0xFFFFFFFF
	64 GB	64-bit	0x30_00000000 - 0x3F_FFFFFFFF
Loosely Integrated Memory	—	—	Start Address is 0x0800_0000
Scratch	—	—	Start Address is 0x0A00_0000

AXI Switch MPU

AXI switch Memory Protection Unit (MPU) provides FPGA (Non-CPU) Initiators read/write/execute access to the Memory subsystem and Fabric Memory. Users can create protection for memory regions accessed by FPGA Initiators by clicking any one of the FPGA Initiators and clicking the **Add Memory Region...** button. User can delete any memory region by selecting the memory region and clicking **Delete** button.

Figure 2-36. AXI Switch MPU Tab



✓ Use Processor PMP and AXI Switch MPU Configurations

FIC_0 Fabric Initiator

Name	Memory Type	Offset Address	Range	High Address	Lock PMP	Read	Write	Execute
FIC0_FIC_0_64GB_0	FIC_0_64GB	0x20_0000_0000	4_KB	0x20_0000_0FFF	☐	✓	☐	☐
FIC0_User_Crypto_128KB_1	User_Crypto_128KB	0x2200_0000	4_KB	0x2200_0FFF	☐	☐	☐	☐

The following table lists the different types of memory regions that user can create.

Memory Type	Memory Size	Address Size	Address Range
DDR Non-Cached	256 MB	32-bit	0xC0000000 - 0xCFFFFFFF
	16 GB	64-bit	0x14_00000000 - 0x17_FFFFFFFF

Memory Partition and Protection (continued)

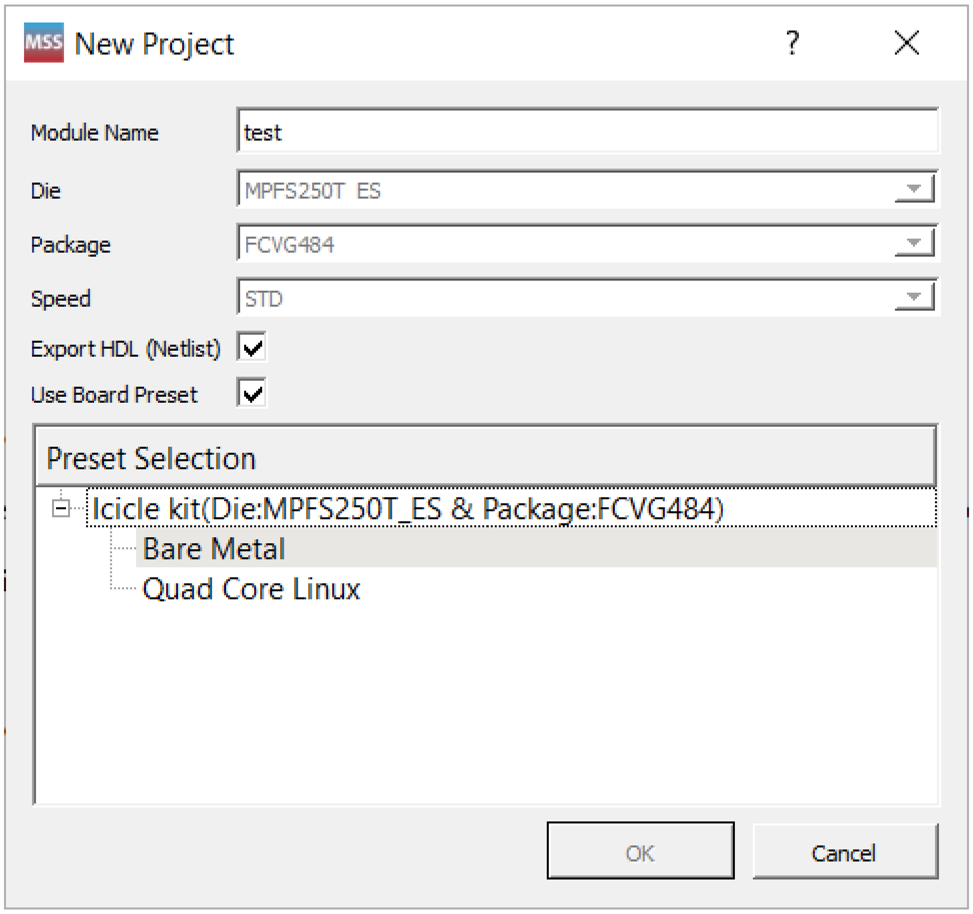
Memory Type	Memory Size	Address Size	Address Range
FIC_0 Memory	512 MB	32-bit	0x60000000 - 0x7FFFFFFF
	64 GB	64-bit	0x20_00000000 - 0x2F_FFFFFFFF
FIC_1 Memory	512 MB	32-bit	0xE0000000 - 0xFFFFFFFF
	64 GB	64-bit	0x30_00000000 - 0x3F_FFFFFFFF
FIC_3 Memory	512 MB	32-bit	0x40000000 - 0x5FFFFFFF
User Crypto Memory	128 KB	32-bit	0x22000000 - 0x2201FFFF

3. Creating a Project and Configuring MSS (Ask a Question)

To create a project and configure MSS:

1. Launch the PolarFire SoC MSS Configurator (`pfsoc_mss`) using one of the following ways:
   - Libero SoC installation directory
   - Standalone MSS installation area
   - Windows Start menu
2. Create a new project using **Project > New**.
3. Enter a module name (for example, `PFSOC_MSS_C0`), and then select the appropriate die, package, and speed. The module name you enter appears in the following places:
   - File names of the PolarFire SoC MSS Configurator generated outputs at the specified output/generation directory
   - MSS component file (`<module_name>.cxz`)
   - MSS XML configuration file (`<module_name>_mss_cfg.xml`)
   - MSS configuration file corresponding to the current MSS configuration that is generated (`<module_name>.cfg`)
   - MSS configuration report file (`<module_name>_Report.html`)
   - Component/module name of the MSS component (`cxz`) that can be imported to a Libero SoC project

Figure 3-1. MSS — Module Name Dialog Box



Module Name: test
Die: MPFS250T_ES
Package: FCVG484
Speed: STD
Export HDL (Netlist): ✓
Use Board Preset: ✓

Icicle kit(Die:MPFS250T_ES & Package:FCVG484)
- Bare Metal
- Quad Core Linux

Presets for Icicle Kit

When you create a New Project, you can import one of the available presets for Icicle kit. These presets are staged in data folder and are in 'CFG' format. All the parameter/values from the CFG file are loaded, except for the following four parameters:

- Module Name
- Die
- Package
- Speed

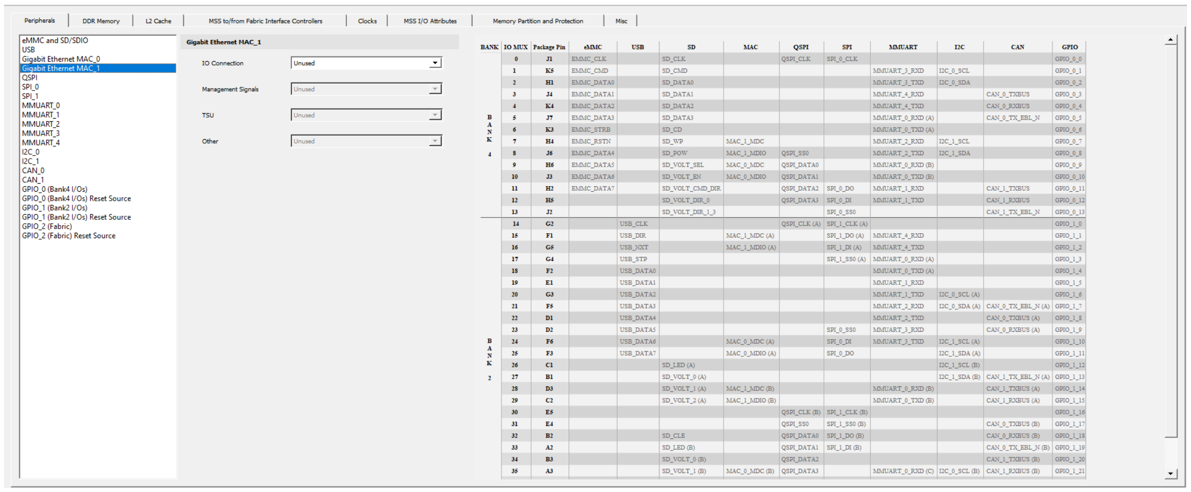
User-selected values are used for Module Name, Die, Package, and Speed parameters in the GUI. This is done to facilitate changing Module Name/Die/Package/Speed (from the Icicle kit preset) to meet users need.

The default speed grade is STD including for the Icicle Kits. If you open a previous release `.cfg`, which does not have speed grade (PolarFire SoC MSS Configurator v2021.2 or earlier) in v2021.3, it has STD as the default speed grade.

The list of presets are shown using a tree widget. By default, the 'Default Configuration' preset is selected in GUI and user must explicitly select one of the presets for Icicle kit to load them. Once a preset is selected, users cannot edit them using **Edit Settings** option as the preset tree widget is unavailable in the **Edit Settings** dialog box.

The following figure shows the MSS configurator tabs.

Figure 3-2. MSS Configurator Tabs



Gigabit Ethernet MAC_1

IO Connection: Unused
Management Signals: Unused
TSU: Unused
Other: Unused

BANK	IO MUX	Package Pin	eMMC	USB	SD	MAC	QSPI	SPI	MMUART	I2C	CAN	GPIO
BANK 4	0	J1	EMMC_CLK		SD_CLK		QSPI_CLK	SPI_0_CLK				GPIO_0_0
	1	K5	EMMC_CMD		SD_CMD				MMUART_3_RXD	I2C_0_SCL		GPIO_0_1
	2	H1	EMMC_DATA0		SD_DATA0				MMUART_3_TXD	I2C_0_SDA		GPIO_0_2
	3	J4	EMMC_DATA1		SD_DATA1				MMUART_4_RXD		CAN_0_TXBUS	GPIO_0_3
	4	K4	EMMC_DATA2		SD_DATA2				MMUART_4_TXD		CAN_0_RXBUS	GPIO_0_4
	5	J7	EMMC_DATA3		SD_DATA3				MMUART_0_RXD (A)		CAN_0_TX_EBL_N	GPIO_0_5
	6	K3	EMMC_STRB		SD_CD				MMUART_0_TXD (A)			GPIO_0_6
	7	H4	EMMC_RSTN		SD_WP	MAC_1_MDC			MMUART_2_RXD	I2C_1_SCL		GPIO_0_7
	8	J6	EMMC_DATA4		SD_POW	MAC_1_MDIO	QSPI_SS0		MMUART_2_TXD	I2C_1_SDA		GPIO_0_8
	9	H6	EMMC_DATA5		SD_VOLT_SEL	MAC_0_MDC	QSPI_DATA0		MMUART_0_RXD (B)			GPIO_0_9
	10	J3	EMMC_DATA6		SD_VOLT_EN	MAC_0_MDIO	QSPI_DATA1		MMUART_0_TXD (B)			GPIO_0_10
	11	H2	EMMC_DATA7		SD_VOLT_CMD_DIR		QSPI_DATA2	SPI_0_DO	MMUART_1_RXD		CAN_1_TXBUS	GPIO_0_11
	12	H5			SD_VOLT_DIR_0		QSPI_DATA3	SPI_0_DI	MMUART_1_TXD		CAN_1_RXBUS	GPIO_0_12
	13	J2			SD_VOLT_DIR_1_3			SPI_0_SS0			CAN_1_TX_EBL_N	GPIO_0_13
BANK 2	14	G2		USB_CLK			QSPI_CLK (A)	SPI_1_CLK (A)				GPIO_1_0
	15	F1		USB_DIR		MAC_1_MDC (A)		SPI_1_DO (A)	MMUART_4_RXD			GPIO_1_1
	16	G5		USB_NXT		MAC_1_MDIO (A)		SPI_1_DI (A)	MMUART_4_TXD			GPIO_1_2
	17	G4		USB_STP				SPI_1_SS0 (A)	MMUART_0_RXD (A)			GPIO_1_3
	18	F2		USB_DATA0					MMUART_0_TXD (A)			GPIO_1_4
	19	E1		USB_DATA1					MMUART_1_RXD			GPIO_1_5
	20	G3		USB_DATA2					MMUART_1_TXD	I2C_0_SCL (A)		GPIO_1_6
	21	F5		USB_DATA3					MMUART_2_RXD	I2C_0_SDA (A)	CAN_0_TX_EBL_N (A)	GPIO_1_7
	22	D1		USB_DATA4					MMUART_2_TXD		CAN_0_TXBUS (A)	GPIO_1_8
	23	D2		USB_DATA5				SPI_0_SS0	MMUART_3_RXD		CAN_0_RXBUS (A)	GPIO_1_9
	24	F6		USB_DATA6		MAC_0_MDC (A)		SPI_0_DI	MMUART_3_TXD	I2C_1_SCL (A)		GPIO_1_10
	25	F3		USB_DATA7		MAC_0_MDIO (A)		SPI_0_DO		I2C_1_SDA (A)		GPIO_1_11
	26	C1			SD_LED (A)					I2C_1_SCL (B)		GPIO_1_12
	27	B1			SD_VOLT_0 (A)					I2C_1_SDA (B)	CAN_1_TX_EBL_N (A)	GPIO_1_13
	28	D3			SD_VOLT_1 (A)	MAC_1_MDC (B)			MMUART_0_RXD (B)		CAN_1_TXBUS (A)	GPIO_1_14
	29	C2			SD_VOLT_2 (A)	MAC_1_MDIO (B)			MMUART_0_TXD (B)		CAN_1_RXBUS (A)	GPIO_1_15
	30	E5					QSPI_CLK (B)	SPI_1_CLK (B)				GPIO_1_16
	31	E4					QSPI_SS0	SPI_1_SS0 (B)			CAN_0_TXBUS (B)	GPIO_1_17
	32	B2			SD_CLE		QSPI_DATA0	SPI_1_DO (B)			CAN_0_RXBUS (B)	GPIO_1_18
	33	A2			SD_LED (B)		QSPI_DATA1	SPI_1_DI (B)			CAN_0_TX_EBL_N (B)	GPIO_1_19
	34	B3			SD_VOLT_0 (B)		QSPI_DATA2				CAN_1_TXBUS (B)	GPIO_1_20
	35	A3			SD_VOLT_1 (B)	MAC_0_MDC (B)	QSPI_DATA3		MMUART_0_RXD (C)	I2C_0_SCL (B)	CAN_1_RXBUS (B)	GPIO_1_21

4. Configure **Clocks**, **Fabric Interface Controllers**, **I/O Configuration**, **DDR Memory**, and **Misc settings**.
5. Click the **Save** option to save the MSS configuration to a `.cfg` file.
6. From the **Save MSS Configuration** dialog box:
   - Browse to a directory and create a folder. For example, create `C:\Microsemi\PFSOC_MSS_Configuration`.
   - Enter a file name (for example, `PFSOC_MSS_C0)` and click **Save**.
     Note: The file name you enter is for the stand-alone MSS project only and is not used as the component name.

The MSS Configuration is created and saved to the file specified and the Log window shows the following message: `INFO: Successfully saved MSS configuration in C:/Microsemi/PFSOC_MSS_Configuration/PFSOC_MSS_C0.cfg file`.

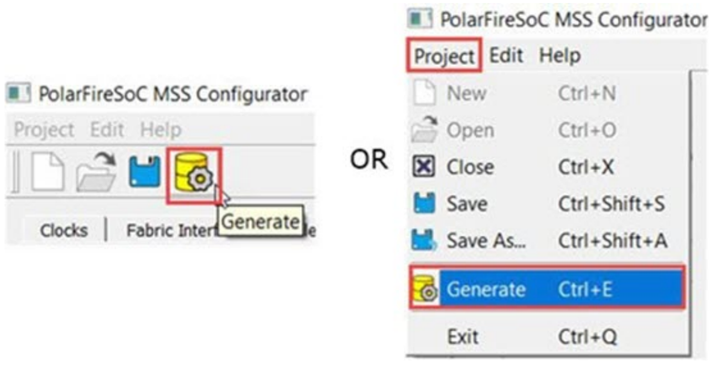
4. Generating, Importing, and Exporting the MSS Component (Ask a Question)

The following sections describe the steps for generating, importing, and exporting the MSS component.

4.1. Generating the MSS Component and Report (Ask a Question)

To generate the MSS component, use the **Generate** option (see the following figure).

Figure 4-1. Generate Option



The configuration file (`module_name.xml`) required for the firmware project and the configuration report file (`module_name.html`) are also generated at this time.

The Log window shows the following messages indicating the generated files:

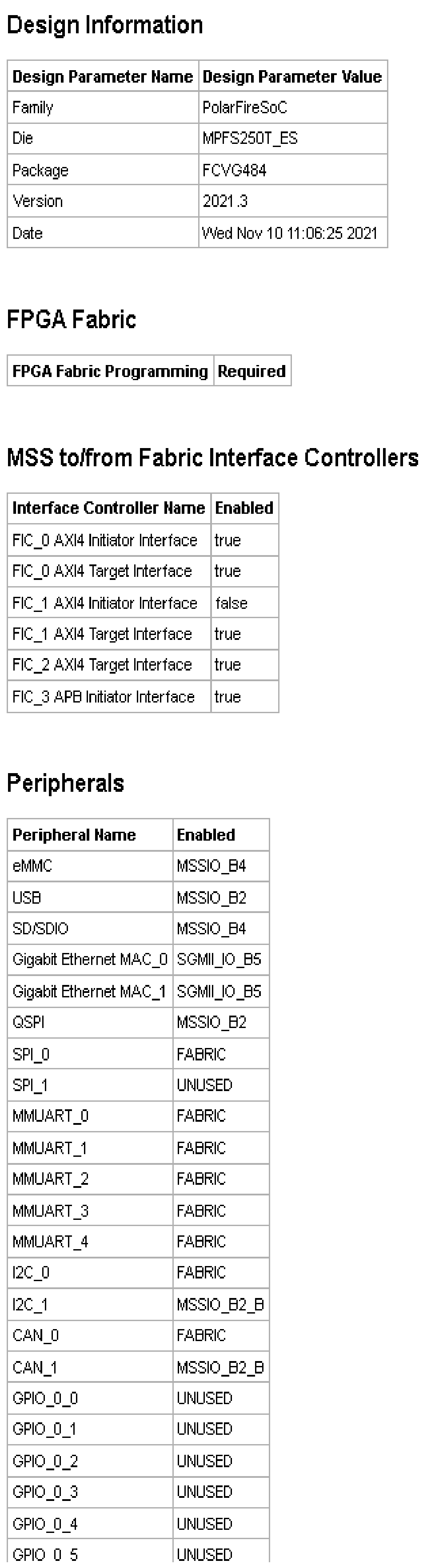
```
INFO: Successfully generated MSS configuration report to 'C:/Microsemi/
PFSOC_MSS_Configuration\PFSOC_MSS_C0_Report.html'
```

```
INFO: Successfully generated MSS component file to 'C:/Microsemi/PFSOC_MSS_Configuration/
PFSOC_MSS_C0.cxz'
```

The report file (`module_name.html`) consists of following sections:

- Design Information – This section consists of design parameters like device family name, die, package, configurator version, and the date the report was generated.
- FPGA Fabric – This section mentions whether FPGA Fabric programming is required or not.
- Fabric Interface Controllers – Consists information about status of the interface controllers.
- Peripherals – Contains information about which peripherals are being used or unused.
- DDR Memory – Shows the memory type.
- List of Ports – Depicts information about all the ports with direction.
- I/O REFCLK Port Settings – Shows all the information about Reference clock ports.
- MSSIO Port Settings – Shows all the information about MSSIO ports.
- DDRIO Port Settings – Shows all the information about DDRIO ports.
- SGMII I/O Port Settings – Shows all the information about SGMII ports.

Figure 4-2. Configuration Report for PolarFire SoC MSS Configurator



Design Information

Design Parameter Name	Design Parameter Value
Family	PolarFireSoC
Die	MPFS250T_ES
Package	FCVG484
Version	2021.3
Date	Wed Nov 10 11:06:25 2021

FPGA Fabric

FPGA Fabric Programming	Required

MSS to/from Fabric Interface Controllers

Interface Controller Name	Enabled
FIC_0 AXI4 Initiator Interface	true
FIC_0 AXI4 Target Interface	true
FIC_1 AXI4 Initiator Interface	false
FIC_1 AXI4 Target Interface	true
FIC_2 AXI4 Target Interface	true
FIC_3 APB Initiator Interface	true

Peripherals

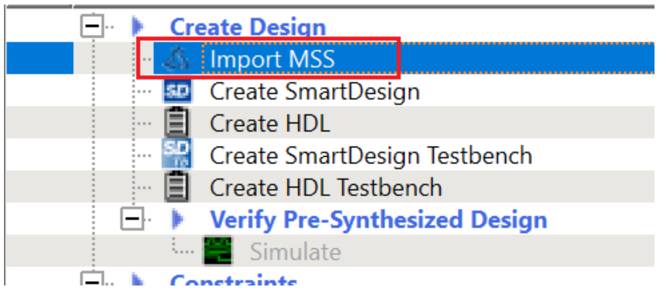
Peripheral Name	Enabled
eMMC	MSSIO_B4
USB	MSSIO_B2
SD/SDIO	MSSIO_B4
Gigabit Ethernet MAC_0	SGMII_IO_B5
Gigabit Ethernet MAC_1	SGMII_IO_B5
QSPI	MSSIO_B2
SPI_0	FABRIC
SPI_1	UNUSED
MMUART_0	FABRIC
MMUART_1	FABRIC
MMUART_2	FABRIC
MMUART_3	FABRIC
MMUART_4	FABRIC
I2C_0	FABRIC
I2C_1	MSSIO_B2_B
CAN_0	FABRIC
CAN_1	MSSIO_B2_B
GPIO_0_0	UNUSED
GPIO_0_1	UNUSED
GPIO_0_2	UNUSED
GPIO_0_3	UNUSED
GPIO_0_4	UNUSED
GPIO_0_5	UNUSED

4.2. Importing the MSS CXZ File to Libero SoC (Ask a Question)

To import the `PFSOC_MSS_C0.cxz` file:

1. Use the **Import MSS** option shown in the following figure.

Figure 4-3. Import MSS to Libero



2. From Design Hierarchy, drag the MSS component to SmartDesign canvas.
3. Build the hierarchy.

Note: Any changes required in the MSS configuration must be performed in the PolarFire SoC MSS Configurator, and the updated `MSS CXZ` file must be re-imported and used in Libero SmartDesign.

4.3. Importing the MSS XML File to SoftConsole (Ask a Question)

Copy the XML file from:

```
<$Directory>:/Microsemi/PFSOC_MSS_Configuration/PFSOC_MSS_C0_mss_cfg.xml
```

to:

```
<$Installation
Directory>:\Microchip\<$SoftConsole_Workspace>\Project_Name\src\platform\config\xml
```

Note: This step can also be performed using the **Import** option from SoftConsole.

4.4. Exporting the FPGA Design Hardware Platform Information (Ask a Question)

When using PolarFire SoC, the overall application runs an embedded software application on the RISC-V cores that may use the FPGA fabric to expand the number of I/O peripherals, accelerate software functions using FPGA logic, or control FPGA fabric functions. In these cases, the processor communicates with the FPGA fabric via the MSS Fabric Interface Controllers (FIC) and interrupt ports. The embedded software application must contain the following information to establish this communication properly:

- Fabric blocks like LSRAM, DMA Controller, and PCIe are connected to the AXI interconnect IP on the Fabric side. MSS communicates with these fabric blocks via Fabric Interface Controllers, which connects to the AXI Interconnect IP. The memory addresses of these fabric blocks are specified in the AXI Interconnect IP Configurator. These memory addresses must be specified in the software application.
- In the Libero SoC design, the user must drive the required MSS interrupt ports and other interrupts can be grounded. The corresponding Interrupt Request (IRQ) handler routines must be invoked in the software application for interrupt handling.

Libero SoC tool does not export the FPGA fabric peripheral memory map, interrupt mapping, or peripheral clock frequencies. Therefore, add this information manually in your embedded software projects. For example, if fabric blocks such as LSRAM and DMA Controller are used in the design and interfaced with the MSS through a FIC, then the memory addresses of these fabric blocks must be specified in the user application code for accessing them from MSS.

5. Simulating an FPGA Design Interacting with MSS (Ask a Question)

The MSS simulation model has been designed to verify that the connectivity to the MSS has been properly established with the FPGA fabric logic.

The MSS simulation model can be used to verify:

- The Fabric—MSS connectivity using the Fabric Interface Controllers (FICs).
- The Fabric—MSS interrupt (M2F and F2M) interface.

For information about how to set up and run the simulation for the PolarFire SoC MSS, see MSS Simulation User Guide for PolarFire SoC.

6. Programming the Application Bitstream (Ask a Question)

To program the application bit stream:

1. Use Libero SoC or FlashProExpress to program the FPGA fabric array, sNVM, eNVM and any security settings.
2. Use Libero SoC to program any eNVM client.

Note: SoftConsole must be used to program the Boot mode.

Alternatively, you can also perform the following steps:

1. Use SoftConsole to program the eNVM with the First Stage Boot image.
   a. Select the project you want programmed to eNVM in SoftConsole’s **Project Explorer** pane.
   b. Click the **PolarFire SoC Boot Mode 1** external tool.

Programming progress messages appear in the Console.

For more information, see PolarFire SoC Software Development and Tool Flow User Guide.

7. Sample Project (Ask a Question)

For PolarFire SoC Icicle Kit reference design and supporting files, see PolarFire SoC Icicle Kit Reference Design GitHub repository.

8. References (Ask a Question)

See the following reference documents for further details:

- Configuration of the MSS clocks.
  - For detailed information about the MSS clocking features, see PolarFire Family Clocking Resources User Guide.
- Configuration of the MSS interfaces to the FPGA fabric.
  - For detailed information about the MSS Fabric Interface Controller (FIC) features, see PolarFire SoC FPGA MSS Technical Reference Manual.
- Selection and assignment of the MSS peripherals to the MSS dedicated I/Os and/or the FPGA fabric dedicated peripheral interfaces.
  - For detailed information about the MSS Peripherals features, see PolarFire SoC FPGA MSS Technical Reference Manual.
- Configuration of the MSS Bank voltages and I/O standards and attributes.
  - For detailed information about the MSS Banks and I/Os features, see PolarFire FPGA and PolarFire SoC FPGA User I/O User Guide.
- Configuration of the MSS DDR memories (DDR3/L, DDR4, LPDDR3, and LPDDR4).
  - For detailed information about the MSS DDR4, DDR3, LPDDR3, and LPDDR4 features, see PolarFire Family Memory Controller User Guide.
- Configuration of the MSS debug features.
  - For detailed information about the MSS debug features, see PolarFire SoC FPGA MSS Technical Reference Manual.

9. Revision History (Ask a Question)

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

Revision	Date	Description
P	05/2025	The following change is made in this revision. • Added definitions for `Boot_fail_error_M2F` and `Boot_Fail_Clear_F2M` signals in the Misc section.
N	08/2024	This document is released with Libero SoC Design Suite v2024.2 without changes from v2024.1.
M	02/2024	This document is released with Libero SoC Design Suite v2024.1 without changes from v2023.2.
L	08/2023	The following change is made in this revision. • Added new section I2C Port Configuration for Fabric I/O.
K	04/2023	The following change is made in this revision. • Updated section Clocks.
J	12/2022	The following list of changes are made in this revision. • Updated section MSS To/From Fabric Interface Controllers. • Updated section Peripherals. • Updated section Bank2 I/Os. • Updated section Bank4 I/Os. • Updated section Bank5 I/Os (REFCLK and SGMII). • Updated section Bank2 and Bank4 I/Os Related to SD and eMMC Muxing. • Updated section DDR Memory. • Updated section Misc. • Updated section L2 Cache. • Updated section Memory Partition and Protection. • Updated section Creating a Project and Configuring MSS.
H	08/2022	The following list of changes are made in this revision. • Updated the Batch Mode section. • Updated the Interactive Mode section. • Updated the Bank2 I/Os section. • Updated the Generating the MSS Component and Report section.
G	04/2022	The following list of changes are made in this revision. • Updated Clocks with support for -1 speed grade devices. • Added information related to Feedback ports in Misc.
F	12/2021	The following list of changes are made in this revision. • Updated Creating a Project and Configuring MSS. • Removed note from Batch Mode. • Added information related to Locked Down I/Os in Misc.
E	08/2021	The following list of changes are made in this revision. • References: Updated the MSS Fabric Interface Controller features reference.

Revision History (continued)

Revision	Date	Description
D	08/2021	The following list of changes are made in this revision. • Clocks: Updated the Clocks Tab with addition of eMMC/SD/SDIO and CAN clock sources. • MSS To/From Fabric Interface Controllers: Fabric Interface Controller tab was renamed to 'MSS to/from Fabric Interface Controllers'. • Peripherals: Added information about eMMC or SD/SDIO setting. • MSS I/O Attributes: MSS REFCLK I/O, Bank4 and Bank2 I/Os and SGMII I/Os were moved to MSS I/O Attributes. • Misc: Updated the section with Feedback Ports option.
C	04/2021	The following list of changes are made in this revision. • Peripherals: Added note related to the availability of SD pins. • Crypto: Updated the section with Streaming Interface Option details. • Memory Partition and Protection: Added this section.
B	12/2020	Revision B is released in December 2020. Following is a list of changes made in this revision: • Updated the Using the PolarFire SoC MSS Configurator GUI section. • Updated the Creating a Project and Configuring MSS section. • Updated the Generating the MSS Component and Report section. • Updated the Clocks section. • Added a note in the SGMII section.
A	9/2020	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

Trademarks

ISBN: 979-8-3371-1225-1

Legal Notice

Microchip Devices Code Protection Feature

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

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