SmartFusion 2 and IGLOO 2 FPGA High-Speed DDR Interfaces User Guide

Introduction (Ask a Question)

This user guide describes the high-speed memory interfaces in SmartFusion® 2 System-on-Chip (SoC) Field Programmable Gate Array (FPGA) and IGLOO® 2 FPGA devices. The high-speed interfaces Microcontroller/ Memory Subsystem Double-Data Rate (MDDR) subsystem and fabric DDR (FDDR) subsystem provide access to DDR memories for high-speed data transfers. The DDR subsystem's functionality, configurations, and their use models are discussed in this user guide.

This user guide contains the following chapters:

1. MDDR Subsystem
2. Fabric DDR Subsystem
3. DDR Bridge
4. Soft Memory Controller Fabric Interface Controller

Additional Documentation (Ask a Question)

The following table describes additional documentation available for the SmartFusion 2 and IGLOO 2 devices. For more information, see the SmartFusion 2 Documentation Page and IGLOO 2 Documentation Page online.

Table 1. Additional Documents

Document	Description
SmartFusion 2 System-on-Chip FPGAs Product Brief and *IGLOO 2 FPGA Product Brief*	This product brief provides an overview of SmartFusion® 2 and IGLOO® 2 family, features, and development tools.
IGLOO 2 and SmartFusion 2 Datasheet	This datasheet contains SmartFusion 2 and IGLOO 2 DC and switching characteristics.
IGLOO 2 Pin Descriptions Datasheet	This document contains IGLOO 2 pin descriptions, package outline drawings, and links to pin tables in Excel format.
SmartFusion 2 Pin Descriptions Datasheet	This document contains SmartFusion 2 pin descriptions, package outline drawings, and links to pin tables in Excel format.
IGLOO 2 FPGA and SmartFusion 2 SoC FPGA Fabric User Guide	SmartFusion 2 and IGLOO 2 FPGAs integrate fourth generation flash-based FPGA fabric. The FPGA fabric is comprised of Logic Elements which consist of a 4 input Look Up Table (LUT), includes embedded memories and Mathblocks for DSP processing capabilities. This document describes the SmartFusion 2 and IGLOO 2SmartFusion 2 and IGLOO 2 FPGA fabric architecture, embedded memories, Mathblocks, fabric routing, and I/Os.
SmartFusion 2 Microcontroller Subsystem	SmartFusion 2 devices integrate a hard microcontroller subsystem (MSS). The MSS consists of a Arm® Cortex®-M3 processor with Embedded Trace Macrocell (ETM), instruction cache, embedded memories, DMA engines, communication peripherals, timers, Real-Time Counter (RTC), general purpose I/Os, and FPGA fabric interfaces. This document describes the SmartFusion 2 MSS and its internal peripherals.
IGLOO 2 High Performance Memory Subsystem User Guide	IGLOO 2 devices integrate a hard High Performance Memory Subsystem (HPMS) consists of embedded memories, DMA engines, and FPGA fabric interfaces. This document describes the IGLOO 2 HPMS and its internal peripherals.

..........continued

Document	Description
IGLOO 2 and SmartFusion 2 High Speed Serial Interfaces User Guide	SmartFusion 2 and IGLOO 2 devices integrate hard high-speed serial interfaces (PCIe®, XAUI/XGXS, SerDes). This document describes the SmartFusion 2 and IGLOO 2 high-speed serial interfaces.
SmartFusion 2 and IGLOO 2 Clocking Resources User Guide	SmartFusion 2 and IGLOO 2 clocking resources include on-chip oscillators, FPGA fabric global network, and Clock Conditioning Circuitry (CCCs) with dedicated Phase-Locked Loops (PLLs). These clocking resources provide flexible clocking schemes to the on-chip hard IP blocks—HPMS, fabric DDR (FDDR) subsystem, and high-speed serial interfaces (PCIe, XAUI/XGXS, SerDes)—and logic implemented in the FPGA fabric.
SmartFusion 2 and IGLOO 2 Low Power Design User Guide	In addition to low static power consumption during normal operation, the SmartFusion 2 and IGLOO 2 devices support an ultra-low-power Static mode (Flash*Freeze mode) with power consumption less than 1 mW. Flash*Freeze mode retains all the SRAM and register data which enables fast recovery to Active mode. This document describes the SmartFusion 2 and IGLOO 2 Flash*Freeze mode entry and exit mechanisms.
SmartFusion 2 and IGLOO 2 FPGA Security and Reliability User Guide	The SmartFusion 2 and IGLOO 2 devices incorporate essentially all the security features that made third generation Microchip SoC devices the gold standard for security in the PLD industry. Also included are unique design and data security features and use models new to the PLD industry. SmartFusion 2 and IGLOO 2 Flash-based FPGA fabric has zero FIT configuration rate due to its Single Event Upset (SEU) immunity, which is critical in reliability applications. This document describes the SmartFusion 2 and IGLOO 2 security features and Error Detection and Correction (EDAC) capabilities.
SmartFusion 2 SoC and IGLOO 2 FPGA System Controller User Guide	The system controller manages programming of the SmartFusion 2 and IGLOO 2 devices and handles system service requests. The subsystems, interfaces, and system services in the system controller are discussed in this user guide.
SmartFusion 2 SoC and IGLOO 2 FPGA System Controller User Guide	Describes different programming modes supported in the SmartFusion 2 and IGLOO 2 devices. High level schematics of these programming methods are also provided as a reference. Important board-level considerations are discussed.
Libero® System-on-Chip (SoC) User Guide	Libero® SoC is the most comprehensive and powerful FPGA design and development software available, providing start-to-finish design flow guidance and support for novice and experienced users alike. Libero SoC combines Microchip SoC Products Group tools with such EDA powerhouses as Synplify and ModelSim®. This user guide discusses the usage of the software and design flow.

MICROCHIP

Table of Contents

MICROCHIP

1. MDDR Subsystem (Ask a Question)

The MDDR is a hardened ASIC block for interfacing the DDR2, DDR3, and LPDDR1 memories. The MDDR subsystem is used to access DDR memories for high-speed data transfers and code execution, and includes a DDR memory controller, DDR PHY, and arbitration logic to support multiple masters. DDR memory connected to the MDDR subsystem can be accessed by the MSS/HPMS masters and master logic implemented in the FPGA fabric (FPGA fabric master).

The MSS/HPMS masters communicate with the MDDR subsystem through an MSS/HPMS DDR bridge that provides an efficient access path. FPGA fabric masters communicate with the MDDR subsystem through AXI or AHB interfaces.

1.1 Features (Ask a Question)

The features of the MDDR subsystem are:

- Integrated on-chip DDR memory controller and PHY
- Capable of supporting LPDDR1, DDR2, and DDR3 memory devices
- Up to 667 Mbps (333.33 MHz DDR) performance
- Supports memory densities upto 2 GB
- Supports 8/16/32-bit DDR Standard Dynamic Random Access Memory (SDRAM) data bus width modes
- Supports a maximum of 8 memory banks
- Supports single rank memory
- Single Error Correction and Double Error Detection (SECDED) enable/disable feature
- Supports DRAM burst lengths of 4, 8, or 16, depending on the bus-width mode and DDR type configuration
- Support for sequential and interleaved burst ordering
- Programs internal control for ZQ short calibration cycles for DDR3 configurations
- Supports dynamic scheduling to optimize bandwidth and latency
- Supports self refresh entry and exit on command
- Supports deep power-down entry and exit on command
- Flexible address mapper logic to allow application specific mapping of row, column, bank, and rank bits
- Configurable support for 1T or 2T timing on the DDR SDRAM control signals
- Supports autonomous DRAM power-down entry and exit caused by lack of transaction arrival for programmable time

The following illustration shows the system level block diagram of the MDDR subsystem.

Figure 1-1. System Level MDDR Block Diagram
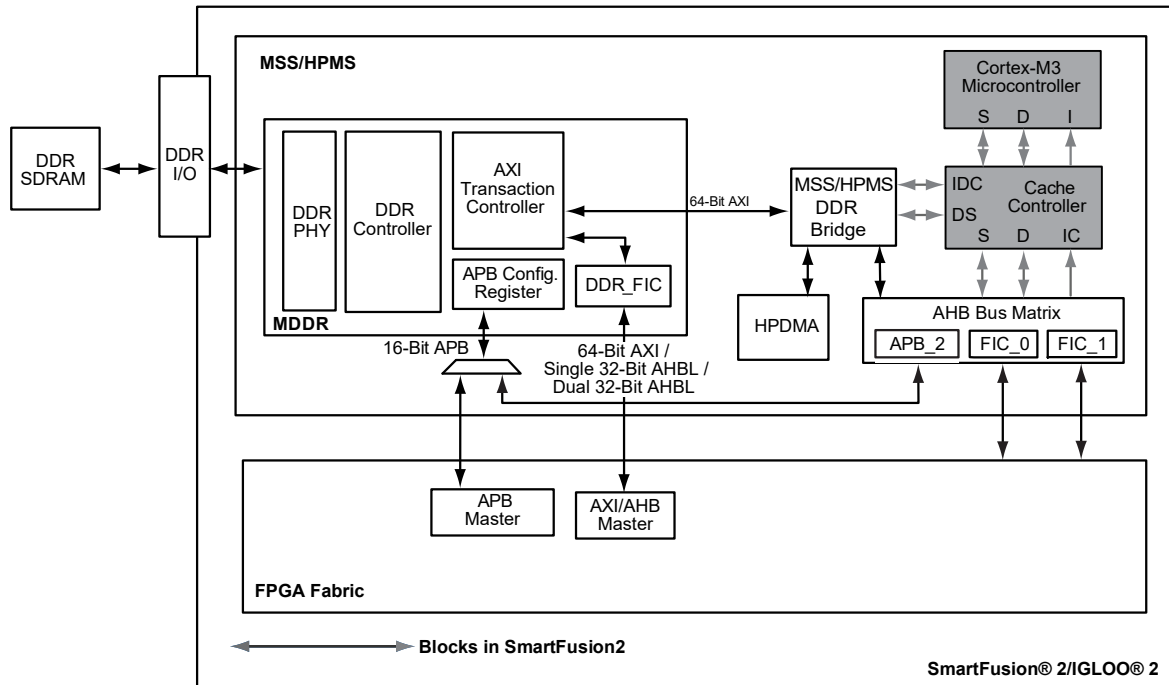
The MDDR subsystem accepts data transfer requests from AXI or AHB interfaces. Any read/write transactions to the DDR memories can occur from the following four paths:

- High performance DMA (HPDMA) controller can access DDR memories through the MSS/HPMS DDR bridge for high speed data transactions.
- Other MSS/HPMS masters (for example, FIC_0, FIC_1, and PDMA) can access DDR memories through the MSS/HPMS DDR bridge.
- AXI or AHBL masters in the FPGA fabric can access DDR memories through DDR_FIC interface.

Important:
- The Cortex®-M3 processor can access DDR memories through the MSS DDR bridge for data and code execution in SmartFusion 2.
- The maximum DDR3 data rate supported by MDDR is 333 MHz/667 Mbps. Therefore, Write Leveling is not mandatory and the interface works if the board layout includes length matching and follows
AN4153: Board and Layout Design Guidelines for SmartFusion2 SoC and IGLOO2 FPGAs. For Read Leveling, Libero SOC auto-generates predefined static delay ratios for MDDR initialization. These delay values are sufficient if the board layout follows the SmartFusion 2/IGLOO 2 board-level guidelines.

1.2   Memory Configurations (Ask a Question)

The SmartFusion 2 and IGLOO 2 FPGA MDDR subsystem supports a wide range of common memory types, configurations, and densities, as shown in the following table. If SECDED mode is enabled in the MDDR controller, the external memory module must be connected to the following:

- Data lines MDDR_DQ_ECC[3:0] when data width is x32
- Data lines MDDR_DQ_ECC[1:0] when data width is x16

- Data line MDDR_DQ_ECC[0] when data width is x8

Table 1-1. Supported Memory (DDR2, DDR3, and LPDDR1) Configurations

Memory Depth	Width	Width (in SECDED Mode)	SmartFusion® 2 and IGLOO® 2 Devices			
			M2S/M2GL 005/010/025/060/090 M2S/M2GL150-FCV484	M2S/M2GL 050 (FCS325, VF400, FG484)	M2S/M2GL 050 (FG896)	M2S/M2GL150 (FC1152)
128M or Less	×32	×36	—	—	✔	✔
	×16	×18	✔	✔	✔	✔
	×8	×9	✔	—	—	✔
256M	×32	×36	—	—	✔	✔
	×16	×18	✔	✔	✔	✔
	×8	×9	✔	—	—	✔
512M	×32	×36	—	—	✔	✔
	×16	×18	✔	✔	✔	✔
	×8	×9	✔	—	—	✔
1G	×32	×36	—	—	✔	✔
	×16	×18	✔	✔	✔	✔
	x8	×9	✔	—	—	✔

1.3 Performance (Ask a Question)

The following table shows the maximum data rates supported by MDDR subsystem for supported memory types.

For more Information, refer to the "DDR Memory Interface Characteristics" section in IGLOO2 FPGA and SmartFusion2 SoC FPGA Datasheet.

Table 1-2. DDR Speeds

Memory Type	Maximum Data Rate (Mbps)
LPDDR1	400 Mbps (200 MHz)
DDR2	667 Mbps (333.33 MHz)
DDR3	667 Mbps (333.33 MHz)

1.4 I/O Utilization (Ask a Question)

The following table lists the I/O utilization for the SmartFusion 2 and IGLOO 2 devices corresponding to supported bus widths. The remaining I/Os in Bank 0 are used for general purposes.

Table 1-3. I/O Utilization for SmartFusion® 2 and IGLOO® 2 Devices

MDDR Bus Width	M2S/ M2GL005/010/025/060/090 M2S/M2GL150-FCV484	M2S/M2GL 050 (FCS325, VF400, FG484)	M2S/M2GL 050 (FG896)	M2S/M2GL 150 (FC1152)
36-bit	—	—	Bank0 (85 pins)	Bank2 (85 pins)
32-bit	—	—	Bank0 (76 pins)	Bank2 (76 pins)
18-bit	Bank0 (59 pins)	Bank0 (59 pins)	Bank0 (59 pins)	Bank2 (59 pins)
16-bit	Bank0 (53 pins)	Bank0 (53 pins)	Bank0 (53 pins)	Bank2 (53 pins)
9-bit	Bank0 (47 pins)	—	—	Bank2 (47 pins)
8-bit	Bank0 (41 pins)	—	—	Bank2 (41 pins)

MICROCHIP

> **Important:** If MDDR is configured for LPDDR, one more I/O also available for every 8-bit as the LPDDR does not have DQS_N.

For general purpose use of the unused I/Os in the MDDR bank, select one of the I/O standards with the same voltage level as the DDR I/Os.

Self refresh must be disabled if the MDDR banks contain a mix of I/Os used for DDR and for general purpose fabric I/Os. For more information, see 1.5.4.3.5. Power Saving Modes.

1.5 Functional Description (Ask a Question)

This section provides the functional description of the MDDR subsystem.

1.5.1 Architecture Overview (Ask a Question)

The following illustration shows a functional block diagram of the MDDR subsystem. The main components include the DDR fabric interface controller (DDR_FIC), AXI transaction handler, DDR memory controller, and DDR PHY.

Figure 1-2. MDDR Subsystem Functional Block Diagram
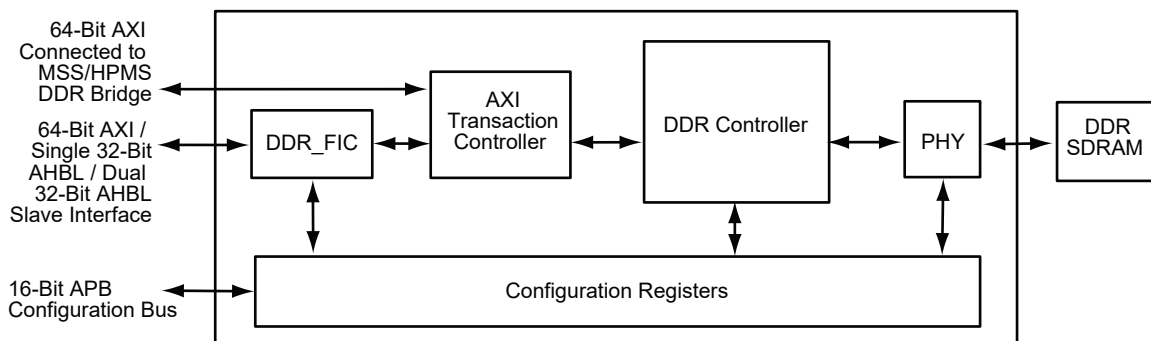
The DDR_FIC facilitates communication between the FPGA fabric masters and AXI transaction controller. The DDR_FIC is configured to provide either one 64-bit AXI slave interface or two independent 32-bit AHB-Lite (AHBL) slave interfaces to the FPGA fabric masters.

The AXI transaction controller receives read and write requests from AXI masters (MSS/HPMS DDR bridge and DDR_FIC) and schedules for the DDR controller by translating them into DDR controller commands.

The DDR controller receives the commands from the AXI transaction controller. These commands are queued internally and scheduled for access to the DDR SDRAM while satisfying DDR SDRAM constraints, transaction priorities, and dependencies between the transactions. The DDR controller in turn issues commands to the PHY module, which launches and captures data to and from the DDR SDRAM.

DDR PHY receives commands from the DDR controller and generates DDR memory signals required to access the external DDR memory.

The 16-bit APB configuration bus provides an interface to configure the MDDR subsystem registers. The MDDR subsystem operates on MDDR_CLK. MSS/HPMS CCC generates the MDDR_CLK using MPLL. For more details on MSS/HPMS CCC refer UG0449: SmartFusion2 and IGLOO2 Clocking Resources User Guide.

1.5.2 Port List (Ask a Question)

The following table describes the MDDR subsystem interface signals.

Table 1-4. MDDR Subsystem Interface Signals

Signal Name	Type	Polarity	Description
APB_S_PCLK	In	—	APB clock. This clock drives all the registers of the APB interface.
APB_S_PRESET_N	In	Low	APB reset signal. This is an active-low signal. This drives the APB interface and is used to generate the soft reset for the DDR controller as well.
MDDR_DDR_CORE_RESET_N	In	Low	Global reset. This resets the DDR_FIC/DDRC/PHY/DDRAXI logic.
MDDR_DDR_AXI_S_RMW	In	High	AXI mode only indicates whether all bytes of a 64-bit lane are valid for all beats of an AXI transfer. • 0: Indicates that all bytes in all beats are valid in the burst and the controller should default to write commands • 1: Indicates that some bytes are invalid and the controller should default to RMW commands. This is classed as an AXI write address channel sideband signal and is valid with the AWVALID signal
HPMS_DDR_FIC_SUBSYSTEM_CLK or, MSS_DDR_FIC_SUBSYSTEM_CLK	Out	—	This output clock is derived from the MDDR_CLK and is based on the DDR_FIC divider ratio. This is the clock that should be used for the AXI or AHB slave interfaces to move data in and out of the MDDR.
HPMS_DDR_FIC_SUBSYSTEM_LOCK or, MSS_DDR_FIC_SUBSYSTEM_LOCK	Out	—	This indicates the lock from FCCC which generates HPMS_DDR_FIC_SUBSYSTEM_CLK for IGLOO2 and MSS_DDR_FIC_SUBSYSTEM_LOCK in SmartFusion 2.
Bus Interfaces			
AXI_SLAVE[1]	Bus	—	AXI slave interface 1.0 bus
AHB0_SLAVE[2]	Bus	—	AHB0 slave interface 3.0 bus
AHB1_SLAVE[3]	Bus	—	AHB1 slave interface 3.0 bus
APB_SLAVE	Bus	—	APB slave interface 3.0 bus
DRAM Interface			
MDDR_CAS_N	Out	Low	DRAM CASN
MDDR_CKE	Out	High	DRAM CKE
MDDR_CLK	Out	—	DRAM single-ended clock – for differential pads
MDDR_CLK_N	Out	—	DRAM single-ended clock – for differential pads
MDDR_CS_N	Out	Low	DRAM CSN
MDDR_ODT	Out	High	DRAM ODT. • 0: Termination OFF • 1: Termination ON
MDDR_RAS_N	Out	Low	DRAM RASN
MDDR_ RESET_N	Out	Low	DRAM reset for DDR3
MDDR_WE_N	Out	Low	DRAM WEN
MDDR_ADDR[15:0]	Out	—	Dram address bits
MDDR_BA[2:0]	Out	—	Dram bank address
MDDR_DM_RDQS[3:0]	In/out	—	DRAM data mask – from bidirectional pads
MDDR_DQS[3:0]	In/out	—	DRAM single-ended data strobe output – for bidirectional pads
MDDR_DQS_N[3:0]	In/out	—	DRAM single-ended data strobe output – for bidirectional pads
MDDR_DQ[31:0]	In/out	—	DRAM data input/output – for bidirectional pads

...........continued

Signal Name	Type	Polarity	Description
MDDR_DQ_ECC[3:0]	In/out	—	DRAM data input/output for SECDED
MDDR_DM_RDQS_ECC	In/out	High	DRAM single-ended data strobe output – for bidirectional pads
MDDR_DQS_ECC	In/out	High	DRAM single-ended data strobe output – for bidirectional pads
MDDR_DQS_ECC_N	In/out	Low	DRAM data input/output – for bidirectional pads
MDDR_DQS_TMATCH_0_IN	In	High	DQS enables input for timing match between DQS and system clock. For simulations, tie to MDDR_DQS_TMATCH_0_OUT.
MDDR_DQS_TMATCH_1_IN	In	High	DQS enables input for timing match between DQS and system clock. For simulations, tie to MDDR_DQS_TMATCH_1_OUT.
MDDR_DQS_TMATCH_0_OUT	Out	High	DQS enables output for timing match between DQS and system clock. For simulations, tie to MDDR_DQS_TMATCH_0_IN.
MDDR_DQS_TMATCH_1_OUT	Out	High	DQS enables output for timing match between DQS and system clock. For simulations, tie to MDDR_DQS_TMATCH_1_IN.
MDDR_DQS_TMATCH_ECC_IN	In	High	DQS enables input for timing match between DQS and system clock. For simulations, tie to MDDR_DQS_TMATCH_ECC_OUT.
MDDR_DQS_TMATCH_ECC_OUT	Out	High	DQS enables output for timing match between DQS and system clock. For simulations, tie to MDDR_DQS_TMATCH_ECC_IN.

Notes:

1. AXI or AHB interface, depending on configuration.
2. MDDR_DQS_N[3:0] signals are not available for LPDDR.
3. TMATCH_IN and TMATCH_OUT pins are required to be connected together outside the device. They are used for gate training as part of the read data capture operation. The two pins create an internal DQS Enable signal that is used to calibrate the flight path. DQS needs to be gated to prevent false triggering of the FIFO write clock. This DQS Enable signal is derived from the system clock and physically matches the clock output buffer and DQS input buffer to compensate for I/O buffer uncertainty due to Process-Voltage-Temperature (PVT) changes. Without this connection, the circuit is not operable.

1.5.2.1 AXI Slave Interface (Ask a Question)

The following table describes the MDDR AXI slave interface signals. These signals will be available only if the MDDR interface is configured for AXI mode. For more AXI protocol details, see the AMBA AXI v1.0 protocol specification.

Table 1-5. AXI Slave Interface Signals

Signal Name	Direction	Polarity	Description
MDDR_DDR_AXI_S_ARREADY	Output	High	Indicates whether or not the slave is ready to accept an address and associated control signals. • 1: Slave ready • 0: Slave not ready
MDDR_DDR_AXI_S_AWREADY	Output	High	Indicates that the slave is ready to accept an address and associated control signals. • 1: Slave ready • 0: Slave not ready

...........continued

Signal Name	Direction	Polarity	Description
MDDR_DDR_AXI_S_BID[3:0]	Output	—	Indicates response ID. The identification tag of the write response.
MDDR_DDR_AXI_S_BRESP[1:0]	Output	—	Indicates write response. This signal indicates the status of the write transaction. • 00: Normal access okay • 01: Exclusive access okay • 10: Slave error • 11: Decode error
MDDR_DDR_AXI_S_BVALID	Output	High	Indicates whether a valid write response is available. • 1: Write response available • 0: Write response not available
MDDR_DDR_AXI_S_RDATA[63:0]	Output	—	Indicates read data.
MDDR_DDR_AXI_S_RID[3:0]	Output	—	Read ID tag. This signal is the ID tag of the read data group of signals.
MDDR_DDR_AXI_S_RLAST	Output	High	Indicates the last transfer in a read burst.
MDDR_DDR_AXI_S_RRESP[1:0]	Output	—	Indicates read response. This signal indicates the status of the read transfer. • 00: Normal access • 01: Exclusive access • 10: Slave error • 11: Decode error
MDDR_DDR_AXI_S_RVALID	Output	—	Indicates whether the required read data is available and the read transfer can complete. • 1: Read data available • 0: Read data not available
MDDR_DDR_AXI_S_WREADY	Output	High	Indicates whether the slave can accept the write data. • 1: Slave ready • 0: Slave not ready
MDDR_DDR_MDDR_DDR_AXI_S_ARADDR[31:0]	Input	—	Indicates initial address of a read burst transaction. **Note:** DDR_FIC AXI interface supports only 64-bit aligned addresses.
MDDR_DDR_AXI_S_ARBURST[1:0]	Input	—	Indicates burst type. The burst type, coupled with the size information, details how the address for each transfer within the burst is calculated. • 00: FIXED - Fixed-address burst FIFO type (Not Supported) • 01: INCR - Incrementing-address burst normal sequential memory • 10: WRAP - Incrementing-address burst that wraps to a lower address at the wrap boundary • 11: Reserved

...........continued

Signal Name	Direction	Polarity	Description
MDDR_DDR_AXI_S_ARID[3:0]	Input	—	Indicates identification tag for the read address group of signals.
MDDR_DDR_AXI_S_ARLEN[3:0]	Input	—	Indicates burst length. The burst length gives the exact number of transfers in a burst. • 0000: 1 • 0001: 2 • 0010: 3 • 0011: 4 • 0100: 5 • 0101: 6 • 0110: 7 • 0111: 8 • 1000: 9 • 1001: 10 • 1010: 11 • 1011: 12 • 1100: 13 • 1101: 14 • 1110: 15 • 1111: 16
MDDR_DDR_AXI_S_ARLOCK[1:0]	Input	—	Indicates lock type. This signal provides additional information about the atomic characteristics of the read transfer. • 00: Normal access • 01: Exclusive access • 10: Locked access • 11: Reserved
MDDR_DDR_AXI_S_ARSIZE[1:0]	Input	—	Indicates the maximum number of data bytes to transfer in each data transfer, within a burst. • 00: 10 : Not Supported • 11: 8
MDDR_DDR_AXI_S_ARVALID	Input	High	Indicates the validity of read address and control information. • 1: Address and control information valid • 0: Address and control information not valid
MDDR_DDR_AXI_S_AWADDR[31:0]	Input	—	Indicates write address. The write address bus gives the address of the first transfer in a write burst transaction. **Note:** DDR_FIC AXI interface supports only 64-bit aligned addresses.

...........continued

Signal Name	Direction	Polarity	Description
MDDR_DDR_AXI_S_AWBURST[1:0]	Input	—	Indicates burst type. The burst type, coupled with the size information, details how the address for each transfer within the burst is calculated. • 00: FIXED - Fixed-address burst FIFO-type (Not Supported) • 01: INCR - Incrementing-address burst normal sequential memory • 10: WRAP - Incrementing-address burst that wraps to a lower address at the wrap boundary • 11: Reserved
MDDR_DDR_AXI_S_AWID[3:0]	Input	—	Indicates identification tag for the write address group of signals.
MDDR_DDR_AXI_S_AWLEN[3:0]	Input	—	Indicates burst length. The burst length gives the exact number of transfers in a burst. This information determines the number of data transfers associated with the address. • 0000: 1 • 0001: 2 • 0010: 3 • 0011: 4 • 0100: 5 • 0101: 6 • 0110: 7 • 0111: 8 • 1000: 9 • 1001: 10 • 1010: 11 • 1011: 12 • 1100: 13 • 1101: 14 • 1110: 15 • 1111: 16
MDDR_DDR_AXI_S_AWLOCK[1:0]	Input	—	Indicates lock type. This signal provides additional information about the atomic characteristics of the write transfer. • 00: Normal access • 01: Exclusive access • 10: Locked access • 11: Reserved
MDDR_DDR_AXI_S_AWSIZE[1:0]	Input	—	Indicates the maximum number of data bytes to transfer in each data transfer, within a burst. • 00 to 10 : Not Supported • 11: 8

...........continued

Signal Name	Direction	Polarity	Description
MDDR_DDR_AXI_S_AWVALID	Input	High	Indicates whether or not valid write address and control information are available. • 1: Address and control information available • 0: Address and control information not available
MDDR_DDR_AXI_S_BREADY	Input	High	Indicates whether or not the master can accept the response information. • 1: Master ready • 0: Master not ready
MDDR_DDR_AXI_S_RREADY	Input	High	Indicates whether or not the master can accept the read data and response information. • 1: Master ready • 0: Master not ready
MDDR_DDR_AXI_S_WDATA[63:0]	Input	—	Indicates write data.
MDDR_DDR_AXI_S_WID[3:0]	Input	—	Indicates response ID. The identification tag of the write response.
MDDR_DDR_AXI_S_WLAST	Input	High	Indicates the last transfer in a write burst.
MDDR_DDR_AXI_S_WSTRB[7:0]	Input	—	Indicates which byte lanes to update in memory.
MDDR_DDR_AXI_S_WVALID	Input	High	Indicates whether or not valid write data and strobes are available. • 1: Write data and strobes available • 0: Write data and strobes not available

1.5.2.2 AHB Slave Interface (Ask a Question)

The following table describes the MDDR AHB slave interface signals. These signals are available only if MDDR interface is configured for single or dual AHB mode. For more AHB protocol details, see the AMBA AHB v3.0 protocol specification.

Table 1-6. AHB Slave Interface Signals

Signal Name	Direction	Polarity	Description
MDDR_DDR_AHBx_S_HREADYOUT	Output	High	Indicates that a transfer has finished on the bus. The signal is asserted Low to extend a transfer. Input to Fabric master.
MDDR_DDR_AHBx_S_HRESP	Output	High	Indicates AHB transfer response to Fabric master.
MDDR_DDR_AHBx_S_HRDATA[31:0]	Output	—	Indicates AHB read data to Fabric master.
MDDR_DDR_AHBx_S_HSEL	Input	High	Indicates AHB slave select signal from Fabric master.
MDDR_DDR_AHBx_S_HADDR[31:0]	Input	—	Indicates AHB address initiated by Fabric master.

...........continued

Signal Name	Direction	Polarity	Description
MDDR_DDR_AHBx_S_HBURST[2:0]	Input	—	Indicates AHB burst type from Fabric master. • 000: Single burst • 001: Incrementing burst of undefined length • 010: 4-beat wrapping burst • 011: 4-beat incrementing burst • 100: 8-beat wrapping burst • 101: 8-beat incrementing burst • 110: 16-beat wrapping burst • 111: 16-beat incrementing burst
MDDR_DDR_AHBx_S_HSIZE[1:0]	Input	—	Indicates AHB transfer size from Fabric master. • 00: 8 Byte • 01: 16 Halfword • 10: 32 Word
MDDR_DDR_AHBx_S_HTRANS[1:0]	Input	—	Indicates AHB transfer type from Fabric master. • 00: IDLE • 01: BUSY • 10: NONSEQUENTIAL • 11: SEQUENTIAL
MDDR_DDR_AHBx_S_HMASTLOCK	Input	High	Indicates AHB master lock signal from Fabric master.
MDDR_DDR_AHBx_S_HWRITE	Input	High	Indicates AHB write control signal from Fabric master.
MDDR_DDR_AHBx_S_HREADY	Input	High	Indicates that a transfer has finished on the bus. Fabric master can drive this signal Low to extend a transfer.
MDDR_DDR_AHBx_S_HWDATA[31:0]	Input	—	Indicates AHB write data from Fabric master.

Important: AHBx indicates AHB0 or AHB1.

1.5.2.3 APB Slave Interface (Ask a Question)

The following table describes the MDDR APB slave interface signals. For more information about APB protocol, see the AMBA APB v3.0 protocol specification.

Table 1-7. MDDR APB Slave Interface Signals

Signal Name	Direction	Polarity	Description
MDDR_APB_S_PREADY	Output	High	Indicates APB Ready signal to Fabric master.
MDDR_APB_S_PSLVERR	Output	High	Indicates error condition on an APB transfer to Fabric master.
MDDR_APB_S_PRDATA[15:0]	Output	—	Indicates APB read data to Fabric master.
MDDR_APB_S_PENABLE	Input	High	Indicates APB enable from Fabric master. The enable signal is used to indicate the second cycle of an APB transfer.
MDDR_APB_S_PSEL	Input	High	Indicates APB slave select signal from Fabric master
MDDR_APB_S_PWRITE	Input	High	Indicates APB write control signal form Fabric master
MDDR_APB_S_PADDR[10:2]	Input	—	Indicates APB address initiated by Fabric master.

MICROCHIP

..........continued

Signal Name	Direction	Polarity	Description
MDDR_APB_S_PWDATA[15:0]	Input	—	Indicates APB write data from Fabric master.

1.5.3 Initialization (Ask a Question)

After power-up, the MDDR needs to have all of the configuration registers written to establish the operating modes of the blocks. When using the System Builder design flow through Libero SoC, this is all handled for the user through the use of the System Builder module. All of the configuration register values are selected by the user and stored in a special portion of the embedded Non-Volatile memory (eNVM). Before the MDDR subsystem is active, it goes through an initialization phase and this process starts with a reset sequence. For DDR3 memories, the initialization phase also includes ZQ calibration and DRAM training.

1.5.3.1 Reset Sequence (Ask a Question)

The following illustration shows the reset sequence for the MDDR subsystem from the Power-on Reset stage. The MDDR subsystem comes out of reset after MPLL Lock is asserted by the MSS/HPMS_CCC. De-assertion of MDDR_AXI_RESET_N signifies the end of the reset sequence. The MDDR reset is generated by asserting the MDDR_CTLR_SOFTRESET bit in SOFT_RESET_CR to 1. For more information about SOFT_RESET_CR, see UG0448: IGLOO2 FPGA High Performance Memory Subsystem. The DDR controller performs external DRAM memory reset and initialization as per the JEDEC specification, including reset, refresh, and mode registers.

1.5.3.2 DDRIO Calibration (Ask a Question)

Each DDRIO has an ODT feature, which is calibrated depending on the DDR I/O standard. DDR I/O calibration occurs after the DDR I/Os are enabled. If the impedance feature is enabled, impedance can be programmed to the desired value in three ways:

- Calibrate the ODT/driver impedance with a calibration block (recommended)
- Calibrate the ODT/driver impedance with fixed calibration codes
- Configure the ODT/driver impedance to the desired value directly

The system register, MDDR_IO_CALIB_CR, can be configured for changing the ODT value to the desired value.

The I/O calibration is always enabled when the DDR subsystem is configured for DDR2 and DDR3 memories.

The I/O calibration is disabled or enabled using the DDR configurator when the DDR subsystem is configured for LPDDR memories.

> **Important:** If I/O calibration is enabled, all I/Os in the DDR bank are calibrated even though the DDR controller is not using all I/Os in the bank.

For more information on DDR I/O calibration, see the Configurable ODT and Driver Impedance section of the I/Os chapter in the UG0445: IGLOO2 FPGA and SmartFusion2 SoC FPGA Fabric User Guide.

Figure 1-3. Reset Sequence
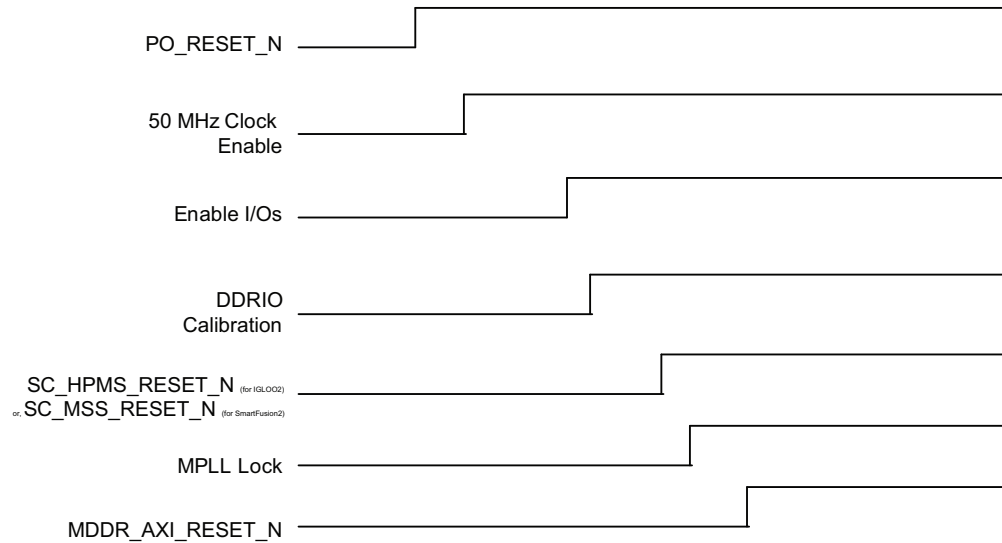
1.5.3.3 ZQ Calibration (Ask a Question)

This is applicable for DDR3 only. The ZQ calibration command is used to calibrate DRAM output drivers (R_{ON}) and On-Die Termination (ODT) values. The DDR3 SDRAM needs a longer time to calibrate R_{ON} and ODT at initialization and a relatively smaller time to perform periodic calibrations.

The DDR controller performs ZQ calibration by issuing a ZQ Calibration Long (ZQCL) command and ZQ Calibration Short (ZQCS) command.

ZQCL is used to perform initial calibration during the power-up initialization sequence. This command is allowed for a period of t_{ZQinit}, as specified by memory vendor. The value of t_{ZQinit} can be modified through register bits of REG_DDRC_T_ZQ_LONG_NOP in Table 1-59.

The ZQCS command is used to perform periodic calibration to account for voltage and temperature variations. A shorter timing window is provided to perform calibration and transfer of values as defined by timing parameter tZQCS. The tZQCS parameter can be modified through register bits of REG_DDRC_T_ZQ_SHORT_NOP in Table 1-60.

Other activities are not performed by the controller for the duration of t_{ZQinit} and tZQCS. All DRAM banks are precharged and tRP is met before ZQCL or ZQCS commands are issued by the DDR controller.

1.5.3.4 DRAM Training (Ask a Question)

High-Speed DDR3 memories typically require the DDR controller to implement Write-Leveling, Read DQS Gate Training, and Read Data Eye Training. However, MDDR only supports a maximum data rate of 333 MHz/667 Mbps, which means the clock period and data window are relatively large compared to high-speed DDR3 memory interfaces. Therefore dynamic write-leveling and read training are not performed. The following sections describe how write-leveling and read training are addressed by the MDDR.

1.5.3.4.1 Write Leveling (Ask a Question)

Dynamic write-leveling is not required for the MDDR controller. The board-layout needs to follow AN4153 SmartFusion2 and IGLOO2 Board Design Guidelines Application Note to keep the skew between DQS and CK within the JEDEC DDR3 tDQSS limit of ±750 ps at each memory device.

333 MHz/667 Mbps is the maximum DDR3 rate MDDR supports. Leveling is not mandatory and the interface will work if the board layout guidelines are followed and length matching is done.

1.5.3.4.2 Read Leveling (Ask a Question)

MDDR does not perform dynamic Read DQS Gate Training and Data Eye Training. Instead, these functions are achieved by using built-in static delay values automatically generated by Libero SoC for the MDDR automatic register initialization.

1.5.3.4.3 Read Gate (Ask a Question)

The DQS gate is aligned by the Libero SoC auto-generated MDDR initialization code containing fixed delay ratios to account for board round-trip time between FPGA and the DDR3 memory. The TMATCH_OUT and TMATCH_IN signals are shorted close to the FPGA balls to remove the FPGA output and input delays from the round trip delay time. Therefore, the fixed delay ratios represent only the board delay.

The fixed delay ratios work in combination with board layouts which follow the SmartFusion2/IGLOO2 Board Design Guidelines (refer AN4153 SmartFusion2 and IGLOO2 Board Design Guidelines Application Note).

1.5.3.4.4 DQS Alignment within Data Eye (Ask a Question)

The incoming read DQS is internally centered within the read DQ data window using a static delay ratio. This static delay is applied by the Libero SoC auto-generated MDDR initialization code. The fixed delay ratios work in combination with board layouts which follow the SmartFusion 2/IGLOO 2 Board Design Guidelines (refer AN4153 SmartFusion2 and IGLOO2 Board Design Guidelines Application Note).

> **Important:** The Libero SoC auto-generated delay ratio for read DQS data eye centering is written to the required register.

1.5.3.5 DDR Memory Initialization Time (Ask a Question)

The time to initialize the DDR memory depends on the following factors:

- Power-up and register initialization by system controller. It depends on the Power-on Reset delay configuration in the Libero project (**Project** > **Project Settings** > **Device settings**).
- DDR controller and PHY configuration registers initialization. In SmartFusion 2 devices, the Cortex-M3 initializes these registers. In IGLOO 2 devices, the ConfigMaster in the FPGA fabric initializes these registers.
- DDR memory initialization by the DDR Controller according to the JEDEC standard (mode register configuration and training).
- DDR memory settling time configured in the System Builder memory configuration window.

1.5.4 Details of Operation (Ask a Question)

This section provides a functional description of each block in the MDDR subsystem.

1.5.4.1 DDR_FIC (Ask a Question)

The following illustration shows the DDR_FIC block diagram.

Figure 1-4. DDR_FIC Block Diagram
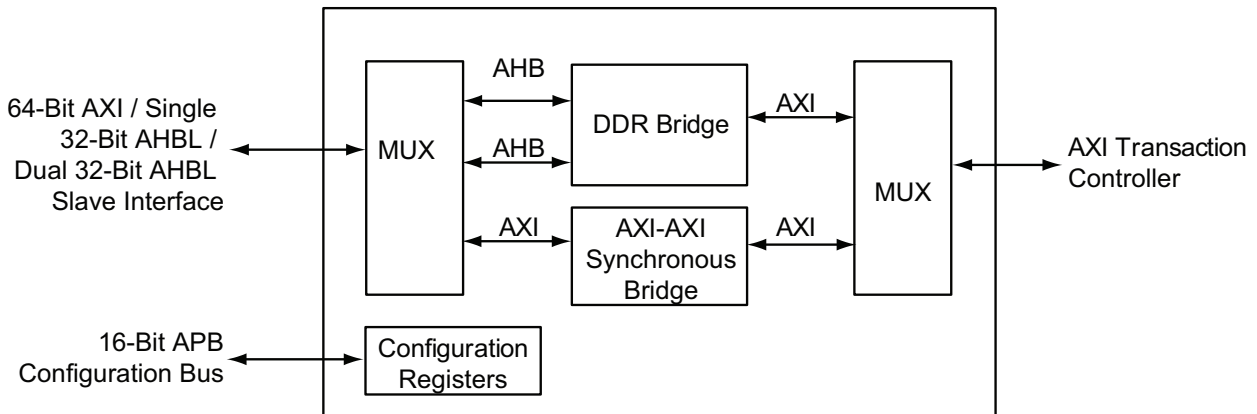
Fabric masters can access the MDDR subsystem in the following ways:

- Single AXI-64 interface
- Single AHB-32 interface
- Dual AHB-32 bit interfaces

If the AXI-64 interface is selected, the DDR_FIC acts as an AXI to AXI synchronous bridge. In this mode, DDR_FIC provides FPGA fabric masters to access the MDDR subsystem through locked transactions. For this purpose, a user configurable 20-bit down counter keeps track of the duration of the locked transfer. If the transfer is not completed before the down counter reaches zero, a single clock cycle pulse interrupt is generated to the fabric interface.

If single or dual AHB-32 interfaces are selected, DDR_FIC converts the single/dual 32-bit AHBL master transactions from the FPGA fabric to 64-bit AXI transactions. In this mode the DDR bridge, embedded as part of the DDR_FIC, is enabled. The DDR bridge has an arbiter, which arbitrates read and write requests from the two AHB masters on a round robin priority scheme. For a detailed description, see the 3.4. DDR Bridge Control Registers in MDDR and FDDR.

The DDR_FIC input interface is clocked by the FPGA fabric clock and the MDDR is clocked by MDDR_CLK from the MSS/HPMS CCC. Clock ratios between MDDR_CLK and DDR_FIC clock can vary. The following table lists supported ratios. Clock ratios are configured through Libero System-on-Chip (SoC) software or through system register MSSDDR_FACC1_CR. For more information, see the 1.11. MDDR Configuration Registers.

Table 1-8. MDDR_CLK to FPGA Fabric Clock Ratios

DIVISOR_A[1:0]	FIC64_DIVISOR[2:0]	MDDR_CLK: FPGA FABRIC Clock Ratio
00	000	1:1
00	001	2:1
00	010	4:1
00	100	8:1
00	101	16:1
01	000	2:1
01	001	4:1
01	010	8:1
01	100	16:1
11	000	3:1
11	001	6:1
11	010	12:1

1.5.4.2 AXI Transaction Controller (Ask a Question)

The AXI transaction controller receives 64-bit AXI transactions from various masters (MSS/HPMS DDR bridge and DDR_FIC) and translates them into DDR controller transactions. The following illustration shows the block diagram of the AXI transaction controller interfaced with the DDR controller.

The AXI transaction controller performs arbitration of the read/write requests initiated by AXI compliant masters.

Figure 1-5. AXI Transaction Controller Block Diagram
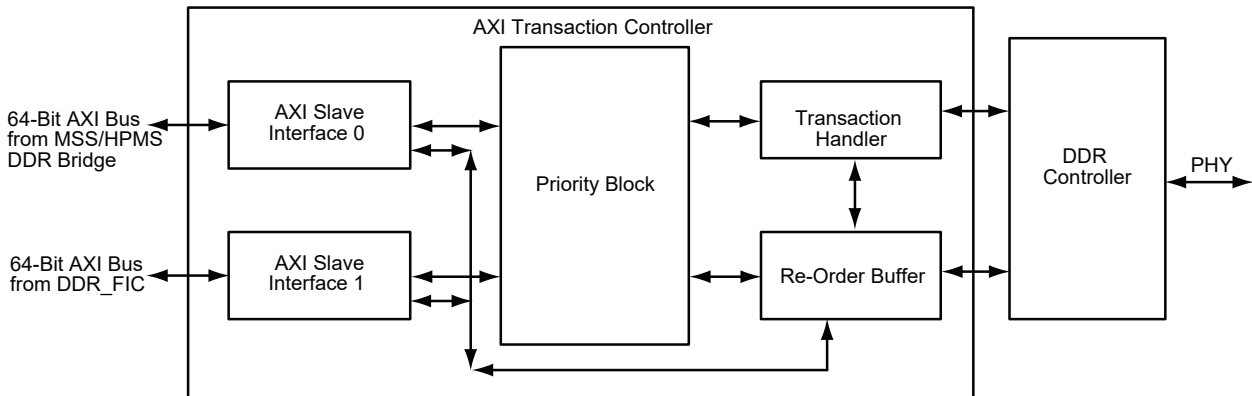
The AXI transaction controller comprises four major blocks:

- AXI slave interface
- Priority block
- Transaction handler
- Reorder buffer

1.5.4.2.1 AXI Slave Interfaces (Ask a Question)

The AXI transaction controller has two 64-bit AXI slave interfaces: one from the MSS/HPMS DDR bridge and the other from DDR_FIC. Each of the AXI slave ports is 64 bits wide and is in compliance with the standard AXI protocol. Each transaction has an ID related to the master interface. Transactions with the same ID are completed in order, while the transactions with different read IDs are completed in any order, depending on when the instruction is executed by the DDR controller. If a master requires ordering between transactions, the same ID should be used.

The AXI slave interface has individual read and write ports. The read port queues read AXI transactions and it can hold up to four read transactions. The write port handles only one write transaction at a time and generates the handshaking signals on the AXI interface.

1.5.4.2.2 Priority Block (Ask a Question)

The priority block prioritizes AXI read/write transactions and provides control to the transaction handler. AXI read transactions have higher priority. The default priority ordering is listed as follows:

1. Reads from the slave port of the MSS/HPMS DDR bridge
2. Reads from the slave port of DDR_FIC
3. Writes from the slave port of the MSS/HPMS DDR bridge
4. Writes from the slave port of DDR_FIC

The fabric master through DDR_FIC is programmed to have a higher priority by configuring the PRIORITY_ID and PRIORITY_ENABLE_BIT bit fields in the Table 1-75 register. Priority levels to other masters can be programmed as well, as shown in the following table.

Table 1-9. Priority Level Configuration

Transactions	Default Priorities (Type-0)	Priorities	
SmartFusion® 2	—	PRIORITY_ENABLE_BIT=01 (Type 1)	PRIORITY_ENABLE_BIT=10/11 (Type 2/3)
Reads from I - Cache	1	2	2
Reads from DSG bus	2	3	3
Reads from HPDMA/AHB bus	3	4	4
Reads from Fabric master having the ID as PRIORITY_ID	4	3	1
Writes from DSG bus	5	5	5
Writes from HPDMA/AHB bus	6	7	7
Writes from Fabric master having the ID as PRIORITY_ID	7	6	6
IGLOO® 2		PRIORITY_ENABLE_BIT=01/10/11 (Type-1/2/3)	
Reads from HPDMA/AHB bus	1	—	2
Reads from Fabric master having the ID as PRIORITY_ID	2	—	1
Writes from HPDMA/AHB bus	3	—	4
Writes from Fabric master having the ID as PRIORITY_ID	4	—	3

1.5.4.2.3 Transaction Handler (Ask a Question)

The transaction handler converts AXI transactions into DDR controller commands. The transaction handler works on one transaction at a time from the read/write port queue that is selected by the priority block.

The transaction handler has a write command controller and read command controller for write and read transactions.

The write command controller fetches the command from the AXI slave write port and sends a pure write instruction to the DDR controller. If SECDED is enabled, a Read Modified Write (RMW) instruction is sent to the DDR controller.

The read command controller generates read transactions to the DDR controller.

1.5.4.2.4 Reorder Buffer (Ask a Question)

The reorder buffer receives data from the DDR controller and orders the data as requested by the AXI master when a single AXI transaction is split into multiple DDR controller transactions, depending on the transfer size.

1.5.4.3 DDR Controller (Ask a Question)

The DDR controller receives requests from the AXI transaction controller, performs the address mapping from system addresses to DRAM addresses (rank, bank, row, and column), and prioritizes requests to minimize the latency of reads (especially high priority reads) and maximize page hits. It also ensures that DRAM is properly initialized, all requests are made to DRAM legally (accounting for associated DRAM constraints), refreshes are inserted as required, and the DRAM enters and exits various power-saving modes appropriately. The following illustration shows the DDR controller connections in the MDDR subsystem.

Figure 1-6. DDR Controller Block Diagram

The following sections describe key functions of the DDR controller.

1.5.4.3.1 Address Mapping (Ask a Question)

Read and write requests to the DDR controller requires a system address. The controller is responsible for mapping this system address with rank, bank, row, and column address to DRAM.

The address mapper maps linear request addresses to DDR memory addresses by selecting the source bit that maps to each and every applicable DDR memory address bit. The address map interface registers are configured to map source address bits to DRAM address (for more information, see 1.5.5.6. Address Mapping in Configuring the MDDR features).

1.5.4.3.2 Transaction Scheduling (Ask a Question)

The DDR controller schedules the read and write transactions to DDR memory. The DDR controller classifies the transactions into three types, based on the commands from the AXI transaction controller:

- Low priority reads (LPR)
- High priority reads (HPR)
- Writes (WR)

Each type of transaction has a queue and the queued transactions can be in normal state or in critical state. The transactions in a queue moves from normal state to critical state when that transaction is not serviced for a count of MAX_STARVE_X32 clocks. The MAX_STARVE_X32 values for each queue is configured using the DDR controller performance registers (refer 1.5.5.10. Performance). The DDR controller completes the critical transactions with high priority.

1.5.4.3.3 Write Combine (Ask a Question)

The DDR controller combines multiple writes to the same address into a single write to DDR memory. When a new write collides with the queued write, the DDR controller overwrites the data for the queued write with that from the new write and only performs one write transaction. The write combine functionality can be disabled by setting the register bit REG_DDRC_DIS_WC in Table 1-54 to 1.

1.5.4.3.4 SECDED (Ask a Question)

The DDR controller supports built-in SECDED capability for correcting single-bit errors and detecting two-bit errors. The SECDED feature can be enabled in the **System Builder** > **memory controller configuration window**. When SECDED is enabled, the DDR controller adds 8 bits of SECDED data to every 64 bits of data.

The DDR controller computes ECC for every 64-bit data. When SECDED is enabled, a write operation computes and stores a SECDED code along with the data, and a read operation reads and checks the data against the stored SECDED code. It is therefore, possible to receive single/dual bit errors when reading uninitialized memory locations. To avoid this, all the memory locations must be written before being read.

Microchip

For a non 64-bit write operation, the DDR controller performs a 256-bit read modify write (RMW) operation. This read modify write operation is always performed on 256-bit aligned addresses.

For example, if the DDR controller receives a 32-bit write operation to address 0x4, then the DDR controller performs the following operations:

1. Reads the 256-bit data from 0x0 (256-bit aligned address for 0x4).

2. Modifies 32-bits (bit33 to bit64) of that 256-bit data with the user 32-bit data.

3. Computes the ECC and writes 288-bits (256-bit data + 32-bit ECC) to address 0x0.

The following illustration shows the DDR controller burst transactions to DRAM for unaligned 64-bit AXI write transaction. The DDR controller is configured for DDR3 memory, 32-bit burst width, and burst length 8.

Figure 1-7. DDR RMW Operation (32-Bit DDR Bus Width and Burst Length 8)

The following illustration shows the DDR controller burst transactions to DRAM for unaligned 64-bit AXI write transaction. The DDR controller is configured for DDR3 memory, 16-bit burst width, and burst length 8.

Figure 1-8. DDR RMW Operation (16-Bit DDR Bus Width and Burst Length 8)

The following illustration shows the DDR controller burst transactions to DRAM for unaligned 64-bit AXI write transaction. The DDR controller is configured for DDR3 memory, 8-bit burst width, and burst length 8.

Figure 1-9. DDR RMW Operation (8-Bit DDR Bus Width and Burst Length 8)
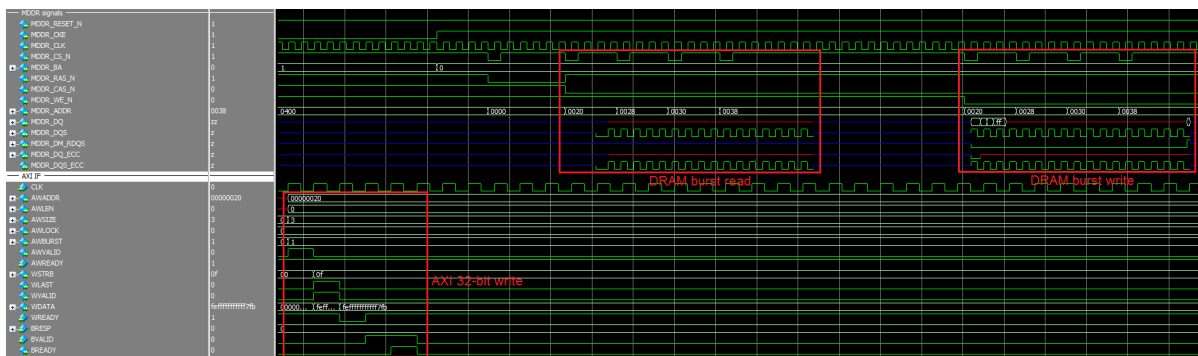
For more information on the SECDED feature of SmartFusion 2 MDDR, see DG0618: Error Detection and Correction on SmartFusion2 Devices using DDR Memory.

The SECDED bits are interlaced with the data bits, as listed in the following table.

Table 1-10. SECDED DQ Lines at DDR

Mode	SECDED Data Pins			
	M2S/ M2GL005/010/025/060/090 M2S/M2GL150-FCV484	M2S/M2GL 050 (FCS325, VF400, FG484)	M2S/M2GL 050 (FG896)	M2S/M2GL 150 (FC1152)
Full bus width	—	—	MDDR_DQ_ECC[3:0]	MDDR_DQ_ECC[3:0]
Half bus width	MDDR_DQ_ECC[1:0]	MDDR_DQ_ECC[1:0]	MDDR_DQ_ECC[1:0]	MDDR_DQ_ECC[1:0]
Quarter bus width	MDDR_DQ_ECC[0]	—	—	MDDR_DQ_ECC[0]

When the controller detects a correctable SECDED error, it does the following:

1. Generates an interrupt signal which can be monitored by reading the interrupt status register, Table 1-98. The ECCINT interrupt is mapped to the group0 interrupt signal MSS_INT_M2F[12] in SmartFusion 2 or HPMS_INT_M2F[12] in IGLOO 2 of the Fabric Interface Interrupt Controller (FIIC).

2. Sends the corrected data to the read requested MSS/HPMS FPGA fabric master as part of the read data.

3. Sends the SECDED error information to the Table 1-86 register.

4. Performs a read-modify-write operation to correct the data present in the DRAM.

When the controller detects an uncorrectable error, it does the following:

1. Generates an interrupt signal which can be monitored by reading the interrupt status register, Table 1-98. The ECCINT interrupt is mapped to the group0 interrupt signal MSS_INT_M2F[12] in SmartFusion 2 or HPMS_INT_M2F[12] in IGLOO 2 of the FIIC.

2. Sends the data with error to the read requested MSS/HPMS FPGA fabric master as part of the read data.

3. Sends the SECDED error information to the Table 1-79 register.

The following SECDED registers are monitored for identifying the exact location of an error in the DDR SDRAM.

1. DDRC_LUE_ADDRESS_1_SR and DDRC_LUE_ADDRESS_2_SR gives the row/bank/column information of the SECDED unrecoverable error.

2. DDRC_LCE_ADDRESS_1_SR and DDRC_LCE_ADDRESS_2_SR gives the row/bank/column information of the SECDED error correction.

3. DDRC_LCB_NUMBER_SR indicates the location of the bit that caused the single-bit error in the SECDED case (encoded value).

4. DDRC_ECC_INT_SR indicates whether the SECDED interrupt is because of a single-bit error or double-bit error. The interrupt is cleared by writing zeros to Table 1-99.

1.5.4.3.5 Power Saving Modes (Ask a Question)

The DDR controller can operate DDR memories in three power saving modes:

Precharge Power-Down (DDR2, DDR3, LPDDR1)

- If power-down is enabled in the **System Builder MDDR configuration** or REG_DDRC_POWERDOWN_EN = 1, the DDR controller automatically keeps DDR memory in precharge power-down mode when the period specified by the power-down entry time or REG_DDRC_POWERDOWN_TO_X32 register has passed, while the controller is idle (except for issuing refreshes).

- The controller automatically performs the precharge power-down exit on any of the following conditions:
  - A refresh cycle is required to any rank in the system
  - The controller receives a new request from the core logic
  - REG_DDRC_POWERDOWN_EN is set to 0

Self Refresh (DDR2, DDR3, LPDDR1)

- The DDR controller keeps the DDR memory devices in Self-refresh mode whenever the self refresh is enabled and the REG_DDRC_SELFREF_EN bit in Table 1-28 is set and no reads or writes are pending in the controller.

- The controller takes the DDR memory out of Self-refresh mode whenever the REG_DDRC_SELFREF_EN input is deasserted or new commands are received by the controller.

- When the DDR self refresh is enabled, the DDR I/O bank may go into recalibration and a glitch may occur in the MDDR bank I/Os, which are being used for general purpose rather than for the DDR memory. The DDR I/Os ODT is periodically calibrated for PVT changes and will be effected only when the I/Os are in tri-state (DDR I/Os are tri stated only in self-refresh mode).

Deep Power-Down (LPDDR1)

- This is supported only for LPDDR1. The DDR controller puts the DDR SDRAM devices in deep power-down mode whenever the REG_DDRC_DEEPPOWERDOWN_EN bit in the Table 1-30 register is set and no reads or writes are pending in the DDR controller.

- The DDR controller automatically exits deep power-down mode and reruns the initialization sequence when the REG_DDRC_DEEPPOWERDOWN_EN bit is reset to 0. The contents of DDR memory may lost upon entry into deep power-down mode.

1.5.4.3.6 DRAM Initialization (Ask a Question)

After Reset, the DDR controller initializes DDR memories through an initialization sequence, depending on the type of DDR memory used. For more information on the initialization process, see the JEDEC specification.

1.5.5 MDDR Subsystem Features Configuration (Ask a Question)

The MDDR subsystem registers must be initialized before accessing DDR memory through the MDDR subsystem. When using the System Builder flow through Libero SoC, all of the necessary registers are initialized automatically by the resulting module.

This section provides the registers features of the MDDR. All registers are listed with their bit definitions in the 1.11. MDDR Configuration Registers section.

1.5.5.1 Memory Type (Ask a Question)

Table 1-31 must be configured to select the memory type (DDR2, DDR3, or LPDDR1) to access from MDDR subsystem.

1.5.5.2 Bus Width Configurations (Ask a Question)

The MDDR supports various bus widths listed in the following table. The MDDR can be programmed to work in full, half, or quarter bus width mode by configuring the Table 1-31 and Table 1-113 registers when the controller is in soft reset.

Table 1-11. Supported Bus Widths

Bus Width	M2GL005/M2GL010/M2GL025/M2GL090	M2GL050 (FCS325, VF400, FG484)	M2GL050 (FG896)	M2GL150 (FC1152)
Full bus width	—	—	✓	✓
Half bus width	✓	✓	✓	✓
Quarter bus width	✓	—	—	✓

1.5.5.3 Burst Mode (Ask a Question)

The DDR controller performs the burst write operations to DDR memory, depending on the burst mode selection. Burst mode is selected as sequential or interleaving by configuring the REG_DDRC_BURST_MODE bit in the Table 1-69 register to 1 or 0. Burst length can be selected as 4, 8, or 16 by configuring the REG_DDRC_BURST_RDWR bit in the Table 1-63 register.

Supported burst modes for DDR SDRAM types and PHY widths are listed in the following table. For M2GL050 devices, only sequential burst mode and a burst length of 8 are supported.

Table 1-12. Supported Burst Modes

Bus Width	Memory Type	Sequential/Interleaving	
		4	8
32	LPDDR1	✓	✓
	DDR2	✓	✓
	DDR3	—	✓
16	LPDDR1	—	✓
	DDR2	—	✓
	DDR3	—	✓
8	LPDDR1	—	✓
	DDR3	—	✓
	DDR2	—	—

Note: The burst length 16 is supported for LPDDR1 if bus width is 16 except M2GL050.

1.5.5.4 Configuring Dynamic DRAM Constraints (Ask a Question)

Timing parameters for DDR memories must be configured according to the DDR memory specification. Dynamic DRAM constraints are subdivided into three basic categories:

- Bank constraints affect the transactions that are scheduled to a given bank
- Rank constraints affect the transactions that are scheduled to a given rank
- Global constraints affect all transactions

1.5.5.5 Dynamic DRAM Bank Constraints (Ask a Question)

The timing constraints which affect the transactions to a bank are listed in the following table. The control bit field must be configured as per the DDR memory vendor specification.

Table 1-13. Dynamically Enforced Bank Constraints

Timing Constraint of DDR Memory	Control Bit	Description
Row cycle time (tRC)	REG_DDRC_T_RC bit in Table 1-44	Minimum time between two successive activates to a given bank.
Row precharge command period (tRP)	REG_DDRC_T_RP bit in Table 1-51	Minimum time from a precharge command to the next command affecting that bank.
Minimum bank active time (tRAS(min))	REG_DDRC_T_RAS_MIN bit in Table 1-48	Minimum time from an activate command to a precharge command to the same bank.
Maximum bank active time (tRAS(max))	REG_DDRC_T_RAS_MAX bit Table 1-48	Maximum time from an activate command to a precharge command to the same bank.
RAS-to-CAS delay (tRCD)	REG_DDRC_T_RCD bit in Table 1-51	Minimum time from an activate command to a Read or Write command to the same bank.
Write command period (tWR)	REG_DDRC_WR2PRE bit in Table 1-46	Minimum time from a Write command to a precharge command to the same bank.
Read-to-precharge delay (tRTP)	REG_DDRC_RD2PRE bit in Table 1-46	Minimum time from a Read command to a precharge command to the same bank. Set this to the current value of additive latency plus half of the burst length.

1.5.5.5.1 Dynamic DRAM Rank Constraints (Ask a Question)

The timing constraints that affect the transactions to a rank are listed in the following table. The control bit field must be configured as per the DDR memory vendor specification.

Table 1-14. Dynamically-Enforced Bank Constraints

Timing Constraints of DDR Memory	Control Bit	Description
Nominal refresh cycle time ($t_{RFC(nom)}$) or t_{REFI})	REG_DDRC_T_RFC_NOM_X32 bit in Table 1-29	Average time between refreshes for a given rank. The actual time between any two refresh commands may be larger or smaller than this; this represents the maximum time allowed between refresh commands to a given rank when averaged over a large period of time.
Minimum refresh cycle time $t_{RFC(min)}$	REG_DDRC_T_RFC_MIN bit in Table 1-28	Minimum time from refresh to refresh or activate.
RAS-to-rAS delay (t_{RRD})	REG_DDRC_T_RRD bit in Table 1-51	Minimum time between activates from bank A to bank B.
RAS-to-CAS delay (t_{CCD})	REG_DDRC_T_CCD bit in Table 1-51	Minimum time between two reads or two writes (from bank A to bank B).
Four active window (t_{FAW})	REG_DDRC_T_FAW bit in Table 1-44	Sliding time window in which a maximum of: 4 bank activates are allowed in an 8-bank design. In a 4-bank design, set this register to 0x1.

1.5.5.5.2 Dynamic DRAM Global Constraints (Ask a Question)

The timing constraints that affect global transactions are listed in the following table. The control bit field must be configured as per the DDR memory vendor specification.

Table 1-15. Dynamic DRAM Global Constraints

Timing Constraint	Control Bit	Description
Read-to-write turnaround time (tRTW)	REG_DDRC_RD2WR bit in Table 1-49	Minimum time to allow between issuing any Read command and issuing any WRITE command
Write-to-read turnaround time (tRTR)	REG_DDRC_WR2RD bit in Table 1-49	Minimum time to allow between issuing any Write command and issuing any Read command
Write latency	REG_DDRC_WRITE_LATENCY bit in Table 1-45	Time after a Write command that write data should be driven to DRAM.

The DDR memories require delays after initializing the mode registers. The following registers must be configured for the delay requirements for the DDR memories. The DDR controller uses these delay values while initializing the DDR memories.

- Table 1-38 (recommended value is 0x4242)
- Table 1-39 (recommended value is 0x8)

1.5.5.6 Address Mapping (Ask a Question)

The DDR controller maps linear request addresses to DDR memory addresses by selecting the source bit that maps to each and every applicable DDR memory address bit.

Each DDR memory address bit has an associated register vector to determine its source. The source address bit number is determined by adding the internal base of a given register to the programmed value for that register, as described in the following equation.

[Internal base] + [register value] = [source address bit number]

For example, reading the description for REG_DDRC_ADDRMAP_COLB3, the internal base is 3; so when the full data bus is in use, the column bit 4 is determined by 3 + [register value].

If this register is programmed to 2, then the source address bit is: 3 + 2 = 5.

The DDR configurator assigns values to the address mapping registers depending on the selected number of columns, rows and banks. The following illustration provides the default mapping of the memory row, bank, and column address to the user interface address domain.

Figure 1-10. Address Mapping

The following are the address mapping registers:

- Table 1-32
- Table 1-33

- Table 1-34
- Table 1-54
- Table 1-35
- Table 1-36

While configuring the registers, ensure that two DDR memory address bits are not determined by the same source address bit.

> **Important:** Some registers map multiple source address bits (REG_DDRC_ADDRMAP_ROW_B0_11).

To arrive at the right address for the DDR controller, the system address or AXI address bits [4:0] are mapped by the MDDR.

- In full bus width mode, the system address bits [4:0] are used to map the lower column address bits (C0, C1, C2)
- In half bus width mode, the system address bits [4:0] are used to map the lower column address bits (C0, C1, C2, C3)
- In quarter bus width mode, the system address bits [4:0] are used to map the lower column address bits (C0, C1, C2, C3, C4)

The MDDR configurator uses (Row, Bank, and Column) address mapping as shown in the following example.

1.5.5.6.1 Example (Ask a Question)

In this example, the Address map registers are configured to access a 512 MB DDR3 SDRAM memory (MT41J512M8RA) from the MDDR subsystem as shown in 1.9.2. Example 2: Connecting 32-Bit DDR3 to MDDR_PADs with SECDED. The 512M x 8-bit DDR3 memory module has 3 bank address lines, 16 rows, and 10 columns.

- The column address bits 3 to 9 are mapped for system address bit[5] to system address bit[11]. To map the column 3-bit (C3) to address [5], the field is configured to 3, as the base value is 2. Similarly, the other column address bits are configured:
  - DDRC_ADDR_MAP_COL_1_CR = 0x3333
  - DDRC_ADDR_MAP_COL_2_CR = 0x3FFF
  - DDRC_ADDR_MAP_COL_3_CR = 0x3300
- The bank address bits 0 to 2 are mapped for system address bit[12] to system address bit[14]. To map the bank bit0 to address [12], the field is configured to A, as the base value is 2. Similarly, the other bank address bits are configured:
  - DDRC_ADDR_MAP_BANK_CR = 0xAAA
- The row address bits 0 to 15 are mapped for system address bit[15] to system address bit[27]. To map the bank bit0 to address [15], the field is configured to 9, as the base value is 6. Similarly, the other bank address bits are configured:
  - DDRC_ADDR_MAP_ROW_1_CR = 0x9999
  - DDRC_ADDR_MAP_ROW_2_CR = 0x9FF
    Note: The MDDR can access the 2 GB address space (0x00000000 - 0x7FFFFFFF). But in this example, 512 MB (0x00000000 - 0x1FFFFFFF) DDR3 SDRAM is connected to the 16 address lines of MDDR. The memory visible in the other memory space is mirrored of this 512 MB memory.

1.5.5.7 DDR Mode Registers (Ask a Question)

After reset, the DDR controller initializes the mode registers of DDR memory with the values in the following registers. The mode registers must be configured according to the specification of the external DDR memory when the controller is in soft reset.

- Table 1-40
- Table 1-41
- Table 1-42
- Table 1-43

The T_MOD and T_MRD bits in Table 1-47 must be configured to the required delay values. T_MOD and T_MRD are delays between loading the mode registers.

1.5.5.8 SECDED (Ask a Question)

To enable SECDED mode, set the REG_DDRC_MODE bits to 101 in Table 1-31. The Table 1-113 register must be configured to enable data slice 4 of the PHY.

The register value REG_DDRC_LPR_NUM_ENTRIES in the performance register, Table 1-63, must be increased by 1 to the value used in Normal mode (without SECDED).

Important: MDDR has 36 DQ lines. These data lines are split into the following data slices:
- Data slice0 represents first 8 DQ lines (DQ0 to DQ7)
- Data slice1 represents next 8 DQ lines (DQ8 to DQ15)
- Data slice2 represents next 8 DQ lines (DQ16 to DQ23)
- Data slice3 represents next 8 DQ lines (DQ24 to DQ31)
- Data slice4 represents the remaining 4 DQ lines (DQ32 to DQ35)

1.5.5.9 Read Write Latencies (Ask a Question)

The read and write latencies between DDR controller and DDR PHY can be configured. Configure the Table 1-45 register for adding latencies for read and writes.

1.5.5.10 Performance (Ask a Question)

The DDR controller has several performance registers which are used to increase the speed of the read and write transactions to DDR memory.

The DDR controller has a transaction store, shared for low and high priority transactions. The Table 1-63 register can be configured for allocating the transaction store between the low and high priority transactions. For example, if the REG_DDRC_LPR_NUM_ENTRIES field in Table 1-63 field is configured to 0, the controller allocates more time to high priority transactions. The ratio for LPR: HPR is 1:7 (as the transaction store depth is 8).

The Table 1-64, Table 1-66, and Table 1-68 registers can be configured for the minimum clock values for treating the transactions in the HPR, LPR, and WR queue as critical and non-critical.

To force all incoming transactions to low priority, configure the Table 1-69 register. By default it is configured to force all the incoming transactions to low priority.

1.5.5.11 Refresh Controls (Ask a Question)

The DDR controller automatically issues refresh commands to DDR memory for every tRFC (min). The DDR controller can be programmed to issue single refreshes at a time (REG_DDRC_REFRESH_BURST = 0) to minimize the worst-case impact of a forced refresh cycle. It can be programmed to burst the maximum number of refreshes allowed for DDR (REFRESH_BURST = 7, for performing 8 refreshes at a time) to minimize the bandwidth lost when refreshing the pages.

1.5.5.12 1T or 2T Timing (Ask a Question)

The DRAM can be used in 1T or 2T Timing mode by configuring the Table 1-70 register. The address bus can be clocked using 1T or 2T clocking. With 1T, the DDR controller can issue a new command on every clock cycle. In 2T timing, the DDR controller holds the address and command bus valid for two clock cycles. This reduces the efficiency of the bus to one command per two clocks, but it doubles the amount of setup and hold time. The data bus remains the same for all of the variations in the address bus and the default configuration is 1T timing mode.

1.5.5.13 ODT Controls (Ask a Question)

The ODT for a specific rank of memory can be enabled or disabled by configuring the Table 1-52 and Table 1-53 registers. These must be configured before taking the controller out of soft reset. They are applied to every read or write issued by the controller.

1.5.5.14 Soft Resets (Ask a Question)

Set the REG_DDRC_SOFT_RSTB bit of Table 1-27 to 0 to reset the DDR controller. To release the DDR controller from reset, set the REG_DDRC_SOFT_RSTB bit of Table 1-74 to 1.

1.5.5.15 MDDR Memory Map (Ask a Question)

The address map to access the DDR memory from MSS/HPMS masters through MDDR is 0xA0000000-0xDFFFFFFF, which is 1 GB. But the MDDR can support up to 2 GB of memory, out of which only 1 GB of this memory is accessible at a time from the MSS/HPMS masters through the AHB bus matrix. DDR_FIC can access the entire 2 GB memory.

To enable MSS/HPMS masters to access 2 GB, the DDR address space (0x00000000-0x7FFFFFFF) is divided into 8 DDR regions, as shown in Table 1-16. Each region is 256 MB (4 regions together form 1 GB). The HPMS masters can access any of these four regions at a time, depending on the Address Space Mapping mode configured for that particular master using the DDRB_CR register in SYSREG. For SmartFusion 2, the DDRB_CR register has four 4-bit fields (DDR_IDC_MAP, DDR_SW_MAP, DDR_HPD_MAP, and DDR_DS_MAP). For IGLOO 2, the DDRB_CR register has two 4-bit fields (DDR_SW_MAP, DDR_HPD_MAP) whose bits can be configured to select the DDR Address Space Mapping modes from 0 to 12.

The Address Space Mapping modes for a 2 GB memory are shown in Table 1-17. For example, if the DDR_SW_MAP is configured as 0001, then the AHB bus matrix can access 0, 1, 2, and 3 regions of DDR that is, the accessible DDR memory from AHB bus matrix is 0x00000000-0x4FFFFFFF which is 1 GB.

Table 1-16. DDR Memory Regions

DDR Memory Region	DDR Memory Space
0	0×00000000-0×0FFFFFFF
1	0×10000000-0×1FFFFFFF
2	0×20000000-0×2FFFFFFF
3	0×30000000-0×3FFFFFFF
4	0×40000000-0×4FFFFFFF
5	0×50000000-0×5FFFFFFF
6	0×60000000-0×6FFFFFFF
7	0×70000000-0×7FFFFFFF

Table 1-17. Accessed DDR Memory Regions Based on Mode Settings for a 2 GB Memory

Address Space Mapping Modes	DDR Memory Regions Visible at MSS/HPMS DDR Address Space for Different Modes			
	MSS/HPMS DDR Space 0 (0×A0000000-0×AFFFFFFF)	MSS/HPMS DDR Space 1 (0×B0000000-0×BFFFFFFF)	MSS/HPMS DDR Space 2 (0×C0000000-0×CFFFFFFF)	MSS/HPMS DDR Space 3 (0×D0000000-0×DFFFFFFF)
0000	Region 2	Region 3	Region 4	Region 5

...........continued

Address Space Mapping Modes	DDR Memory Regions Visible at MSS/HPMS DDR Address Space for Different Modes			
	MSS/HPMS DDR Space 0 (0×A0000000-0×AFFFFFFF)	MSS/HPMS DDR Space 1 (0×B0000000-0×BFFFFFFF)	MSS/HPMS DDR Space 2 (0×C0000000-0×CFFFFFFF)	MSS/HPMS DDR Space 3 (0×D0000000-0×DFFFFFFF)
0001	Region 0	Region 1	Region 2	Region 3
0010	Region 0	Region 1	Region 2	Region 3
0011	Region 4	Region 5	Region 6	Region 7
0110	Region 0	Region 1	Region 2	Region 3
0111	Region 0	Region 1	Region 4	Region 5
1000	Region 0	Region 1	Region 6	Region 7

If 1 GB of DDR memory is connected to MDDR, only 4 regions are available (0-4). The following table shows the DDR regions available for address mode settings.

**Table 1-18.** Accessed DDR Memory Regions Based on Mode Settings for a 1 GB Memory

Address Space Mapping Modes	DDR Memory Regions Visible at HPMS DDR Address Space for Different Modes			
	MSS/HPMS DDR Space 0 (0×A0000000-0×AFFFFFFF)	MSS/HPMS DDR Space 1 (0×B0000000-0×BFFFFFFF)	MSS/HPMS DDR Space 2 (0×C0000000-0×CFFFFFFF)	MSS/HPMS DDR Space 3 (0×D0000000-0×DFFFFFFF)
0000	Region 2	Region 3	Region 0	Region 1
0001	Region 0	Region 1	Region 2	Region 3
0010	Region 0	Region 1	Region 2	Region 3

1.6 How to Use MDDR in IGLOO 2 Device (Ask a Question)

This section describes how to use MDDR in the IGLOO 2 devices. To configure the IGLOO 2 device features and then build a complete system, use the System Builder graphical design wizard in the Libero Software.

The following image shows the initial System Builder window where you can select the features that you require. For details on how to launch the System Builder wizard and a detailed information on how to use it, see IGLOO2 System Builder User Guide. You can also use CoreABC based initialization as described in IGLOO2 Standalone Peripheral Initialization User Guide.

Figure 1-11. System Builder—Device Features Window
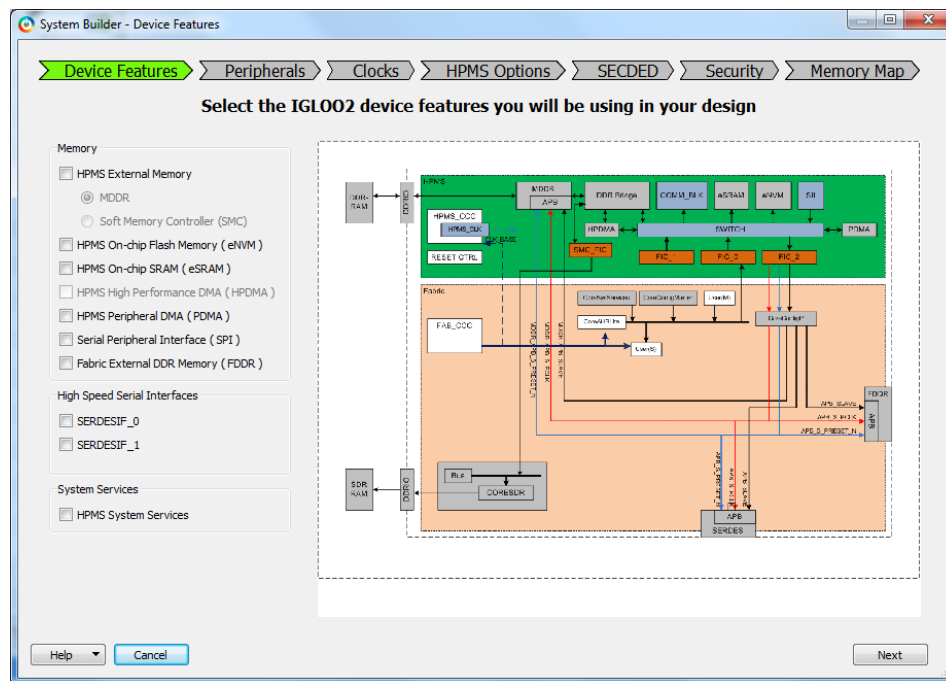
For more information about how to use MDDR in the SmartFusion 2 devices, see 1.12. How to Use the MDDR in SmartFusion 2.

1.6.1 Configuring MDDR (Ask a Question)

The following steps configure the MDDR:

1. Check the **HPMS External DDR Memory (MDDR)** check box under the **Device Features** tab and leave the other check boxes unchecked. The following image shows the **System Builder** > **Device Features**.

Figure 1-12. MDR Initialization Path
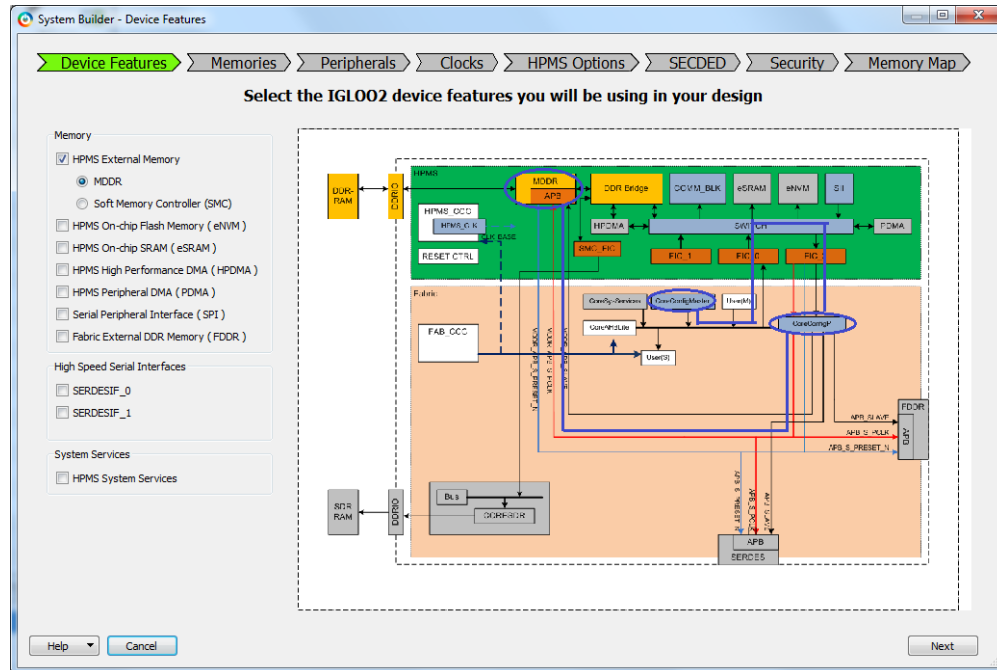
2. Selecting the MDDR under **HPMS External Memory** check box in the System Builder performs the following actions:
   – Instantiates the required IPs like CoreConfigMaster and CoreConfigP that initialize the MDDR Controller.
   – Establishes the initialization path:
     CoreConfigMaster → FIC_0 → eNVM → FIC_2 → CoreConfigP → APB bus of the MDDR subsytem
     • CoreConfigMaster (AHB Master) accesses the DDR configuration data stored in eNVM through FIC_0.
     • The configuration data is sent to CoreConfigIP through the FIC_2 master port.
     • CoreConfigP sends the configuration data to APB bus of the MDDR subsystem.
3. Navigate to the **Memories** tab. Select the memory settings under the **General** tab depending on the application requirement, as shown in Figure 1-13.
   – Memory type can be selected as DDR2, DDR3, or LPDDR.
   – Data width can be selected as 32-bit, 16-bit, or 8-bit. See Table 1-12 for supported data widths for various IGLOO2 device packages.
   – SECDED (ECC) can be enabled or disabled.
   – Arbitration Scheme can be selected from Type-0 to Type-3. See Table 1-9 for arbitration scheme details.
   – The highest priority ID of fabric master ranges from 0 to 15 if the selected arbitration scheme is other than Type-0.
   – For address mapping, the register settings that perform mapping to system address bits for row, bank, and column combinations are automatically computed by the configurator using the address mapping option. The following table lists the supported range for row, bank, and column.

Table 1-19. Supported Address Width Range for Row, Bank, and Column Addressing in DDR/LPDDR

Width	DDR2	DDR3	LPDDR
Row Address	12–16	12–16	12–16
Bank Address	2–3	2–3	2–3
Column Address	9–12	9–12	9–12

For more information see 1.5.5.6. Address Mapping.

– Select the **I/O Drive Strength** as **Half Drive Strength** or **Full Drive Strength**, as shown in Figure 1-13. The following table lists how the DDR I/O standard is configured based on this setting.

Table 1-20. DDR I/O Standard is Configured Based on I/O Drive Strength Setting

| I/O Drive Strength	Memory Type	
	DDR2	DDR3
Half Drive Strength	SSTL18I	SSTL15I
Full Drive Strength	SSTL18II	SSTL15II

Figure 1-13. I/O Drive Strength Setting
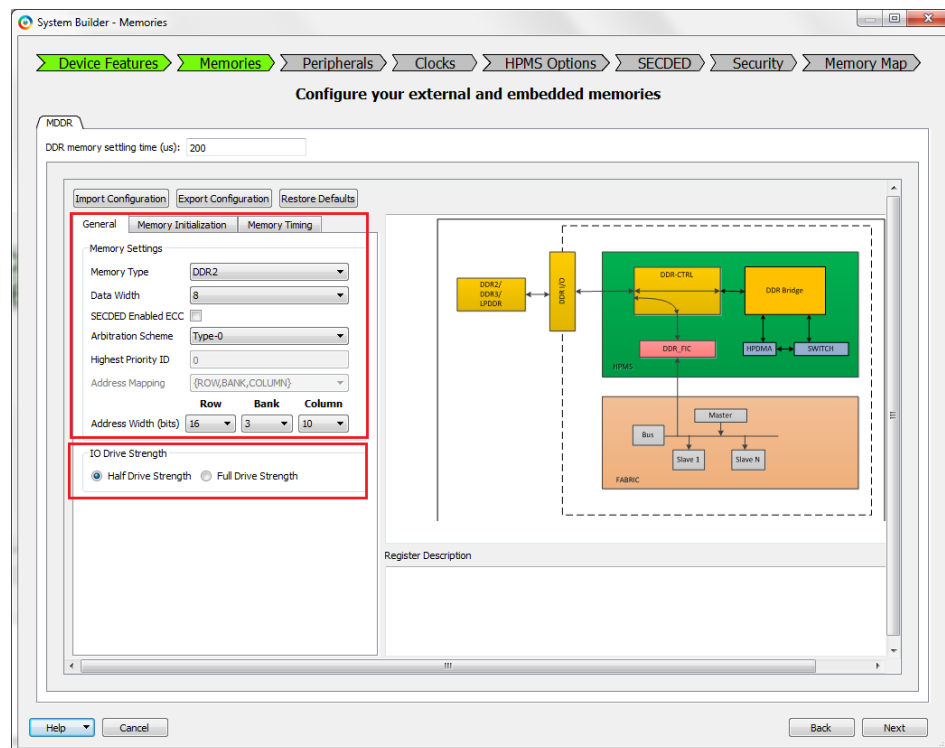
4. For only LPDDR memory, the I/O standard and I/O calibration settings are available as shown in the following illustration.

– Select I/O standard as LVCMOS18 or LPDDRI. For the Microchip M2GL_EVAL_KIT board, select LPDDRI(SSTL18) because the board is designed to use the LPDDRI I/O standard.

Important: If LVCMOS18 is selected, all I/Os are configured to LVCMOS1.8 except CLK/CLK_N.CLK and CLK_N, which are configured to the LPDDRI standard because they are differential signals.

- Select I/O calibration as ON or OFF. If I/O calibration is selected as ON, then the IGLOO2 MDDR_IMP_CALIB pin must be pulled down with a resistor. For more information on resistor values, see the Impedance Calibration section in the DS0124: IGLOO2 Pin Descriptions Datasheet.

**Figure 1-14.** Selecting I/O Standard as LVCMOS18 or LPDDRI
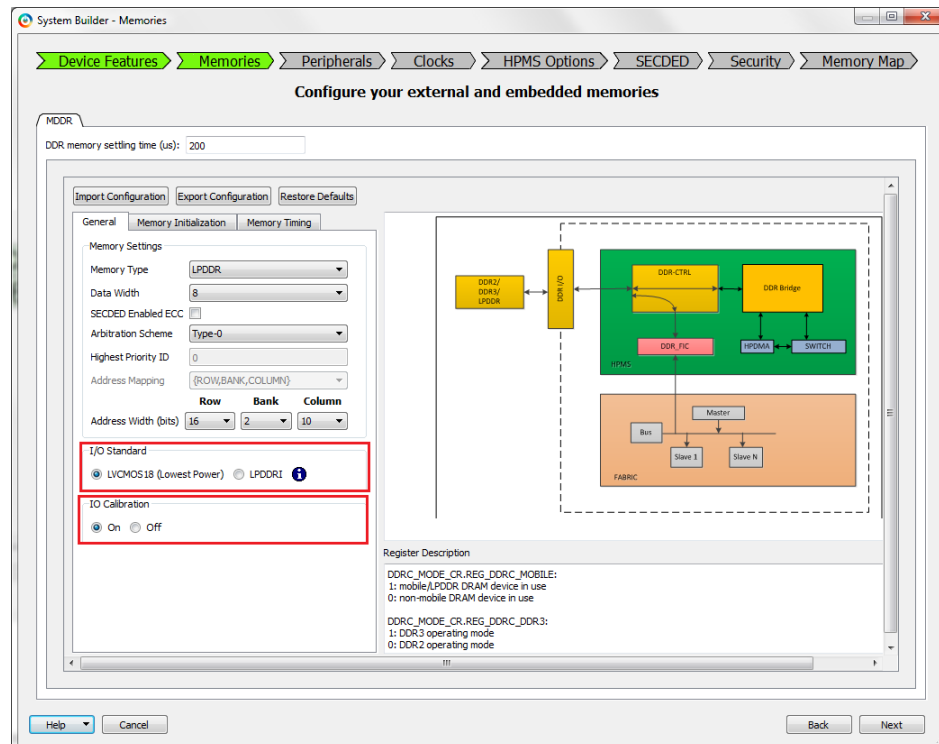
5. Depending on the application requirement, select the memory initialization settings under the **Memory Initialization** tab as shown in Figure 1-15.
6. Select the following performance-related settings:
   - Burst length can be selected as 4, 8, or 16. See Table 1-12 for supported burst lengths.
   - Burst order can be selected as sequential or interleaved. See Table 1-12 for supported burst orders.
   - Timing mode can be selected as 1T or 2T. For more information, see 1.5.5.12. 1T or 2T Timing.
   - CAS latency is the delay in clock cycles between the internal READ command and the availability of the first bit of output data. Select the CAS latency according to the DDR memory (mode register) datasheet.
7. Select the following power saving mode settings. For more information, see 1.5.4.3.5. Power Saving Modes.
   - Self-refresh enabled
   - Auto refresh burst count

- Power-down enabled
- Stop the clock (supported only for LPDDR)
- Deep power-down enabled (supported only for LPDDR)
- Power down entry time

8. Select the additional performance settings for DDR3 memory.
   - Additive CAS Latency is defined by EMR[5:3] register of DDR2 memory and by MR1[4:3] register of DDR3 memory. It enables the DDR2 or DDR3 SDRAM to allow a READ or WRITE command from DDR Controller after the ACTIVATE command for the same bank prior to tRCD (MIN). This configuration is part of DDR2 Extended Mode register and DDR3 mode register1.
   - CAS Write Latency (CWL) is defined by DDR3 MR2[5:3] and is the delay in clock cycles from the releasing of the internal write to the latching of the first data in. The overall WRITE latency (WL) is equal to CWL + AL (by default CWL is set to 5 clock cycles).

9. Select the following ZQ Calibration settings for DDR3 memory. For more details, refer to 1.5.3.3. ZQ Calibration.
   - Zqinit
   - ZQCS
   - ZQCS Interval

10. Select the other following settings.
    - The local ODT setting is not supported for LPDDR memory. For the DDR2/DDR3 memory type, the user can choose any option for "Local ODT". User can enable or disable "LOCAL ODT" during read transaction.
    - Drive strength setting is defined by EMR[7:5] register bits of LPDDR memory with drop down options of **Full**, **Half**, **Quarter**, and **One-eighth** drive strength; EMR[1] register bit of DDR2 memory with drop down options of **Full** and **Weak** drive strength; and MR1 register bits M5 and M1 of DDR3 memory with drop down options of `RZQ/6' and `RZQ/7'.
    - The partial array self-refresh coverage setting is defined by EMR[2:0] register bits of LPDDR memory with drop down options of **Full**, **Quarter**, **One-eighth**, and **One-sixteenth**. This feature improves power savings by selecting the amount of memory to be refreshed during self-refresh.
    - RTT (Nominal) setting is defined by EMR[6] and EMR[2] register bits of DDR2 memory, which determines what ODT resistance is enabled with drop down options of **RTT disabled**, **50 ohms**, **75 Ω**, and **150 Ω**, and it is defined by MR1[9], MR1[6] and MR1[2] register bits of DDR3 memory. In DDR3 memory, RTT nominal termination is allowed during standby conditions and WRITE operations, not during READ operations with drop down options of **RZQ/2**, **RZQ/4**, and **RZQ/6**.
    - RTT_WR (Dynamic ODT) setting is defined by MR2[10:9] register bits of DDR3 memory. This is applicable only during WRITE operations. If dynamic ODT (Rtt_WR) is enabled, DRAM switches from normal ODT (RTT_nom) to dynamic ODT (Rtt_WR) when beginning WRITE burst and subsequently switches back to normal ODT at the end of WRITE burst. The drop down options provided to the user are **off**, **RZQ/4**, and **RZQ/2**.
    - The auto self-refresh setting is defined by MR2[6] register bit of DDR3 memory with drop down options `Manual' and `Auto'. The self-refresh temperature setting is defined by MR2[7] register bit of DDR2 memory with drop down options of `Normal' and `Extended'.

Figure 1-15. Memory Initialization Configuration
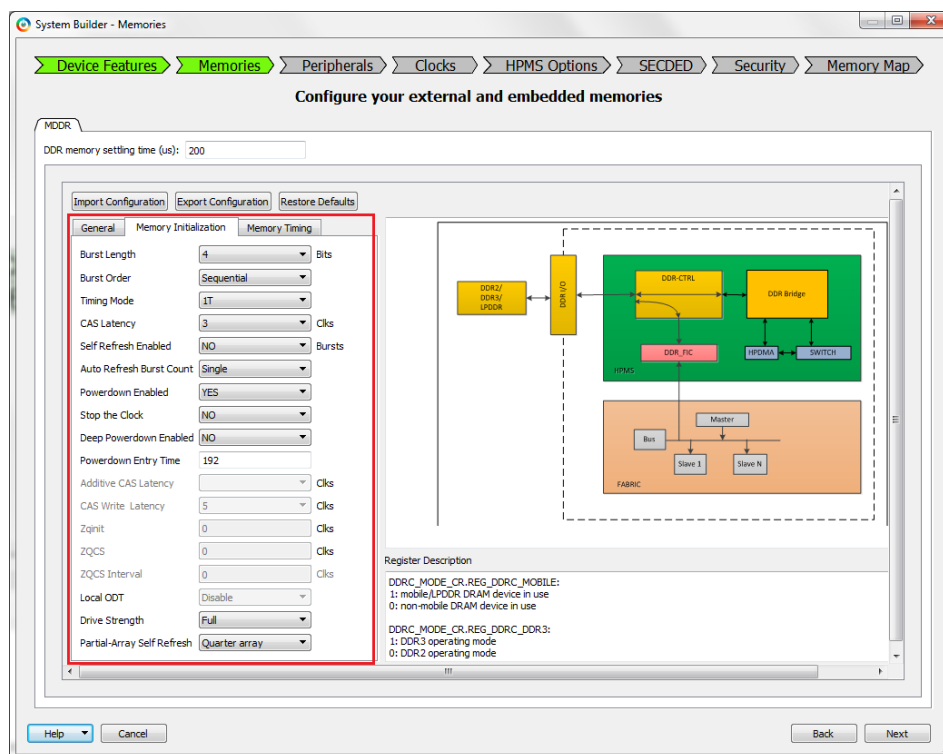
11. Select the memory timing settings under the **Memory Timing** tab according to the DDR memory vendor datasheet, as shown in the following image. For more information, see 1.5.5.4. Configuring Dynamic DRAM Constraints.

Figure 1-16. Memory Timing Configuration
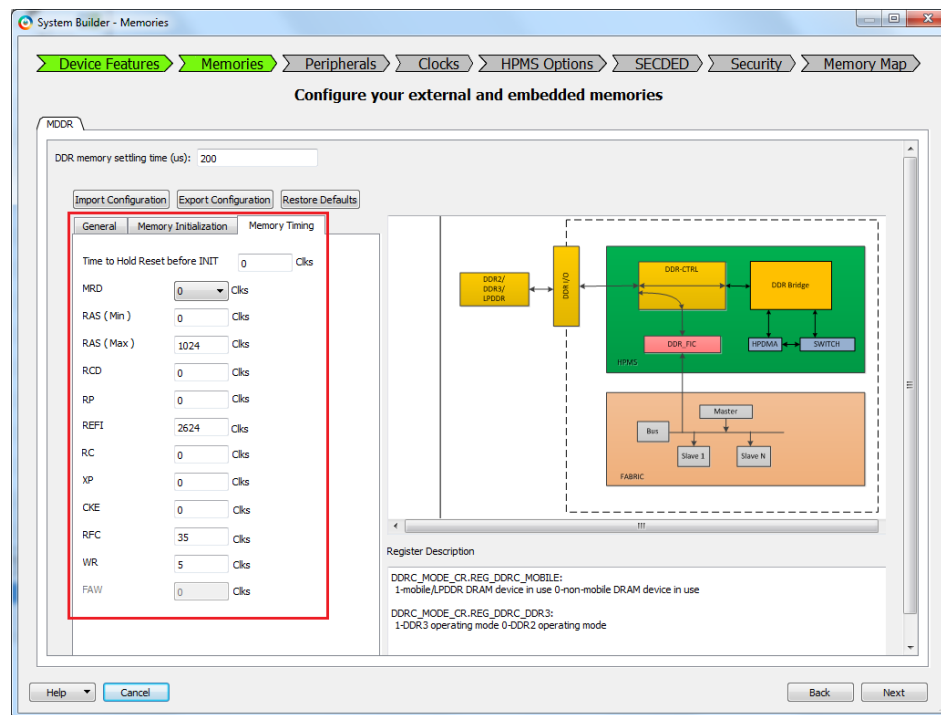
The configurator also provides the option to import and export the register configurations. The configuration settings are stored in eNVM. Configuration files for accessing LPDDR memory on IGLOO 2 evaluation kit can be downloaded from:

www.microsemi.com/soc/documents/LPDDR_Emcraft_Config.zip.

The following is an example of MDDR register configurations for operating the LPDDR memory (MT46H64M16LF) with clock 166 MHz.

• Device Memory Settling Time (µs): 200

The DDR memories require settling time for the memory to initialize before accessing it. The LPDDR memory model MT46H64M16LF needs 200 µs settling time.

• General
    – Memory Type: LPDDR
    – Data Width: 16
• Memory Initialization
    – Burst length: 8
    – Burst Order: Interleaved
    – Timing Mode: 1T
    – CAS Latency: 3
    – Self Refresh Enabled: No
    – Auto Refresh Burst Count: 8
    – PowerDown Enabled: Yes
    – Stop the clock: No
    – Deep PowerDown enabled: No

- No Activity clocks for Entry: 320
- Memory Timing
  - Time To Hold Reset Before INIT: 67584 clks
  - MRD: 4 clks
  - RAS (Min): 8 clks
  - RAS (Max): 8192 clks
  - RCD: 6 clks
  - RP: 7 clks
  - REFI: 3104 clks
  - RC: 3 clks
  - XP: 3 clks
  - CKE: 3 clks
  - RFC: 79 clks
  - FAW: 0 clks
    Navigate to the **Peripherals** tab. The **Peripherals** tab allows configuration of the Fabric AMBA Master and Fabric AMBA Slave required for the design. Drag and drop the required master/slave to the corresponding subsystem. The following image shows the Peripherals tab. Drag and drop the Fabric Master core to the HPMS DDR FIC Subsystem. This allows to the interface to be configured as AXI or single AHB-Lite. On completing the configuration, the selected interface is enabled. The user logic in the FPGA fabric can access the DDR memory through the MDDR using these interfaces.

Figure 1-17. System Builder - Peripherals Tab
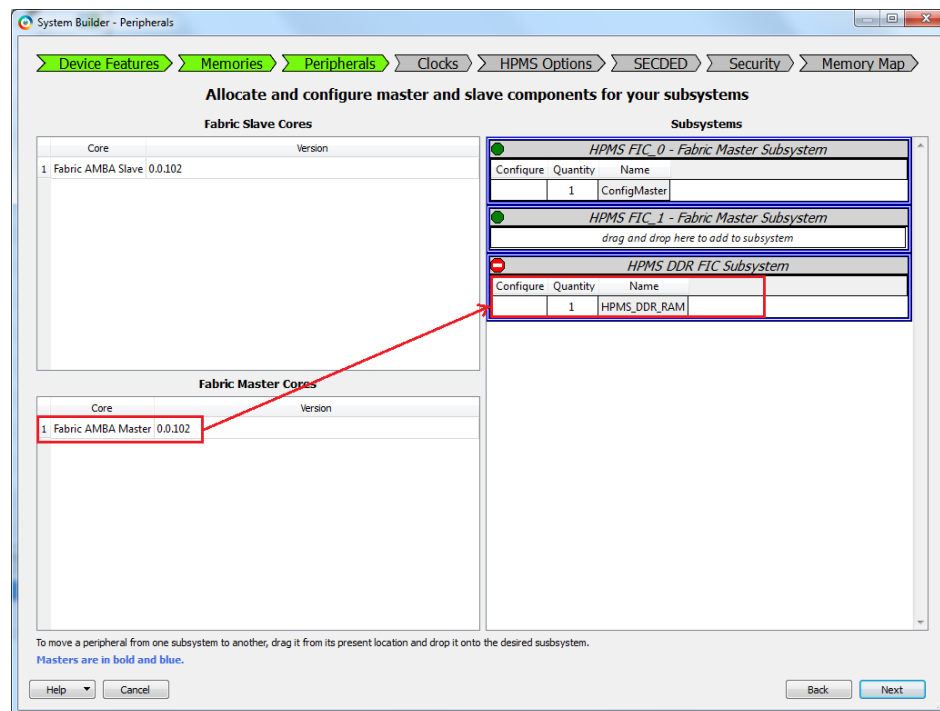
Navigate to the **Clocks** tab. The **Clocks** tab allows configuration of the System Clock and subsystem clocks. The MDDR subsystem operates on MDDR_CLK, which comes from

HPMS_CCC. The MDDR_CLK must be selected as multiples of 1, 2, 3, 4, 6, or 8 of HPMS_CLK. This clock can be configured using the HPMS_CCC configurator. The maximum frequency of MDDR_CLK is 333.33 MHz. The following illustration shows the MDDR_CLK configuration.

Figure 1-18. MDDR_CLK Configuration

DDR_FIC_CLK drives the DDR_FIC slave interface and defines the frequency at which the FPGA fabric subsystem connected to this interface is intended to run. DDR_FIC_CLK can be configured as a ratio of MDDR_CLK (1, 2, 3, 4, 6, 8, 12, 16, or 32) using the Clocks configurator. The maximum frequency of DDR_FIC_CLK is 200 MHz. The following illustration shows the DDR_FIC_CLK configuration.

Figure 1-19. DDR_FIC_CLK Configuration

If the MDDR_CLK ratio to HPMS_CLK is a multiple of 3, DDR_FIC_CLKs ratio to MDDR_CLK must also be a multiple of 3, and vice versa. The configuration issues an error if this requirement is not met. This limitation is imposed by the internal implementation of the HPMS CCC.

1.6.1.1 I/O Configuration (Ask a Question)

In the I/O Editor window, as shown in the following illustration, configure I/O settings such as ODT and drive strength.

Figure 1-20. I/O Editor Window

© 2023 Microchip Technology Inc. and its subsidiaries

1.6.2 Accessing MDDR from FPGA Fabric through the AXI Interface (Ask a Question)

The AXI master in the FPGA fabric accesses the DDR memory through the MDDR subsystem. The following illustration shows the MDDR subsystem with the AXI interface. The MDDR registers are configured from the FPGA fabric using the CoreConfigMaster IP through the CoreConfigP IP APB interface.

Figure 1-21. MDDR with AXI Interfaces
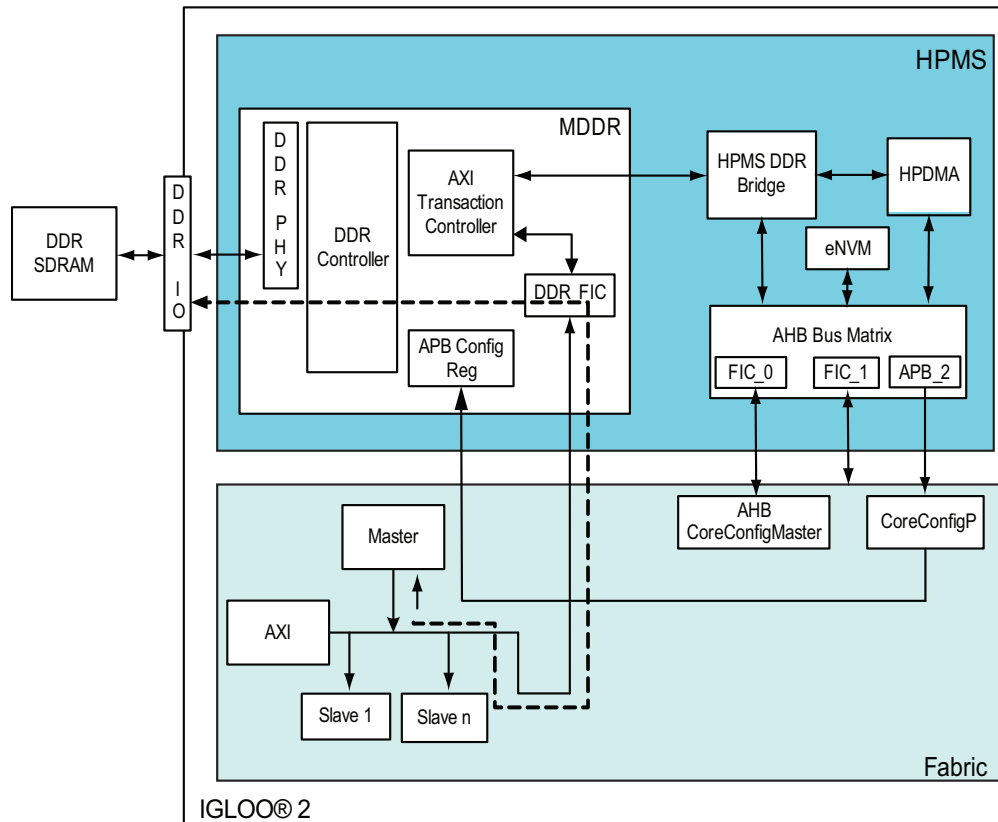
Read, write, and read-modify-write transactions are initiated by the AXI master to read from or write the data to the DDR memory after initializing the MDDR registers.

The following steps describe how to access the MDDR from AXI master in the FPGA fabric:

1. Go to the **System Builder** > **Device Features** tab, check the **HPMS External DDR Memory** check box, and select **MDDR**. Leave the rest of the check boxes unchecked. The following illustration shows the **System Builder** > **Device Features tab**.

Figure 1-22. System Builder - Device Features Tab
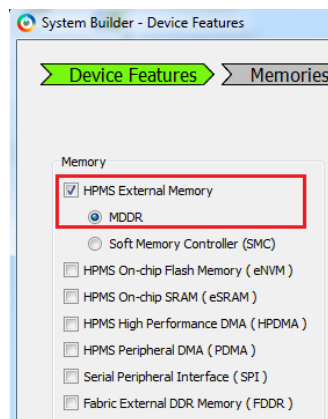
2. Configure the **HPMS External Memory** in the **Memories** tab as shown in the following illustration. In this example, the design is created to access DDR3 memory with a 32-bit data width and no ECC.

3. Set the **DDR memory settling time** to 200 µs and click **Import Register Configuration**.

Figure 1-23. Memory Configuration
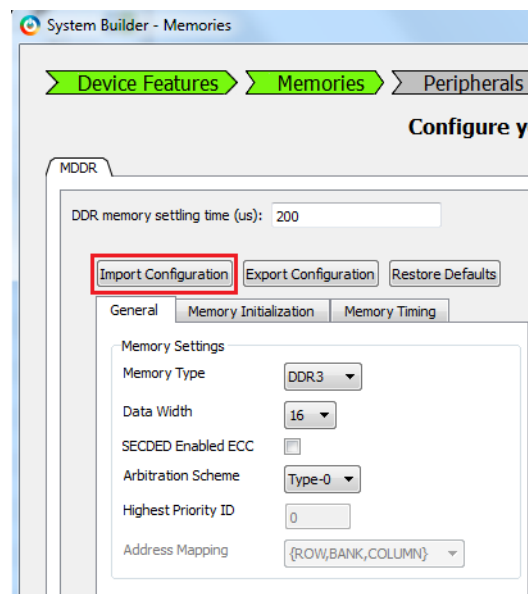
4. Navigate to the **Peripherals** tab.

5. In the **Peripherals** tab, drag the **Fabric Master Core** and drop on to the **HPMS DDR FIC Subsystem**. You can see that the master is added to the subsystem. The following image shows the **Peripherals** tab with the **AMBA_MASTER_0** added.

6. Click the **Configure** icon to open the **AMBA Master - Configuration** dialog. The following image shows the **Peripherals** tab with the **Configure** icon highlighted.

Figure 1-24. Peripherals Tab with the Master Added and Configure Icon Highlighted
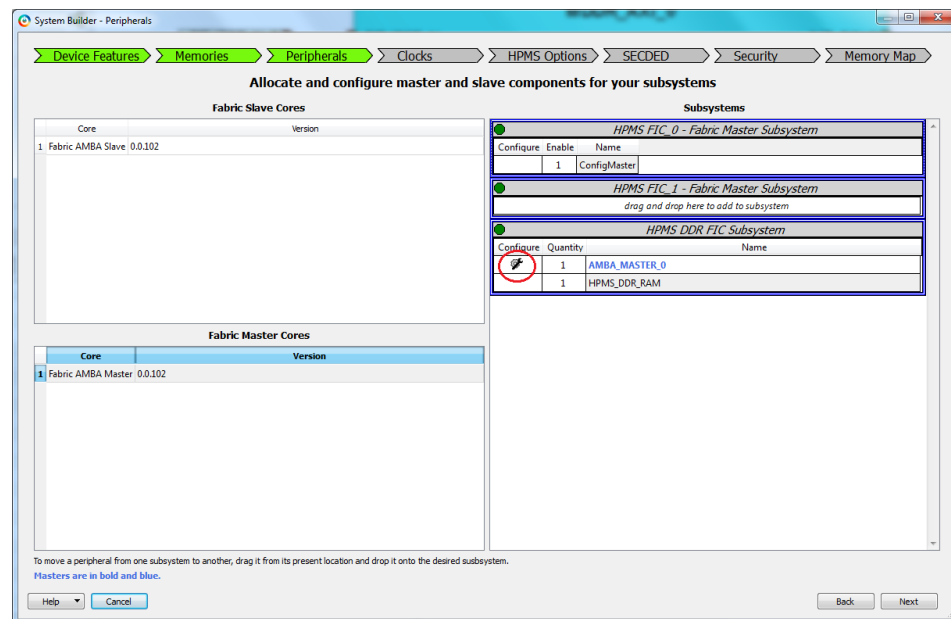
7. In the **Configuring AMBA_MASTER_0** dialog, select the **Interface Type** as **AXI** and then click **OK**. The following image shows the **AMBA Master - Configuration** dialog.

Figure 1-25. AMBA Master Configuration
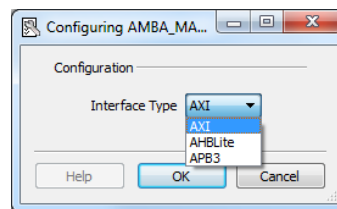
8. Configure the System Clock and Subsystem clocks in the **Clocks** tab. The following image shows the Clocks configuration dialog.
    –   Select the On-chip 25/50 MHz RC oscillator.
    –   Configure HPMS_CCC for MDDR_CLK and DDR_FIC_CLK.
9. Configure **HPMS_CLK**, **DDR_FIC_CLK**, **APB_0_CLK**, **FIC_0_CLK** to 111 MHz and **MDDR clock** as 333 MHz.

Figure 1-26. System Clocks Configuration
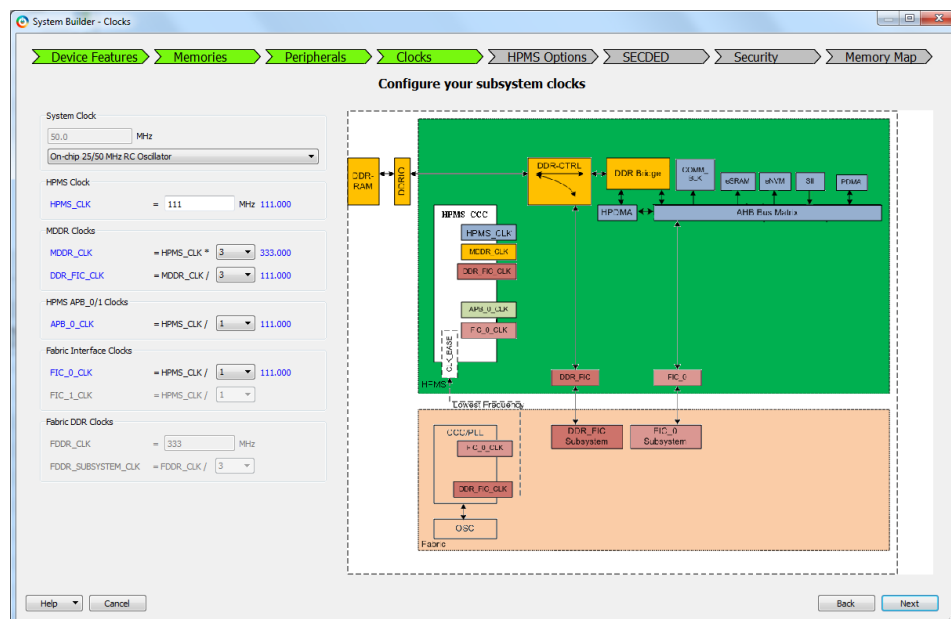
10. Navigate to the **Memory Map** tab giving the required data in the rest of the **System Builder** tabs.

11. Instantiate your AXI master logic in the SmartDesign canvas to access the MDDR subsystem through the AXI interface. Ensure that the AXI master logic accesses the MDDR after configuring the MDDR registers (INIT_DONE indicates the successful MDDR initialization).

12. Connect the AXI_Master logic signals as follows:
    – RESET_N to INIT_DONE
    – CLK to HPMS_DDR_FIC_SUBSYSTEM_CLK
    – LOCK to HPMS_DDR_FIC_SUBSYSTEM_LOCK
    – AXI_S_RMW to MDDR_DDR_AXI_S_RMW

The following illustration shows the rest of the connections in the top level design.

Figure 1-27. SmartDesign Connections (Top Level View)
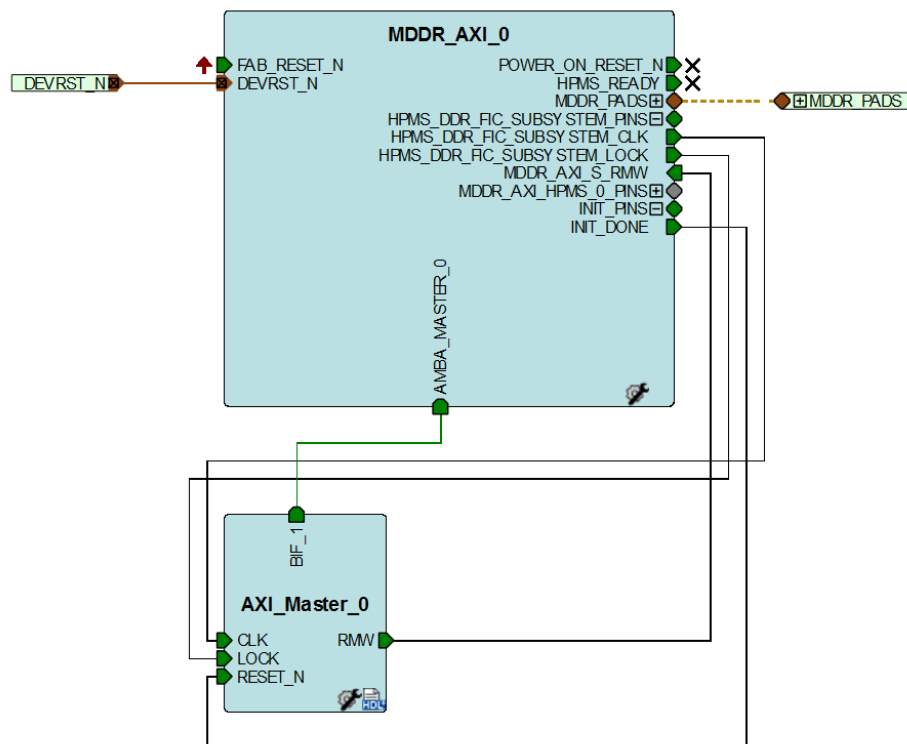
For MDDR AXI throughput, see AC422: SmartFusion2 - Optimizing DDR Controller for Improved Efficiency - Libero v11.7 Application Note.

1.6.3 Accessing MDDR from FPGA Fabric Through the AHB Interface (Ask a Question)

The MDDR subsystem is used to access the DDR memory using the AHB-Lite interface. The following illustration shows the MDDR with AHB-Lite interface.

Figure 1-28. MDDR with Single AHB-Lite Interface
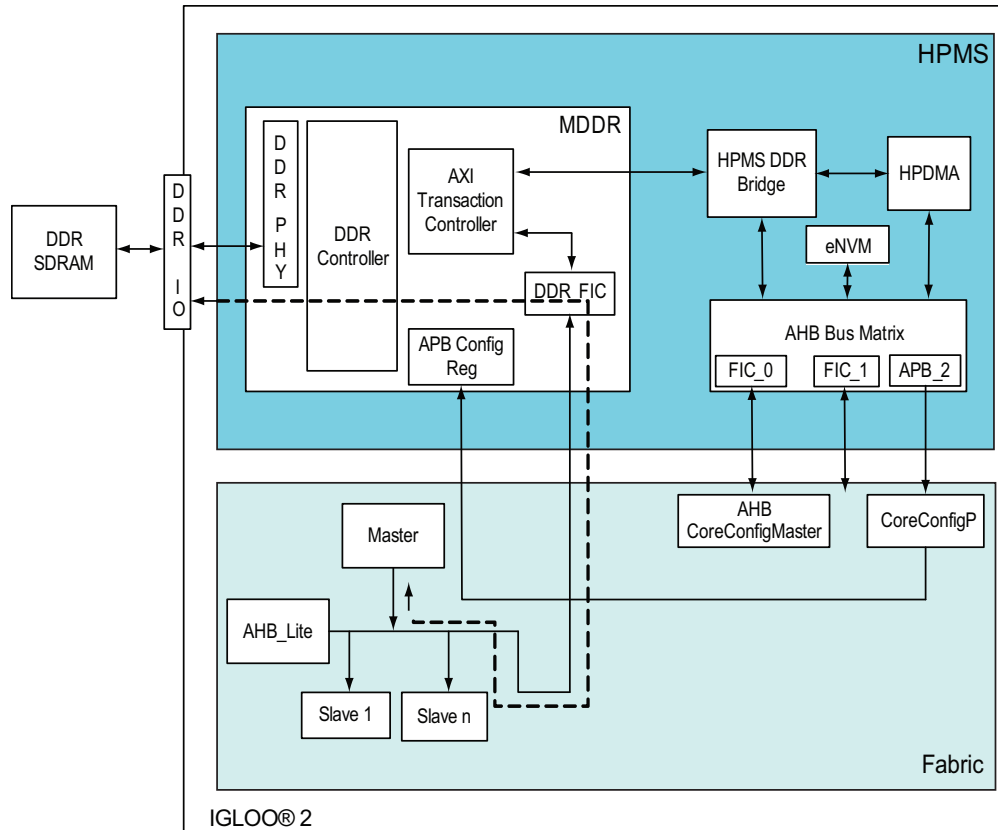
The procedure for accessing the MDDR from AHB master in the FPGA fabric is the same as in 1.6.2. Accessing MDDR from FPGA Fabric through the AXI Interface—except for the following:

- Configure the AMBA Master Interface Type as AHB-Lite in the HPMS DDR FIC Subsystem in the Peripherals tab of the System Builder wizard.

Table 1-21 lists the MDDR throughput for the following configuration:

- Fabric Interface: AHB
- MDDR Mode: DDR3
- Fabric Clock to MDDR Clock Ratio: 1:4
- PHY Width: 16 and 32
- Clock Frequency: 80 MHz

The other parameters are configured similar to the MDDR configuration in AC422: SmartFusion2 - Optimizing DDR Controller for Improved Efficiency - Libero v11.7 Application Note.

Table 1-21. MDDR Throughput (for AHB)

MDDR-Fabric Interface-Memory	Frequency Ratio (CLK_BASE:FDDR_CLK)	PHY Width	Write Transaction BW (MB/sec)	Read Transaction BW (MB/sec)
MDDR_AHB-DDR3	1:4 80 MHz:320 MHz	PHY_16	80 MB	79 MB
		PHY_32	80 MB	79 MB

1.6.4 Accessing MDDR from the HPDMA (Ask a Question)

The HPDMA controller can access DDR SDRAM connected to the MDDR subsystem through the HPMS DDR bridge. The following illustration shows the MDDR with HPDMA.

Figure 1-29. MDDR with HPDMA
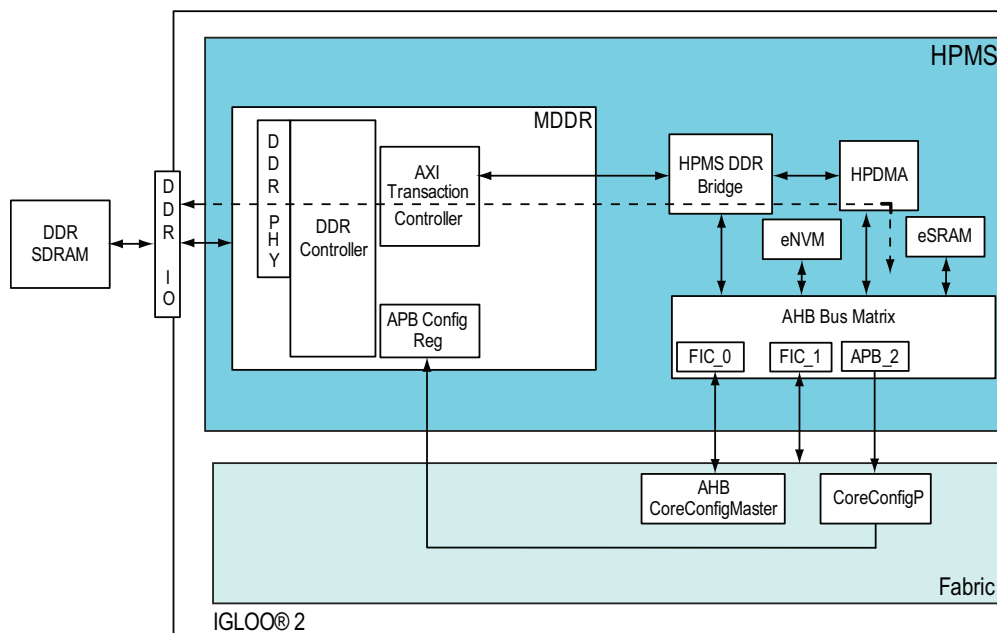
The following steps describe how to access the MDDR from HPDMA:

1. Open the **System Builder** > **Device Features** tab. Check the **HPMS External DDR Memory** check box, select **MDDR** and **HPMS High Performance DMA (HPDMA)** check boxes, leaving the rest of the check boxes unchecked. The following image shows the **System Builder** > **Device Features tab**.

   Figure 1-30. System Builder - Device Features Tab
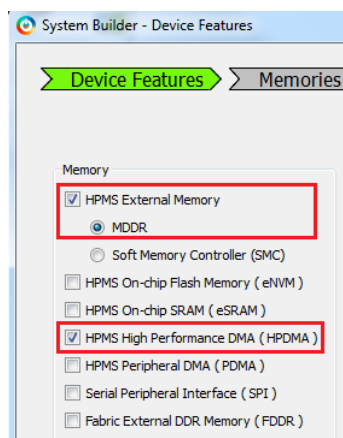
2. Configure the **HPMS External Memory** in **Memories** tab as shown in the following image. In this example, the design is created to access the DDR3 memory with a 32-bit data width and no ECC.

3. Set the **DDR memory settling time** to 200 µs and click **Import Register Configuration**.

Figure 1-31. Memory Configurations
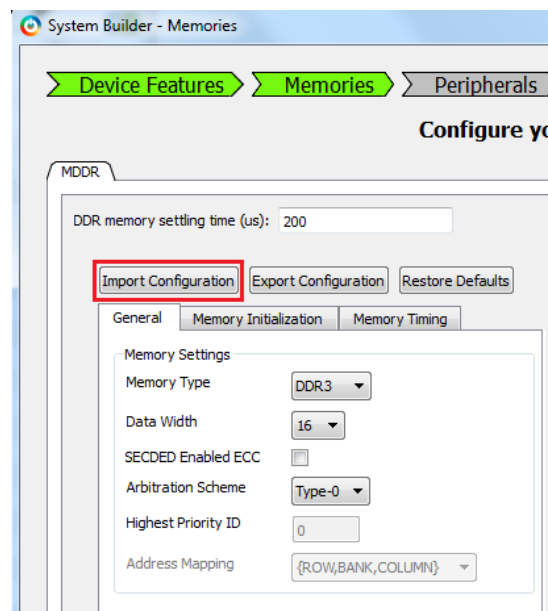
4. Configure the **System Clock** and Subsystem clocks in the **Clocks** tab. The following image shows the Clocks configuration dialog.
   – Select the On-chip 25/50 MHz RC Oscillator
   – Configure HPMS_CCC for MDDR_CLK

5. Configure **HPMS_CLK**, **APB_0_CLK**, **FIC_0_CLK** clocks as 111 MHz and the **MDDR_CLK** clock as 333 MHz.

**Figure 1-32.** Clocks Configuration
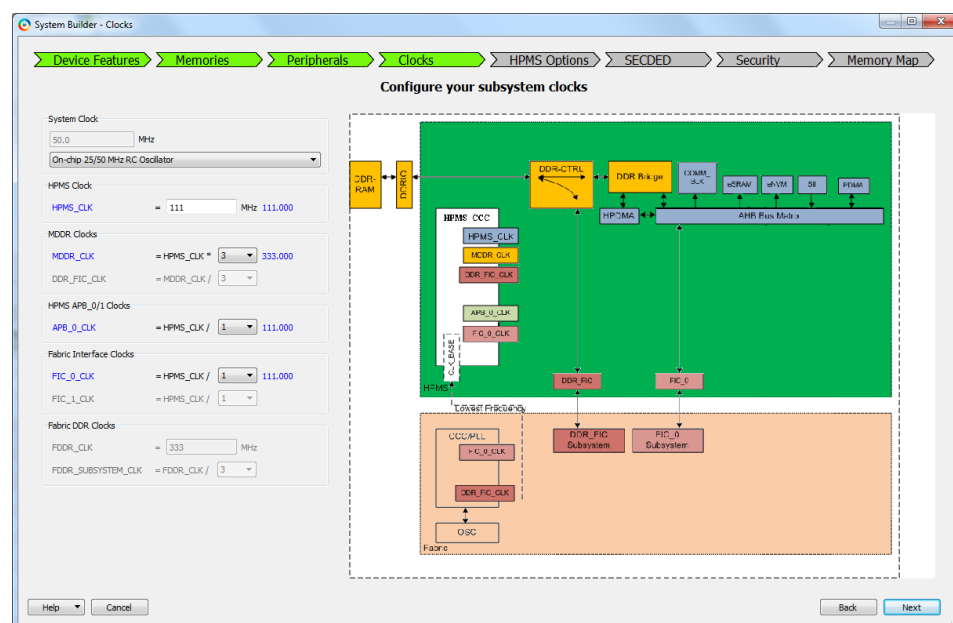
6. Navigate to the **Memory Map** tab giving the required data in the rest of the **System Builder** tabs.

For more Information on how to use HPDMA, refer to the HPDMA chapter in UG0448: IGLOO2 High Performance Memory Subsystem User Guide.

1.7 Timing Diagrams (Ask a Question)

This section shows the operation of the DDR controller with AXI interface with timing diagrams. The DDR3 16-bit micron memory model is used to perform the read and write transactions from MDDR Fabric Interface (DDR_FIC). The AXI/AHB clock is configured for 166 MHz and MDDR clock is configured for 332 MHz, that is, FIC clock to MDDR clock ratio is 1:2.

Figure 1-33. AXI Single Write Transaction and Corresponding DDR Controller Commands
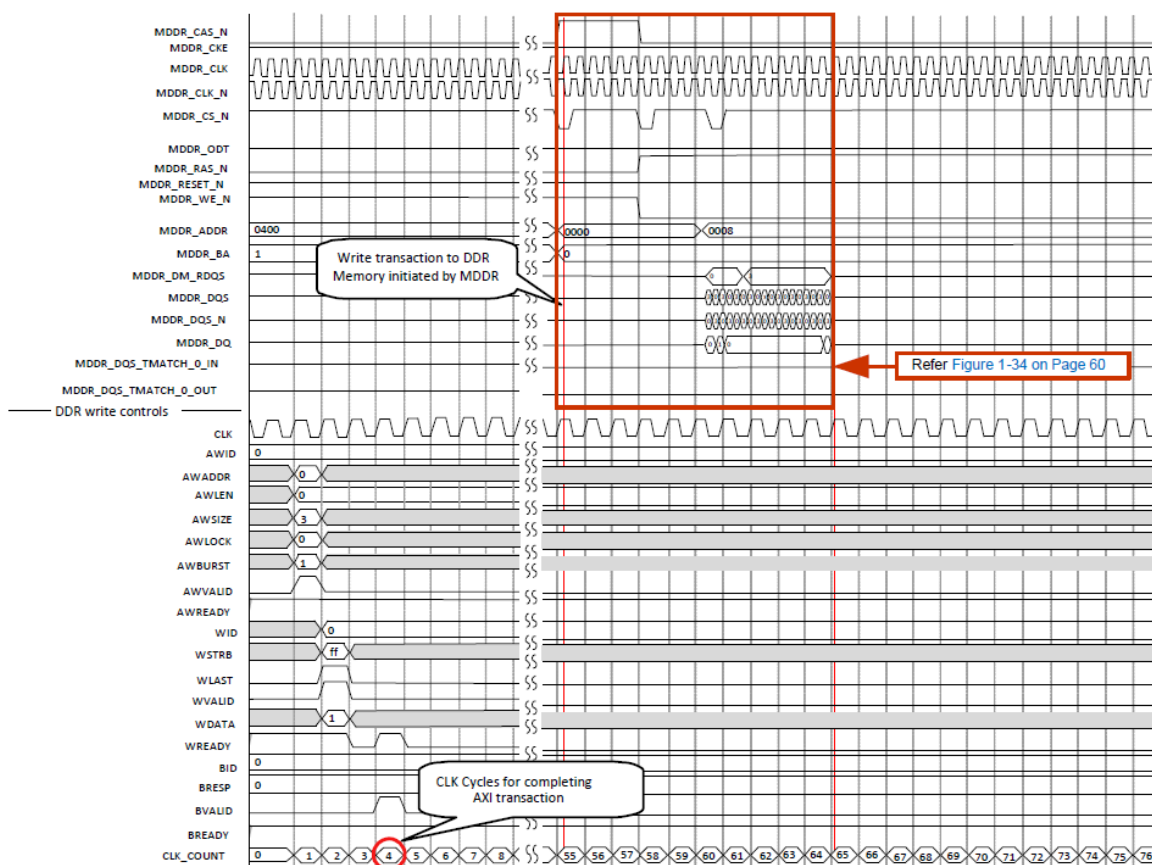
Figure 1-34. DDR Controller Command Sequence for Single AXI Write Transaction
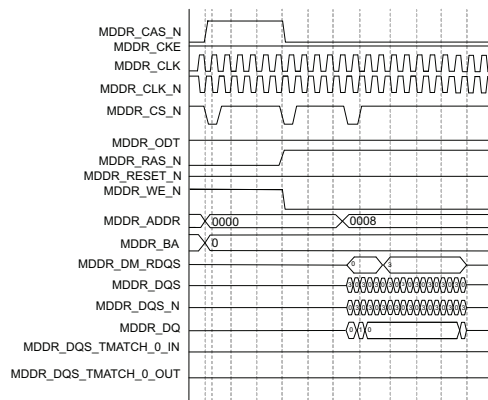
Figure 1-35. AXI Single Read Transaction and Corresponding DDR Controller Commands
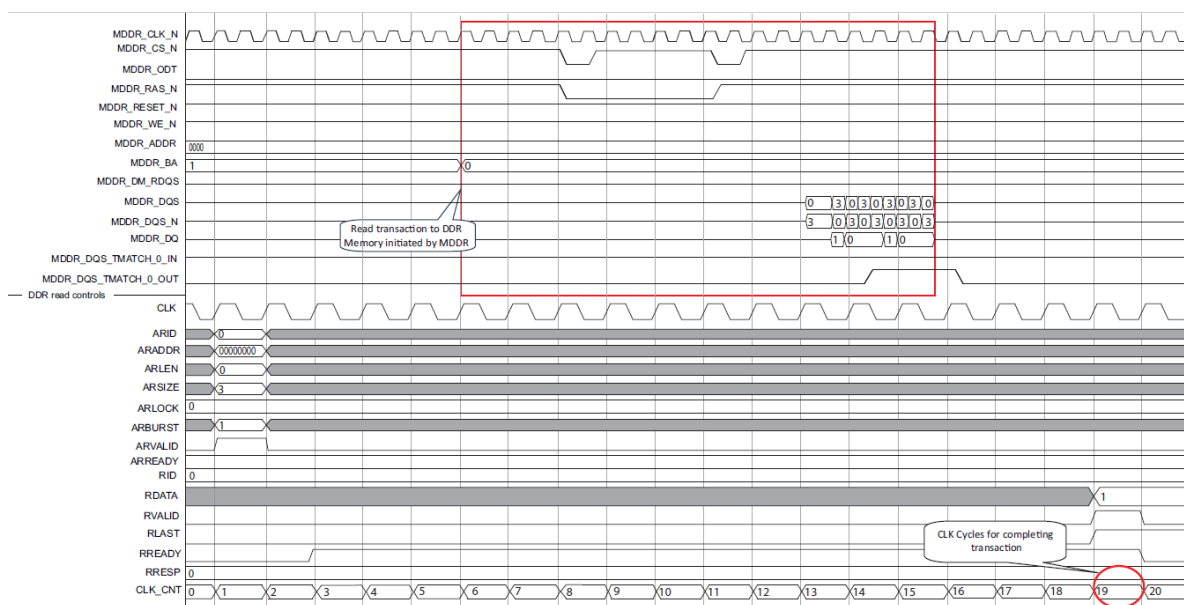
Figure 1-36. AXI INCR16 Write Transaction and Corresponding DDR Controller Commands
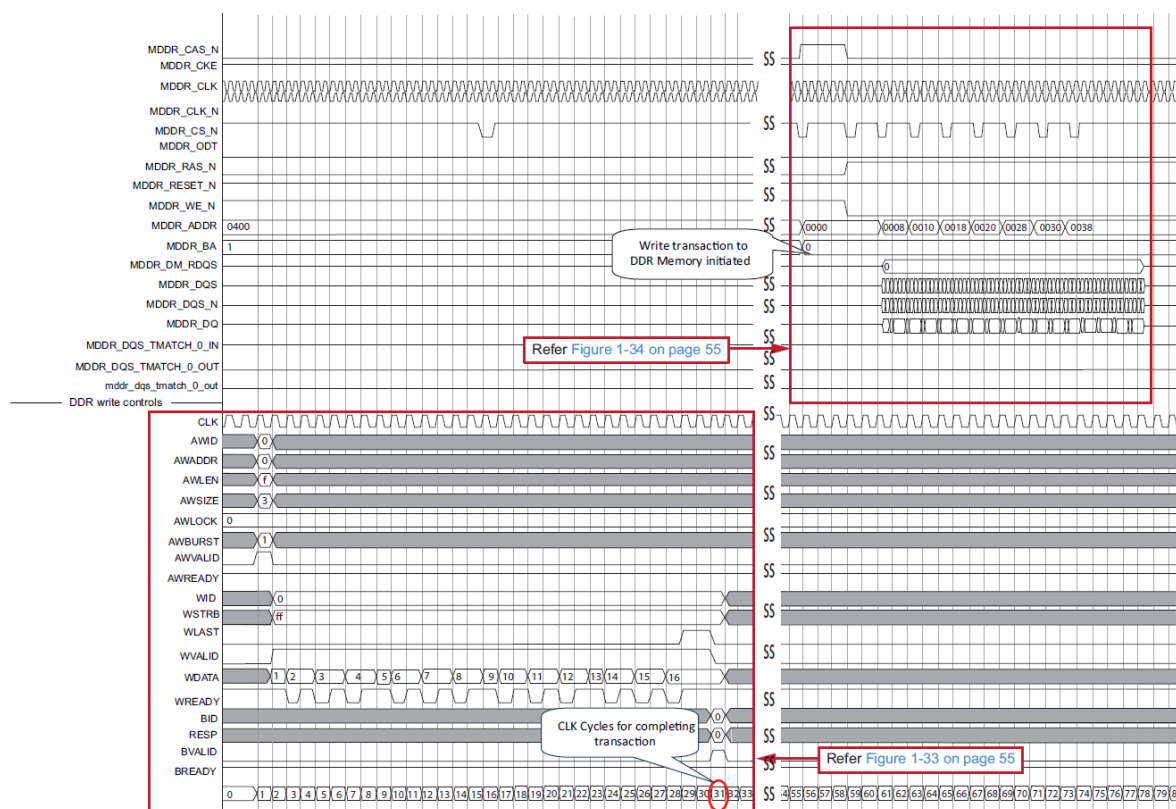
Figure 1-37. AXI INCR16 Write Transaction
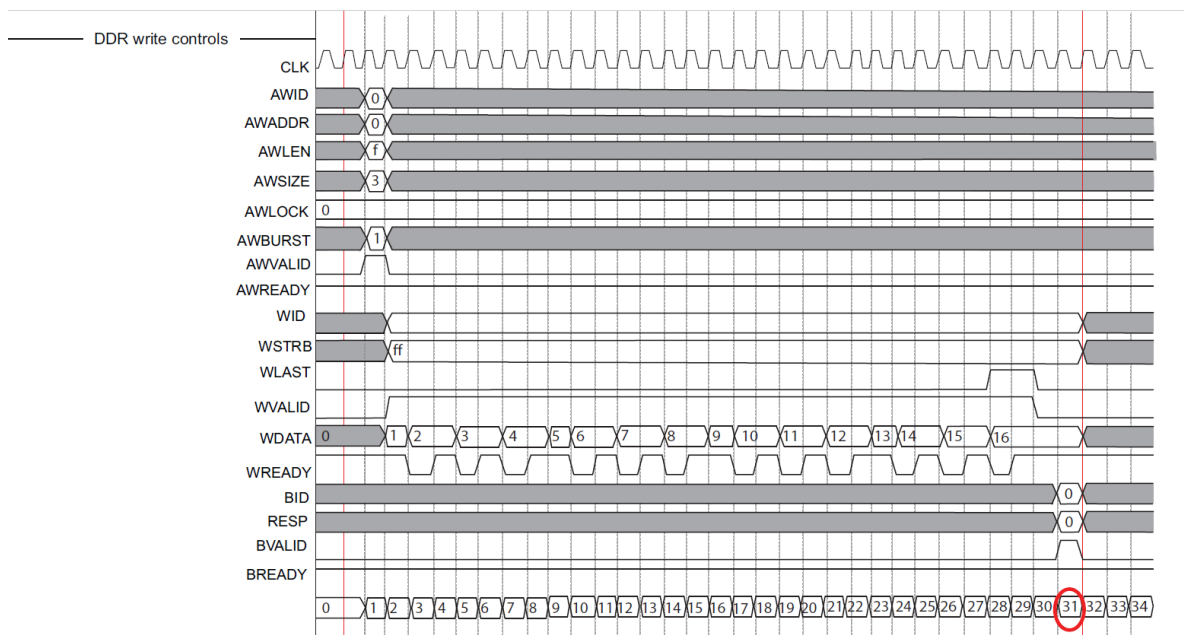
MICROCHIP

Figure 1-38. DDR Controller Command Sequence for AXI INCR16 Write Transaction
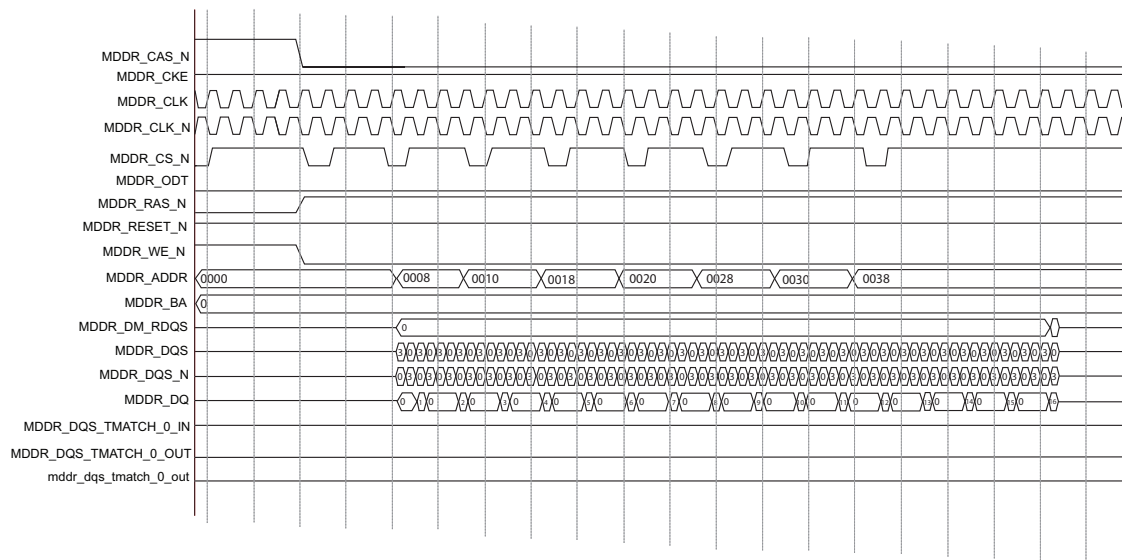
Figure 1-39. AXI INCR-16 Read Transaction and Corresponding DDR Controller Commands

Figure 1-40. DDR Controller Command Sequence for AXI INCR-16 Read Transaction

The following table summarizes the number of cycles to complete the AXI/AHB transactions to MDDR.
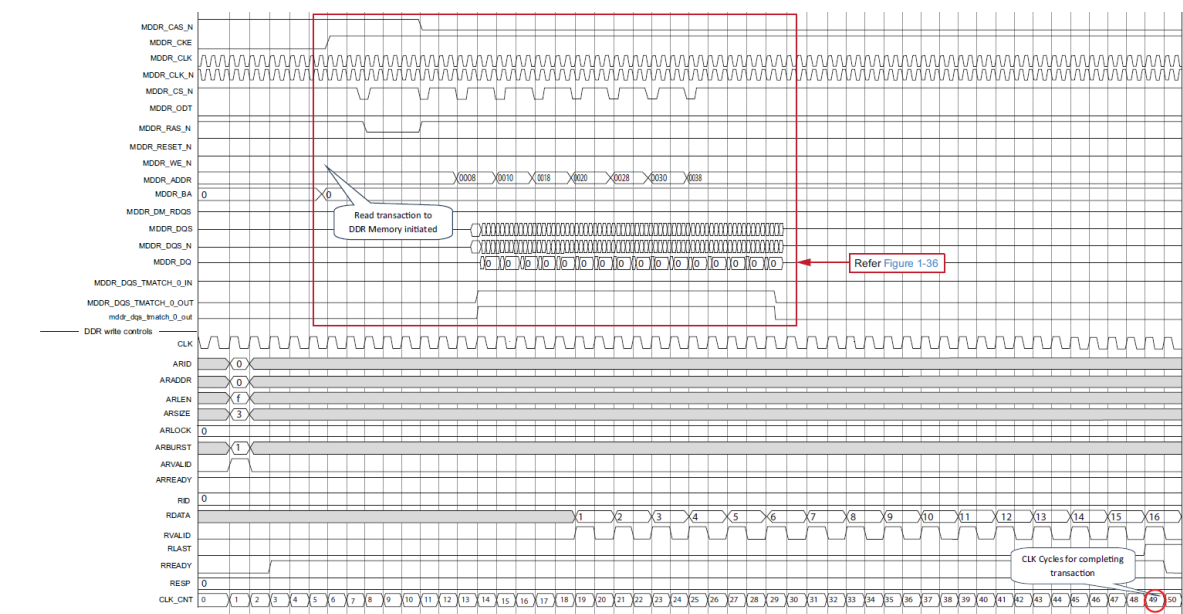
Table 1-22. Number of Cycles for AXI/AHB Transactions to MDDR

Transaction Type	Write Cycles	Read Cycles
AXI Single	4	19
AXI INCR16 Burst	31	49

1.8 Timing Optimization Technique for AXI (Ask a Question)

The AXI mode of the MDDR or FDDR provides the highest throughput interface to the external memory device. The best interface ratio for clocking is 2:1 ratio which keeps the fabric clock and fabric interface running at the same rate as the external memory device. For these types of interfaces the following technique provides an optimization method for timing closure when using the 2:1 interface. Timing closure can be achieved by Timing Optimization Technique when the timing closure is not met with the design.

The optimization method resides between an existing AXI master and the DDR_FIC AXI slave interface and no changes are required to the AXI master design. The following illustration shows a diagram of the technique, which uses a negative edge register on the VALID lines.

Figure 1-41. AXI Timing Optimization Logic
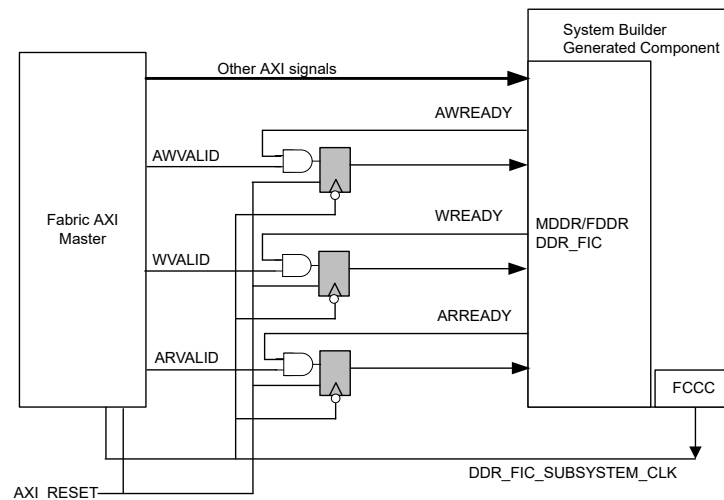
The AXI data lines into the DDR_FIC can now be relaxed with additional half AXI clock cycle as the AXI valid signals are delayed by half AXI clock cycle. The following illustration shows the AXI transaction with the optimization logic.

Figure 1-42. Timing Diagram
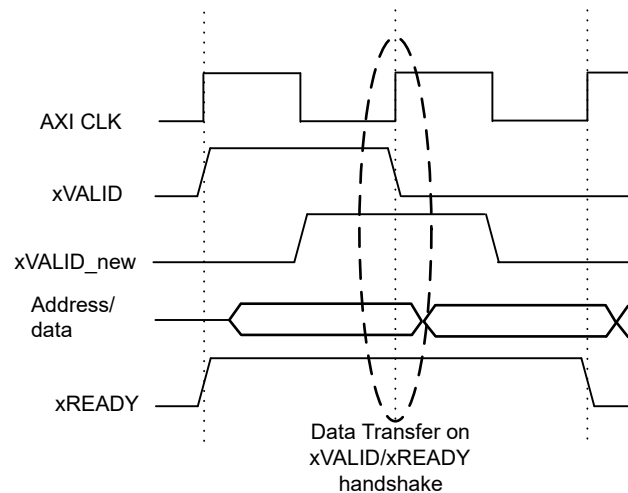
The following SDC constraints need to be added to the timing SDC file. It applies the proper timing relaxation on the DDR_FIC_AXI signals.

For FDDR:

```
/* The following constraints provide a relaxation constraint on the signals of 1.5 clock
periods.
The user should adjust the ddr_clock_frequency to match their application. */
set ddr_clock_frequency 333
set delay1 [ expr 3000/$ddr_clock_frequency ]
set max_delay $delay1 -to [ get_pins { \
*/INST_FDDR_IP:F_ARADDR* \
*/INST_FDDR_IP:F_ARBURST* \
*/INST_FDDR_IP:F_ARID* \
*/INST_FDDR_IP:F_ARLEN*\
*/INST_FDDR_IP:F_ARLOCK* \
```

```
*/INST_FDDR_IP:F_ARSIZE* \
*/INST_FDDR_IP:F_AWADDR* \
*/INST_FDDR_IP:F_AWBURST* \
*/INST_FDDR_IP:F_AWID* \
*/INST_FDDR_IP:F_AWLEN* \
*/INST_FDDR_IP:F_AWLOCK* \
*/INST_FDDR_IP:F_AWSIZE* \
*/INST_FDDR_IP:F_WDATA* \
*/INST_FDDR_IP:F_WID* \
*/INST_FDDR_IP:F_WLAST \
*/INST_FDDR_IP:F_WSTRB* \
*/INST_FDDR_IP:F_BREADY* \
*/INST_FDDR_IP:F_RMW_AXI \
*/INST_FDDR_IP:F_RREADY* \
} ]
/* The following constraints provide a relaxation constraint on the signals of 1 clock
period. */
set delay2 [ expr 2000/$ddr_clock_frequency ]
set_max_delay $delay2 -to [ get_pins { \
*/INST_FDDR_IP:F_ARVALID* \
*/INST_FDDR_IP:F_AWVALID* \
*/INST_FDDR_IP:F_WVALID \
} ]
```

For MDDR:

```
/* The following constraints provide a relaxation constraint on the signals of 1.5 clock
periods.
The user should adjust the ddr_clock_frequency to match their application. */
set ddr_clock_frequency 333
set delay1 [ expr 3000/$ddr_clock_frequency ]
set_max_delay1 $delay1 -to [ get_pins { \
*/INST_MSS_*_IP:F_ARADDR* \
*/INST_MSS_*_IP:F_ARBURST* \
*/INST_MSS_*_IP:F_ARID* \
*/INST_MSS_*_IP:F_ARLEN*\
*/INST_MSS_*_IP:F_ARLOCK* \
*/INST_MSS_*_IP:F_ARSIZE* \
*/INST_MSS_*_IP:F_AWADDR* \
*/INST_MSS_*_IP:F_AWBURST* \
*/INST_MSS_*_IP:F_AWID* \
*/INST_MSS_*_IP:F_AWLEN* \
*/INST_MSS_*_IP:F_AWLOCK* \
*/INST_MSS_*_IP:F_AWSIZE* \
*/INST_MSS_*_IP:F_WDATA* \
*/INST_MSS_*_IP:F_WID* \
*/INST_MSS_*_IP:F_WLAST \
*/INST_MSS_*_IP:F_WSTRB* \
*/INST_MSS_*_IP:F_BREADY \
*/INST_MSS_*_IP:F_RMW_AXI \
*/INST_MSS_*_IP:F_RREADY \
} ]
/* The following constraints provide a relaxation constraint on the signals of 1 clock
period. */
set delay2 [ expr 2000/$ddr_clock_frequency ]
set_max_delay $delay2 -to [ get_pins { \
*/INST_MSS_*_IP:F_ARVALID* \
*/INST_MSS_*_IP:F_AWVALID* \
*/INST_MSS_*_IP:F_WVALID \
} ]
```

1.9 DDR Memory Device Examples (Ask a Question)

This section describes how to connect DDR memories to IGLOO 2 MDDR_PADs with examples.

Important: For more information on requirement of termination resistors, refer to the Datasheets/Application Notes of the memory manufacturers.

1.9.1 Example 1: Connecting 32-Bit DDR2 to MDDR_PADs (Ask a Question)

The following illustration shows DDR2 SDRAM connected to the MDDR of a IGLOO 2 device. Micron's MT47H64M16 is a 128 MB density device with x16 data width. The MDDR is configured in full bus width mode and without SECDED. The total amount of DDR2 memory connected to MDDR is 256 MB.

Figure 1-43. x16 DDR2 SDRAM Connected to MDDR
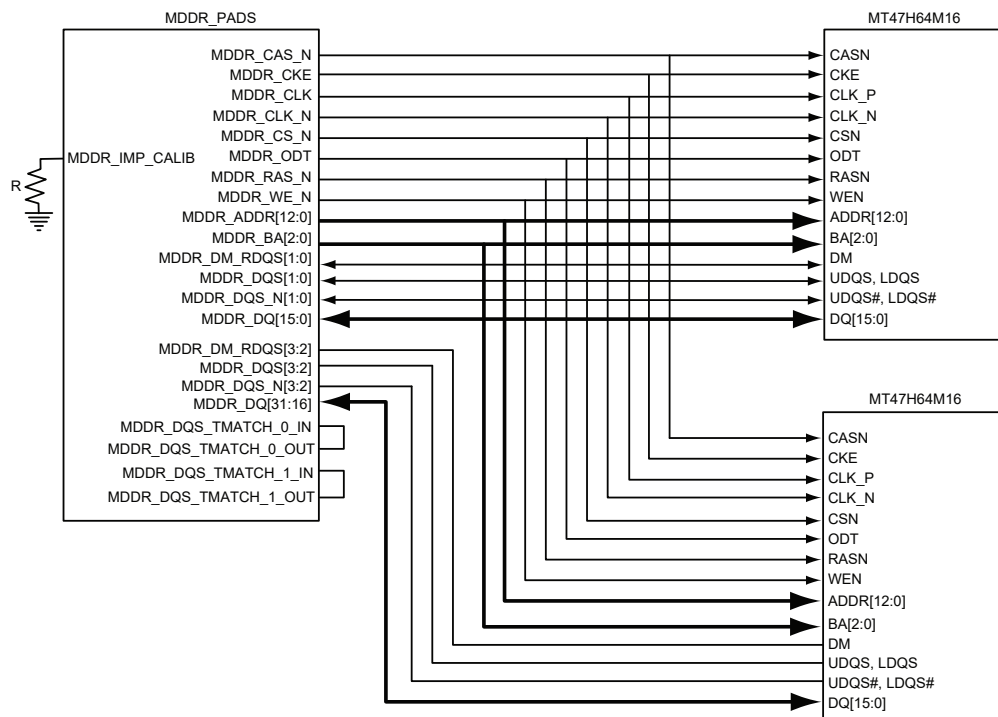
1.9.2 Example 2: Connecting 32-Bit DDR3 to MDDR_PADs with SECDED (Ask a Question)

The following illustration shows DDR3 SDRAM connected to the MDDR of a IGLOO 2 device. Micron's MT41J512M8RA is a 512 MB density device with x8 data width. The MDDR is configured in full bus width mode with SECDED enabled. The SDRAM connected to MDDR_DQ_ECC[3:0] is used to store SECDED bits. The total amount of DDR3 memory (excluding memory for SECDED) connected to MDDR is 2 GB.

Figure 1-44. ×8 DDR3 SDRAM Connection to MDDR
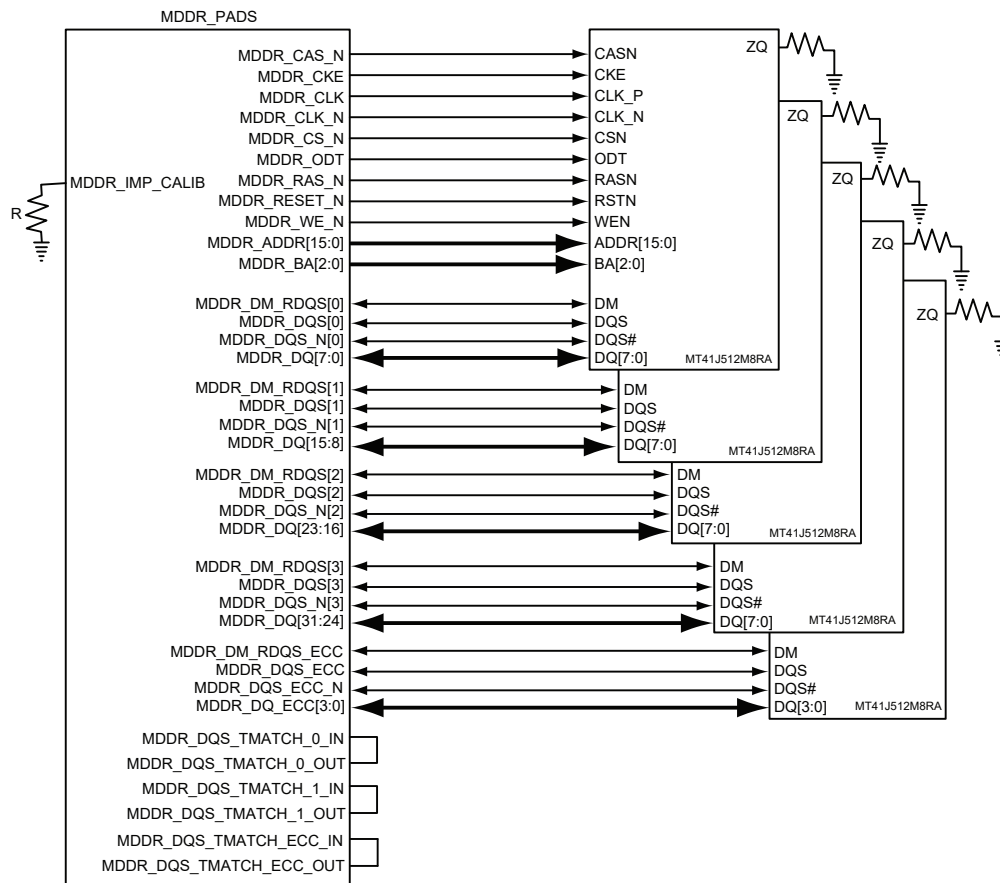
1.9.3 Example 3: Connecting 16-Bit LPDDR to MDDR_PADs with SECDED (Ask a Question)

The following illustration shows LPDDR1 SDRAM connected to the MDDR of a IGLOO 2 device. The Micron's MT46H32M16LF is a 64 MB density device with x16 data width. The MDDR is configured in full bus width mode with SECDED enabled. The SDRAM connected to MDDR_DQ_ECC[1:0] is used to store SECDED bits. The total amount of LPDDR1 memory (excluding memory for SECDED) connected to MDDR is 64 MB.

Figure 1-45. ×16 LPDDR1 SDRAM Connection to MDDR
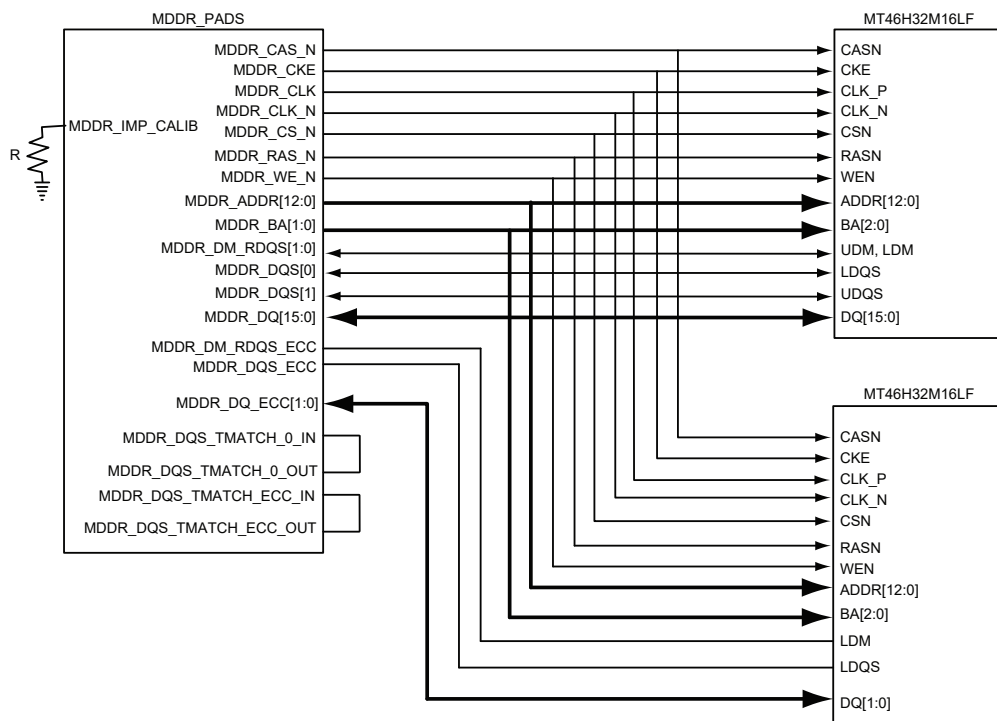
1.10   Board Design Considerations (Ask a Question)

MDDR/FDDR subsystems are interfaced with DDR memories through DDRIO. DDRIO is a multi-standard I/O optimized for LPDDR, DDR2, and DDR3 performance. The following table lists the I/O standards and calibration resistance requirements for MDDR/FDDR to interface with DDR memories.

Table 1-23. I/O Standards and Calibration Resistance Requirements for MDDR/FDDR

Memory Type	I/O Standard	Calibration Resistor
LPDDR	LVCMOS18 LPDDRI(SSTL18)	Not Required* Required
DDR2	SSTL18	Required
DDR3	SSTL15	Required

For more information on I/O Standards and Calibration Resistance Requirements, see AN4153: Board and Layout Design Guidelines for SmartFusion2 SoC and IGLOO2 FPGAs.

> **Important:** For LVCMOS18 I/O Standard, the user can optionally calibrate the I/O. If calibration is desired, the user must install the appropriate resistor on the PCB.

1.11   MDDR Configuration Registers (Ask a Question)

This section provides MDDR subsystem registers along with the address offset, functionality, and bit definitions. The registers are categorized based on the controller blocks in the MDDR subsystem.

The following table lists the categories of registers and their offset addresses. The base address of the MDDR subsystem registers is 0x40020800.

Table 1-24. Address Table for Register Interfaces

Registers	Address Offset Space
DDR Controller Configuration Register	0×000:0×1FC
PHY Configuration Register Summary	0×200:0×3FC
DDR_FIC Configuration Register Summary	0×400:0×4FC
Reserved	0×500:0×7FC

1.11.1 SYSREG Configuration Register Summary (Ask a Question)

In addition to the specific MDDR subsystem registers, the registers listed in the following table also control the behavior of the MDDR subsystem. These registers are located in the SYSREG section of the user's guide and are listed here for convenience. Refer to the "System Register Block" in the UG0448: IGLOO2 High Performance Memory Subsystem User Guide for a detailed description of each register and associated bits.

Table 1-25. SYSREG Configuration Register Summary

Register Name	Register Type	Flash Write Protect	Reset Source	Description
MDDR_CR	RW-P	Register	PORESET_N	MDDR Configuration register
MDDR_IO_CALIB_CR	RW-P	Register	PORESET_N	MDDR I/O Calibration Control register
HPMSDDR_PLL_STATUS_LOW_CR	RW-P	Register	CC_RESET_N	Used to control the corresponding configuration input of the MPLL.
HPMSDDR_PLL_STATUS_HIGH_CR	RW-P	Register	CC_RESET_N	Used to control the corresponding configuration input of the MPLL register
HPMSDDR_FACC1_CR	RW-P	Field	CC_RESET_N	HPMS DDR Fabric Alignment Clock Controller 1 Configuration register
HPMSDDR_FACC2_CR	RW-P	Field	CC_RESET_N	HPMS DDR Fabric Alignment Clock Controller 2 Configuration register
HPMSDDR_CLK_CALIB_STATUS	RW-P	Register	SYSRESET_N	Used to start an FPGA fabric calibration test circuit.
DDRB_CR	RW-P	Register	SYSRESET_N	HPMS DDR bridge configuration register
HPMSDDR_PLL_STATUS	RO	—	—	HPMS DDR PLL Status register
MDDR_IO_CALIB_STATUS	RO	—	PORESET_N	DDR I/O Calibration Status register
HPMSDDR_CLK_CALIB_STATUS	RO	—	SYSRESET_N	HPMS DDR Clock Calibration Status register
SOFT_RESET_CR	RW-P	Bit	SYSRESET_N	Soft reset control register

1.11.2 DDR Controller Configuration Register Summary (Ask a Question)

Table 1-26. DDR Controller Configuration Register

Register Name	Address Offset	Register Type	Reset Source	Description
DDRC_DYN_SOFT_RESET_CR	0×000	RW/RO	PRESET_N	DDRC Reset register
DDRC_DYN_REFRESH_1_CR	0×008	RW	PRESET_N	DDRC Refresh Control register
DDRC_DYN_REFRESH_2_CR	0×00C	RW	PRESET_N	DDRC Refresh Control register
DDRC_DYN_POWERDOWN_CR	0×010	RW	PRESET_N	DDRC Power-Down Control register
Reserved	0×014	—	—	—
DDRC_MODE_CR	0×018	RW	PRESET_N	DDRC Mode register

..........continued

Register Name	Address Offset	Register Type	Reset Source	Description
DDRC_ADDR_MAP_BANK_CR	0×01C	RW	PRESET_N	DDRC Bank Address Map register
Reserved	0×020	—	—	—
DDRC_ADDR_MAP_COL_1_CR	0×024	RW	PRESET_N	DDRC Column Address Map register
DDRC_ADDR_MAP_COL_2_CR	0×028	RW	PRESET_N	DDRC Column Address Map register
DDRC_ADDR_MAP_ROW_1_CR	0×02C	RW	PRESET_N	DDRC Row Address Map register
DDRC_ADDR_MAP_ROW_2_CR	0×030	RW	PRESET_N	DDRC Row Address Map register
DDRC_INIT_1_CR	0×034	RW	PRESET_N	DDRC Initialization Control register
DDRC_CKE_RSTN_CYCLES_1_CR	0×038	RW	PRESET_N	DDRC Initialization Control register
DDRC_ CKE_RSTN_CYCLES_2_CR	0×03C	RW	PRESET_N	DDRC Initialization Control register
DDRC_INIT_MR_CR	0×040	RW	PRESET_N	DDRC MR Initialization register
DDRC_INIT_EMR_CR	0×044	RW	PRESET_N	DDRC EMR Initialization register
DDRC_INIT_EMR2_CR	0×048	RW	PRESET_N	DDRC EMR2 Initialization register
DDRC_INIT_EMR3_CR	0×04C	RW	PRESET_N	DDRC EMR3 Initialization register
DDRC_DRAM_BANK_TIMING_PARAM_CR	0×050	RW	PRESET_N	DDRC DRAM Bank Timing Parameter register
DDRC_DRAM_RD_WR_LATENCY_CR	0×054	RW	PRESET_N	DDRC DRAM Write Latency register
DDRC_DRAM_RD_WR_PRE_CR	0×058	RW	PRESET_N	DDRC DRAM Read-Write Precharge Timing register
DDRC_DRAM_MR_TIMING_PARAM_CR	0×05C	RW	PRESET_N	DDRC DRAM Mode Register Timing Parameter register
DDRC_DRAM_RAS_TIMING_CR	0×060	RW	PRESET_N	DDRC DRAM RAS Timing Parameter register
DDRC_DRAM_RD_WR_TRNARND_TIME_CR	0×064	RW	PRESET_N	DDRC DRAM Read Write Turn-around Timing register
DDRC_DRAM_T_PD_CR	0×068	RW	PRESET_N	DDRC DRAM Power-Down Parameter register
DDRC_DRAM_BANK_ACT_TIMING_CR	0×06C	RW	PRESET_N	DDRC DRAM Bank Activate Timing Parameter register
DDRC_ODT_PARAM_1_CR	0×070	RW	PRESET_N	DDRC ODT Delay Control register
DDRC_ODT_PARAM_2_CR	0×074	RW	PRESET_N	DDRC ODT Hold/Block cycles register
DDRC_ADDR_MAP_COL_3_CR	0×078	RW	PRESET_N	Upper byte is DDRC Column Address Map register and lower byte controls debug features.
DDRC_MODE_REG_RD_WR_CR	0×07C	RW	PRESET_N	DDRC Mode Register Read/Write Command register

..........continued

Register Name	Address Offset	Register Type	Reset Source	Description
DDRC_MODE_REG_DATA_CR	0×080	RW	PRESET_N	DDRC Mode Register Write Data Register
DDRC_PWR_SAVE_1_CR	0×084	RW	PRESET_N	DDRC Power Save register
DDRC_PWR_SAVE_2_CR	0×088	RW	PRESET_N	DDRC Power Save register
DDRC_ZQ_LONG_TIME_CR	0×08C	RW	PRESET_N	DDRC ZQ Long Time Calibration register
DDRC_ZQ_SHORT_TIME_CR	0×090	RW	PRESET_N	DDRC ZQ Short Time Calibration register
DDRC_ZQ_SHORT_INT_REFRESH_MARGIN_1_CR	0×094	RW	PRESET_N	DDRC ZQ Short Time Calibration register
DDRC_ZQ_SHORT_INT_REFRESH_MARGIN_2_CR	0×098	RW	PRESET_N	DDRC ZQ Short Time Calibration register
DDRC_PERF_PARAM_1_CR	0×09C	RW	PRESET_N	DDRC Performance Parameter register
DDRC_HPR_QUEUE_PARAM_1_CR	0×0A0	RW	PRESET_N	DDRC Performance Parameter register
DDRC_HPR_QUEUE_PARAM_2_CR	0×0A4	RW	PRESET_N	DDRC Performance Parameter register
DDRC_LPR_QUEUE_PARAM_1_CR	0×0A8	RW	PRESET_N	DDRC Performance Parameter register
DDRC_LPR_QUEUE_PARAM_2_CR	0×0AC	RW	PRESET_N	DDRC Performance Parameter register
DDRC_WR_QUEUE_PARAM_CR	0×0B0	RW	PRESET_N	DDRC Performance Parameter register
DDRC_PERF_PARAM_2_CR	0×0B4	RW	PRESET_N	DDRC Performance Parameter register
DDRC_PERF_PARAM_3_CR	0×0B8	RW	PRESET_N	DDRC Performance Parameter register
DDRC_DFI_RDDATA_EN_CR	0×0BC	RW	PRESET_N	DDRC DFI Read Command Timing register
DDRC_DFI_MIN_CTRLUPD_TIMING_CR	0×0C0	RW	PRESET_N	DDRC DFI Controller Update Min Time register
DDRC_DFI_MAX_CTRLUPD_TIMING_CR	0×0C4	RW	PRESET_N	DDRC DFI Controller Update Max Time register
Reserved	—	—	—	—
Reserved	—	—	—	—
Reserved	—	—	—	—
Reserved	—	—	—	—
Reserved	—	—	—	—
DDRC_DYN_SOFT_RESET_ALIAS_CR	0×0DC	RW	PRESET_N	DDRC reset register
DDRC_AXI_FABRIC_PRI_ID_CR	0×0E0	RW	PRESET_N	DDRC AXI Interface Fabric Priority ID Register
DDRC_SR	0×0E4	RO	PRESET_N	DDRC Status register
SECDED Registers				
DDRC_SINGLE_ERR_CNT_STATUS_SR	0×0E8	RO	PRESET_N	DDRC single error count Status register
DDRC_DOUBLE_ERR_CNT_STATUS_SR	0×0EC	RO	PRESET_N	DDRC double error count status register
DDRC_LUE_SYNDROME_1_SR	0×0F0	RO	PRESET_N	DDRC last uncorrected error syndrome register

...........continued

Register Name	Address Offset	Register Type	Reset Source	Description
DDRC_LUE_SYNDROME_2_SR	0×0F4	RO	PRESET_N	DDRC last uncorrected error syndrome register
DDRC_LUE_SYNDROME_3_SR	0×0F8	RO	PRESET_N	DDRC last uncorrected error syndrome register
DDRC_LUE_SYNDROME_4_SR	0×0FC	RO	PRESET_N	DDRC last uncorrected error syndrome register
DDRC_LUE_SYNDROME_5_SR	0×100	RO	PRESET_N	DDRC last uncorrected error syndrome register
DDRC_LUE_ADDRESS_1_SR	0×104	RO	PRESET_N	DDRC last uncorrected error address register
DDRC_LUE_ADDRESS_2_SR	0×108	RO	PRESET_N	DDRC last uncorrected error address register
DDRC_LCE_SYNDROME_1_SR	0×10C	RO	PRESET_N	DDRC last corrected error syndrome register
DDRC_LCE_SYNDROME_2_SR	0×110	RO	PRESET_N	DDRC last corrected error syndrome register
DDRC_LCE_SYNDROME_3_SR	0×114	RO	PRESET_N	DDRC last corrected error syndrome register
DDRC_LCE_SYNDROME_4_SR	0×118	RO	PRESET_N	DDRC last corrected error syndrome register
DDRC_LCE_SYNDROME_5_SR	0×11C	RO	PRESET_N	DDRC last corrected error syndrome register
DDRC_LCE_ADDRESS_1_SR	0×120	RO	PRESET_N	DDRC last corrected error address register
DDRC_LCE_ADDRESS_2_SR	0×124	RO	PRESET_N	DDRC last corrected error address register
DDRC_LCB_NUMBER_SR	0×128	RO	PRESET_N	DDRC last corrected bit number register
DDRC_LCB_MASK_1_SR	0×12C	RO	PRESET_N	DDRC last corrected bit mask status register
DDRC_LCB_MASK_2_SR	0×130	RO	PRESET_N	DDRC last corrected bit mask status register
DDRC_LCB_MASK_3_SR	0×134	RO	PRESET_N	DDRC last corrected bit mask status register
DDRC_LCB_MASK_4_SR	0×138	RO	PRESET_N	DDRC last corrected bit mask status register
DDRC_ECC_INT_SR	0×13C	RO	PRESET_N	DDRC SECDED interrupt status register
DDRC_ECC_INT_CLR_REG	0×140	RW	PRESET_N	DDRC SECDED interrupt clear register

1.11.3 DDR Controller Configuration Register Bit Definitions (Ask a Question)

Table 1-27. DDRC_DYN_SOFT_RESET_CR

Bit Number	Name	Reset Value	Description
[31:3]	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.

...........continued

Bit Number	Name	Reset Value	Description
2	AXIRESET	0×1	Set when main AXI reset signal is asserted. Reads and writes to the dynamic registers should not be carried out. This is a read only bit.
1	RESET_APB_REG	0×0	Full soft reset If this bit is set when the soft reset bit is written as 1, all APB registers reset to the power-up state.
0	REG_DDRC_SOFT_RSTB	0×0	This is a soft reset • 0: Puts the controller into reset • 1: Takes the controller out of reset The controller should be taken out of reset only when all other registers have been programmed. Asserting this bit does NOT reset all the APB configuration registers. Once the soft reset bit is asserted, the APB register should be modified as required.

Table 1-28. DDRC_DYN_REFRESH_1_CR

Bit Number	Name	Reset Value	Description
[31:15]	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
[14:7]	REG_DDRC_T_RFC_MIN	0×23	tRFC(min) – Minimum time from refresh to refresh or activate (specification: 75 ns to 195 ns). Unit: clocks.
6	REG_DDRC_REFRESH_UPDATE_LEVEL	0×0	Toggle this signal to indicate that the refresh register(s) have been updated. The value is automatically updated when exiting soft reset, so it does not need to be toggled initially.
5	REG_DDRC_SELFREF_EN	0×0	If 1, then the controller puts the DRAM into self refresh when the transaction store is empty.
[4:0]	REG_DDRC_REFRESH_TO_X32	0×8	Speculative refresh

Table 1-29. DDRC_DYN_REFRESH_2_CR

Bit Number	Name	Reset Value	Description
[31:15]	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
[14:3]	REG_DDRC_T_RFC_NOM_X32	0×52	tREFI: Average time between refreshes (specification: 7.8 µs). Unit: multiples of 32 clocks.

...........continued

Bit Number	Name	Reset Value	Description
[2:0]	REG_DDRC_REFRESH_BURST	0×0	The programmed value plus one is the number of refresh timeouts that is allowed to accumulate before traffic is blocked and the refreshes are forced to execute. Closing pages to perform a refresh is a one-time penalty that must be paid for each group of refreshes; therefore, performing refreshes in a burst reduces the per-refresh penalty of these page closings. Higher numbers for burst_of_N_refresh slightly increases utilization; lower numbers decreases the worst-case latency associated with refreshes. • 0x0: Single refresh • 0x1: Burst-of-2 • 0x7: Burst-of-8 refresh

Table 1-30. DDRC_DYN_POWERDOWN_CR

Bit Number	Name	Reset Value	Description
[31:2]	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
1	REG_DDRC_POWERDOWN_EN	0×1	If true, the controller goes into power-down after a programmable number of cycles (REG_DDRC_POWERDOWN_TO_X32). This register bit may be reprogrammed during the course of normal operation.
0	REG_DDRC_DEEPPOWERDOWN_EN	0×0	• 1: Controller puts the DRAM into deep power-down mode when the transaction store is empty • 0: Brings controller out of deep power-down mode. Present only in designs that have mobile support.

Table 1-31. DDRC_MODE_CR

Bit Number	Name	Reset Value	Description
[31:9]	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
8	REG_DDRC_DDR3	0×0	• 1: DDR3 operating mode • 0: DDR2 operating mode
7	REG_DDRC_MOBILE	0×0	• 1: Mobile/LPDDR1 DRAM device in use • 0: Non-mobile DRAM device in use
6	REG_DDRC_SDRAM	0×0	• 1: SDRAM mode • 0: Non-SDRAM mode. Only present in designs that support SDRAM and/or mSDR devices
5	REG_DDRC_TEST_MODE	0×0	• 1: Reserved • 0: Controller is in normal mode

..........continued

Bit Number	Name	Reset Value	Description
[4:2]	REG_DDRC_MODE	0×0	DRAM SECDED mode • 000: No SECDED • 101: SECDED enabled • All other selections are reserved
[1:0]	REG_DDRC_DATA_BUS_WIDTH	0×0	• 00: Full DQ bus width to DRAM • 01: Half DQ bus width to DRAM • 10: Quarter DQ bus width to DRAM • 11: Reserved **Note:** The half bus width modes are only supported when the DRAM bus width is a multiple of 16.

Table 1-32. DDRC_ADDR_MAP_BANK_CR

Bit Number	Name	Reset Value	Description
[31:12]	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
[11:8]	REG_DDRC_ADDRMAP_BANK_B0	0×0	Selects the address bits used as bank address bit 0. • Valid Range: 0 to 14 • Internal Base: 2 The selected address bit for each of the bank address bits is determined by adding the internal base to the value of this field.
[7:4]	REG_DDRC_ADDRMAP_BANK_B1	0×0	Selects the address bits used as bank address bit 1. • Valid Range: 0 to 14 • Internal Base: 3 The selected address bit for each of the bank address bits is determined by adding the internal base to the value of this field.
[3:0]	REG_DDRC_ADDRMAP_BANK_B2	0×0	Selects the address bits used as bank address bit 2. • Valid Range: 0 to 14 and 15 • Internal Base: 4 The selected address bit is determined by adding the internal base to the value of this field. If set to 15, bank address bit 2 is set to 0.

Table 1-33. DDRC_ADDR_MAP_COL_1_CR

Bit Number	Name	Reset Value	Description
[31:16]	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.

..........continued

Bit Number	Name	Reset Value	Description
[15:12]	REG_DDRC_ADDRMAP_COL_B2	0×0	• Full bus width mode: Selects column address bit 3 • Half bus width mode: Selects column address bit 4 • Quarter bus width mode: Selects column address bit 5 • Valid range: 0 to 7 • Internal base: 2 The selected address bit is determined by adding the internal base to the value of this field.
[11:8]	REG_DDRC_ADDRMAP_COL_B3	0×0	• Full bus width mode: Selects column address bit 4 • Half bus width mode: Selects column address bit 5 • Quarter bus width mode: Selects column address bit 6 • Valid range: 0 to 7 • Internal base: 3 The selected address bit is determined by adding the internal base to the value of this field.
[7:4]	REG_DDRC_ADDRMAP_COL_B4	0×0	• Full bus width mode: Selects column address bit 5 • Half bus width mode: Selects column address bit 6 • Quarter bus width mode: Selects column address bit 7 • Valid Range: 0 to 7 • Internal base: 4 The selected address bit for each of the column address bits is determined by adding the internal base to the value of this field.
[3:0]	REG_DDRC_ADDRMAP_COL_B7	0×0	• Full bus width mode: Selects column address bit 8 • Half bus width mode: Selects column address bit 9 • Quarter bus width mode: Selects column address bit 11 • Valid range: 0 to 7, and 15 • Internal base: 7 The selected address bit is determined by adding the internal base to the value of this field. If set to 15, column address bit 9 is set to 0. **Note:** Per JEDEC DDR2 specification, column address bit 10 is reserved for indicating auto-precharge, and hence no source address bit can be mapped to column address bit 10.

Table 1-34. DDRC_ADDR_MAP_COL_2_CR

Bit Number	Name	Reset Value	Description
[31:16]	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.

MICROCHIP

..........continued

Bit Number	Name	Reset Value	Description
[15:12]	REG_DDRC_ADDRMAP_COL_B8	0×0	• Full bus width mode: Selects column address bit 9 • Half bus width mode: Selects column address bit 11 • Quarter bus width mode: Selects column address bit 12 • Valid range: 0 to 7, and 15 • Internal base: 8 The selected address bit is determined by adding the internal base to the value of this field. If set to 15, column address bit 9 is set to 0. **Note:** Per JEDEC DDR2 specification, column address bit 10 is reserved for indicating auto-precharge, and hence no source address bit can be mapped to column address bit 10.
[11:8]	REG_DDRC_ADDRMAP_COL_B9	0×0	• Full bus width mode: Selects column address bit 11 • Half bus width mode: Selects column address bit 12 • Quarter bus width mode: Selects column address bit 13 • Valid range: 0 to 7, and 15 • Internal base: 9 The selected address bit is determined by adding the internal base to the value of this field. If set to 15, column address bit 9 is set to 0.
[7:4]	REG_DDRC_ADDRMAP_COL_B10	0×0	• Full bus width mode: Selects column address bit 12 • Half bus width mode: Selects column address bit 13 • Quarter bus width mode: Unused. Should be set to 15 • Valid range: 0 to 7, and 15 • Internal base: 10 The selected address bit is determined by adding the internal base to the value of this field. If set to 15, column address bit 10 is set to 0.
[3:0]	REG_DDRC_ADDRMAP_COL_B11	0×0	• Full bus width mode: Selects column address bit 13 • Half bus width mode: Unused. To make it unused, this should be tied to 0xF • Quarter bus width mode: Unused. To make it unused, this should be tied to 0xF • Valid range: 0 to 7, and 15 • Internal base: 11 The selected address bit is determined by adding the internal base to the value of this field. If set to 15, column address bit 11 is set to 0.

Table 1-35. DDRC_ADDR_MAP_ROW_1_CR

Bit Number	Name	Reset Value	Description
[31:16]	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.

..........continued

Bit Number	Name	Reset Value	Description
[15:12]	REG_DDRC_ADDRMAP_ROW_B0	0×0	Selects the address bits used as row address bit 0. • Valid range: 0 to 11 • Internal base: 6 The selected address bit for each of the row address bits is determined by adding the internal base to the value of this field.
[11:8]	REG_DDRC_ADDRMAP_ROW_B1	0×0	Selects the address bits used as row address bit 1. • Valid range: 0 to 11 • Internal base: 7 The selected address bit for each of the row address bits is determined by adding the internal base to the value of this field.
[7:4]	REG_DDRC_ADDRMAP_ROW_B2_11	0×0	Selects the address bits used as row address bits 2 to 11. • Valid Range: 0 to 11 • Internal Base: 8 for row address bit 2 • 9 for row address bit 3 • 10 for row address bit 4 • .... • 15 for row address bit 9 • 16 for row address bit 10 • 17 for row address bit 11 The selected address bit for each of the row address bits is determined by adding the internal base to the value of this field.
[3:0]	REG_DDRC_ADDRMAP_ROW_B12	0×0	Selects the address bit used as row address bit 12. • Valid Range: 0 to 11, and 15 • Internal Base: 18 The selected address bit is determined by adding the internal base to the value of this field. If set to 15, row address bit 12 is set to 0.

Table 1-36. DDRC_ADDR_MAP_ROW_2_CR

Bit Number	Name	Reset Value	Description
[31:12]	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
[11:8]	REG_DDRC_ADDRMAP_ROW_B13	0×0	Selects the address bits used as row address bit 13. Valid range: 0 to 11, and 15 Internal base: 19 The selected address bit is determined by adding the internal base to the value of this field. If set to 15, row address bit 13 is set to 0.

MICROCHIP

...........continued

Bit Number	Name	Reset Value	Description
[7:4]	REG_DDRC_ADDRMAP_ROW_B14	0×0	Selects the address bit used as row address bit 14. Valid range: 0 to 11, and 15 Internal base: 20 The selected address bit is determined by adding the internal base to the value of this field. If set to 15, row address bit 14 is set to 0.
[3:0]	REG_DDRC_ADDRMAP_ROW_B15	0×0	Selects the address bit used as row address bit 15. Valid range: 0 to 11, and 15 Internal base: 21 The selected address bit is determined by adding the internal base to the value of this field. If set to 15, row address bit 15 is set to 0.

Table 1-37. DDRC_INIT_1_CR

Bit Number	Name	Reset Value	Description
[31:12]	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
[11:8]	REG_DDRC_PRE_OCD_X32	0×0	Wait period before driving the on chip driver calibration (OCD) Complete command to DRAM. Units are in counts of a global timer that pulses every 32 clock cycles. There is no known specific requirement for this. It may be set to zero.
[7:1]	REG_DDRC_FINAL_WAIT_X32	0×0	Cycles to wait after completing the DRAM initialization sequence before starting the dynamic scheduler. Units are in counts of a global timer that pulses every 32 clock cycles. There is known specific requirement for this; it may be set to zero.
0	REG_DDRC_SKIP_OCD	0×1	This register must be kept at 1. • 1: Indicates the controller is to skip the on-chip driver calibration (OCD) adjustment step during DDR2 initialization. OCD_Default and OCD_Exit are performed instead • 0: Not supported

Table 1-38. DDRC_CKE_RSTN_CYCLES_1_CR

Bit Number	Name	Reset Value	Description
[31:16]	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.

...........continued

Bit Number	Name	Reset Value	Description
[15:8]	REG_DDRC_PRE_CKE_X1024	0×0	The 10-bit REG_DDRC_PRE_CKE_X1024 [9:0] value is spit across the two registers: DDRC_CKE_RSTN_CYCLES_1_CR and DDRC_CKE_RSTN_CYCLES_2_CR. [7:0] bits of REG_DDRC_PRE_CKE_X1024. Cycles to wait after reset before driving CKE High to start the DRAM initialization sequence. Units: 1,024 clock cycles. DDR2 specifications typically require this to be programmed for a delay of ≥ 200 µs.
[7:0]	REG_DDRC_DRAM_RSTN_X1024	0×0	Number of cycles to assert DRAM reset signal during initialization sequence. This is only present for implementations supporting DDR3 devices.

Table 1-39. DDRC_ CKE_RSTN_CYCLES_2_CR

Bit Number	Name	Reset Value	Description
[31:16]	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
[11:2]	REG_DDRC_POST_CKE_X1024	0×0	Cycles to wait after driving CKE High to start the DRAM initialization sequence. Units: 1,024 clocks. • DDR: Typically requires a 400 ns delay, requiring this value to be programmed to 2 at all clock speeds • SDR: Typically requires this to be programmed for a delay of 100 µs to 200 µs
[1:0]	REG_DDRC_PRE_CKE_X1024	0×0	This field represents the upper 2 bits of the 10-bit REG_DDRC_PRE_CKE_X1024 value split across the 2 registers DDRC_CKE_RSTN_CYCLES_1_CR and DDRC_CKE_RSTN_CYCLES_2_CR. [9:8] bits of REG_DDRC_PRE_CKE_X1024. Cycles to wait from the start of reset assertion before driving CKE High to start the DRAM initialization sequence. Units: 1,024 clock cycles DDR2 specifications typically require this to be programmed for a delay of ≥ 200 µs.

Table 1-40. DDRC_INIT_MR_CR

Bit Number	Name	Reset Value	Description
[31:16]	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
[15:0]	REG_DDRC_MR	0×095A	Value to be loaded into the DRAM Mode register. Bit 8 is for the DLL and the setting here is ignored. The controller sets appropriately. During DRAM initialization procedure, the controller will send the mode register setting to DRAM. The mode register sets the DRAM burst length, burst type, CAS latency (CL), and operating mode.

Table 1-41. DDRC_INIT_EMR_CR

Bit Number	Name	Reset Value	Description
[31:16]	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
[15:0]	REG_DDRC_EMR	0×0402	Value to be loaded into DRAM EMR registers. Bits [9:7] are for OCD and the setting in this bits is ignored. The controller sets those bits appropriately.

Table 1-42. DDRC_INIT_EMR2_CR

Bit Number	Name	Reset Value	Description
[31:16]	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
[15:0]	REG_DDRC_EMR2	0×0	Value to be loaded into DRAM EMR2 registers.

Table 1-43. DDRC_INIT_EMR3_CR

Bit Number	Name	Reset Value	Description
[31:16]	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
[15:0]	REG_DDRC_EMR3	0×0	Value to be loaded into DRAM EMR3 registers.

Table 1-44. DDRC_DRAM_BANK_TIMING_PARAM_CR

Bit Number	Name	Reset Value	Description
[31:12]	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
[11:6]	REG_DDRC_T_RC	0×0	tRC: Minimum time between activates to same bank (specification: 65 ns for DDR2-400 and smaller for faster parts). Unit: clocks.
[5:0]	REG_DDRC_T_FAW	0×0	tFAW: Valid only in burst-of-8 mode. At most 4 banks must be activated in a rolling window of tFAW cycles. Unit: clocks

Table 1-45. DDRC_DRAM_RD_WR_LATENCY_CR

Bit Number	Name	Reset Value	Description
[31:10]	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
[9:5]	REG_DDRC_WRITE_LATENCY	0×0	Number of clocks between the write command to write data enable PHY.
[4:0]	REG_DDRC_READ_LATENCY	0×0	Time from read command to read data on DRAM interface. Unit: clocks This signal is present for designs supporting LPDDR1 DRAM only. It is used to calculate when the DRAM clock may be stopped.

MICROCHIP

Table 1-46. DDRC_DRAM_RD_WR_PRE_CR

Bit Number	Name	Reset Value	Description
[31:10]	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
[9:5]	REG_DDRC_WR2PRE	0×0	Minimum time between write and precharge to same bank (specifications: WL + BL/2 + tWR = approximately 8 cycles + 15 ns = 14 clocks @ 400 MHz and less for lower frequencies). Unit: Clocks where: WL = Write latency BL = Burst length. This must match the value programmed in the BL bit of the mode register to the DRAM. tWR = Write recovery time. This comes directly from the DRAM specs.
[4:0]	REG_DDRC_RD2PRE	0×0	tRTP – Minimum time from read to precharge of same bank (specification: tRTP for BL = 4 and tRTP + 2 for BL = 8. tRTP = 7.5 ns). Unit: clocks.

Table 1-47. DDRC_DRAM_MR_TIMING_PARAM_CR

Bit Number	Name	Reset Value	Description
[31:13]	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
[12:3]	REG_DDRC_T_MOD	0×0	Present for DDR3 only (replaces REG_DDRC_T_MRD functionality when used with DDR3 devices). The mode register set command updates delay in number of clock cycles. This is required to be programmed even when a design that supports DDR3 is running in DDR2 mode (minimum is the larger of 12 clock cycles or 15 ns).
[2:0]	REG_DDRC_T_MRD	0×0	tMRD: Cycles between load mode commands. Not used in DDR3 mode.

Table 1-48. DDRC_DRAM_RAS_TIMING_CR

Bit Number	Name	Reset Value	Description
[31:11]	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
[10:5]	REG_DDRC_T_RAS_MAX	0×0	tRAS(max): Maximum time between activate and precharge to same bank. Maximum time that a page can be kept open (specification: 70 µs). Minimum value of this register is 1. Zero is invalid. Unit: Multiples of 1,024 clocks.
[4:0]	REG_DDRC_T_RAS_MIN	0×0	tRAS(min): Minimum time between activate and precharge to the same bank (specification: 45 ns). Unit: clocks.

Table 1-49. DDRC_DRAM_RD_WR_TRNARND_TIME_CR

Bit Number	Name	Reset Value	Description
[31:10]	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.

..........continued

Bit Number	Name	Reset Value	Description
[9:5]	REG_DDRC_RD2WR	0×0	RL + BL/2 + 2 – WL Minimum time from READ command to WRITE command. Include time for bus turnaround and all per-bank, per-rank, and global constraints. Unit: clocks. where, WL = Write latency BL = Burst length. This must match the value programmed in the BL bit of the mode register to the DRAM. RL = Read latency = CAS latency.
[4:0]	REG_DDRC_WR2RD	0×0	WL + tWTR + BL/2 Minimum time from WRITE command to READ command. Includes time for bus turnaround and recovery times and all per-bank, per- rank, and global constraints. Unit: clocks. where, WL: Write latency. BL: Burst length. This should match the value programmed in the BL bit of the mode register to the DRAM. tWTR: Internal WRITE to READ command delay. This comes directly from the DRAM specifications.

Table 1-50. DDRC_DRAM_T_PD_CR

Bit Number	Name	Reset Value	Description
[31:9]	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
[8:4]	REG_DDRC_T_XP	0×0	tXP: Minimum time after power-down exit to any operation. Units: clocks
[3:0]	REG_DDRC_T_CKE	0×0	Minimum number of cycles of CKE High/Low during power-down and self refresh. Unit: clocks

Table 1-51. DDRC_DRAM_BANK_ACT_TIMING_CR

Bit Number	Name	Reset Value	Description
[31:14]	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
[13:10]	REG_DDRC_T_RCD	0×0	tRCD: Minimum time from activate to READ or WRITE command to same bank (specification: 15 ns for DDR2-400 and lower for faster devices). Unit: clocks.
[9:7]	REG_DDRC_T_CCD	0×0	tCCD: Minimum time between two reads or two writes (from bank A to bank B) (specification: 2 cycles) is this value + 1. Unit: clocks.
[6:4]	REG_DDRC_T_RRD	0×0	tRRD: Minimum time between activates from bank A to bank B (specification: 10 ns or less). Unit: clocks.
[3:0]	REG_DDRC_T_RP	0×0	tRP: Minimum time from precharge to activate of same bank. Unit: clocks.

Table 1-52. DDRC_ODT_PARAM_1_CR

Bit Number	Name	Reset Value	Description
[31:12]	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.

...........continued

Bit Number	Name	Reset Value	Description
[11:8]	REG_DDRC_RD_ODT_DELAY	0×0	The delay, in clock cycles, from issuing a READ command to setting ODT values associated with that command. Recommended value for DDR2 is CL − 4.
[7:4]	REG_DDRC_WR_ODT_DELAY	0×0	The delay, in clock cycles, from issuing a WRITE command to setting ODT values associated with that command. The recommended value for DDR2 is CL − 5. Where CL is CAS latency. DDR ODT has a 2-cycle on-time delay and a 2.5-cycle off-time delay. ODT setting should remain constant for the entire time that DQS is driven by the controller.
[3:2]	REG_DDRC_RANK0_WR_ODT	0×0	0: Indicates which remote ODTs should be turned on during a write to rank 0. Each rank has a remote ODT (in the DRAM) which can be turned on by setting the appropriate bit here. Set this bit to 1 to enable its ODT. 1: Uppermost bit is unused
[1:0]	REG_DDRC_RANK0_RD_ODT	0×0	0: Indicates which remote ODTs should be turned on during a read to rank 0. Each rank has a remote ODT (in the DRAM) which can be turned on by setting the appropriate bit here. Set this bit to 1 to enable its ODT. 1: Uppermost bit is unused

1.11.3.1 DDRC_ODT_PARAM_2_CR (Ask a Question)

Table 1-53. DDRC_ODT_PARAM_2_CR

Bit Number	Name	Reset Value	Description
[31:10]	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
[9:6]	REG_DDRC_RD_ODT_HOLD	0×0	Cycles to hold ODT for a READ command. • 0: ODT signal is ON for 1 cycle • 1: ODT signal is ON for 2 cycles, and so on
[5:2]	REG_DDRC_WR_ODT_HOLD	0×0	Cycles to hold ODT for a WRITE command. • 0: ODT signal is ON for 1 cycle • 1: ODT signal is ON for 2 cycles, and so on
[1:0]	REG_DDRC_WR_ODT_BLOCK	0×0	• 00: Block read/write scheduling for 1-cycle when write requires changing ODT settings • 01: Block read/write scheduling for 2 cycles when write requires changing ODT settings • 10: Block read/write scheduling for 3 cycles when write requires changing ODT settings • 11: Reserved

MICROCHIP

Table 1-54. DDRC_ADDR_MAP_COL_3_CR

Bit Number	Name	Reset Value	Description
[31:16] [7:6]	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
[15:12]	REG_DDRC_ADDRMAP_COL_B5	0×0	• Full bus width mode: Selects column address bit 6 • Half bus width mode: Selects column address bit 7 • Quarter bus width mode: Selects column address bit 8 Valid range: 0 to 7 Internal base: 5 The selected address bit for each of the column address bits is determined by adding the internal base to the value of this field.
[11:8]	REG_DDRC_ADDRMAP_COL_B6	0×0	• Full bus width mode: Selects column address bit 7 • Half bus width mode: Selects column address bit 8 • Quarter bus width mode: Selects column address bit 9 Valid range: 0 to 7 Internal base: 6 The selected address bit for each of the column address bits is determined by adding the internal base to the value of this field.
5	REG_DDRC_DIS_WC	0×0	When 1, disable write combine.
4	REG_DDRC_DIS_ACT_BYPASS	0×0	Only present in designs supporting activate bypass. When 1, disable bypass path for high priority read activates
3	REG_DDRC_DIS_RD_BYPASS	0×0	Only present in designs supporting read bypass. When 1, disable bypass path for high priority read page hits.
2	REG_DDRC_DIS_PRE_BYPASS	0×0	Only present in designs supporting precharge bypass. When 1, disable bypass path for high priority precharges
1	REG_DDRC_DIS_COLLISION_PAGE_OPT	0×0	When this is set to '0', auto-precharge is disabled for the flushed command in a collision case. Collision cases are write followed by read to same address, read followed by write to same address, or write followed by write to same address with REG_DDRC_DIS_WC bit = 1 (where same address comparisons exclude the two address bits representing the critical word).
0	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.

Table 1-55. DDRC_MODE_REG_RD_WR_CR

Bit Number	Name	Reset Value	Description
[31:4]	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
3	REG_DDRC_MR_WR	0×0	When 1 is written and DDRC_REG_MR_WR_BUSY is Low, a mode register read or write operation is started. There is no need for the CPU to set this back to zero. This bit always reads as zero.

..........continued

Bit Number	Name	Reset Value	Description
[2:1]	REG_DDRC_MR_ADDR	0×0	Address of the Mode register that is to be written to. • 00: MR0 • 01: MR1 • 10: MR2 • 11: MR3
0	REG_DDRC_MR_TYPE	0×0	Indicates whether the Mode register operation is read or write. • 1: Read • 0: Write

Table 1-56. DDRC_MODE_REG_DATA_CR

Bit Number	Name	Reset Value	Description
[31:16]	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
[15:0]	REG_DDRC_MR_DATA	0×0	Mode register write data

Table 1-57. DDRC_PWR_SAVE_1_CR

Bit Number	Name	Reset Value	Description
[31:13]	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
[12:6]	REG_DDRC_POST_SELFREF_GAP_X32	0×10	Minimum time to wait after coming out of self refresh before doing anything. This must be larger than all the constraints that exist (Specifications: maximum of tXSNR and tXSRD and tXSDLL, which is 512 clocks). Unit: Multiples of 32 clocks.
[5:1]	REG_DDRC_POWERDOWN_TO_X32	0×06	After this many clocks of NOP or DESELECT, the controller puts the DRAM into power-down. This must be enabled in the Master Control register. Unit: Multiples of 32 clocks.
0	REG_DDRC_CLOCK_STOP_EN	0×0	• 1: Stops the clock to the PHY whenever a clock is not required by LPDDR1 • 0: Clock will never be stopped This is only present for implementations supporting mobile/LPDDR1 devices.

Table 1-58. DDRC_PWR_SAVE_2_CR

Bit Number	Name	Reset Value	Description
[31:12]	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
11	REG_DDRC_DIS_PAD_PD	0×0	• 1: Disable the pad power-down feature • 0: Enable the pad power-down feature Used only in non-DFI designs.

...........continued

Bit Number	Name	Reset Value	Description
[10:3]	REG_DDRC_DEEPPOWERDOWN_TO_X1024	0×0	Not supported.
[2:0]	REG_DDRC_PAD_PD	0×0	If pads have a power-saving mode, this is the greater of the time for the pads to enter power-down or the time for the pads to exit power-down. Used only in non-DFI designs. Unit: clocks.

Table 1-59. DDRC_ZQ_LONG_TIME_CR

Bit Number	Name	Reset Value	Description
[31:10]	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
[9:0]	REG_DDRC_T_ZQ_LONG_NOP	0×0	Number of cycles of NOP required after a ZQCL (ZQ calibration long) command is issued to DRAM. Units: Clock cycles. This is only present for implementations supporting DDR3 devices

Table 1-60. DDRC_ZQ_SHORT_TIME_CR

Bit Number	Name	Reset Value	Description
[31:10]	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
[9:0]	REG_DDRC_T_ZQ_SHORT_NOP	0×0	Number of cycles of NOP required after a ZQCS (ZQ calibration short) command is issued to DRAM. Units: Clock cycles. This is only present for implementations supporting DDR3 devices.

Table 1-61. DDRC_ZQ_SHORT_INT_REFRESH_MARGIN_1_CR

Bit Number	Name	Reset Value	Description
[31:16]	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
[15:4]	REG_DDRC_T_ZQ_SHORT_INTERVAL_X1024	0×0	20 bits are split into two registers. [11:0] bits of REG_DDRC_T_ZQ_SHORT_INTERVAL_ X1024. Average interval to wait between automatically issuing ZQ calibration short (ZQCS) commands to DDR3 devices. Not considered if REG_DDRC_DIS_AUTO_ZQ = 1. Units: 1,024 clock cycles This is only present for implementations supporting DDR3 devices.
[3:0]	REG_DDRC_REFRESH_MARGIN	0×02	Threshold value in number of clock cycles before the critical refresh or page timer expires. A critical refresh is to be issued before this threshold is reached. Microchip recommends using the default value. Unit: Multiples of 32 clocks.

MICROCHIP

Table 1-62. DDRC_ZQ_SHORT_INT_REFRESH_MARGIN_2_CR

Bit Number	Name	Reset Value	Description
[31:8]	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
[7:0]	REG_DDRC_T_ZQ_SHORT_INTERVAL_X1024	0×0	20 bits are split into two registers. [19:12] bits of REG_DDRC_T_ZQ_SHORT_INTERVAL_X1024. Average interval to wait between automatically issuing ZQ calibration short (ZQCS) commands to DDR3 devices. Not considered if REG_DDRC_DIS_AUTO_ZQ = 1. Units: 1,024 clock cycles This is only present for implementations supporting DDR3 devices.

Table 1-63. DDRC_PERF_PARAM_1_CR

Bit Number	Name	Reset Value	Description
[31:16]	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
[15:13]	REG_DDRC_BURST_RDWR	0×0	• 001: Burst length of 4 • 010: Burst length of 8 • 100: Burst length of 16 All other values are reserved. This controls the burst size used to access the DRAM. This must match the BL mode register setting in the DRAM. The DDRC and AXI controllers are optimized for a burst length of 8. The recommended setting is 8. A burst length of 16 is only supported for LPDDR1. Setting to 16 when using LPDDR1 in half/quarter bus mode may boost performance. For systems that tend to do many single cycle random transactions, a burst length of 4 may slightly improve system performance.
12	Reserved	0×0	This bit must always be set to zero.
[11:5]	REG_DDRC_RDWR_IDLE_GAP	0×04	When the preferred transaction store is empty for this many clock cycles, switch to the alternate transaction store if it is non-empty. The read transaction store (both high and low priority) is the default preferred transaction store and the write transaction store is the alternate store. When "Prefer write over read" is set, this is reversed.
4	REG_DDRC_PAGECLOSE	0×0	• 1: Bank is closed and kept closed if no transactions are available for it. This is different from auto-precharge: – Explicit precharge commands are used, and not read/write with auto-precharge and – Page is not closed after a read/write if there is another read/write pending to the same page • 0: Bank remains open until there is a need to close it (to open a different page, or for page timeout or refresh timeout)
3	Reserved		This bit must always be set to zero.

..........continued

Bit Number	Name	Reset Value	Description
[2:0]	REG_DDRC_LPR_NUM_ENTRIES	0×03	Number of entries in the low priority transaction store is this value plus 1. READ_CAM_DEPTH – (REG_DDRC_LPR_NUM_ENTRIES + 1) is the number of entries available for the high priority transaction store. READ_CAM_DEPTH = Depth of the read transaction store, that is, 8. Setting this to maximum value allocates all entries to low priority transaction store. Setting this to 0 allocates 1 entry to low priority transaction store and the rest to high priority transaction store. **Note:** In designs with ECC, number of lpr and wr credits issued to the core is 1 less than the non-ECC case. 1 entry each is reserved in wr and lpr cam for storing the RMW requests arising out of Single bit Error Correction RMW operation.

Table 1-64. DDRC_HPR_QUEUE_PARAM_1_CR

Bit Number	Name	Reset Value	Description
[31:16]	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
15	REG_DDRC_HPR_MAX_STARVE_X32	0×0	Lower 1 bit of REG_DDRC_HPR_MAX_STARVE_X32. Number of clocks that the HPR queue can be starved before it goes critical. Unit: 32 clocks.
[14:4]	REG_DDRC_HPR_MIN_NON_CRITICAL	0×0	Number of clocks that the HPR queue is guaranteed to be non-critical. Unit: 32 clocks.
[3:0]	REG_DDRC_HPR_XACT_RUN_LENGTH	0×0	Number of transactions that are serviced once the HPR queue goes critical is the smaller of this value and number of transactions available. Units: Transactions.

Table 1-65. DDRC_HPR_QUEUE_PARAM_2_CR

Bit Number	Name	Reset Value	Description
[31:11]	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
[10:0]	REG_DDRC_HPR_MAX_STARVE_X32	0×0	[11:1] bits of REG_DDRC_HPR_MAX_STARVE_X32 Number of clocks that the HPR queue can be starved before it goes critical. Unit: 32 clocks.

Table 1-66. DDRC_LPR_QUEUE_PARAM_1_CR

Bit Number	Name	Reset Value	Description
[31:16]	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
15	REG_DDRC_LPR_MAX_STARVE_X32	0×0	12 bits are split into two registers. Lower 1 bit of REG_DDRC_LPR_MAX_STARVE_X32. Number of clocks that the LPR queue can be starved before it goes critical. Unit: 32 clocks.

...........continued

Bit Number	Name	Reset Value	Description
[14:4]	REG_DDRC_LPR_MIN_NON_CRITICAL	0×0	Number of clocks that the LPR queue is guaranteed to be non-critical. Unit: 32 clocks.
[3:0]	REG_DDRC_LPR_XACT_RUN_LENGTH	0×0	Number of transactions that are serviced once the LPR queue goes critical is the smaller of this value and number of transactions available. Units: Transactions.

Table 1-67. DDRC_LPR_QUEUE_PARAM_2_CR

Bit Number	Name	Reset Value	Description
[31:16]	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
[10:0]	REG_DDRC_LPR_MAX_STARVE_X32	0×0	12 bits are split into two registers. [11:1] bits of REG_DDRC_HPR_MAX_STARVE_X32. Number of clocks that the LPR queue can be starved before it goes critical. Unit: 32 clocks.

Table 1-68. DDRC_WR_QUEUE_PARAM_CR

Bit Number	Name	Reset Value	Description
[31:15]	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
[14:4]	REG_DDRC_W_MIN_NON_CRITICAL	0×0	Number of clocks that the write queue is guaranteed to be non-critical. Unit: 32 clocks.
[3:0]	REG_DDRC_W_XACT_RUN_LENGTH	0×0	Number of transactions that are serviced once the WR queue goes critical is the smaller of this value and number of transactions available. Units: Transactions.

Table 1-69. DDRC_PERF_PARAM_2_CR

Bit Number	Name	Reset Value	Description
[31:12]	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
11	REG_DDRC_BURSTCHOP	0×0	Not supported in this version of the DDRC controller always reads as zero.
10	REG_DDRC_BURST_MODE	0×0	1: Interleaved burst mode 0: Sequential burst mode The burst mode programmed in the DRAM mode register and the order of the input data to the controller should both match the value programmed in the REG_DDRC_BURST_MODE register.
[9:2]	REG_DDRC_GO2CRITICAL_HYSTERESIS	0×0	Indicates the number of cycles that CO_GS_GO2CRITICAL_RD or CO_GS_GO2CRITICAL_WR must be asserted before the corresponding queue moves to the critical state in the DDRC.

...........continued

Bit Number	Name	Reset Value	Description
1	REG_DDRC_PREFER_WRITE	0×0	If set, the bank selector prefers writes over reads.
0	REG_DDRC_FORCE_LOW_PRI_N	0×0	Active Low signal. When asserted ('0'), all incoming transactions are forced to low priority. Forcing the incoming transactions to low priority implicitly turns off bypass.

Table 1-70. DDRC_PERF_PARAM_3_CR

Bit Number	Name	Reset Value	Description
[31:1]	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
0	REG_DDRC_EN_2T_TIMING_MODE	0×0	1: DDRC uses 2T timing 0: DDRC uses 1T timing

Table 1-71. DDRC_DFI_RDDATA_EN_CR

Bit Number	Name	Reset Value	Description
[31:5]	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
[4:0]	REG_DDRC_DFI_T_RDDATA_EN	0×0	Time from the assertion of a READ command on the DFI interface to the assertion of the DDRC_DFI_RDDATA_EN signal. Program this to (RL – 1), where RL is the read latency of the DRAM. For LPDDR1 this should be set to RL. Units: Clocks

Table 1-72. DDRC_DFI_MIN_CTRLUPD_TIMING_CR

Bit Number	Name	Reset Value	Description
[31:10]	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
[9:0]	REG_DDRC_DFI_T_CTRLUP_MIN	0×03	Specifies the minimum number of clock cycles that the DDRC_DFI_CTRLUPD_REQ signal must be asserted. Lowest value to assign to this variable is 0x3. Units: Clocks

Table 1-73. DDRC_DFI_MAX_CTRLUPD_TIMING_CR

Bit Number	Name	Reset Value	Description
[31:10]	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
[9:0]	REG_DDRC_DFI_T_CTRLUP_MAX	0×40	Specifies the maximum number of clock cycles that the DDRC_DFI_CTRLUPD_REQ signal can assert. Lowest value to assign to this variable is 0x40. Units: Clocks

Table 1-74. DDRC_DYN_SOFT_RESET_ALIAS_CR

Bit Number	Name	Reset Value	Description
[31:3]	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
2	AXIRESET	0×1	Set when main AXI reset signal is asserted. Reads and writes to the dynamic registers should not be carried out. This is a read only bit.
1	RESET_APB_REG	0×0	Full soft reset If this bit is set when the soft reset bit is written as '1', all APB registers reset to the power-up state.
0	REG_DDRC_SOFT_RSTB	0×0	This is a soft reset. • 0: Puts the controller into reset • 1: Takes the controller out of reset The controller should be taken out of reset only when all other registers have been programmed. Asserting this bit does NOT reset all the APB configuration registers. Once the soft reset bit is asserted, the APB register should be modified as required.

Table 1-75. DDRC_AXI_FABRIC_PRI_ID_CR

Bit Number	Name	Reset Value	Description
[31:6]	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
[5:4]	PRIORITY_ENABLE_BIT	0×0	This is to set the priority of the fabric master ID. 01/10/11: Indicates that the ID is higher priority. 00: None of the master IDs from the fabric has a higher priority.
[3:0]	PRIORITY_ID	0×0	If the Priority Enable bit is 1, this ID will have a higher priority over other IDs.

Table 1-76. DDRC_SR

Bit Number	Name	Reset Value	Description
[31:6]	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.

Microchip

...........continued

Bit Number	Name	Reset Value	Description
[5:3]	DDRC_CORE_REG_OPERATING_MODE	0×0	Operating mode. This is 3 bits wide in designs with mobile support and 2-bits in all other designs. Non-mobile designs: • 000: Init • 010: Power-down • 001: Normal • 011: Self Refresh Mobile designs: • 000: Init • 001: Normal • 010: Power-down • 011: Self refresh • 1XX: Deep power-down
0	DDRC_REG_MR_WR_BUSY	0×0	• 1: Indicates that a mode register write operation is in progress • 0: Indicates that the core can initiate a mode register write operation Core must initiate an MR write operation only if this signal is Low. This signal goes High in the clock after the controller accepts the write request. It goes Low when the MR write command is issued to the DRAM. Any MR write command that is received when DDRC_REG_MR_WR_BUSY is High, is not accepted.

Table 1-77. DDRC_SINGLE_ERR_CNT_STATUS_SR

Bit Number	Name	Reset Value	Description
[31:6	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
[15:0]	DDRC_SINGLE_ERR_CNT_STATUS_REG	0×0	Single error count status. If the count reaches 0xFFFF, it is held and only cleared after DDRC_ECC_INT_CLR_REG is written over by the system.

Table 1-78. DDRC_DOUBLE_ERR_CNT_STATUS_SR

Bit Number	Name	Reset Value	Description
[15:0]	DDRC_DOUBLE_ERR_CNT_STATUS_REG	0×0	Double error count status. If the count reaches 0xFFFF then it is held and only cleared after DDRC_ECC_INT_CLR_REG is written over by the system.
[31:6]	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.

Table 1-79. DDRC_LUE_SYNDROME_1_SR

Bit Number	Name	Reset Value	Description
[31:16]	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
[15:0]	DDRC_REG_ECC_SYNDROMES	0×0	72 bits are split into five registers. [15:0] bits of DDRC_REG_ECC_SYNDROMES. First data which has SECDED error in it. 72 bits consists of the following: SECDED: [71:64] – SECDED [63:00] – Data In the same clock cycle, if one lane has a correctable error and the other lane has an uncorrectable error, the syndrome for the uncorrectable error is sent on this bus. If more than one data lane has an error in it, the lower data lane is selected. The priority applied when there are multiple errors in the same cycle is as follows: • Uncorrectable error, lower lane • Uncorrectable error, upper lane • Correctable error, lower lane • Correctable error, upper lane Only present in designs that support SECDED. This is cleared after DDRC_ECC_INT_CLR_REG is written over by the system.

Table 1-80. DDRC_LUE_SYNDROME_2_SR

Bit Number	Name	Reset Value	Description
[31:16]	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.

...........continued

Bit Number	Name	Reset Value	Description
[15:0]	DDRC_REG_ECC_SYNDROMES	0×0	72 bits are split into five registers. [31:16] bits of DDRC_REG_ECC_SYNDROMES. First data which has SECDED error in it. 72 bits consists of the following: SECDED: • [71:64] – SECDED • [63:00] – Data In the same clock cycle, if one lane has a correctable error and the other lane has an uncorrectable error, the syndrome for the uncorrectable error is sent on this bus. If more than one data lane has an error in it, the lower data lane is selected. The priority applied when there are multiple errors in the same cycle is as follows: • Uncorrectable error, lower lane • Uncorrectable error, upper lane • Correctable error, lower lane • Correctable error, upper lane Only present in designs that support SECDED. This is cleared after DDRC_ECC_INT_CLR_REG is written over by the system.

Table 1-81. DDRC_LUE_SYNDROME_3_SR

Bit Number	Name	Reset Value	Description
[31:16]	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
[15:0]	DDRC_REG_ECC_SYNDROMES	0×0	72 bits are split into five registers. [47:32] bits of DDRC_REG_ECC_SYNDROMES. First data which has SECDED error in it. 72 bits consists of the following: SECDED: • [71:64] – SECDED • [63:00] – Data In the same clock cycle, if one lane has a correctable error and the other lane has an uncorrectable error, the syndrome for the uncorrectable error is sent on this bus. If more than one data lane has an error in it, the lower data lane is selected. The priority applied when there are multiple errors in the same cycle is as follows: • Uncorrectable error, lower lane • Uncorrectable error, upper lane • Correctable error, lower lane • Correctable error, upper lane Only present in designs that support SECDED. This is cleared after DDRC_ECC_INT_CLR_REG is written over by the system.

Table 1-82. DDRC_LUE_SYNDROME_4_SR

Bit Number	Name	Reset Value	Description
[31:16]	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
[15:0]	DDRC_REG_ECC_SYNDROMES	0×0	72 bits are split into five registers. [63:48] bits of DDRC_REG_ECC_SYNDROMES. First data which has SECDED error in it. 72 bits consists of the following: SECDED: • [71:64] – SECDED • [63:00] – Data In the same clock cycle, if one lane has a correctable error and the other lane has an uncorrectable error, the syndrome for the uncorrectable error is sent on this bus. If more than one data lane has an error in it, the lower data lane is selected. The priority applied when there are multiple errors in the same cycle is as follows: • Uncorrectable error, lower lane • Uncorrectable error, upper lane • Correctable error, lower lane • Correctable error, upper lane Only present in designs that support SECDED. This is cleared after DDRC_ECC_INT_CLR_REG is written over by the system.

Table 1-83. DDRC_LUE_SYNDROME_5_SR

Bit Number	Name	Reset Value	Description
[16:8]	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.

...........continued

Bit Number	Name	Reset Value	Description
[7:0]	DDRC_REG_ECC_SYNDROMES	0×0	72 bits are split into five registers. [71:64] bits of DDRC_REG_ECC_SYNDROMES. First data which has SECDED error in it. 72 bits consists of the following: SECDED: • [71:64] – SECDED • [63:00] – Data In the same clock cycle, if one lane has a correctable error and the other lane has an uncorrectable error, the syndrome for the uncorrectable error is sent on this bus. If more than one data lane has an error in it, the lower data lane is selected. The priority applied when there are multiple errors in the same cycle is as follows: • Uncorrectable error, lower lane • Uncorrectable error, upper lane • Correctable error, lower lane • Correctable error, upper lane Only present in designs that support SECDED. This is cleared after DDRC_ECC_INT_CLR_REG is written over by the system.

Table 1-84. DDRC_LUE_ADDRESS_1_SR

Bit Number	Name	Reset Value	Description
[31:15]	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
[14:12]	DDRC_REG_ECC_BANK	0×0	Bank where the SECDED error occurred. Only present in designs that support SECDED.
[11:0]	DDRC_REG_ECC_COL	0×0	Column where the SECDED error occurred. Col[0] is always set to 0, coming out of the controller. This bit is overwritten by the register module and indicates whether the error came from upper or lower lane. Only present in designs that support SECDED.

Table 1-85. DDRC_LUE_ADDRESS_2_SR

Bit Number	Name	Reset Value	Description
[31:16]	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
15:0	DDRC_REG_ECC_ROW	0×0	Row where the SECDED error occurred. Only present in designs that support SECDED.

Table 1-86. DDRC_LCE_SYNDROME_1_SR

Bit Number	Name	Reset Value	Description
[31:16]	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
[15:0]	DDRC_REG_ECC_SYNDROMES	0×0	72 bits are split into five registers. [15:0] bits of DDRC_REG_ECC_SYNDROMES. First data which has SECDED error in it. 72 bits consists of the following: SECDED: • [71:64] – SECDED • [63:00] – Data In the same clock cycle, if one lane has a correctable error and the other lane has an uncorrectable error, the syndrome for the uncorrectable error is sent on this bus. If more than one data lane has an error in it, the lower data lane is selected. The priority applied when there are multiple errors in the same cycle is as follows: • Uncorrectable error, lower lane • Uncorrectable error, upper lane • Correctable error, lower lane • Correctable error, upper lane Only present in designs that support SECDED. This is cleared after DDRC_ECC_INT_CLR_REG is written over by the system.

Table 1-87. DDRC_LCE_SYNDROME_2_SR

Bit Number	Name	Reset Value	Description
[31:16]	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.

...........continued

Bit Number	Name	Reset Value	Description
[15:0]	DDRC_REG_ECC_SYNDROMES	0×0	72 bits are split into five registers. [31:16] bits of DDRC_REG_ECC_SYNDROMES. First data which has SECDED error in it. 72 bits consists of the following: SECDED: • [71:64] – SECDED • [63:00] – Data In the same clock cycle, if one lane has a correctable error and the other lane has an uncorrectable error, the syndrome for the uncorrectable error is sent on this bus. If more than one data lane has an error in it, then the lower data lane is selected. The priority applied when there are multiple errors in the same cycle is as follows: • Uncorrectable error, lower lane • Uncorrectable error, upper lane • Correctable error, lower lane • Correctable error, upper lane Only present in designs that support SECDED. This is cleared after DDRC_ECC_INT_CLR_REG is written over by the system.

Table 1-88. DDRC_LCE_SYNDROME_3_SR

Bit Number	Name	Reset Value	Description
[31:16]	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
[15:0]	DDRC_REG_ECC_SYNDROMES	0×0	72 bits are split into five registers. [47:32] bits of DDRC_REG_ECC_SYNDROMES. First data which has SECDED error in it. 72 bits consists of the following: SECDED: • [71:64] – SECDED • [63:00] – Data In the same clock cycle, if one lane has a correctable error and the other lane has an uncorrectable error, the syndrome for the uncorrectable error is sent on this bus. If more than one data lane has an error in it, the lower data lane is selected. The priority applied when there are multiple errors in the same cycle is as follows: • Uncorrectable error, lower lane • Uncorrectable error, upper lane • Correctable error, lower lane • Correctable error, upper lane Only present in designs that support SECDED. This is cleared after DDRC_ECC_INT_CLR_REG is written over by the system.

MICROCHIP

Table 1-89. DDRC_LCE_SYNDROME_4_SR

Bit Number	Name	Reset Value	Description
[31:16]	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
[15:0]	DDRC_REG_ECC_SYNDROMES	0×0	72 bits are split into five registers. [63:48] bits of DDRC_REG_ECC_SYNDROMES. First data which has SECDED error in it. 72 bits consists of the following: SECDED: • [71:64] – SECDED • [63:00] – Data In the same clock cycle, if one lane has a correctable error and the other lane has an uncorrectable error, the syndrome for the uncorrectable error is sent on this bus. If more than one data lane has an error in it, the lower data lane is selected. The priority applied when there are multiple errors in the same cycle is as follows: • Uncorrectable error, lower lane • Uncorrectable error, upper lane • Correctable error, lower lane • Correctable error, upper lane Only present in designs that support SECDED. This is cleared after DDRC_ECC_INT_CLR_REG is written over by the system.

Table 1-90. DDRC_LCE_SYNDROME_5_SR

Bit Number	Name	Reset Value	Description
[16:8]	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.

...........continued

Bit Number	Name	Reset Value	Description
[7:0]	DDRC_REG_ECC_SYNDROMES	0×0	72 bits are split into five registers. [71:64] bits of DDRC_REG_ECC_SYNDROMES. First data which has SECDED error in it. 72 bits consists of the following SECDED: • [71:64] – SECDED • [63:00] – Data In the same clock cycle, if one lane has a correctable error and the other lane has an uncorrectable error, the syndrome for the uncorrectable error is sent on this bus. If more than one data lane has an error in it, the lower data lane is selected. The priority applied when there are multiple errors in the same cycle is as follows: • Uncorrectable error, lower lane • Uncorrectable error, upper lane • Correctable error, lower lane • Correctable error, upper lane Only present in designs that support SECDED. This is cleared after DDRC_ECC_INT_CLR_REG is written over by the system.

Table 1-91. DDRC_LCE_ADDRESS_1_SR

Bit Number	Name	Reset Value	Description
[31:15]	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
[14:12]	DDRC_REG_ECC_BANK	0×0	Bank where the SECDED error occurred.
[11:0]	DDRC_REG_ECC_COL	0×0	Column where the SECDED error occurred. Col[0] is always set to 0 coming out of the controller. This bit is overwritten by the register module and indicates whether the error came from upper or lower lane.

Table 1-92. DDRC_LCE_ADDRESS_2_SR

Bit Number	Name	Reset Value	Description
[31:16]	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
[15:0]	DDRC_REG_ECC_ROW	0×0	Row where the SECDED error occurred.

Table 1-93. DDRC_LCB_NUMBER_SR

Bit Number	Name	Reset Value	Description
[31:7]	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
[6:0]	DDRC_LCB_BIT_NUM	0×0	Indicates the location of the bit that caused a single-bit error in SECDED case (encoded value). If more than one data lane has an error in it, the lower data lane is selected. This register is 7 bits wide in order to handle 72 bits of the data present in a single lane. This does not indicate CORRECTED_BIT_NUM in the case of device correction SECDED. The encoding is only present in designs that support SECDED.

Table 1-94. DDRC_LCB_MASK_1_SR

Bit Number	Name	Reset Value	Description
[31:16]	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
[15:0]	DDRC_LCB_MASK	0×0	64 bits are split into four registers. [15:0] bits of DDRC_LCB_MASK. Indicates the mask of the corrected data. • 1: On any bit indicates that the bit has been corrected by the DRAM SECDED logic • 0: On any bit indicates that the bit has NOT been corrected by the DRAM SECDED logic Valid when any bit of DDRC_REG_ECC_CORRECTED_ERR is High. This mask doesn't indicate any correction that has been made in the SECDED check bits. If there are errors in multiple lanes, this signal will have the mask for the lowest lane.

Table 1-95. DDRC_LCB_MASK_2_SR

Bit Number	Name	Reset Value	Description
[31:16]	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
[15:0]	DDRC_LCB_MASK	0×0	64 bits are split into four registers. [31:16] bits of DDRC_LCB_MASK. Indicates the mask of the corrected data. • 1: On any bit indicates that the bit has been corrected by the DRAM SECDED logic • 0: On any bit indicates that the bit has NOT been corrected by the DRAM SECDED logic Valid when any bit of DDRC_REG_ECC_CORRECTED_ERR is High. This mask does not indicate any correction that has been made in the SECDED check bits. If there are errors in multiple lanes, this signal will have the mask for the lowest lane.

Table 1-96. DDRC_LCB_MASK_3_SR

Bit Number	Name	Reset Value	Description
[31:16]	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.

..........continued

Bit Number	Name	Reset Value	Description
[15:0]	DDRC_LCB_MASK	0×0	64 bits are split into four registers. [47:32] bits of DDRC_LCB_MASK. Indicates the mask of the corrected data. • 1: On any bit indicates that the bit has been corrected by the DRAM SECDED logic • 0: On any bit indicates that the bit has NOT been corrected by the DRAM SECDED logic Valid when any bit of DDRC_REG_ECC_CORRECTED_ERR is High. This mask does not indicate any correction that has been made in the SECDED check bits. If there are errors in multiple lanes, this signal will have the mask for the lowest lane.

Table 1-97. DDRC_LCB_MASK_4_SR

Bit Number	Name	Reset Value	Description
[31:16]	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
[15:0]	DDRC_LCB_MASK	0×0	64 bits are split into four registers. [63:48] bits of DDRC_LCB_MASK. Indicates the mask of the corrected data. • 1: On any bit indicates that the bit has been corrected by the DRAM SECDED logic • 0: On any bit indicates that the bit has NOT been corrected by the DRAM SECDED logic Valid when any bit of DDRC_REG_ECC_CORRECTED_ERR is High. This mask does not indicate any correction that has been made in the SECDED check bits. If there are errors in multiple lanes, this signal will have the mask for the lowest lane.

Table 1-98. DDRC_ECC_INT_SR

Bit Number	Name	Reset Value	Description
[31:3]	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
[2:0]	DDRC_ECC_STATUS_SR	0×0	• Bit 0: 1 Indicates the SECDED interrupt is due to a single error • Bit 1: 1 Indicates the SECDED interrupt is due to a double error • Bit 2: Always 1

Table 1-99. DDRC_ECC_INT_CLR_REG

Bit Number	Name	Reset Value	Description
[31:1]	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.

...........continued

Bit Number	Name	Reset Value	Description
0	DDRC_ECC_INT_CLR_REG	0×0	This register should be written by the processor when it has read the SECDED error status information. This helps to clear all the SECDED status information, such as error counters and other SECDED registers. The read value of this register is always 0.

1.11.4 PHY Configuration Register Summary (Ask a Question)

The following table summarizes the PHY register map.

➡ **Important:** The PHY register values must not be changed from Libero SoC defaults, as write leveling is not performed.

Table 1-100. PHY Configuration Register Summary

Register Name	Offset	Type	Reset Source	Description
PHY_DYN_BIST_TEST_CR	0×200	RW	PRESET_N	PHY BIST test configuration register
PHY_DYN_BIST_TEST_ERRCLR_1_CR	0×204	RW	PRESET_N	PHY BIST test error clear register
PHY_DYN_BIST_TEST_ERRCLR_2_CR	0×208	RW	PRESET_N	PHY BIST test error clear register
PHY_DYN_BIST_TEST_ERRCLR_3_CR	0×20C	RW	PRESET_N	PHY BIST test error clear register
PHY_BIST_TEST_SHIFT_PATTERN_1_CR	0×210	RW	PRESET_N	PHY BIST test shift pattern register
PHY_BIST_TEST_SHIFT_PATTERN_2_CR	0×214	RW	PRESET_N	PHY BIST test shift pattern register
PHY_BIST_TEST_SHIFT_PATTERN_3_CR	0×218	RW	PRESET_N	PHY BIST test shift pattern register
PHY_DYN_LOOPBACK_CR	0×21C	RW	PRESET_N	PHY loopback test configuration register
PHY_BOARD_LOOPBACK_CR	0×220	RW	PRESET_N	PHY Board loopback test configuration register
PHY_CTRL_SLAVE_RATIO_CR	0×224	RW	PRESET_N	PHY control slice DLL slave ratio register
PHY_CTRL_SLAVE_FORCE_CR	0×228	RW	PRESET_N	PHY control slice DLL slave force register
PHY_CTRL_SLAVE_DELAY_CR	0×22C	RW	PRESET_N	PHY control slice DLL slave delay register
PHY_DATA_SLICE_IN_USE_CR	0×230	RW	PRESET_N	PHY data slice in use register
PHY_LVL_NUM_OF_DQ0_CR	0×234	RW	PRESET_N	PHY receiver on off control register
PHY_DQ_OFFSET_1_CR	0×238	RW	PRESET_N	Selection register of offset value from DQS to DQ
PHY_DQ_OFFSET_2_CR	0×23C	RW	PRESET_N	Selection register of offset value from DQS to DQ
PHY_DQ_OFFSET_3_CR	0×240	RW	PRESET_N	Selection register of offset value from DQS to DQ
PHY_DIS_CALIB_RST_CR	0×244	RW	PRESET_N	Calibration reset disabling register
PHY_DLL_LOCK_DIFF_CR	0×248	RW	PRESET_N	Selects the maximum number of delay line taps
PHY_FIFO_WE_IN_DELAY_1_CR	0×24C	RW	PRESET_N	Delay value for FIFO WE
PHY_FIFO_WE_IN_DELAY_2_CR	0×250	RW	PRESET_N	Delay value for FIFO WE
PHY_FIFO_WE_IN_DELAY_3_CR	0×254	RW	PRESET_N	Delay value for FIFO WE
PHY_FIFO_WE_IN_FORCE_CR	0×258	RW	PRESET_N	Overwriting delay value selection reg for FIFO WE

...........continued

Register Name	Offset	Type	Reset Source	Description
PHY_FIFO_WE_SLAVE_RATIO_1_CR	0×25C	RW	PRESET_N	Ratio value for FIFO WE slave DLL
PHY_FIFO_WE_SLAVE_RATIO_2_CR	0×260	RW	PRESET_N	Ratio value for FIFO WE slave DLL
PHY_FIFO_WE_SLAVE_RATIO_3_CR	0×264	RW	PRESET_N	Ratio value for FIFO WE slave DLL
PHY_FIFO_WE_SLAVE_RATIO_4_CR	0×268	RW	PRESET_N	Ratio value for FIFO WE slave DLL
PHY_GATELVL_INIT_MODE_CR	0×26C	RW	PRESET_N	Init ratio selection register
PHY_GATELVL_INIT_RATIO_1_CR	0×270	RW	PRESET_N	Init ratio value configuration register
PHY_GATELVL_INIT_RATIO_2_CR	0×274	RW	PRESET_N	Init ratio value configuration register
PHY_GATELVL_INIT_RATIO_3_CR	0×278	RW	PRESET_N	Init ratio value configuration register
PHY_GATELVL_INIT_RATIO_4_CR	0×27C	RW	PRESET_N	Init ratio value configuration register
PHY_LOCAL_ODT_CR	0×280	RW	PRESET_N	PHY ODT control register
PHY_INVERT_CLKOUT_CR	0×284	RW	PRESET_N	PHY DRAM clock polarity change register
PHY_RD_DQS_SLAVE_DELAY_1_CR	0×288	RW	PRESET_N	Delay value for read DQS
PHY_RD_DQS_SLAVE_DELAY_2_CR	0×28C	RW	PRESET_N	Delay value for read DQS
PHY_RD_DQS_SLAVE_DELAY_3_CR	0×290	RW	PRESET_N	Delay value for read DQS
PHY_RD_DQS_SLAVE_FORCE_CR	0×294	RW	PRESET_N	Overwriting delay value selection reg for read DQS
PHY_RD_DQS_SLAVE_RATIO_1_CR	0×298	RW	PRESET_N	Ratio value for read DQS slave DLL
PHY_RD_DQS_SLAVE_RATIO_2_CR	0×29C	RW	PRESET_N	Ratio value for read DQS slave DLL
PHY_RD_DQS_SLAVE_RATIO_3_CR	0×2A0	RW	PRESET_N	Ratio value for read DQS slave DLL
PHY_RD_DQS_SLAVE_RATIO_4_CR	0×2A4	RW	PRESET_N	Ratio value for read DQS slave DLL
PHY_WR_DQS_SLAVE_DELAY_1_CR	0×2A8	RW	PRESET_N	Delay value for write DQS
PHY_WR_DQS_SLAVE_DELAY_2_CR	0×2AC	RW	PRESET_N	Delay value for write DQS
PHY_WR_DQS_SLAVE_DELAY_3_CR	0×2B0	RW	PRESET_N	Delay value for write DQS
PHY_WR_DQS_SLAVE_FORCE_CR	0×2B4	RW	PRESET_N	Overwriting delay value selection reg for write DQS
PHY_WR_DQS_SLAVE_RATIO_1_CR	0×2B8	RW	PRESET_N	Ratio value for write DQS slave DLL
PHY_WR_DQS_SLAVE_RATIO_2_CR	0×2BC	RW	PRESET_N	Ratio value for write DQS slave DLL
PHY_WR_DQS_SLAVE_RATIO_3_CR	0×2C0	RW	PRESET_N	Ratio value for write DQS slave DLL
PHY_WR_DQS_SLAVE_RATIO_4_CR	0×2C4	RW	PRESET_N	Ratio value for write DQS slave DLL
PHY_WR_DATA_SLAVE_DELAY_1_CR	0×2C8	RW	PRESET_N	Delay value for write DATA
PHY_WR_DATA_SLAVE_DELAY_2_CR	0×2CC	RW	PRESET_N	Delay value for write DATA
PHY_WR_DATA_SLAVE_DELAY_3_CR	0×2D0	RW	PRESET_N	Delay value for write DATA
PHY_WR_DATA_SLAVE_FORCE_CR	0×2D4	RW	PRESET_N	Overwriting delay value selection reg for write DATA
PHY_WR_DATA_SLAVE_RATIO_1_CR	0×2D8	RW	PRESET_N	Ratio value for write DATA slave DLL
PHY_WR_DATA_SLAVE_RATIO_2_CR	0×2DC	RW	PRESET_N	Ratio value for write DATA slave DLL
PHY_WR_DATA_SLAVE_RATIO_3_CR	0×2E0	RW	PRESET_N	Ratio value for write DATA slave DLL
PHY_WR_DATA_SLAVE_RATIO_4_CR	0×2E4	RW	PRESET_N	Ratio value for write DATA slave DLL
PHY_WRLVL_INIT_MODE_CR	0×2E8	RW	PRESET_N	Initialization ratio selection register used by write leveling
PHY_WRLVL_INIT_RATIO_1_CR	0×2EC	RW	PRESET_N	Configuring register for initialization ratio used by write leveling
PHY_WRLVL_INIT_RATIO_2_CR	0×2F0	RW	PRESET_N	Configuring register for initialization ratio used by write leveling
PHY_WRLVL_INIT_RATIO_3_CR	0×2F4	RW	PRESET_N	Configuring register for initialization ratio used by write leveling

Microchip

...........continued

Register Name	Offset	Type	Reset Source	Description
PHY_WRLVL_INIT_RATIO_4_CR	0×2F8	RW	PRESET_N	Configuring register for initialization ratio used by write leveling
PHY_WR_RD_RL_CR	0×2FC	RW	PRESET_N	Configurable register for delays to read and write
PHY_DYN_RDC_FIFO_RST_ERR_CNT_CLR_CR	0×300	RW	PRESET_N	Reset register for counter
PHY_RDC_WE_TO_RE_DELAY_CR	0×304	RW	PRESET_N	Configurable register for delay between WE and RE
PHY_USE_FIXED_RE_CR	0×308	RW	PRESET_N	Selection register for generating read enable to FIFO
PHY_USE_RANK0_DELAYS_CR	0×30C	RW	PRESET_N	Delay selection. This applies to multi-rank designs only
PHY_USE_LVL_TRNG_LEVEL_CTRL_CR	0×310	RW	PRESET_N	Training control register
PHY_DYN_CONFIG_CR	0×314	RW	PRESET_N	PHY dynamically controlled register
PHY_RD_WR_GATE_LVL_CR	0×318	RW	PRESET_N	Training mode selection register
PHY_DYN_RESET_CR	0×31C	RW	PRESET_N	This register will bring the PHY out of reset
PHY_LEVELLING_FAILURE_SR	0×320	RO	PRESET_N	Leveling failure status register
PHY_BIST_ERROR_1_SR	0×324	RO	PRESET_N	BIST error status register
PHY_BIST_ERROR_2_SR	0×328	RO	PRESET_N	BIST error status register
PHY_BIST_ERROR_3_SR	0×32C	RO	PRESET_N	BIST error status register
PHY_WRLVL_DQS_RATIO_1_SR	0×330	RO	PRESET_N	Write level DQS ratio status register
PHY_WRLVL_DQS_RATIO_2_SR	0×334	RO	PRESET_N	Write level DQS ratio status register
PHY_WRLVL_DQS_RATIO_3_SR	0×338	RO	PRESET_N	Write level DQS ratio status register
PHY_WRLVL_DQS_RATIO_4_SR	0×33C	RO	PRESET_N	Write level DQS ratio status register
PHY_WRLVL_DQ_RATIO_1_SR	0×340	RO	PRESET_N	Write level DQ ratio status register
PHY_WRLVL_DQ_RATIO_2_SR	0×344	RO	PRESET_N	Write level DQ ratio status register
PHY_WRLVL_DQ_RATIO_3_SR	0×348	RO	PRESET_N	Write level DQ ratio status register
PHY_WRLVL_DQ_RATIO_4_SR	0×34C	RO	PRESET_N	Write level DQ ratio status register
PHY_RDLVL_DQS_RATIO_1_SR	0×350	RO	PRESET_N	Read level DQS ratio status register
PHY_RDLVL_DQS_RATIO_2_SR	0×354	RO	PRESET_N	Read level DQS ratio status register
PHY_RDLVL_DQS_RATIO_3_SR	0×358	RO	PRESET_N	Read level DQS ratio status register
PHY_RDLVL_DQS_RATIO_4_SR	0×35C	RO	PRESET_N	Read level DQS ratio status register
PHY_FIFO_1_SR	0×360	RO	PRESET_N	FIFO status register
PHY_FIFO_2_SR	0×364	RO	PRESET_N	FIFO status register
PHY_FIFO_3_SR	0×368	RO	PRESET_N	FIFO status register
PHY_FIFO_4_SR	0×36C	RO	PRESET_N	FIFO status register
PHY_MASTER_DLL_SR	0×370	RO	PRESET_N	Master DLL status register
PHY_DLL_SLAVE_VALUE_1_SR	0×374	RO	PRESET_N	Slave DLL status register
PHY_DLL_SLAVE_VALUE_2_SR	0×378	RO	PRESET_N	Slave DLL status register
PHY_STATUS_OF_IN_DELAY_VAL_1_SR	0×37C	RO	PRESET_N	IN delay status register
PHY_STATUS_OF_IN_DELAY_VAL_2_SR	0×380	RO	PRESET_N	IN delay status register
PHY_STATUS_OF_OUT_DELAY_VAL_1_SR	0×384	RO	PRESET_N	OUT delay status register
PHY_STATUS_OF_OUT_DELAY_VAL_2_SR	0×388	RO	PRESET_N	OUT delay status register
PHY_DLL_LOCK_AND_SLAVE_VAL_SR	0×38C	RO	PRESET_N	DLL lock status register
PHY_CTRL_OUTPUT_FILTER_SR	0×390	RO	PRESET_N	Control output filter status register
PHY_RD_DQS_SLAVE_DLL_VAL_1_SR	0×398	RO	PRESET_N	Read DQS slave DLL status register

...........continued

Register Name	Offset	Type	Reset Source	Description
PHY_RD_DQS_SLAVE_DLL_VAL_2_SR	0×39C	RO	PRESET_N	Read DQS slave DLL status register
PHY_RD_DQS_SLAVE_DLL_VAL_3_SR	0×3A0	RO	PRESET_N	Read DQS slave DLL status register
PHY_WR_DATA_SLAVE_DLL_VAL_1_SR	0×3A4	RO	PRESET_N	Write DATA slave DLL status register
PHY_WR_DATA_SLAVE_DLL_VAL_2_SR	0×3A8	RO	PRESET_N	Write DATA slave DLL status register
PHY_WR_DATA_SLAVE_DLL_VAL_3_SR	0×3AC	RO	PRESET_N	Write DATA slave DLL status register
PHY_FIFO_WE_SLAVE_DLL_VAL_1_SR	0×3B0	RO	PRESET_N	FIFO WE slave DLL status register
PHY_FIFO_WE_SLAVE_DLL_VAL_2_SR	0×3B4	RO	PRESET_N	FIFO WE slave DLL status register
PHY_FIFO_WE_SLAVE_DLL_VAL_3_SR	0×3B8	RO	PRESET_N	FIFO WE slave DLL status register
PHY_WR_DQS_SLAVE_DLL_VAL_1_SR	0×3BC	RO	PRESET_N	Write DQS slave DLL status register
PHY_WR_DQS_SLAVE_DLL_VAL_2_SR	0×3C0	RO	PRESET_N	Write DQS slave DLL status register
PHY_WR_DQS_SLAVE_DLL_VAL_3_SR	0×3C4	RO	PRESET_N	Write DQS slave DLL status register
PHY_CTRL_SLAVE_DLL_VAL_SR	0×3C8	RO	PRESET_N	DLL controller status register

1.11.5   PHY Configuration Register Bit Definitions (Ask a Question)

Table 1-101. PHY_DYN_BIST_TEST_CR

Bit Number	Name	Reset Value	Description
[31:5]	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
4	REG_PHY_AT_SPD_ATPG	0×0	• 1: Test with full clock speed but lower coverage • 0: Test with lower clock speed but higher coverage
3	REG_PHY_BIST_ENABLE	0×0	Enable the internal BIST generation and checker logic when this port is set High. Setting this port as '0' will stop the BIST generator / checker. In order to run BIST tests, this port must be set along with REG_PHY_LOOPBACK.
[2:1]	REG_PHY_BIST_MODE	0×0	The mode bits select the pattern type generated by the BIST generator. All the patterns are transmitted continuously once enabled. • 00: Constant pattern (0 repeated on each DQ bit) • 01: Low frequency pattern (00001111 repeated on each DQ bit) • 10: PRBS pattern ($2^7 - 1$ PRBS pattern repeated on each DQ bit) Each DQ bit always has same data value except when early shifting in PRBS mode is requested.
0	REG_PHY_BIST_FORCE_ERR	0×0	This register bit is used to check that the BIST checker is not giving a false pass. When this port is set to 1, the data bit gets inverted before sending out to the external memory and BIST checker must return a mismatch error.

Table 1-102. PHY_DYN_BIST_TEST_ERRCLR_1_CR

Bit Number	Name	Reset Value	Description
[31:16]	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
[15:0]	REG_PHY_BIST_ERR_CLR	0×0	[15:0] bits of REG_PHY_BIST_ERR_CLR. Clear the mismatch error flag from the BIST checker. • 1: Sticky error flag is cleared • 0: No effect

Table 1-103. PHY_DYN_BIST_TEST_ERRCLR_2_CR

Bit Number	Name	Reset Value	Description
[31:16]	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
[15:0]	REG_PHY_BIST_ERR_CLR	0×0	[31:16] bits of REG_PHY_BIST_ERR_CLR. Clear the mismatch error flag from the BIST checker. • 1: Sticky error flag is cleared • 0: No effect

Table 1-104. PHY_DYN_BIST_TEST_ERRCLR_3_CR

Bit Number	Name	Reset Value	Description
[31:12]	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
[11:0]	REG_PHY_BIST_ERR_CLR	0×0	[43:32] bits of REG_PHY_BIST_ERR_CLR. Clear the mismatch error flag from the BIST checker. • 1: Sticky error flag is cleared • 0: No effect

Table 1-105. PHY_BIST_TEST_SHIFT_PATTERN_1_CR

Bit Number	Name	Reset Value	Description
[31:16]	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
[15:0]	REG_PHY_BIST_SHIFT_DQ	0×0	[15:0] bits of REG_PHY_BIST_SHIFT_DQ. Determines whether early shifting is required for a particular DQ bit when REG_PHY_BIST_MODE is 10. • 1: PRBS pattern shifted early by 1 bit • 0: PRBS pattern without any shift

Table 1-106. PHY_BIST_TEST_SHIFT_PATTERN_2_CR

Bit Number	Name	Reset Value	Description
[31:16]	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.

..........continued

Bit Number	Name	Reset Value	Description
[15:0]	REG_PHY_BIST_SHIFT_DQ	0×0	[31:16] bits of REG_PHY_BIST_SHIFT_DQ. Determines whether early shifting is required for a particular DQ bit when REG_PHY_BIST_MODE is 10. • 1: PRBS pattern shifted early by 1 bit • 0: PRBS pattern without any shift

Table 1-107. PHY_BIST_TEST_SHIFT_PATTERN_3_CR

Bit Number	Name	Reset Value	Description
[31:12]	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
[11:0]	REG_PHY_BIST_SHIFT_DQ	0×0	[43:32] bits of REG_PHY_BIST_SHIFT_DQ. Determines whether early shifting is required for a particular DQ bit when REG_PHY_BIST_MODE is 10. • 1: PRBS pattern shifted early by 1 bit • 0: PRBS pattern without any shift

Table 1-108. PHY_DYN_LOOPBACK_CR

Bit Number	Name	Reset Value	Description
[31:1]	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
0	REG_PHY_LOOPBACK	0×0	Loopback testing. • 1: Enable • 0: Disable

Table 1-109. PHY_BOARD_LOOPBACK_CR

Bit Number	Name	Reset Value	Description
[31:10]	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
[9:5]	REG_PHY_BOARD_LPBK_TX	0×0	External board loopback testing. • 1: This slice behaves as a transmitter for board loopback • 0: Default This port must always be set to '0' except when in external board-level loopback test mode.
[4:0]	REG_PHY_BOARD_LPBK_RX	0×0	External board loopback testing. • 1: This slice behaves as a receiver for board loopback • 0: Disable This port must always be set to '0' except when in external board-level loopback test mode.

Table 1-110. PHY_CTRL_SLAVE_RATIO_CR

Bit Number	Name	Reset Value	Description
[31:10]	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
[9:0]	REG_PHY_CTRL_SLAVE_RATIO	0×0	Ratio value for address/command launches timing in PHY_CTRL macro. This is the fraction of a clock cycle represented by the shift to be applied to the read DQS in units of 256ths. In other words, the full-cycle tap value from the master DLL will be scaled by this number over 256 to get the delay value for the slave delay line.

Table 1-111. PHY_CTRL_SLAVE_FORCE_CR

Bit Number	Name	Reset Value	Description
[31:1]	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
0	REG_PHY_CTRL_SLAVE_FORCE	0×0	1: Overwrite the delay/tap value for address/ command timing slave DLL with the value of the REG_PHY_RD_DQS_SLAVE_DELAY bus

Table 1-112. PHY_CTRL_SLAVE_DELAY_CR

Bit Number	Name	Reset Value	Description
[31:9]	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
[8:0]	REG_PHY_CTRL_SLAVE_DELAY	0×0	If REG_PHY_RD_DQS_SLAVE_FORCE is 1, replace delay/tap value for address/command timing slave DLL with this value.

Table 1-113. PHY_DATA_SLICE_IN_USE_CR

Bit Number	Name	Reset Value	Description
[31:5]	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
[4:0]	REG_PHY_DATA_SLICE_IN_USE	0×0	Data bus width selection for read FIFO RE generation. One bit for each data slice. • 1: Data slice is valid • 0: Read data responses are ignored **Note:** The PHY data slice 0 must always be enabled.

Table 1-114. PHY_LVL_NUM_OF_DQ0_CR

Bit Number	Name	Reset Value	Description
[31:8]	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.

...........continued

Bit Number	Name	Reset Value	Description
[7:4]	REG_PHY_GATELVL_NUM_OF_DQ0	0×0	This register value determines the number of samples for dq0_in for each ratio increment by the gate training FSM. NUM_OF_ITERATION = REG_PHY_GATELVL_NUM_OF_DQ0 + 1
[3:0]	REG_PHY_WRLVL_NUM_OF_DQ0	0×0	This register value determines the number of samples for dq0_in for each ratio increment by the write leveling FSM. NUM_OF_ITERATION = REG_PHY_GATELVL_NUM_OF_DQ0 + 1

Table 1-115. PHY_DQ_OFFSET_1_CR

Bit Number	Name	Reset Value	Description
[31:16]	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
[15:0]	REG_PHY_DQ_OFFSET	0×0240	[15:0] bits of REG_PHY_DQ_OFFSET. Offset value from DQS to DQ. Default value: 0×40 (for 90 degree shift). This is only used when REG_PHY_USE_WR_LEVEL = 1.

Table 1-116. PHY_DQ_OFFSET_2_CR

Bit Number	Name	Reset Value	Description
[31:16]	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
[15:0]	REG_PHY_DQ_OFFSET	0×4081	[31:16] bits of REG_PHY_DQ_OFFSET. Offset value from DQS to DQ. Default value: 0×40 (for 90 degree shift). This is only used when REG_PHY_USE_WR_LEVEL = 1.

Table 1-117. PHY_DQ_OFFSET_3_CR

Bit Number	Name	Reset Value	Description
[31:3]	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
[2:0]	REG_PHY_DQ_OFFSET	0×0	[34:32] bits of REG_PHY_DQ_OFFSET. Offset value from DQS to DQ. Default value: 0×40 (for 90 degree shift). This is only used when REG_PHY_USE_WR_LEVEL = 1.

Table 1-118. PHY_DIS_CALIB_RST_CR

Bit Number	Name	Reset Value	Description
[31:1]	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.

...........continued

Bit Number	Name	Reset Value	Description
0	REG_PHY_DIS_CALIB_RST	0×0	Disables the resetting of the read capture FIFO pointers with DLL_CALIB (internally generated signal). The pointers are reset to ensure that the PHY can recover if the appropriate number of DQS edges is not observed after a read command (which can happen when the DQS squelch timing is manually overridden via the debug registers). 0: Enable 1: Disable

Table 1-119. PHY_DLL_LOCK_DIFF_CR

Bit Number	Name	Reset Value	Description
[31:4]	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
[3:0]	REG_PHY_DLL_LOCK_DIFF	0×0	The maximum number of delay line taps variations allowed while maintaining the master DLL lock. This is calculated as total jitter/ delay line tap size. Where total jitter is half of (incoming clock jitter (pp) + delay line jitter (pp)).

Table 1-120. PHY_FIFO_WE_IN_DELAY_1_CR

Bit Number	Name	Reset Value	Description
[31:16]	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
[15:0]	REG_PHY_FIFO_WE_IN_DELAY	0×0	REG_PHY_FIFO_WE_IN_DELAY is a 45 bit register. It has 9 bits for every 8 bits of data {DQ[7:0], DQ[15:8], DQ[23:16], DQ[31:24] and DQ_ECC[3:0]). This is to delay the TMATCH_OUT by fraction of clock cycle. Delay = {(REG_PHY_FIFO_WE_IN_DELAY [8:0]/256) * (CLOCK period)} **Note:** Microchip recommends to use same delay values for every data slice. This register has [15:0] bits of REG_PHY_FIFO_WE_IN_DELAY. Delay value to be used when REG_PHY_FIFO_WE_IN_FORCEX is set to 1.

Table 1-121. PHY_FIFO_WE_IN_DELAY_2_CR

Bit Number	Name	Reset Value	Description
[31:16]	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
[15:0]	REG_PHY_FIFO_WE_IN_DELAY	0×0	[31:16] bits of REG_PHY_FIFO_WE_IN_DELAY. Delay value to be used when REG_PHY_FIFO_WE_IN_FORCEX is set to 1.

Table 1-122. PHY_FIFO_WE_IN_DELAY_3_CR

Bit Number	Name	Reset Value	Description
[31:13]	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
[12:0]	REG_PHY_FIFO_WE_IN_DELAY	0×0	[44:32] bits of REG_PHY_FIFO_WE_IN_DELAY. Delay value to be used when REG_PHY_FIFO_WE_IN_FORCEX is set to 1.

Table 1-123. PHY_FIFO_WE_IN_FORCE_CR

Bit Number	Name	Reset Value	Description
[7:5]	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
[4:0]	REG_PHY_FIFO_WE_IN_FORCE	0×0	1: Overwrite the delay value for the TMATCH_OUT with the value of the REG_PHY_FIFO_WE_IN_DELAY register. REG_PHY_FIFO_WE_IN_FORCE has a bit for every 8bits of data {DQ[7:0], DQ[15:8], DQ[23:16], DQ[31:24] and DQ_ECC[3:0]).

Table 1-124. PHY_FIFO_WE_SLAVE_RATIO_1_CR

Bit Number	Name	Reset Value	Description
[31:16]	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
[15:0]	REG_PHY_FIFO_WE_SLAVE_RATIO	0×0	[15:0] bits of REG_PHY_FIFO_WE_SLAVE_RATIO DDR PHY uses REG_PHY_FIFO_WE_SLAVE_RATIO when read DQS gate training is disabled and PHY_FIFO_WE_IN_FORCE_CR is set to 0. REG_PHY_FIFO_WE_SLAVE_RATIO needs to be configured to 2 × board delay + (PHY_INVERT_CLKOUT) × 0x80. REG_PHY_FIFO_WE_SLAVE_RATIO is a 55-bit register with 11-bits for each data slice. [10:0] bits for data slice 0 [21:11] bits for data slice 1 [32:22] bits for data slice 2 [43:33] bits for data slice 3 [54:44] bits for data slice 4 This register is divided into the following 16-bit registers: REG_PHY_FIFO_WE_SLAVE_RATIO_1_CR REG_PHY_FIFO_WE_SLAVE_RATIO_2_CR REG_PHY_FIFO_WE_SLAVE_RATIO_CR REG_PHY_FIFO_WE_SLAVE_RATIO_4_CR

Table 1-125. PHY_FIFO_WE_SLAVE_RATIO_2_CR

Bit Number	Name	Reset Value	Description
[31:15]	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
[15:0]	REG_PHY_FIFO_WE_SLAVE_RATIO	0×0	[31:16] bits of REG_PHY_FIFO_WE_SLAVE_RATIO

Table 1-126. PHY_FIFO_WE_SLAVE_RATIO_3_CR

Bit Number	Name	Reset Value	Description
[31:15]	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
[15:0]	REG_PHY_FIFO_WE_SLAVE_RATIO	0×0	[47:32] bits of REG_PHY_FIFO_WE_SLAVE_RATIO

Table 1-127. PHY_FIFO_WE_SLAVE_RATIO_4_CR

Bit Number	Name	Reset Value	Description
[31:7]	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
[6:0]	REG_PHY_FIFO_WE_SLAVE_RATIO	0×0	[54:48] bits of REG_PHY_FIFO_WE_SLAVE_RATIO

Table 1-128. PHY_GATELVL_INIT_MODE_CR

Bit Number	Name	Reset Value	Description
[31:1]	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
0	REG_PHY_GATELVL_INIT_MODE	0×0	The user programmable init ratio selection mode. • 1: Selects a starting ratio value based on REG_PHY_GATELVL_INIT_RATIO port • 0: Selects a starting ratio value based on write leveling of the same data slice

Table 1-129. PHY_GATELVL_INIT_RATIO_1_CR

Bit Number	Name	Reset Value	Description
[31:16]	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.

..........continued

Bit Number	Name	Reset Value	Description
[15:0]	REG_PHY_GATELVL_INIT_RATIO	0×0	[15:0] of REG_PHY_GATELVL_INIT_RATIO DDR PHY uses REG_PHY_GATELVL_INIT_RATIO when read DQS gate training is enabled. The read DQS gate training tunes the DQS signal delays. The Libero DDR Configurator does not allow you to configure this register, because by default the read DQS gate training is disabled. The read DQS gate training can be enabled by modifying the register configuration file and importing the file in the System Builder memory controller configuration page. When read DQS gate training is enabled REG_PHY_GATELVL_INIT_RATIO needs to be configured to Ratio = SKEW (in ps) × 256)/MDLL_CLK_PERIOD (in ps) – 0x20. Where, skew is the board routing delay for a data slice. REG_PHY_GATELVL_INIT_RATIO is a 55-bit register with 11-bits for each data slice. • [10:0] bits for data slice 0 • [21:11] bits for data slice 1 • [32:22] bits for data slice 2 • [43:33] bits for data slice 3 • [54:44] bits for data slice 4 This register is divided into the following 16-bit registers: • PHY_GATELVL_INIT_RATIO_1_CR • PHY_GATELVL_INIT_RATIO_2_CR • PHY_GATELVL_INIT_RATIO_3_CR • PHY_GATELVL_INIT_RATIO_4_CR

Table 1-130. PHY_GATELVL_INIT_RATIO_2_CR

Bit Number	Name	Reset Value	Description
[31:16]	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
[15:0]	REG_PHY_GATELVL_INIT_RATIO	0×0	[31:16] of REG_PHY_GATELVL_INIT_RATIO

Table 1-131. PHY_GATELVL_INIT_RATIO_3_CR

Bit Number	Name	Reset Value	Description
[31:16]	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
[15:0]	REG_PHY_GATELVL_INIT_RATIO	0×0	[47:32] of REG_PHY_GATELVL_INIT_RATIO

Table 1-132. PHY_GATELVL_INIT_RATIO_4_CR

Bit Number	Name	Reset Value	Description
[31:7]	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.

...........continued

Bit Number	Name	Reset Value	Description
[6:0]	REG_PHY_GATELVL_INIT_RATIO	0×0	[54:48] of REG_PHY_GATELVL_INIT_RATIO

Table 1-133. PHY_LOCAL_ODT_CR

Bit Number	Name	Reset Value	Description
[31:4]	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
[3:2]	REG_PHY_IDLE_LOCAL_ODT	0×0	The user programmable initialization ratio selection mode. 01: Selects a starting ratio value based on the REG_PHY_GATELVL_INIT_RATIO port 00: Selects a starting ratio value based on write leveling of the same data slice
1	REG_PHY_WR_LOCAL_ODT	0×0	Tied to 0
0	REG_PHY_RD_LOCAL_ODT	0×0	Tied to 0

Table 1-134. PHY_INVERT_CLKOUT_CR

Bit Number	Name	Reset Value	Description
[31:1]	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
0	REG_PHY_INVERT_CLKOUT	0×0	Inverts the polarity of the DRAM clock. • 0: Core clock is passed on to DRAM. Most common usage mode • 1: Inverted core clock is passed on to DRAM. Use this when CLK can arrive at a DRAM device ahead of DQS or coincidence with DQS based on board topology. This effectively delays the CLK to the DRAM device by half a cycle, providing a CLK edge that DQS can align to during leveling.

Table 1-135. PHY_RD_DQS_SLAVE_DELAY_1_CR

Bit Number	Name	Reset Value	Description
[31:16]	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
[15:0]	REG_PHY_RD_DQS_SLAVE_DELAY	0×0	[15:0] bits of REG_PHY_RD_DQS_SLAVE_DELAY If REG_PHY_RD_DQS_SLAVE_FORCE is 1, replace delay/tap value for read DQS slave DLL with this value.

Table 1-136. PHY_RD_DQS_SLAVE_DELAY_2_CR

Bit Number	Name	Reset Value	Description
[31:16]	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.

..........continued

Bit Number	Name	Reset Value	Description
[15:0]	REG_PHY_RD_DQS_SLAVE_DELAY	0×0	[31:16] bits of REG_PHY_RD_DQS_SLAVE_DELAY If REG_PHY_RD_DQS_SLAVE_FORCE is 1, replace delay/tap value for read DQS slave DLL with this value.

Table 1-137. PHY_RD_DQS_SLAVE_DELAY_3_CR

Bit Number	Name	Reset Value	Description
[31:13]	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
[12:0]	REG_PHY_RD_DQS_SLAVE_DELAY	0×0	[44:32] bits of REG_PHY_RD_DQS_SLAVE_DELAY If REG_PHY_RD_DQS_SLAVE_FORCE is 1, replace delay/tap value for read DQS slave DLL with this value.

Table 1-138. PHY_RD_DQS_SLAVE_FORCE_CR

Bit Number	Name	Reset Value	Description
[31:1]	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
0	REG_PHY_RD_DQS_SLAVE_FORCE	0×0	1: Overwrite the delay/tap value for read DQS slave DLL with the value of PHY_RD_DQS_SLAVE_DELAY

Table 1-139. PHY_RD_DQS_SLAVE_RATIO_1_CR

Bit Number	Name	Reset Value	Description
[31:16]	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
[15:0]	REG_PHY_RD_DQS_SLAVE_RATIO	0×0040	[15:0] bits of REG_PHY_RD_DQS_SLAVE_RATIO Ratio value for read DQS slave DLL. This is the fraction of a clock cycle represented by the shift to be applied to the read DQS in units of 256ths. In other words, the full-cycle tap value from the master DLL will be scaled by this number over 256 to get the delay value for the slave delay line. Default value: 0x40

Table 1-140. PHY_RD_DQS_SLAVE_RATIO_2_CR

Bit Number	Name	Reset Value	Description
[31:16]	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.

...........continued

Bit Number	Name	Reset Value	Description
[15:0]	REG_PHY_RD_DQS_SLAVE_RATIO	0×0401	[31:16] bits of REG_PHY_RD_DQS_SLAVE_RATIO Ratio value for read DQS slave DLL. This is the fraction of a clock cycle represented by the shift to be applied to the read DQS in units of 256ths. In other words, the full-cycle tap value from the master DLL will be scaled by this number over 256 to get the delay value for the slave delay line. Default value: 0x40

Table 1-141. PHY_RD_DQS_SLAVE_RATIO_3_CR

Bit Number	Name	Reset Value	Description
[31:16]	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
[15:0]	REG_PHY_RD_DQS_SLAVE_RATIO	0×4010	[47:32] bits of REG_PHY_RD_DQS_SLAVE_RATIO Ratio value for read DQS slave DLL. This is the fraction of a clock cycle represented by the shift to be applied to the read DQS in units of 256ths. In other words, the full-cycle tap value from the master DLL will be scaled by this number over 256 to get the delay value for the slave delay line. Default value: 0×40

Table 1-142. PHY_RD_DQS_SLAVE_RATIO_4_CR

Bit Number	Name	Reset Value	Description
[31:2]	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
[1:0]	REG_PHY_RD_DQS_SLAVE_RATIO	0×0	[49:48] bits of REG_PHY_RD_DQS_SLAVE_RATIO Ratio value for read DQS slave DLL. This is the fraction of a clock cycle represented by the shift to be applied to the read DQS in units of 256ths. In other words, the full-cycle tap value from the master DLL will be scaled by this number over 256 to get the delay value for the slave delay line. Default value: 0×40

Table 1-143. PHY_WR_DQS_SLAVE_DELAY_1_CR

Bit Number	Name	Reset Value	Description
[31:16]	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
[15:0]	REG_PHY_WR_DQS_SLAVE_DELAY	0×0	[15:0] bits of REG_PHY_WR_DQS_SLAVE_DELAY If REG_PHY_WR_DQS_SLAVE_FORCE is 1, replace delay/tap value for read DQS slave DLL with this value.

Table 1-144. PHY_WR_DQS_SLAVE_DELAY_2_CR

Bit Number	Name	Reset Value	Description
[31:16]	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
[15:0]	REG_PHY_WR_DQS_SLAVE_DELAY	0×0	[31:16] bits of REG_PHY_WR_DQS_SLAVE_DELAY If REG_PHY_WR_DQS_SLAVE_FORCE is 1, replace delay/tap value for read DQS slave DLL with this value.

Table 1-145. PHY_WR_DQS_SLAVE_DELAY_3_CR

Bit Number	Name	Reset Value	Description
[31:13]	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
[12:0]	REG_PHY_WR_DQS_SLAVE_DELAY	0×0	[44:32] bits of REG_PHY_WR_DQS_SLAVE_DELAY If REG_PHY_WR_DQS_SLAVE_FORCE is 1, replace delay/tap value for read DQS slave DLL with this value.

Table 1-146. PHY_WR_DQS_SLAVE_FORCE_CR

Bit Number	Name	Reset Value	Description
[31:5]	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
[4:0]	REG_PHY_WR_DQS_SLAVE_FORCE	0×0	1: Overwrite the delay/tap value for read DQS slave DLL with the value of the REG_PHY_WR_DQS_SLAVE_DELAY bus. Bit-4 is for PHY Data slice 4, bit-3 for PHY Data slice 3 and so on.

Table 1-147. PHY_WR_DQS_SLAVE_RATIO_1_CR

Bit Number	Name	Reset Value	Description
[31:16]	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
[15:0]	REG_PHY_WR_DQS_SLAVE_RATIO	0×0	[15:0] bits of REG_PHY_WR_DQS_SLAVE_RATIO Ratio value for read DQS slave DLL. This is the fraction of a clock cycle represented by the shift to be applied to the read DQS in units of 256ths. In other words, the full-cycle tap value from the master DLL will be scaled by this number over 256 to get the delay value for the slave delay line. Default value: 0×40

Table 1-148. PHY_WR_DQS_SLAVE_RATIO_2_CR

Bit Number	Name	Reset Value	Description
[31:16]	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.

...........continued

Bit Number	Name	Reset Value	Description
[15:0]	REG_PHY_WR_DQS_SLAVE_RATIO	0×0	[31:16] bits of REG_PHY_WR_DQS_SLAVE_RATIO Ratio value for read DQS slave DLL. This is the fraction of a clock cycle represented by the shift to be applied to the read DQS in units of 256ths. In other words, the full-cycle tap value from the master DLL will be scaled by this number over 256 to get the delay value for the slave delay line. Default value: 0×40

Table 1-149. PHY_WR_DQS_SLAVE_RATIO_3_CR

Bit Number	Name	Reset Value	Description
[31:16]	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
[15:0]	REG_PHY_WR_DQS_SLAVE_RATIO	0×0	[47:32] bits of REG_PHY_WR_DQS_SLAVE_RATIO Ratio value for read DQS slave DLL. This is the fraction of a clock cycle represented by the shift to be applied to the read DQS in units of 256ths. In other words, the full-cycle tap value from the master DLL will be scaled by this number over 256 to get the delay value for the slave delay line. Default value: 0×40

Table 1-150. PHY_WR_DQS_SLAVE_RATIO_4_CR

Bit Number	Name	Reset Value	Description
[31:2]	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
[1:0]	REG_PHY_WR_DQS_SLAVE_RATIO	0×0	[49:48] bits of REG_PHY_WR_DQS_SLAVE_RATIO Ratio value for read DQS slave DLL. This is the fraction of a clock cycle represented by the shift to be applied to the read DQS in units of 256ths. In other words, the full-cycle tap value from the master DLL will be scaled by this number over 256 to get the delay value for the slave delay line. Default value: 0×40

Table 1-151. PHY_WR_DATA_SLAVE_DELAY_1_CR

Bit Number	Name	Reset Value	Description
[31:16]	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
[15:0]	REG_PHY_WR_DATA_SLAVE_DELAY	0×0	[15:0] bits of REG_PHY_WR_DATA_SLAVE_DELAY If REG_PHY_WR_DATA_SLAVE_FORCE is 1, replace delay/tap value for write data slave DLL with this value.

MICROCHIP

Table 1-152. PHY_WR_DATA_SLAVE_DELAY_2_CR

Bit Number	Name	Reset Value	Description
[31:16]	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
[15:0]	REG_PHY_WR_DATA_SLAVE_DELAY	0×0	[31:16] bits of REG_PHY_WR_DATA_SLAVE_DELAY If REG_PHY_WR_DATA_SLAVE_FORCE is 1, replace delay/tap value for write data slave DLL with this value.

Table 1-153. PHY_WR_DATA_SLAVE_DELAY_3_CR

Bit Number	Name	Reset Value	Description
[31:13]	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
[12:0]	REG_PHY_WR_DATA_SLAVE_DELAY	0×0	[44:32] bits of REG_PHY_WR_DATA_SLAVE_DELAY If REG_PHY_WR_DATA_SLAVE_FORCE is 1, replace delay/tap value for write data slave DLL with this value.

Table 1-154. PHY_WR_DATA_SLAVE_FORCE_CR

Bit Number	Name	Reset Value	Description
[31:5]	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
[4:0]	REG_PHY_WR_DATA_SLAVE_FORCE	0×0	1: Overwrite the delay/tap value for write data slave DLL with the value of the REG_PHY_WR_DATA_SLAVE_DELAY bus. Bit-4 is for PHY Data slice 4, bit-3 for PHY Data slice 3 and so on.

Table 1-155. PHY_WR_DATA_SLAVE_RATIO_1_CR

Bit Number	Name	Reset Value	Description
[31:16]	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
[15:0]	REG_PHY_WR_DATA_SLAVE_RATIO	0×0040	[15:0] bits of REG_PHY_WR_DATA_SLAVE_RATIO Ratio value for write data slave DLL. This is the fraction of a clock cycle represented by the shift to be applied to the write DQ MUXes in units of 256ths. In other words, the full-cycle tap value from the master DLL will be scaled by this number over 256 to get the delay value for the slave delay line. This is only used when REG_PHY_USE_WR_LEVEL = 0.

Table 1-156. PHY_WR_DATA_SLAVE_RATIO_2_CR

Bit Number	Name	Reset Value	Description
[31:16]	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
[15:0]	REG_PHY_WR_DATA_SLAVE_RATIO	0×0401	[31:16] bits of REG_PHY_WR_DATA_SLAVE_RATIO Ratio value for write data slave DLL. This is the fraction of a clock cycle represented by the shift to be applied to the write DQ MUXes in units of 256ths. In other words, the full-cycle tap value from the master DLL will be scaled by this number over 256 to get the delay value for the slave delay line. This is only used when REG_PHY_USE_WR_LEVEL = 0.

Table 1-157. PHY_WR_DATA_SLAVE_RATIO_3_CR

Bit Number	Name	Reset Value	Description
[31:16]	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
[15:0]	REG_PHY_WR_DATA_SLAVE_RATIO	0×0401	[47:32] bits of REG_PHY_WR_DATA_SLAVE_RATIO Ratio value for write data slave DLL. This is the fraction of a clock cycle represented by the shift to be applied to the write DQ MUXes in units of 256ths. In other words, the full-cycle tap value from the master DLL will be scaled by this number over 256 to get the delay value for the slave delay line. This is only used when REG_PHY_USE_WR_LEVEL = 0.

Table 1-158. PHY_WR_DATA_SLAVE_RATIO_4_CR

Bit Number	Name	Reset Value	Description
[31:2]	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
[1:0]	REG_PHY_WR_DATA_SLAVE_RATIO	0×0	[49:48] bits of REG_PHY_WR_DATA_SLAVE_RATIO Ratio value for write data slave DLL. This is the fraction of a clock cycle represented by the shift to be applied to the write DQ MUXes in units of 256ths. In other words, the full-cycle tap value from the master DLL will be scaled by this number over 256 to get the delay value for the slave delay line. This is only used when REG_PHY_USE_WR_LEVEL = 0.

Table 1-159. PHY_WRLVL_INIT_MODE_CR

Bit Number	Name	Reset Value	Description
[31:1]	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.

...........continued

Bit Number	Name	Reset Value	Description
0	REG_PHY_WRLVL_INIT_MODE	0×0	The user programmable init ratio selection mode. 1: Selects a starting ratio value based on REG_PHY_WRLVL_INIT_RATIO PORT 0: Selects a starting ratio value based on write leveling of previous data slice

Table 1-160. PHY_WRLVL_INIT_RATIO_1_CR

Bit Number	Name	Reset Value	Description
[31:16]	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
[15:0]	REG_PHY_WRLVL_INIT_MODE	0×0	[15:0] bits of REG_PHY_WRLVL_INIT_MODE The user programmable initialization ratio used by the write leveling FSM when the REG_PHY_WRLVL_INIT_MODE port is set to 1. The recommended setting of REG_PHY_WRLVL_INIT_RATIO is a half cycle less than the total skew between CLK and DQS at the DRAM.

Table 1-161. PHY_WRLVL_INIT_RATIO_2_CR

Bit Number	Name	Reset Value	Description
[31:16]	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
[15:0]	REG_PHY_WRLVL_INIT_MODE	0×0	[31:16] bits of REG_PHY_WRLVL_INIT_MODE The user programmable initialization ratio used by the write leveling FSM when the REG_PHY_WRLVL_INIT_MODE port is set to 1. The recommended setting of REG_PHY_WRLVL_INIT_RATIO is a half cycle less than the total skew between CLK and DQS at the DRAM.

Table 1-162. PHY_WRLVL_INIT_RATIO_3_CR

Bit Number	Name	Reset Value	Description
[31:16]	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
[15:0]	REG_PHY_WRLVL_INIT_MODE	0×0	[47:32] bits of REG_PHY_WRLVL_INIT_MODE The user programmable initialization ratio used by the write leveling FSM when the REG_PHY_WRLVL_INIT_MODE port is set to 1. The recommended setting of REG_PHY_WRLVL_INIT_RATIO is a half cycle less than the total skew between CLK and DQS at the DRAM.

Table 1-163. PHY_WRLVL_INIT_RATIO_4_CR

Bit Number	Name	Reset Value	Description
[31:2]	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
[1:0]	REG_PHY_WRLVL_INIT_MODE	0×0	[49:48] bits of REG_PHY_WRLVL_INIT_MODE The user programmable init ratio used by the write leveling FSM when the REG_PHY_WRLVL_INIT_MODE PORT is set to 1. The recommended setting of REG_PHY_WRLVL_INIT_RATIO is a half cycle less than the total skew between CLK and DQS at the DRAM.

Table 1-164. PHY_WR_RD_RL_CR

Bit Number	Name	Reset Value	Description
[31:10]	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
[9:5]	REG_PHY_WR_RL_DELAY	0×0	This delay determines when to select the active rank's ratio logic delay for write data and write DQS slave delay lines after PHY receives a write command at the control interface. This is only used for multi-rank designs when REG_PHY_USE_RANK0_DELAYS = 0. This must be programmed as (Write Latency – 4) with a minimum value of 1.
[4:0]	REG_PHY_RD_RL_DELAY	0×0	This delay determines when to select the active rank's ratio logic delay for FIFO_WE and read DQS slave delay lines after PHY receives a read command at the control interface. This is only used for multi-rank designs when REG_PHY_USE_RANK0_DELAYS = 0.

Table 1-165. PHY_DYN_RDC_FIFO_RST_ERR_CNT_CLR_CR

Bit Number	Name	Reset Value	Description
[31:1]	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
0	REG_PHY_RDC_FIFO_RST_ERR_CNT_CLR	0×0	Clear/reset for counter RDC_FIFO_RST_ERR_CNT. • 0: No clear • 1: Clear

Table 1-166. PHY_RDC_WE_TO_RE_DELAY_CR

Bit Number	Name	Reset Value	Description
[31:4]	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.

...........continued

Bit Number	Name	Reset Value	Description
[3:0]	REG_PHY_RDC_WE_TO_RE_DELAY	0×0	Register input: specified in number of clock cycles. This is valid only if USE_FIXED_RE is High. As read capture FIFO depth is limited to 8 entries only, the recommended value for this port is less than 8, even though a higher number may work in some cases, depending upon memory system design.

Table 1-167. PHY_USE_FIXED_RE_CR

Bit Number	Name	Reset Value	Description
[31:1]	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
0	REG_PHY_USE_FIXED_RE	0×0	• 1: PHY generates FIFO read enable after fixed number of clock cycles as defined by REG_PHY_RDC_WE_TO_RE_DELAY[3:0] • 0: PHY uses the NOT_EMPTY method to do the read enable generation **Note:** This port must be set High during the training/leveling process—when DDRC_DFI_WRLVL_EN / DDRC_DFI_RDLVL_EN / DDRC_DFI_RDLVL_GATE_EN PORT is set High.

Table 1-168. PHY_USE_RANK0_DELAYS_CR

Bit Number	Name	Reset Value	Description
[31:1]	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
0	REG_PHY_USE_RANK0_DELAYS	0×0	Delay selection. This applies to multi-rank designs only. 1: Rank 0 delays are used for all ranks 0: Each rank uses its own delay This port must be set High when write latency < 5.

Table 1-169. PHY_USE_LVL_TRNG_LEVEL_CTRL_CR

Bit Number	Name	Reset Value	Description
[31:3]	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
2	REG_PHY_USE_WR_LEVEL	0×0	Write leveling training control. 0: Use register programmed ratio values 1: Use ratio for delay line calculated by write leveling **Note:** This port must be set to 0 when PHY is not working in DDR3 mode.

...........continued

Bit Number	Name	Reset Value	Description
1	REG_PHY_USE_RD_DQS_GATE_LEVEL	0×0	Read DQS gate training control. 0: Use register programmed ratio values 1: Use ratio for delay line calculated by DQS gate leveling This can be used in DDR2 mode also. **Note:** This port must be set to 0 when PHY is not working in DDR2/DDR3 mode
0	REG_PHY_USE_RD_DATA_EYE_LEVEL	0×0	Read data eye training control. 0: Use register programmed ratio values 1: Use ratio for delay line calculated by data eye leveling **Note:** This port must be set to 0 when PHY is not working in DDR3 mode

Table 1-170. PHY_DYN_CONFIG_CR

Bit Number	Name	Reset Value	Description
[31:5]	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
4	REG_PHY_DIS_PHY_CTRL_RSTN	0×0	Disable the PHY control macro reset. • 1: PHY control macro does not get reset • 0: PHY control macro gets reset (default)
3	REG_PHY_LPDDR1	0×0	If the PHY is operating in LPDDR1 mode
2	REG_PHY_BL2	0×0	Burst length control. • 1: Burst length 2 • 0: Other burst length
1	REG_PHY_CLK_STALL_LEVEL	0×0	This port determines whether the delay line clock stalls at High or Low level. The expected input is a very slow clock to avoid asymmetric aging in delay lines. This port is implementation specific and may not be available in all PHYs.
0	REG_PHY_CMD_LATENCY	0×0	Extra command latency. • 0: Default • 1: Command bus has 1 extra cycle of latency

Table 1-171. PHY_RD_WR_GATE_LVL_CR

Bit Number	Name	Reset Value	Description
[31:15]	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
[14:10]	REG_PHY_GATELVL_INC_MODE	0×0	Incremental read DQS gate training mode. One bit for each data slice. • 1: Incremental read gate training • 0: Normal read gate training

...........continued

Bit Number	Name	Reset Value	Description
[9:5]	REG_PHY_WRLVL_INC_MODE	0×0	Incremental write leveling mode. One bit for each data slice. • 1: Incremental write leveling • 0: Normal write leveling
[4:0]	REG_PHY_RDLVL_INC_MODE	0×0	Incremental read data eye training mode. One bit for each data slice. 1: Incremental read data eye training

Table 1-172. PHY_DYN_RESET_CR

Bit Number	Name	Reset Value	Description
[31:1]	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
0	PHY_RESET	0×0	A 1 in this register will bring the PHY out of reset. This is dynamic and synchronized internally before giving to PHY.

Table 1-173. PHY_LEVELLING_FAILURE_SR

Bit Number	Name	Reset Value	Description
[31:15]	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
[14:10]	PHY_REG_RDLVL_INC_FAIL	0×0	Incremental read leveling fail status flag for each PHY data slice. • 1: Incremental read leveling test has failed • 0: Incremental read leveling test has passed
[9:5]	PHY_REG_WRLVL_INC_FAIL	0×0	Incremental write leveling fail status flag for each PHY data slice. • 1: Incremental write leveling test has failed • 0: Incremental write leveling test has passed
[4:0]	PHY_REG_GATELVL_INC_FAIL	0×0	Incremental gate leveling fail status flag for each PHY data slice. • 1: Incremental gate leveling test has failed • 0: Incremental gate leveling test has passed

Table 1-174. PHY_BIST_ERROR_1_SR

Bit Number	Name	Reset Value	Description
[31:16]	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
[15:0]	PHY_REG_BIST_ERR	0×0	[15:0] bits of PHY_REG_BIST_ERR Mismatch error flag from the BIST checker. • 1: Pattern mismatch error • 0: All patterns matched. This is a sticky flag. In order to clear this bit, the REG_PHY_BIST_ERR_CLR must be set High. The bits [8:0] are used for Slice 0, bits [17:9] for slice 1, and so on. The MSB in each slice is used for Mask Bit and lower bits are for DQ bits.

Table 1-175. PHY_BIST_ERROR_2_SR

Bit Number	Name	Reset Value	Description
[31:16]	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
[15:0]	PHY_REG_BIST_ERR	0×0	[31:16] bits of PHY_REG_BIST_ERR Mismatch error flag from the BIST checker. • 1: Pattern mismatch error • 0: All patterns matched. This is a sticky flag. In order to clear this bit, the REG_PHY_BIST_ERR_CLR port must be set High. The bits [8:0] are used for Slice 0, bits [17:9] for slice 1, and so on. The MSB in each slice is used for Mask Bit and lower bits are for DQ bits.

Table 1-176. PHY_BIST_ERROR_3_SR

Bit Number	Name	Reset Value	Description
[31:13]	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
[12:0]	PHY_REG_BIST_ERR	0×0	[44:32] bits of PHY_REG_BIST_ERR Mismatch error flag from the BIST checker. • 1: Pattern mismatch error • 0: All patterns matched. This is a sticky flag. In order to clear this bit, the REG_PHY_BIST_ERR_CLR port must be set High. The bits [8:0] are used for Slice 0, bits [17:9] for slice 1, and so on. The MSB in each slice is used for Mask Bit and lower bits are for DQ bits.

Table 1-177. PHY_WRLVL_DQS_RATIO_1_SR

Bit Number	Name	Reset Value	Description
[31:16]	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
[15:0]	PHY_REG_WRLVL_DQS_RATIO	0×0	[15:0] bits of PHY_REG_WRLVL_DQS_RATIO Ratio value generated by the write leveling FSM for write DQS.

Table 1-178. PHY_WRLVL_DQS_RATIO_2_SR

Bit Number	Name	Reset Value	Description
[31:16]	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
[15:0]	PHY_REG_WRLVL_DQS_RATIO	0×0	[31:16] bits of PHY_REG_WRLVL_DQS_RATIO Ratio value generated by the write leveling FSM for write DQS.

Table 1-179. PHY_WRLVL_DQS_RATIO_3_SR

Bit Number	Name	Reset Value	Description
[31:16]	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
[15:0]	PHY_REG_WRLVL_DQS_RATIO	0×0	[47:32] bits of PHY_REG_WRLVL_DQS_RATIO Ratio value generated by the write leveling FSM for write DQS.

Table 1-180. PHY_WRLVL_DQS_RATIO_4_SR

Bit Number	Name	Reset Value	Description
[31:2]	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
[1:0]	PHY_REG_WRLVL_DQS_RATIO	0×0	[49:48] bits of PHY_REG_WRLVL_DQS_RATIO Ratio value generated by the write leveling FSM for write DQS.

Table 1-181. PHY_WRLVL_DQ_RATIO_1_SR

Bit Number	Name	Reset Value	Description
[31:16]	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
[15:0]	PHY_REG_WRLVL_DQ_RATIO	0×0	[15:0] bits of PHY_REG_WRLVL_DQ_RATIO Ratio value generated by the write leveling FSM for write data.

Table 1-182. PHY_WRLVL_DQ_RATIO_2_SR

Bit Number	Name	Reset Value	Description
[31:16]	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
[15:0]	PHY_REG_WRLVL_DQ_RATIO	0×0	[31:16] bits of PHY_REG_WRLVL_DQ_RATIO Ratio value generated by the write leveling FSM for write data.

Table 1-183. PHY_WRLVL_DQ_RATIO_3_SR

Bit Number	Name	Reset Value	Description
[31:16]	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
[15:0]	PHY_REG_WRLVL_DQ_RATIO	0×0	[47:32] bits of PHY_REG_WRLVL_DQ_RATIO Ratio value generated by the write leveling FSM for write data.

Table 1-184. PHY_WRLVL_DQ_RATIO_4_SR

Bit Number	Name	Reset Value	Description
[31:2]	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.

...........continued

Bit Number	Name	Reset Value	Description
[1:0]	PHY_REG_WRLVL_DQ_RATIO	0×0	[49:48] bits of PHY_REG_WRLVL_DQ_RATIO Ratio value generated by the write leveling FSM for write data.

Table 1-185. PHY_RDLVL_DQS_RATIO_1_SR

Bit Number	Name	Reset Value	Description
[31:16]	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
[15:0]	PHY_REG_RDLVL_DQS_RATIO	0×0	[15:0] bits of PHY_REG_RDLVL_DQS_RATIO Ratio value generated by read data eye training FSM.

Table 1-186. PHY_RDLVL_DQS_RATIO_2_SR

Bit Number	Name	Reset Value	Description
[31:16]	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
[15:0]	PHY_REG_RDLVL_DQS_RATIO	0×0	[31:16] bits of PHY_REG_RDLVL_DQS_RATIO Ratio value generated by read data eye training FSM.

Table 1-187. PHY_RDLVL_DQS_RATIO_3_SR

Bit Number	Name	Reset Value	Description
[31:16]	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
[15:0]	PHY_REG_RDLVL_DQS_RATIO	0×0	[47:32] bits of PHY_REG_RDLVL_DQS_RATIO Ratio value generated by read data eye training FSM.

Table 1-188. PHY_RDLVL_DQS_RATIO_4_SR

Bit Number	Name	Reset Value	Description
[31:2]	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
[1:0]	PHY_REG_RDLVL_DQS_RATIO	0×0	[49:48] bits of PHY_REG_RDLVL_DQS_RATIO Ratio value generated by read data eye training FSM.

Table 1-189. PHY_FIFO_1_SR

Bit Number	Name	Reset Value	Description
[31:16]	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
[15:0]	PHY_REG_RDLVL_FIFOWEIN_RATIO	0×0	[15:0] bits of PHY_REG_RDLVL_FIFOWEIN_RATIO Ratio value generated by read gate training FSM.

Table 1-190. PHY_FIFO_2_SR

Bit Number	Name	Reset Value	Description
[31:16]	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
[15:0]	PHY_REG_RDLVL_FIFOWEIN_RATIO	0×0	[31:16] bits of PHY_REG_RDLVL_FIFOWEIN_RATIO Ratio value generated by read gate training FSM.

Table 1-191. PHY_FIFO_3_SR

Bit Number	Name	Reset Value	Description
[31:16]	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
[15:0]	PHY_RDLVL_FIFOWEIN_RATIO	0×0	[47:32] bits of PHY_REG_RDLVL_FIFOWEIN_RATIO Ratio value generated by read gate training FSM.

Table 1-192. PHY_FIFO_4_SR

Bit Number	Name	Reset Value	Description
[31:11]	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
[10:7]	REG_PHY_RDC_FIFO_RST_ERR_CNT	0×0	Counter for counting how many times the pointers of read capture FIFO differs when they are reset by DLL_CALIB.
[6:0]	PHY_REG_RDLVL_FIFOWEIN_RATIO	0×0	[54:48] bits of PHY_REG_RDLVL_FIFOWEIN_RATIO Ratio value generated by read gate training FSM.

Table 1-193. PHY_MASTER_DLL_SR

Bit Number	Name	Reset Value	Description
[31:9]	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
[8:3]	PHY_REG_STATUS_OF_IN_LOCK_STATE	0×0	Lock status from the output filter module inside the master DLL. (2 bits per MDLL). PHY has three MDLLs. Bit[0] – Fine delay line lock status. • 1: Locked • 0: Unlocked Bit[1] – Coarse delay line lock status. • 1: Locked • 0: Unlocked
[2:0]	PHY_REG_STATUS_DLL_LOCK	0×0	Status signal: • 1: Master DLL is locked • 0: Master DLL is not locked Three bits correspond to three MDLLs.

Table 1-194. PHY_DLL_SLAVE_VALUE_1_SR

Bit Number	Name	Reset Value	Description
[31:16]	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
[15:0]	PHY_REG_STATUS_DLL_SLAVE_VALUE	0×0	[15:0] bits of PHY_REG_STATUS_DLL_SLAVE_VALUE Shows the current coarse and fine delay values measured for a full-cycle shift by each master DLL. This is a 27 bit register, 9 bits for each DLL. [1:0]: Fine value [8:2]: Coarse value

Table 1-195. PHY_DLL_SLAVE_VALUE_2_SR

Bit Number	Name	Reset Value	Description
[31:11]	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
[10:0]	PHY_REG_STATUS_DLL_SLAVE_VALUE	0×0	[26:16] bits of PHY_REG_STATUS_DLL_SLAVE_VALUE Shows the current coarse and fine delay values measured for a full-cycle shift by each master DLL. This is a 27 bit register, 9 bits for each DLL. [1:0]: Fine value [8:2]: Coarse value

Table 1-196. PHY_STATUS_OF_IN_DELAY_VAL_1_SR

Bit Number	Name	Reset Value	Description
[31:16]	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
[15:0]	PHY_REG_STATUS_OF_IN_DELAY_VALUE	0×0	[15:0] bits of PHY_REG_STATUS_OF_IN_DELAY_VALUE The coarse and fine values going into the output filter in the master DLL. This is a 27 bit register, 9 bits for each DLL. {coarse[6:0],fine[1:0]}

Table 1-197. PHY_STATUS_OF_IN_DELAY_VAL_2_SR

Bit Number	Name	Reset Value	Description
[31:11]	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
[10:0]	PHY_REG_STATUS_OF_IN_DELAY_VALUE	0×0	[26:16] bits of PHY_REG_STATUS_OF_IN_DELAY_VALUE The coarse and fine values going into the output filter in the master DLL. This is a 27-bit register, 9 bits for each DLL. {coarse[6:0],fine[1:0]}

Table 1-198. PHY_STATUS_OF_OUT_DELAY_VAL_1_SR

Bit Number	Name	Reset Value	Description
[31:16]	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
[15:0]	PHY_REG_STATUS_OF_OUT_DELAY_VALUE	0×0	[15:0] bits of PHY_REG_STATUS_OF_OUT_DELAY_VALUE The coarse and fine values coming out of the output filter in the master DLL. This is a 27 bit register, 9 bits for each DLL. {coarse[6:0],fine[1:0]}

Table 1-199. PHY_STATUS_OF_OUT_DELAY_VAL_2_SR

Bit Number	Name	Reset Value	Description
[31:11]	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
[10:0]	PHY_REG_STATUS_OF_OUT_DELAY_VALUE	0×0	[26:16] bits of PHY_REG_STATUS_OF_OUT_DELAY_VALUE The coarse and fine values coming out of the output filter in the master DLL. This is a 27 bit register, 9 bits for each DLL. {coarse[6:0],fine[1:0]}

Table 1-200. PHY_DLL_LOCK_AND_SLAVE_VAL_SR

Bit Number	Name	Reset Value	Description
[31:10]	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
9	PHY_REG_STATUS_PHY_CTRL_DLL_LOCK	0×0	PHY_CTRL Master DLL Status bits. 1: Master DLL is locked 0: Master DLL is not locked
[8:0]	PHY_REG_STATUS_PHY_CTRL_DLL_SLAVE_VALUE	0×0	Shows the current coarse and fine delay value going to the PHY_CTRL slave DLL. [1:0]: Fine value [8:2]: Coarse value

Table 1-201. PHY_CTRL_OUTPUT_FILTER_SR

Bit Number	Name	Reset Value	Description
[31:11]	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.

...........continued

Bit Number	Name	Reset Value	Description
[10:9]	PHY_REG_STATUS_PHY_CTRL_OF_IN_LOCK_STATE	0×0	Lock status from the output filter module inside the PHY_CTRL Master DLL. Bit[9] – Fine delay line lock status. 1: Locked 0: Unlocked Bit[10] – Coarse delay line lock status. 1: Locked 0: Unlocked
[8:0]	PHY_REG_STATUS_PHY_CTRL_OF_IN_DELAY_VALUE	0×0	The coarse and fine values going into the output filter in the PHY_CTRL master DLL. [1:0]: Fine value [8:2]: Coarse value

Table 1-202. PHY_RD_DQS_SLAVE_DLL_VAL_1_SR

Bit Number	Name	Reset Value	Description
[31:16]	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
[15:0]	PHY_STATUS_RD_DQS_SLAVE_DLL_VALUE	0×0	[15:0] bits of PHY_STATUS_RD_DQS_SLAVE_DLL_VALUE Delay value applied to read DQS slave DLL.

Table 1-203. PHY_RD_DQS_SLAVE_DLL_VAL_2_SR

Bit Number	Name	Reset Value	Description
[31:16]	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
[15:0]	PHY_REG_STATUS_RD_DQS_SLAVE_DLL_VALUE	0×0	[31:16] bits of PHY_STATUS_RD_DQS_SLAVE_DLL_VALUE Delay value applied to read DQS slave DLL.

Table 1-204. PHY_RD_DQS_SLAVE_DLL_VAL_3_SR

Bit Number	Name	Reset Value	Description
[31:13]	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.

...........continued

Bit Number	Name	Reset Value	Description
[12:0]	PHY_REG_STATUS_RD_DQS_SLAVE_DLL_VALUE	0×0	[44:32] bits of PHY_STATUS_RD_DQS_SLAVE_DLL_VALUE Delay value applied to read DQS slave DLL.

Table 1-205. PHY_WR_DATA_SLAVE_DLL_VAL_1_SR

Bit Number	Name	Reset Value	Description
[31:16]	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
[15:0]	PHY_REG_STATUS_WR_DATA_SLAVE_DLL_VALUE	0×0	[15:0] bits of PHY_REG_STATUS_WR_DATA_SLAVE_DLL_VALUE Delay value applied to write data slave DLL.

Table 1-206. PHY_WR_DATA_SLAVE_DLL_VAL_2_SR

Bit Number	Name	Reset Value	Description
[31:16]	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
[15:0]	PHY_REG_STATUS_WR_DATA_SLAVE_DLL_VALUE	0×0	[31:16] bits of PHY_REG_STATUS_WR_DATA_SLAVE_DLL_VALUE Delay value applied to write data slave DLL.

Table 1-207. PHY_WR_DATA_SLAVE_DLL_VAL_3_SR

Bit Number	Name	Reset Value	Description
[31:13]	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
[12:0]	PHY_REG_STATUS_WR_DATA_SLAVE_DLL_VALUE	0×0	[44:32] bits of PHY_REG_STATUS_WR_DATA_SLAVE_DLL_VALUE Delay value applied to write data slave DLL.

Table 1-208. PHY_FIFO_WE_SLAVE_DLL_VAL_1_SR

Bit Number	Name	Reset Value	Description
[31:16]	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.

...........continued

Bit Number	Name	Reset Value	Description
[15:0]	PHY_REG_STATUS_FIFO_WE_SLAVE_DLL_VALUE	0×0	[15:0] bits of PHY_REG_STATUS_FIFO_WE_SLAVE_DLL_VALUE Delay value applied to FIFO WE slave DLL.

Table 1-209. PHY_FIFO_WE_SLAVE_DLL_VAL_2_SR

Bit Number	Name	Reset Value	Description
[31:16]	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
[15:0]	PHY_REG_STATUS_FIFO_WE_SLAVE_DLL_VALUE	0×0	[31:16] bits of PHY_REG_STATUS_FIFO_WE_SLAVE_DLL_VALUE Delay value applied to FIFO WE slave DLL.

Table 1-210. PHY_FIFO_WE_SLAVE_DLL_VAL_3_SR

Bit Number	Name	Reset Value	Description
[31:13]	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
[12:0]	PHY_REG_STATUS_FIFO_WE_SLAVE_DLL_VALUE	0×0	[44:32] bits of PHY_REG_STATUS_FIFO_WE_SLAVE_DLL_VALUE Delay value applied to FIFO WE slave DLL.

Table 1-211. PHY_WR_DQS_SLAVE_DLL_VAL_1_SR

Bit Number	Name	Reset Value	Description
[31:16]	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
[15:0]	PHY_REG_STATUS_WR_DQS_SLAVE_DLL_VALUE	0×0	[15:0] bits of PHY_REG_STATUS_WR_DQS_SLAVE_DLL_VALUE Delay value applied to write DQS slave DLL.

Table 1-212. PHY_WR_DQS_SLAVE_DLL_VAL_2_SR

Bit Number	Name	Reset Value	Description
[31:16]	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.

...........continued

Bit Number	Name	Reset Value	Description
[15:0]	PHY_REG_STATUS_WR_DQS_SLAVE_DLL_VALUE	0×0	[31:16] bits of PHY_REG_STATUS_WR_DQS_SLAVE_DLL_VALUE Delay value applied to write DQS slave DLL.

Table 1-213. PHY_WR_DQS_SLAVE_DLL_VAL_3_SR

Bit Number	Name	Reset Value	Description
[31:13]	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
[12:0]	PHY_REG_STATUS_WR_DQS_SLAVE_DLL_VALUE	0×0	[44:32] bits of PHY_REG_STATUS_WR_DQS_SLAVE_DLL_VALUE Delay value applied to write DQS slave DLL.

Table 1-214. PHY_CTRL_SLAVE_DLL_VAL_SR

Bit Number	Name	Reset Value	Description
[31:9]	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
[8:0]	PHY_REG_STATUS_PHY_CTRL_SLAVE_DLL_VALUE	0×0	Delay value applied to write DQS slave DLL.

1.11.6 DDR_FIC Configuration Registers Summary (Ask a Question)

Table 1-215. DDR_FIC Configuration Register Summary

Register Name	Address Offset	R/W	Reset Source	Description
DDR_FIC_NB_ADDR_CR	0×400	RW	PRESET_N	Indicates the base address of the non-bufferable address region.
DDR_FIC_NBRWB_SIZE_CR	0×404	RW	PRESET_N	Indicates the size of the non-bufferable address region.
DDR_FIC_BUF_TIMER_CR	0×408	RW	PRESET_N	10-bit timer interface used to configure the timeout register.
DDR_FIC_HPD_SW_RW_EN_CR	0×40C	RW	PRESET_N	Enable write buffer and read buffer register for AHBL master1 and master2.
DDR_FIC_HPD_SW_RW_INVAL_CR	0×410	RW	PRESET_N	Invalidates write buffer and read buffer for AHBL master1 and master2.
DDR_FIC_SW_WR_ERCLR_CR	0×414	RW	PRESET_N	Clear bit for error status by AHBL master1 and master2 write buffer.
DDR_FIC_ERR_INT_ENABLE	0×418	RW	PRESET_N	Used for Interrupt generation.
DDR_FIC_NUM_AHB_MASTERS_CR	0×41C	RW	PRESET_N	Defines whether one or two AHBL 32-bit masters are implemented in fabric.
DDR_FIC_HPB_ERR_ADDR_1_SR	0×420	RO	PRESET_N	Tag of write buffer for which error response is received is placed in this register.

MICROCHIP

..........continued

Register Name	Address Offset	R/W	Reset Source	Description
DDR_FIC_HPB_ERR_ADDR_2_SR	0×424	RO	PRESET_N	Tag of write buffer for which error response is received is placed in this register.
DDR_FIC_SW_ERR_ADDR_1_SR	0×428	RO	PRESET_N	Tag of write buffer for which error response is received is placed in this register.
DDR_FIC_SW_ERR_ADDR_2_SR	0×42C	RO	PRESET_N	Tag of write buffer for which error response is received is placed in this register.
DDR_FIC_HPD_SW_WRB_EMPTY_SR	0×430	RO	PRESET_N	Indicates valid data in read and write buffer for AHBL master1 and master2.
DDR_FIC_SW_HPB_LOCKOUT_SR	0×434	RO	PRESET_N	Write and read buffer status register for AHBL master1 and master2.
DDR_FIC_SW_HPD_WERR_SR	0×438	RO	PRESET_N	Error response register for bufferable write request
DDR_FIC_LOCK_TIMEOUTVAL_1_CR	0×440	RW	PRESET_N	Indicates maximum number of cycles a master can hold the bus for locked transfer.
DDR_FIC_LOCK_TIMEOUTVAL_2_CR	0×444	RW	PRESET_N	Indicates maximum number of cycles a master can hold the bus for locked transfer.
DDR_FIC_LOCK_TIMEOUT_EN_CR	0×448	RW	PRESET_N	Lock timeout feature enable register
DDR_FIC_RDWR_ERR_SR	0×460	RO	PRESET_N	Indicates read address of math error register.

1.11.7 DDR_FIC Configuration Register Bit Definitions (Ask a Question)

Table 1-216. DDR_FIC_NB_ADDR_CR

Bit Number	Name	Reset Value	Description
[31:16]	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
[15:0]	DDR_FIC_NB_ADD	0×0	This indicates the base address of the non-bufferable address region.

Table 1-217. DDR_FIC_NBRWB_SIZE_CR

Bit Number	Name	Reset Value	Description
[31:9]	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
8	DDR_FIC_WCB_SZ	0×0	Configures write buffer and read buffer size as per DDR burst size. This port is common for all buffers. Buffers can be configured to 16 byte or 32 byte size. • 0: Buffer size is configured to 16 bytes • 1: Buffer size is configured to 32 bytes
[7:4]	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.

...........continued

Bit Number	Name	Reset Value	Description
[3:0]	DDR_FIC_NUBF_SZ	0×0	This signal indicates the size of the non-bufferable address region. The region sizes are as follows: • 0000: None (default) • 0001: 64 KB bufferable region • 0010: 128 KB bufferable region • 0011: 256 KB bufferable region • 0100: 512 KB bufferable region • 0101: 1 MB bufferable region • 0110: 2 MB bufferable region • 0111: 4 MB bufferable region • 1000: 8 MB bufferable region • 1001: 16 MB bufferable region • 1010: 32 MB bufferable region • 1011: 64 MB bufferable region • 1100: 128 MB bufferable region • 1101: 256 MB bufferable region • 1110: 512 MB bufferable region • 1111: 1 GB bufferable region

Table 1-218. DDR_FIC_BUF_TIMER_CR

Bit Number	Name	Reset Value	Description
[31:10]	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
[9:0]	DDR_FIC_TIMER	0×0	10-bit timer interface used to configure timeout register. Once timer reaches the timeout value, a flush request is generated by the flush controller in the DDR_FIC. This port is common for all buffers.

Table 1-219. DDR_FIC_HPD_SW_RW_EN_CR

Bit Number	Name	Reset Value	Description
[31:7]	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
6	DDR_FIC_M1_REN	0×0	• 1: Enable read buffer for AHBL master1 • 0: Disable read buffer for AHBL master1
5	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
4	DDR_FIC_M1_WEN	0×0	• 1: Enable write buffer for AHBL master1 • 0: Disable write buffer for AHBL master1
3	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
2	DDR_FIC_M2_REN	0×0	• 1: Enable read buffer for AHBL master2 • 0: Disable read buffer for AHBL master2

..........continued

Bit Number	Name	Reset Value	Description
1	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
0	DDR_FIC_M2_WEN	0×0	• 1: Enable write buffer for AHBL master2 • 0: Disable write buffer for AHBL master2

Table 1-220. DDR_FIC_HPD_SW_RW_INVAL_CR

Bit Number	Name	Reset Value	Description
[31:7]	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
6	DDR_FIC_flshM1	0×0	• 1: Flush read buffer for AHBL master1 • 0: Default
5	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
4	DDR_FIC_invalid_M1	0×0	• 1: Invalidate write buffer for AHBL master1 • 0: Default
3	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
2	DDR_FIC_flshM2	0×0	• 1: Flush write buffer for AHBL master2 • 0: Default
1	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
0	DDR_FIC_invalid_M2	0×0	• 1: Invalidate read buffer for AHBL master2 • 0: Default

Table 1-221. DDR_FIC_SW_WR_ERCLR_CR

Bit Number	Name	Reset Value	Description
[31:9]	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
8	DDR_FIC_LTO_CLR	0×0	Clear signal to lock timeout interrupt.
[7:5]	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
4	DDR_FIC_M2_WR_ERCLR	0×0	Clear bit for error status of AHBL master2 write buffer. Once it goes High, error status is cleared.
[3:1]	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
0	DDR_FIC_M1_WR_ERCLR	0×0	Clear bit for error status posted by AHBL master1 write buffer. Once it goes High, error status is cleared.

Table 1-222. DDR_FIC_ERR_INT_ENABLE

Bit Number	Name	Reset Value	Description
[31:2]	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
1	SYR_SW_WR_ERR	0×0	Status bit. • Goes High when error response is received for bufferable write request. • Goes Low when processor serves interrupt and makes clear bit for AHBL master1.
0	SYR_HPD_WR_ERR	0×0	Status bit. • Goes Low when processor serves the interrupt • Goes High when error response is received for bufferable write request

Table 1-223. DDR_FIC_NUM_AHB_MASTERS_CR

Bit Number	Name	Reset Value	Description
[31:5]	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
4	CFG_NUM_AHB_MASTERS	0×0	Defines whether one or two AHBL 32-bit masters are implemented in the fabric. • 0: One 32-bit AHB master implemented in fabric • 1: Two 32-bit AHB masters implemented in fabric
[3:0]	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.

Table 1-224. DDR_FIC_HPB_ERR_ADDR_1_SR

Bit Number	Name	Reset Value	Description
[31:16]	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
[15:0]	DDR_FIC_M1_ERR_ADD	0×0	32 bits are split into two registers. [15:0] bits of DDR_FIC_M1_ERR_ADD Tag of write buffer for which error response is received is placed in this register. The following values are updated in this register as per buffer size: Buffer size 16 bytes: 28 bit TAG value is loaded to [31:4] and 0000 to [3:0] 32 bytes: upper 27 bits of TAG is loaded to [31:5] and 00000 to [4:0]

Table 1-225. DDR_FIC_HPB_ERR_ADDR_2_SR

Bit Number	Name	Reset Value	Description
[31:16]	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.

...........continued

Bit Number	Name	Reset Value	Description
[15:0]	DDR_FIC_M1_ERR_ADD	0×0	32 bits are split into two registers. [31:16] bits of DDR_FIC_M1_ERR_ADD Tag of write buffer for which error response is received is placed in this register. The following values are updated in this register as per buffer size: Buffer size 16 bytes: 28 bit TAG value is loaded to [31:4] and 0000 to [3:0] 32 bytes: upper 27 bits of TAG is loaded to [31:5] and 00000 to [4:0]

Table 1-226. DDR_FIC_SW_ERR_ADDR_1_SR

Bit Number	Name	Reset Value	Description
[31:16]	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
[15:0]	DDR_FIC_M2_ERR_ADD	0×0	32 bits are split into two registers. Lower 16 bits. Tag of write buffer for which error response is received is placed in this register. The following values are updated in this register as per buffer size: Buffer size: DDR_FIC_M2_ERR_ADD[31:0] 16 bits: TAG, 0000 32 bits: TAG[27:1], 00000

Table 1-227. DDR_FIC_SW_ERR_ADDR_2_SR

Bit Number	Name	Reset Value	Description
[31:16]	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
[15:0]	DDR_FIC_M2_ERR_ADD	0×0	32 bits are split into two registers. [31:16] bits of DDR_FIC_M2_ERR_ADD Tag of write buffer for which error response is received is placed in this register. The following values are updated in this register as per buffer size: Buffer size 16 bytes: 28 bit TAG value is loaded to [31:4] and 0000 to [3:0] 32 bytes: upper 27 bits of TAG is loaded to [31:5] and 00000 to [4:0]

Table 1-228. DDR_FIC_HPD_SW_WRB_EMPTY_SR

Bit Number	Name	Reset Value	Description
[31:7]	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
6	DDR_FIC_M1_RBEMPTY	0×0	1: Read buffer of AHBL master1 does not have valid data.
5	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
4	DDR_FIC_M1_WBEMPTY	0×0	• 0: Default • 1: Write buffer of AHBL master1 does not have valid data

...........continued

Bit Number	Name	Reset Value	Description
3	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
2	DDR_FIC_M2_RBEMPTY	0×0	• 0: Default • 1: Read buffer of AHBL master2 does not have valid data
1	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
0	DDR_FIC_M2_WBEMPTY	0×0	• 1: Write buffer of AHBL master2 does not have valid data • 0: Default

Table 1-229. DDR_FIC_SW_HPB_LOCKOUT_SR

Bit Number	Name	Reset Value	Description
[31:9] [7] [5] [3] [1]	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
8	DDR_FIC_LCKTOUT	0×0	Indicates lock counter in arbiter reached its maximum value. Lock counter (20-bit) starts counting when a locked request gets access to a bus and will be cleared when the lock signal becomes logic 0.
6	DDR_FIC_M2_WDSBL_DN	0×0	High indicates AHBL master2 write buffer is disabled.
4	DDR_FIC_M2_RDSBL_DN	0×0	High indicates AHBL master2 read buffer is disabled.
2	DDR_FIC_M1_WDSBL_DN	0×0	High indicates AHBL master1 read buffer is disabled.
0	DDR_FIC_M1_RDSBL_DN	0×0	High indicates AHBL master1 write buffer is disabled.

Table 1-230. DDR_FIC_SW_HPD_WERR_SR

Bit Number	Name	Reset Value	Description
[31:9]	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
8	DDR_FIC_M1_WR_ERR	0×0	Status bit • Goes High when error response is received for bufferable write request • Goes Low when the processor serves an interrupt and makes a clear bit for AHBL master1
[7:1]	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
0	DDR_FIC_M2_WR_ERR	0×0	Status bit • Goes High when error response is received for bufferable write request • Goes Low when processor serves the interrupt

MICROCHIP

Table 1-231. DDR_FIC_LOCK_TIMEOUTVAL_1_CR

Bit Number	Name	Reset Value	Description
[31:16]	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
[15:0]	CFGR_LOCK_TIMEOUT_REG	0×0	20 bits are split into two registers. [15:0] bits of CFGR_LOCK_TIMEOUT_REG Lock timeout 20-bit register. Indicates maximum number of cycles a master can hold the bus for locked transfer. If master holds the bus for locked transfer more than the required cycles, an interrupt is generated.

Table 1-232. DDR_FIC_LOCK_TIMEOUTVAL_2_CR

Bit Number	Name	Reset Value	Description
[31:4]	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
[3:0]	CFGR_LOCK_TIMEOUT_REG	0×0	20 bits are split into two registers. [19:16] bits of CFGR_LOCK_TIMEOUT_REG Lock timeout 20-bit register. Indicates maximum number of cycles a master can hold the bus for locked transfer. If master holds the bus for locked transfer more than the required cycles, an interrupt is generated.

Table 1-233. DDR_FIC_LOCK_TIMEOUT_EN_CR

Bit Number	Name	Reset Value	Description
[31:1]	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
0	CFGR_LOCK_TIMEOUT_EN	0×0	1: Lock timeout feature is enabled and interrupt is generated 0: Lock timeout feature is disabled and interrupt is not generated

Table 1-234. DDR_FIC_RDWR_ERR_SR

Bit Number	Name	Reset Value	Description
[31:6]	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
[5:0]	DDR_FIC_CFG_RDWR_ERR_SR	0×0	Read address of math error register.

1.12 How to Use the MDDR in SmartFusion 2 (Ask a Question)

This section describes how to use the MDDR subsystem in the design. It contains the following sections:

MICROCHIP

1.12.1 Design Flow Using System Builder (Ask a Question)

This section describes how to use MDDR in the SmartFusion 2 devices using the System Builder graphical design wizard in the Libero Software.

The following image shows the initial System Builder window where you can select the features that you require.

- For information on how to launch and use the System Builder wizard, see SmartFusion2 System Builder User Guide.
- For more information on DDR initialization, see SmartFusion2 DDR Controller and Serial High Speed Controller Initialization Methodology.

Figure 1-46. System Builder - Device Features Window
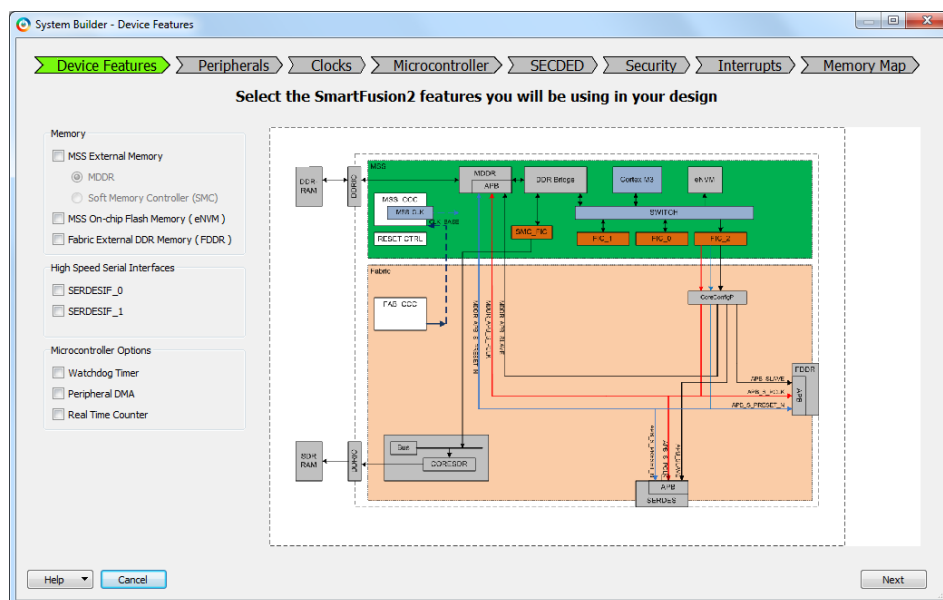
The following steps describe how to configure the MDDR.

1. Check the **MSS External Memory** check box under the **Device Features** tab, select **MDDR** and leave the other check boxes unchecked. The following image shows the **System Builder** > **Device Features**.

Figure 1-47. MSS External DDR Memory Selection
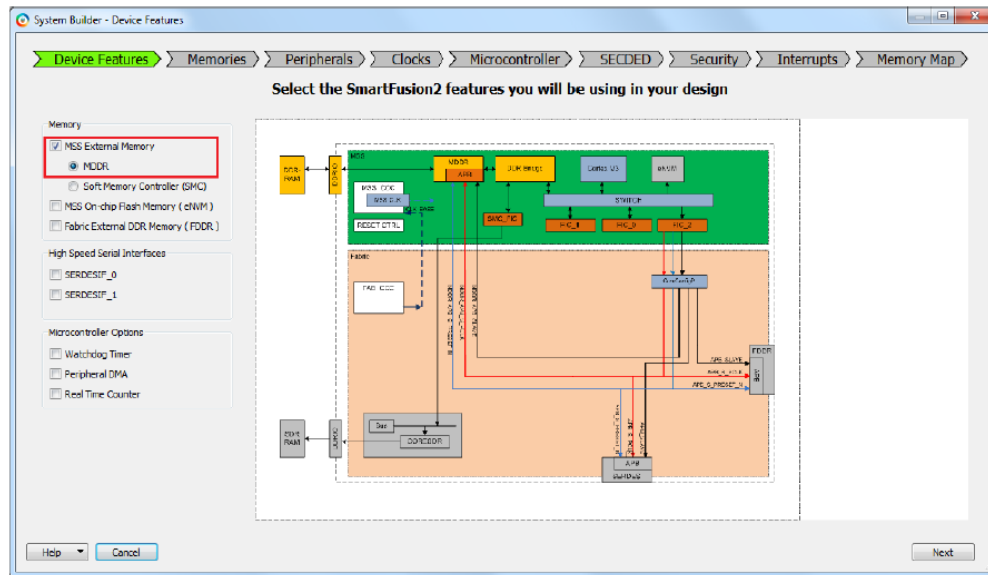
2. Navigate to the **Memories** tab. Depending on the application requirement, select the memory settings under the **General** tab, as shown in the following image.

   – Memory Type can be selected as DDR2, DDR3, or LPDDR.

   – The Data width can be selected as 32-bit, 16-bit, or 8-bit. See Table 1-11 for supported data widths for various SmartFusion 2 device packages.

   – The SECDED (ECC) can be enabled or disabled.

   – Arbitration Scheme can be selected between Type-0 to Type-3. See Table 1-10 for details of arbitration Scheme.

   – The Highest priority ID of fabric master can be entered from 0 to 15, if the Arbitration Scheme selected other than Type-0.

   – Address Mapping - The register settings to perform mapping to system address bits for various Row, Bank, and Column combinations are automatically computed by the configurator using address mapping option. Table 1-19 shows the supported range for Row, Bank, and Column. For more information see 1.5.5.6. Address Mapping.

   – Select the I/O Drive Strength as **Half Drive Strength** or **Full Drive Strength** as shown in Figure 1-48. The DDR I/O standard is configured as listed in Table 1-20 based on this setting.

Table 1-235. DDR I/O Standard Configured Based on I/O Drive Strength Setting

I/O Drive Strength	Memory Type	
	DDR2	DDR3
Half Drive Strength	SSTL18I	SSTL15I
Full Drive Strength	SSTL18II	SSTL15II

Figure 1-48. I/O Drive Strength Setting
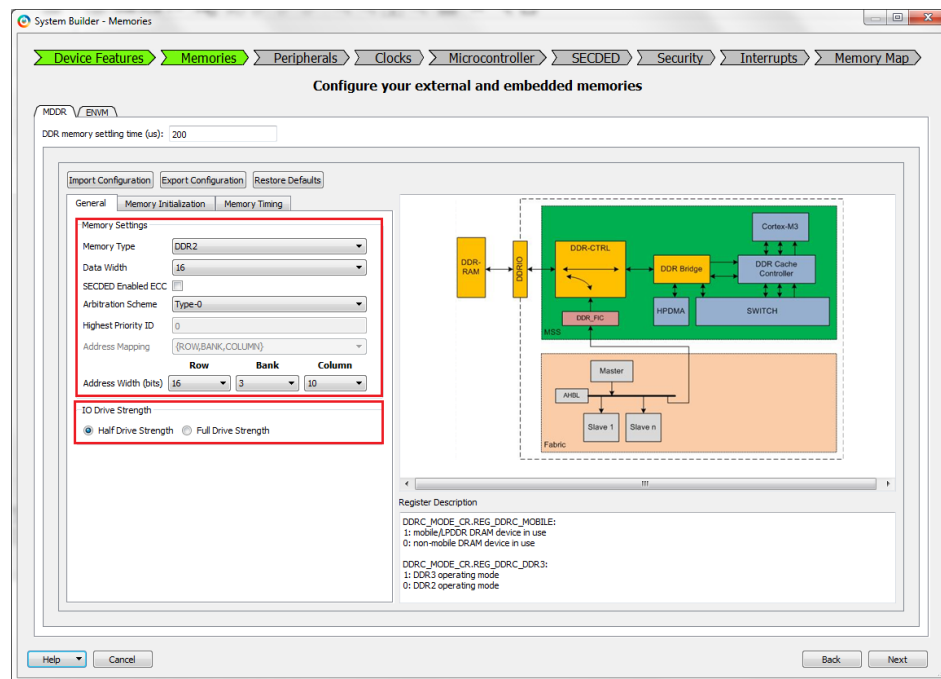
3. For only LPDDR memory, the I/O standard and I/O calibration settings are available as shown in the following image.
   – Select I/O standard as LVCMOS18 or LPDDRI. For Microchip M2S_EVAL_KIT board select LPDDRI(SSTL18) as the board is designed to use LPDDRI I/O standard.

   > **Important:** If LVCMOS18 is selected, all I/Os are configured to LVCMOS1.8 except CLK/CLK_N.CLK and CLK_N are configured to LPDDRI standard because they are differential signals.

   – Select I/O calibration as ON or OFF. If I/O calibration is selected as ON, then the Smartfusion 2 MDDR_IMP_CALIB pin must be pulled down with a resistor. For information on resistor values, see the "Impedance Calibration section" in DS0115: SmartFusion2 Pin Descriptions Datasheet.
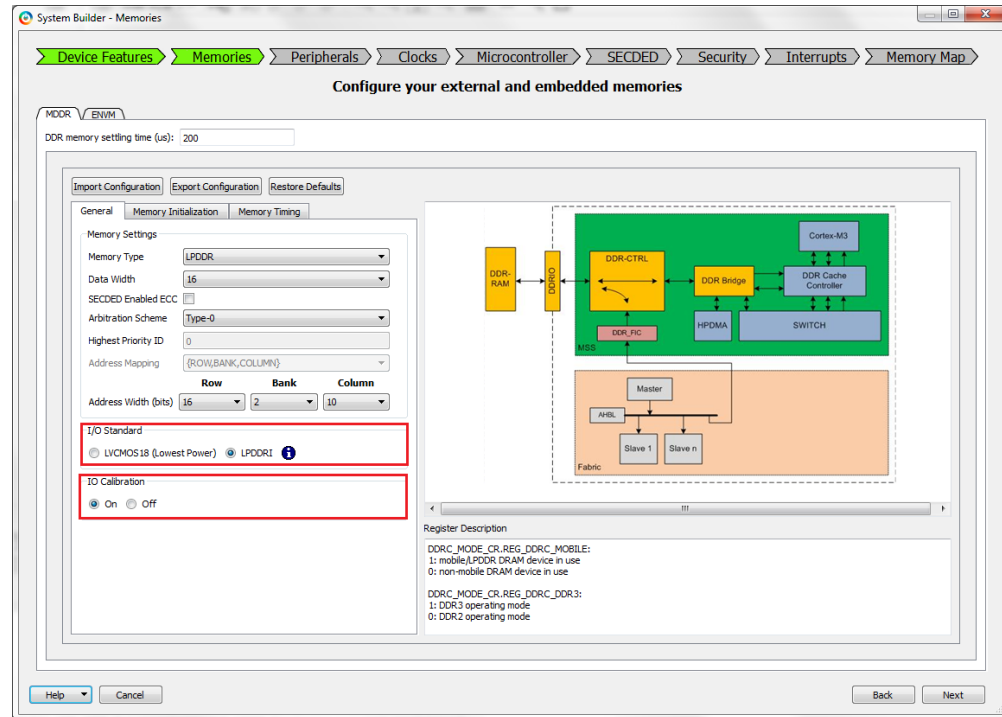
4. Depending on the application requirement, select the Memory Initialization settings under the Memory Initialization tab as shown in Figure 1-50.
    –   Select the below performance related settings:
        •   Burst Length can be selected as 4, 8 or 16. Table 1-11 for supported burst lengths.
        •   Burst order can be selected as sequential or interleaved. See Table 1-13 for supported burst orders.
        •   Timing mode can be selected as 1T or 2T. For more details see 1.5.5.12. 1T or 2T Timing.
        •   CAS latency is the delay, in clock cycles, between the internal READ command and the availability of the first bit of output data. Select the CAS latency according to the DDR memory (Mode register) datasheet.
    –   Select the below power saving mode settings. See 1.5.4.3.5. Power Saving Modes for more details.
        •   Self-Refresh Enabled
        •   Auto Refresh Burst Count
        •   Power down Enabled
        •   Stop the clock: supported only for LPDDR
        •   Deep Power down Enabled: supported only for LPDDR
        •   Power down entry time
    –   Select the additional performance settings for DDR3 memory.
        •   Additive CAS Latency is defined by EMR[5:3] register of DDR2 memory and by MR1[4:3] register of DDR3 memory. It enables the DDR2 or DDR3 SDRAM to allow a READ or WRITE command from DDR Controller after the ACTIVATE command for the same bank prior to tRCD (MIN). This configuration is part of DDR2 Extended Mode register and DDR3 mode register1.

- CAS Write Latency (CWL) is defined by DDR3 MR2[5:3] and is the delay, in clock cycles, from the releasing of the internal write to the latching of the first data in. The overall WRITE latency (WL) is equal to CWL + AL by default CWL is set to 5 clock cycles.

– Select the below ZQ Calibration settings for DDR3 memory. For more details see 1.5.3.3. ZQ Calibration.

- Zqinit

- ZQCS

- ZQCS Interval

– Select Other Settings

- Local ODT setting is not supported for LPDDR memory. For DDR2/DDR3 memory type, user can choose any option for Local ODT. User can enable or disable **Local ODT** during read transaction.

- Drive strength setting is defined by EMR[7:5] register bits of LPDDR memory with drop down options of **Full**, **Half**, **Quarter**, and **One-eighth** drive strength, it is defined by EMR[1] register bit of DDR2 memory with drop down options of **Full** and **Weak** drive strength and it is defined by MR1 register bits M5 and M1 of DDR3 memory with drop down options of **RZQ/6** and **RZQ/7**.

- Partial array self-refresh coverage setting is defined by EMR[2:0] register bits of LPDDR memory with drop down options of **Full**, **Quarter**, **One-eighth**, and **One-sixteenth**. This feature helps in improving power savings during self-refresh by selecting the amount of memory to be refreshed during self-refresh.

- RTT (Nominal) setting is defined by EMR[6] and EMR[2] register bits of DDR2 memory which determines what ODT resistance is enabled with drop down options of `RTT disabled', '50 Ω', '75 Ω' and `150 Ω' and it is defined by MR1[9], MR1[6] and MR1[2] register bits of DDR3 memory. In DDR3 memory RTT nominal termination is allowed during standby conditions and WRITE operations and NOT during READ operations with drop down options of **RZQ/2**, **RZQ/4**, and **RZQ/6**.

- RTT_WR (Dynamic ODT) setting is defined by MR2[10:9] register bits of DDR3 memory. This is applicable only during WRITE operations. If dynamic ODT (Rtt_WR) is enabled, DRAM switches from normal ODT (RTT_nom) to dynamic ODT (Rtt_WR) when beginning WRITE burst and subsequently switches back to normal ODT at the end of WRITE burst. The drop down options provided to the user are **off**, **RZQ/4** and **RZQ/2**.

- Auto self-refresh setting is defined by MR2[6] register bit of DDR3 memory with drop down option of **Manual** and **Auto**. Self-refresh temperature setting is defined by MR2[7] register bit of DDR2 memory with drop down options of **Normal** and **Extended**.

Figure 1-50. DDR Memory initialization Settings
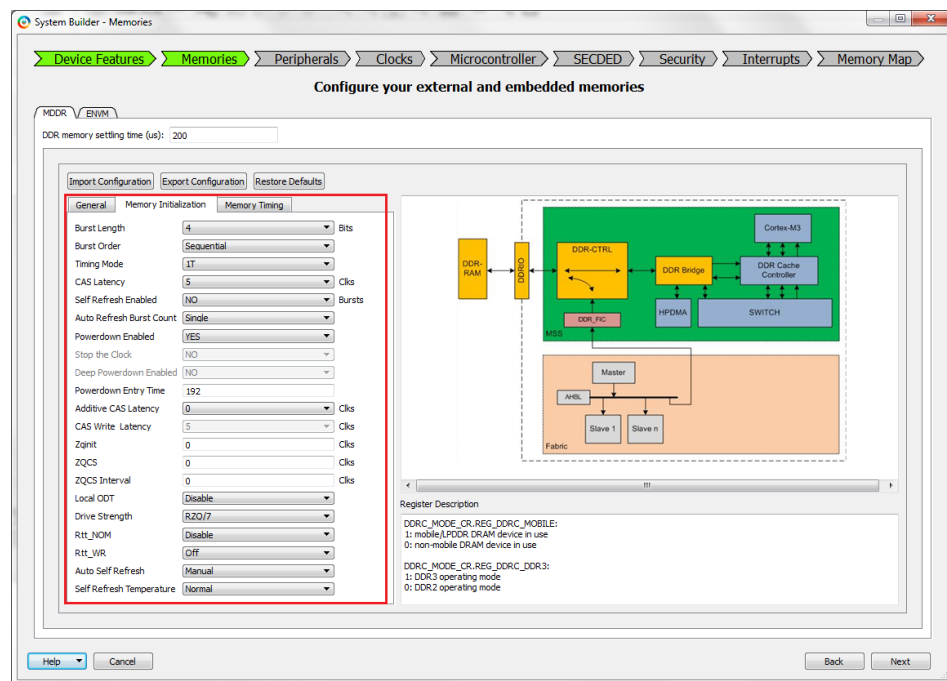
5. Select the memory timing settings under the **Memory Timing** tab according to the DDR memory vendor datasheet as shown in the following illustration. For more information, see .

Figure 1-51. DDR Memory Timing Settings
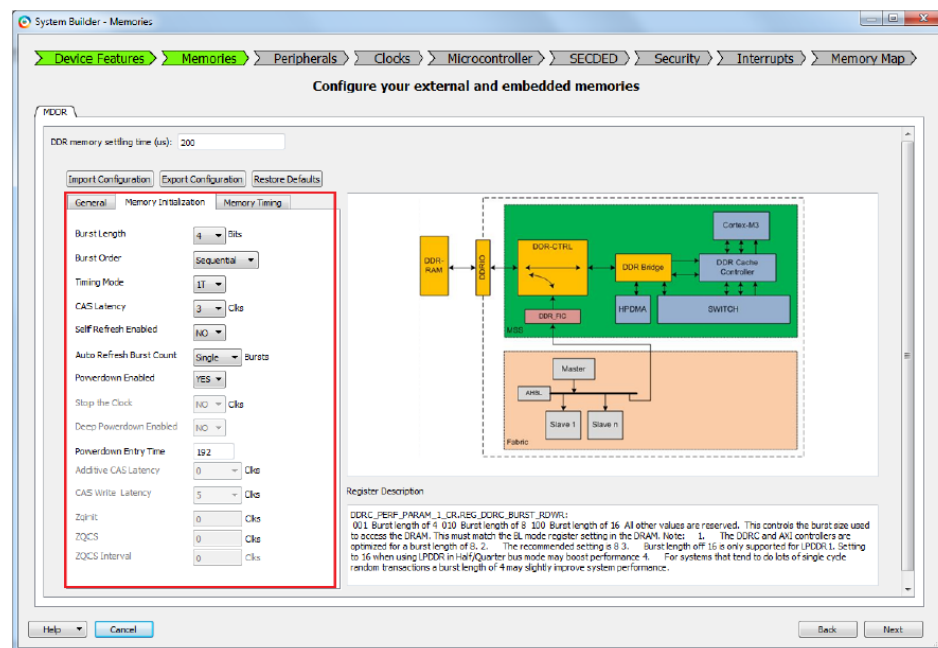
The configurator also provides the option to import and export the register configurations.

Configuration files for accessing DDR3 memory on SmartFusion2 Development kit can be downloaded from www.microsemi.com/soc/documents/MDDR3_16Bit_SB.zip.

Configuration files for accessing LPDDR memory on SmartFusion 2 Starter kit can be downloaded from www.microsemi.com/soc/documents/LPDDR_Emcraft_Config.zip.

> **Important:** The firmware generated by Libero SoC stores these configurations and the MDDR subsystem registers are initialized by the Cortex-M3 processor during the system_init phase of the firmware projects (SoftConsole/IAR/Keil projects generated by Libero SoC).

The following is an example of MDDR register configurations for operating the LPDDR memory (MT46H64M16LF) with clock 166 MHz.
- Device Memory Settling Time (us): 200

The DDR memories require settling time for the memory to initialize before accessing it. the LPDDR memory model MT46H64M16LF needs 200 μs settling time.

General
- Memory Type - Select LPDDR
- Data Width: 16

Memory Initialization:
- Burst length: 8
- Burst Order: Interleaved
- Timing Mode: 1T
- CAS Latency: 3
- Self Refresh Enabled: No
- Auto Refresh Burst Count: 8
- PowerDown Enabled: Yes
- Stop the clock: No
- Deep PowerDown enabled: No
- No Activity clocks for Entry: 320

Memory Timing
- Time To Hold Reset Before INIT - 67584 clks
- MRD: 4 clks
- RAS (Min): 8 clks
- RAS (Max): 8192 clks
- RCD: 6 clks
- RP: 7 clks
- REFI: 3104 clks
- RC: 3 clks
- XP: 3 clks
- CKE: 3 clks
- RFC: 79 clks
- FAW: 0 clks

6. Navigate to the **Peripherals** tab. To access the MDDR from the FPGA fabric, drag and drop the Fabric AMBA Master to the MSS DDR FIC Subsystem and click configure to select the type of interface as AXI or single AHB-Lite. The user logic in the FPGA fabric can access the DDR memory through the MDDR using these interfaces. The following image shows the **Peripherals** tab.

Figure 1-52. MSS DDR FIC Subsystem Configuration
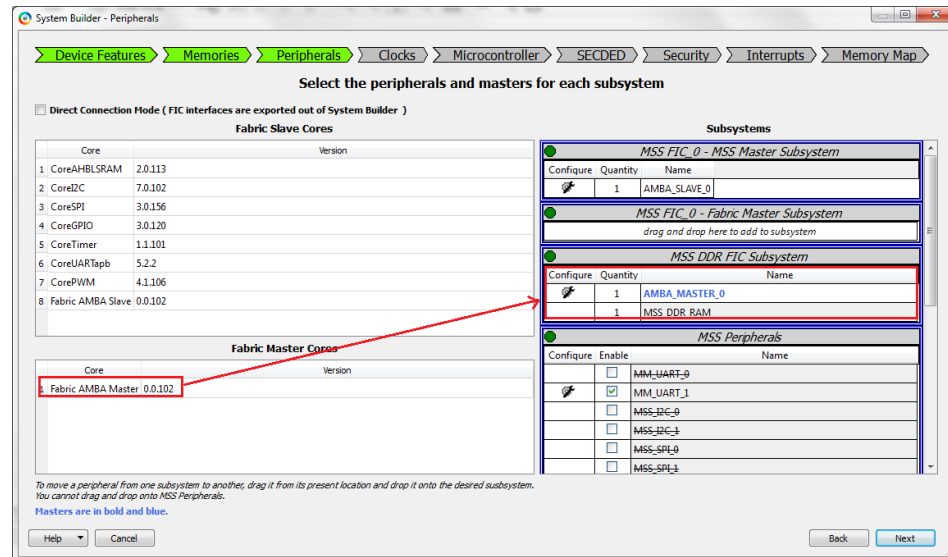
7. Navigate to the **Clocks** tab. The **Clocks** tab allows to configure the system clock and subsystem clocks. The MDDR subsystem operates on MDDR_CLK, which comes from MSS_CCC. The MDDR_CLK must be selected as multiples of 1, 2, 3, 4, 6 or 8-of M3_CLK. The maximum frequency of MDDR_CLK is 333.33 MHz. The following image shows the MDDR_CLK configuration.

Figure 1-53. MDDR Clock Configuration
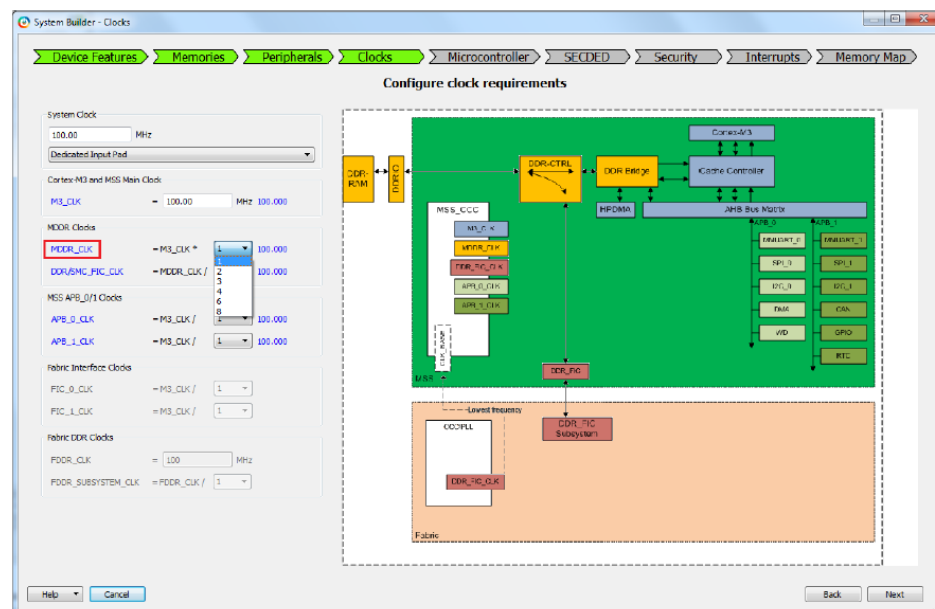
DDR_FIC_CLK drives the DDR_FIC slave interface and defines the frequency at which the FPGA fabric subsystem connected to this interface is intended to run. DDR_FIC_CLK can be configured as a ratio of MDDR_CLK (1, 2, 3, 4, 6, 8, 12, or 16) using the Clocks configurator. The maximum frequency of DDR_FIC_CLK is 200 MHz. The following image shows the DDR_FIC_CLK configuration.

If the MDDR_CLK ratio to M3_CLK is a multiple of 3, DDR_SMC_FIC_CLK's ratio to MDDR_CLK must also be a multiple of 3, and vice versa. The configurator issues an error if this requirement is not met. This limitation is imposed by the internal implementation of the MSS CCC.

Figure 1-54. DDR_FIC Clock Configuration
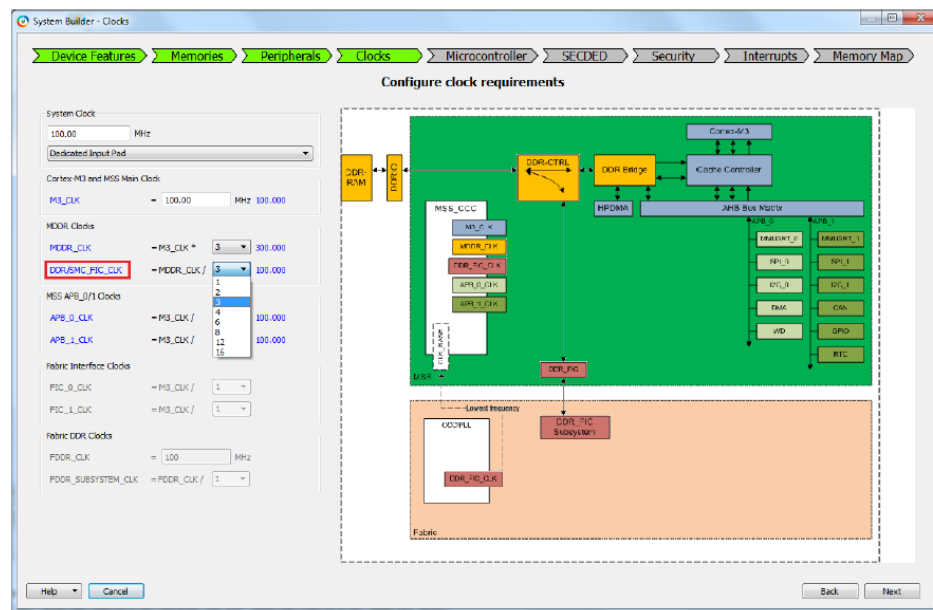
1.12.2   Design Flow Using SmartDesign (Ask a Question)

The following flowchart illustrates the design flow for using the MDDR subsystem to access external DDR memory. The design flow consists of two parts:

- **Libero SoC flow** – This includes configuring the type of DDR memory, choosing fabric master interface type, clocking, and DDR I/O settings.
- **MDDR register initialization** – The MDDR subsystem registers can be initialized using the Cortex-M3 processor or FPGA fabric master. After MSS resets, the MDDR registers must be configured according to application and DDR memory specification. The 1.5.5. MDDR Subsystem Features Configuration provides the details of required register configuration for MDDR features. While configuring the registers, the soft reset to the DDR controller must be asserted.

After releasing the soft reset, the DDR controller performs DDR memory initialization and sets the status bits in Table 1-76.

Figure 1-55. Design Flow
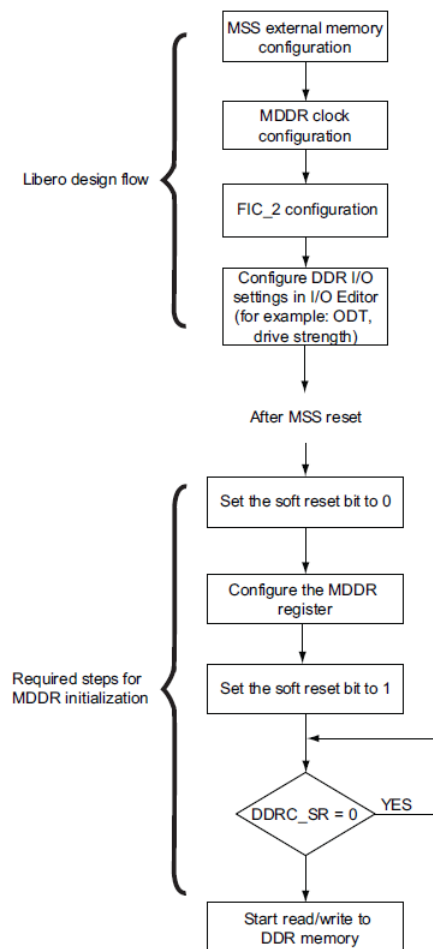
The following sections explain the configuration steps in the flowchart.

1.12.2.1 MSS External Memory Configuration (Ask a Question)

The MDDR subsystem is configured through the MDDR configurator, which is part of the MSS configurator in the Libero SoC design software. The following image shows the MDDR configurator.

Figure 1-56. MDDR Configurator
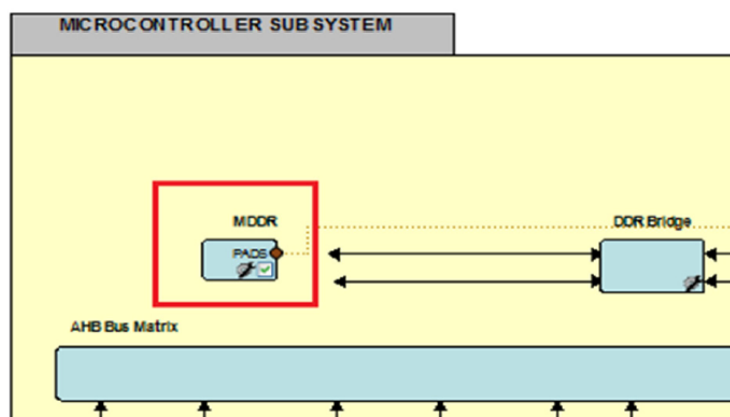
Double-click **MDDR** Configurator, which gives the following choices for the external memory interface type as shown in the following image.

- **Double Data Rate**: This option must be selected to access the external DDR memories (DDR2, DDR3, and LPDDR).
- **Soft Memory Controller**: This option must be selected to access the external memories through SMC_FIC and soft memory controller in FPGA. For more information on using SMC_FIC mode, see 4. Soft Memory Controller Fabric Interface Controller.

Figure 1-57. Memory Interface Configuration
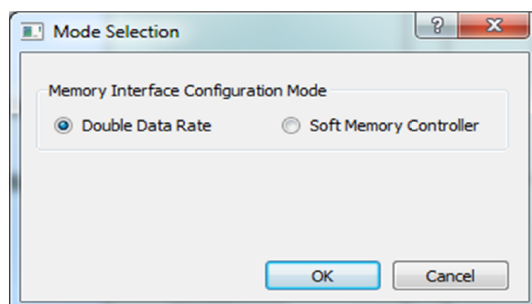
Select **Double Data Rate** and click **OK**. The MSS External Memory Configurator will be displayed as shown in the following image. Select the memory settings as described in the steps 2, 3 and 4 in the 1.12.1. Design Flow Using System Builder.

To access the MDDR from the FPGA fabric, select **From Fabric Interface Settings** and the type of interface as AXI, single AHBLite, or two AHBLite Interfaces. On completion of the configuration, the selected interface is exposed in SmartDesign. The user logic in the FPGA fabric can access the DDR memory through MDDR using these interfaces.

Figure 1-58. MSS External DDR Memory Configurator
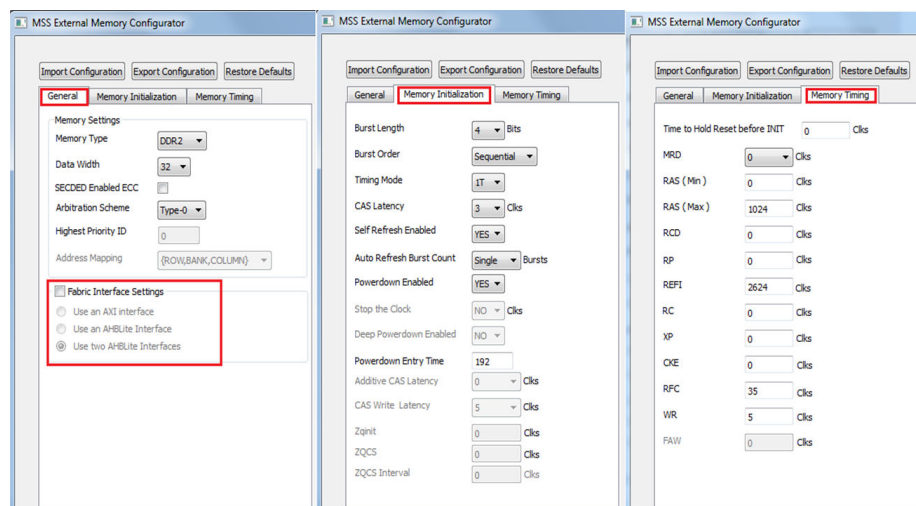
1.12.2.2 MDDR Clock Configuration (Ask a Question)

The MDDR subsystem operates on MDDR_CLK, which comes from MSS_CCC. The MDDR_CLK must be selected as a multiple—1, 2, 3, 4, 6 or 8—of M3_CLK. This clock value can be configured through the MSS_CCC configurator in Libero SoC, as shown in the following figure.

The maximum frequency of MDDR_CLK is 333.33 MHz.

Figure 1-59. MDDR Clock Configuration
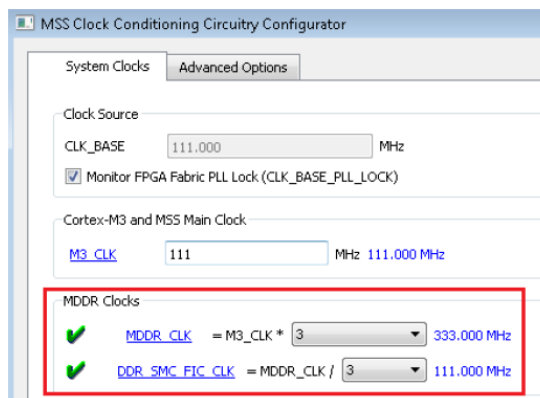
DDR_SMC_FIC_CLK drives the DDR_FIC slave interface and defines the frequency at which the FPGA fabric subsystem connected to this interface is intended to run. DDR_SMC_FIC_CLK can be configured as a ratio of MDDR_CLK (1, 2, 3, 4, 6, 8, 12, or 16) through the MSS_CCC configurator in Libero SoC, as shown in the following image. The maximum frequency of DDR_SMC_CLK is 200 MHz.

Figure 1-60. MDDR Clock Configuration
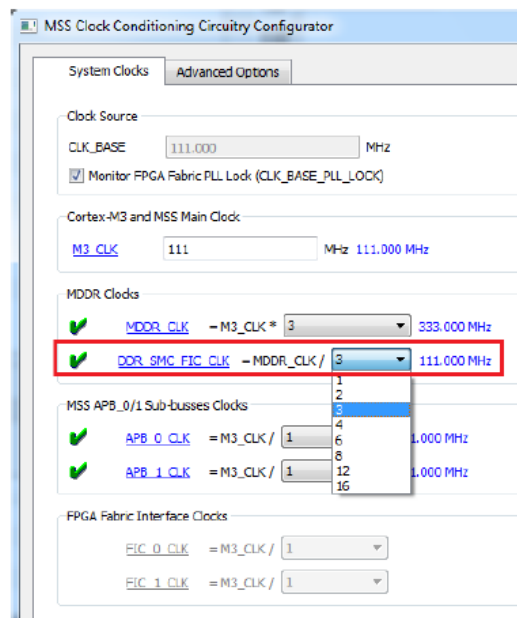
If the MDDR_CLK ratio to M3_CLK is a multiple of 3, DDR_SMC_FIC_CLK's ratio to MDDR_CLK must also be a multiple of 3, and vice versa. The configurator issues an error if this requirement is not met. This limitation is imposed by the internal implementation of the MSS CCC.

1.12.2.2.1 FIC_2 Configuration (Ask a Question)

This is required to initialize the MDDR registers (optional when initializing from MSS). Configure FIC_2 (peripheral initialization) block, as shown in the following image to expose the MDDR APB interface (MDDR_APB_SLAVE interface) in Libero SmartDesign. Use the MDDR_APB_SLAVE interface to connect with the APB master logic in the FPGA fabric.

Figure 1-61. FIC_2 Configuration
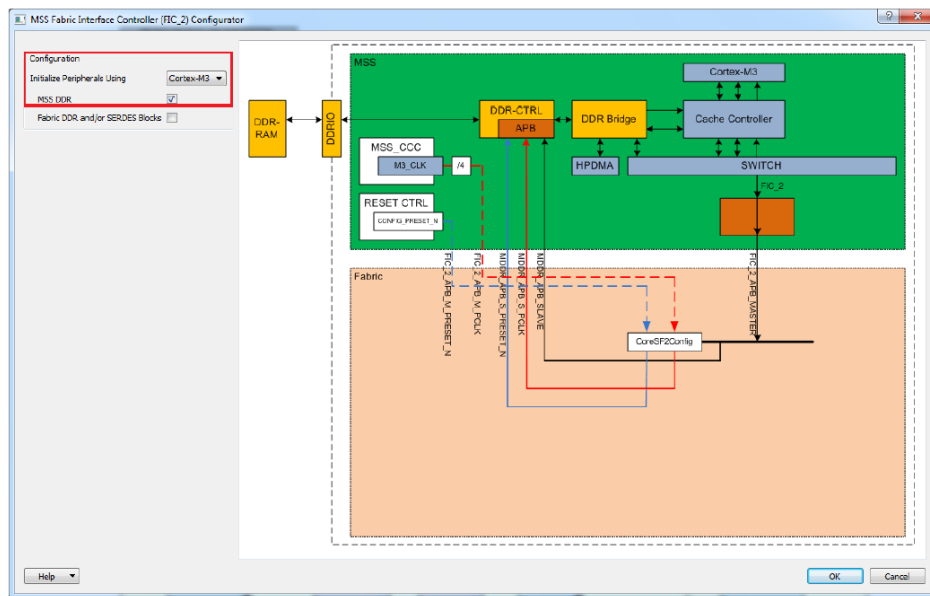
When enabling this option, the MDDR_APB_S_PCLK and FIC_2_APB_M_PCLK signals are exposed in SmartDesign. MDDR_APB_S_PCLK must be connected to FIC_2_APB_M_PCLK. The FIC_2_APB_M_PCLK clock is generated from the MSS_CCC and is identical to M3_CLK/4.

1.12.2.3 I/O Configuration (Ask a Question)

I/O settings such as ODT and drive strength can be configured as shown in the following image using the I/O Editor in the Libero design software.

Figure 1-62. I/O Configuration

For more information about MDDR Subsystem Features Configuration, see 1.5.5. MDDR Subsystem Features Configuration.

1.12.3 Use Model 1: Accessing MDDR from FPGA Fabric Through the AXI Interface (Ask a Question)

The MDDR subsystem can be used to access DDR memory as shown in the following illustration. This use model follows the steps "Design flow using SmartDesign" for using MDDR. The AXI master in the FPGA fabric accesses the DDR memory through the MDDR subsystem. The MDDR registers are configured from FPGA fabric through the APB interface. The APB master in the FPGA fabric asserts a ready signal to indicate that the DDR memory is successfully initialized.

The read, write, and read-modify-write transactions are initiated by the AXI master to read or write the data into the DDR memory after receiving the ready signal from APB master.

Figure 1-63. MDDR with AXI Interface
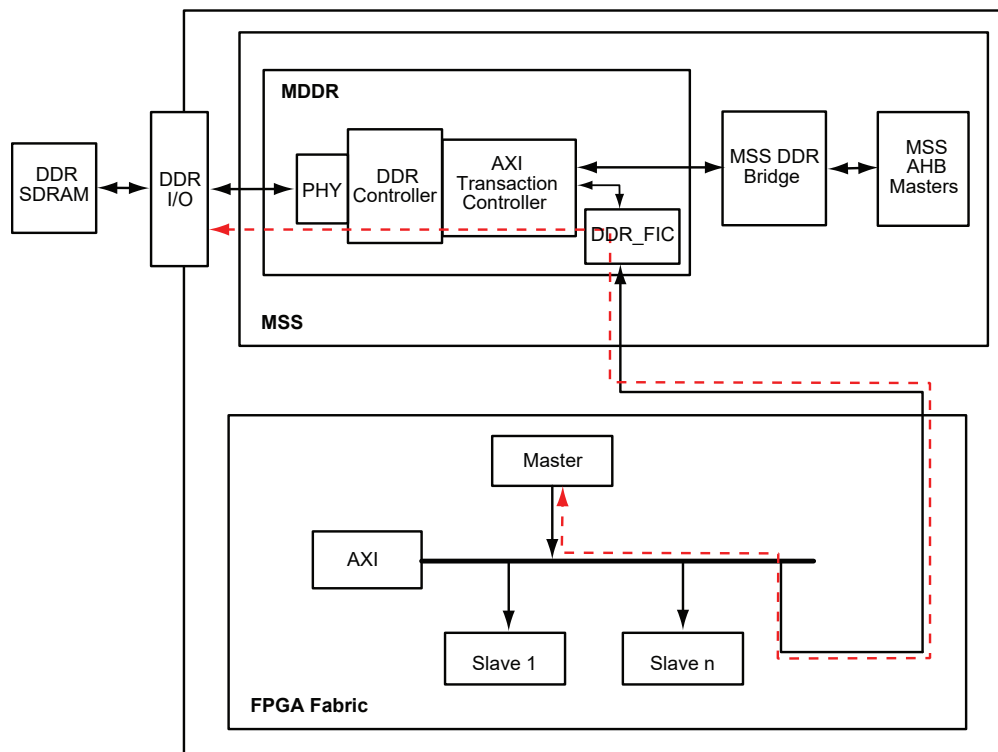
Use the following steps to access the MDDR from the AXI master in the FPGA fabric:

1. Instantiate the SmartFusion2 MSS component onto the SmartDesign canvas.
2. Configure the SmartFusion2 MSS peripheral components as required using the MSS configurator.
3. Configure the MDDR and select the AXI interface, as shown in the following image. In this example, the design is created to access DDR3 memory with a 32-bit data width.

Figure 1-64. MSS External Memory Configuration
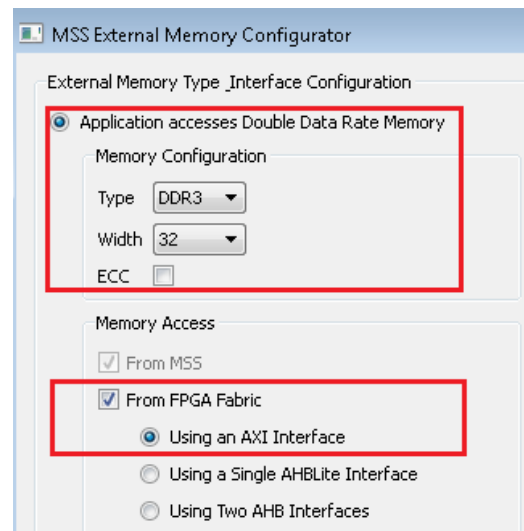
4. Configure FIC_2, as shown in the following image, to enable the MDDR subsystem APB interface for configuring the MDDR registers using APB master in the FPGA fabric.

Figure 1-65. Configuring FIC_2
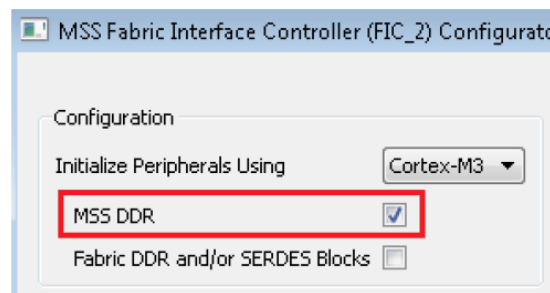
5. Configure the MSS_CCC for MDDR_CLK and DDR_SMC_FIC_CLK. In the following image, the MDDR clock is configured to 333 MHz and M3_CLK is configured to 111 MHz.

Figure 1-66. MDDR Clock Configuration
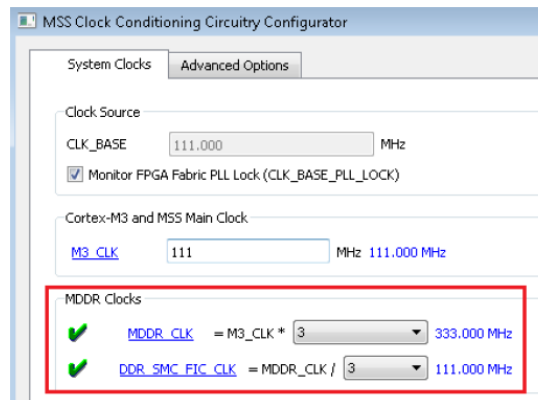
6. Instantiate the clock resources (FCCC and chip oscillators) in the SmartDesign canvas and configure, as required.

7. Instantiate user AXI master logic in the SmartDesign canvas to access the MDDR through the AXI interface. Make sure that the AXI master logic accesses the MDDR after configuring the MDDR registers from the APB master. The AXI master clock should be same as DDR_SMC_FIC_CLK.

8. Instantiate user APB master logic in the SmartDesign canvas to configure the MDDR registers through the APB interface. The APB master logic should initialize the registers after the MSS comes out of reset. The APB clock must be connected to FIC_2_APB_M_PCLK.

9. Connect the AXI master and APB master to the MSS component through CoreAXI and CoreAPB or use the auto connect option in SmartDesign.

10. Make the other connections in the SmartDesign canvas, as shown in the following illustration.

Figure 1-67. SmartDesign Canvas
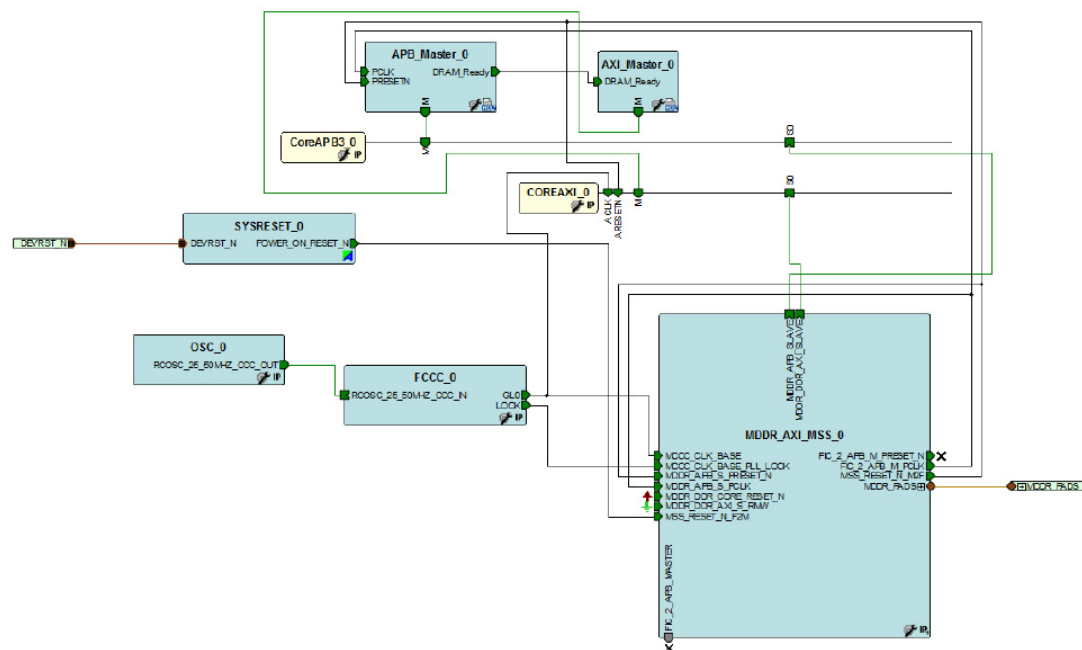
11. To verify the design in Libero SoC software, create a SmartDesign testbench project and instantiate a DDR memory model provided by the DDR memory vendor. Simulate the design and observe the AXI read and write transactions.

> **Important:** The MDDR subsystem can be configured using the Cortex-M3 processor without having an APB master. The System Builder can be used to create the design by following steps in 1.12.1. Design Flow Using System Builder. The System Builder provides "INIT_DONE" to indicate that the DDR memory has been successfully initialized.

1.12.4 Use Model 2: Accessing MDDR from FPGA Fabric Through the AHB Interface (Ask a Question)

The MDDR subsystem can be used to access the DDR memory, as shown in the following illustration. The MDDR register can be configured through the MSS or user logic (AHB master) in the FPGA fabric.

Figure 1-68. MDDR with Single AHB Interface
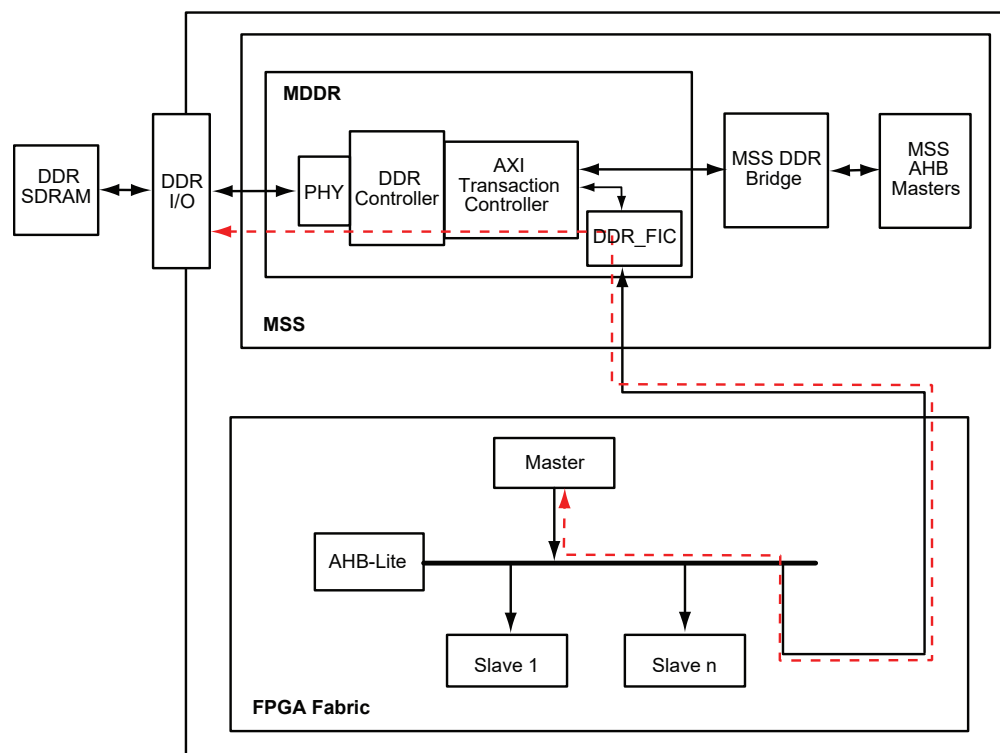
To use a dual rather than single AHB interface to the MDDR, set the CFG_NUM_AHB_MASTERS bit in the Table 1-223 register to 1.

Figure 1-69. MDDR with Dual AHB Interface
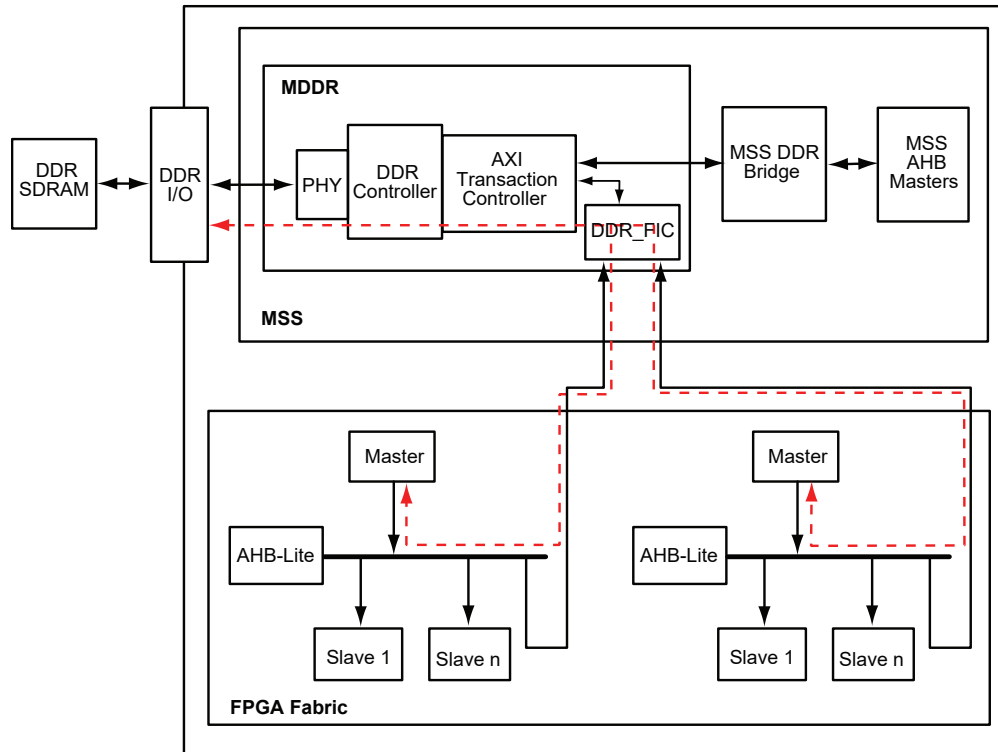
The steps for accessing the MDDR from one or two AHB masters in the FPGA fabric is the same as in 1.12.3. Use Model 1: Accessing MDDR from FPGA Fabric Through the AXI Interface except for the following:

- The single AHB or two AHB interfaces must be selected in the MSS external memory configurator instead of AXI master.

- One or two AHB masters must be connected through CoreAHB's in the SmartDesign canvas.

1.12.5 Use Model 3: Accessing MDDR from Cortex-M3 Processor (Ask a Question)

The Cortex-M3 processor can access the DDR SDRAM connected to the MDDR subsystem through the MSS DDR bridge, as shown in Figure 1-70.This use model follows the steps 1.12.1. Design Flow Using System Builder for using MDDR. Use the following steps to access the MDDR from the Cortex-M3 processor:

Figure 1-70. Accessing MDDR from Cortex-M3 Processor
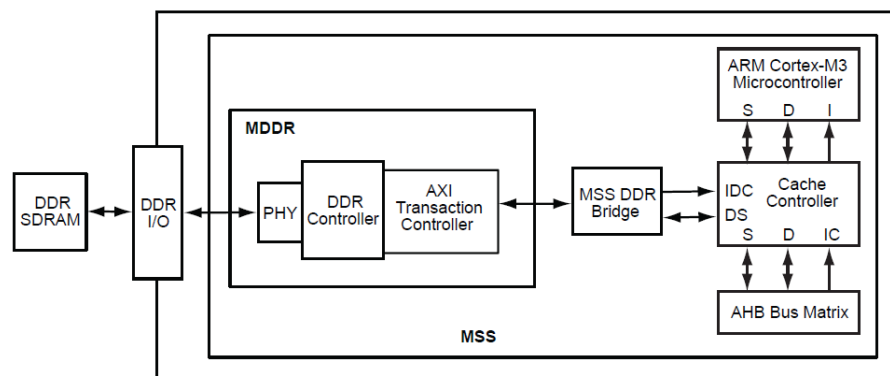
1. Go to the **System Builder** > **Device Features** and check the **MSS External Memory** check box and leave the rest of the check boxes unchecked. Figure 1-16 shows the **System Builder** > **Device Features**.

Figure 1-71. MSS External Memory Configuration
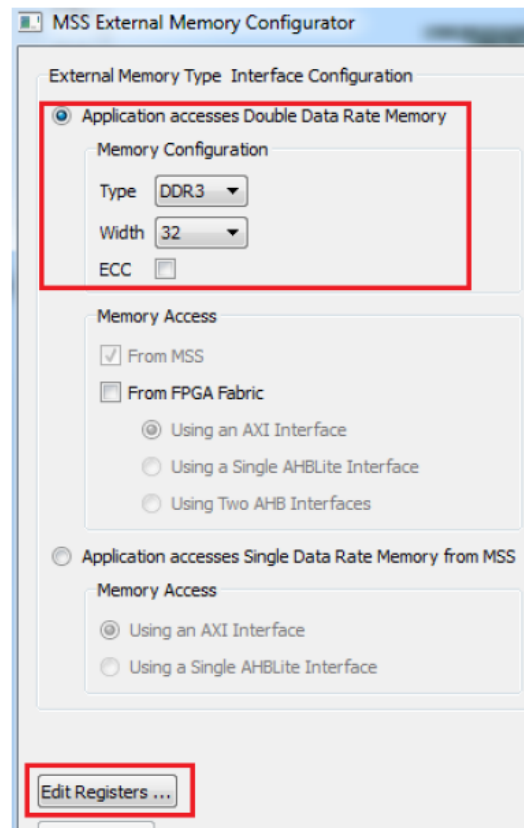
2. Navigate to **Memories** tab and import the DDR configuration file or select the appropriate DDR memory settings. See 1.5.5. MDDR Subsystem Features Configuration to configure the necessary registers.

3. Navigate to **Clocks** tab to configure the MDDR_CLK. In this example, MDDR_CLK is configured to 333 MHz as shown in the following image.

Figure 1-72. Configuring MDDR_CLK
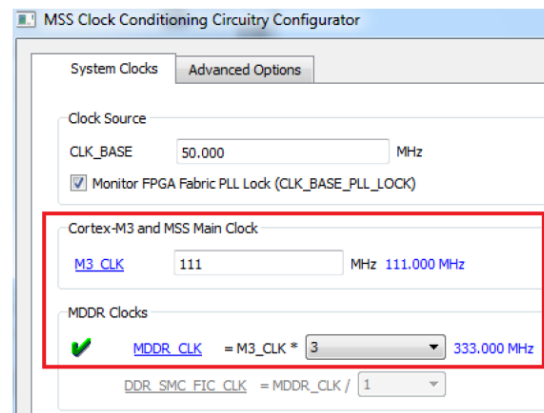
4. Navigate to the **Memory Map** tab giving the required data in the rest of the **System Builder** tabs. Click **Finish**, the system builder creates the design and generates.

5. Connect the clock resources to the MSS component in the SmartDesign canvas.

6. To verify the design in Libero SoC software, create the SmartDesign testbench project and instantiate a DDR memory model provided by the DDR memory vendor.

7. Write BFM commands for read and write transactions. The MDDR_init.bfm file will be generated by Libero SoC software, containing the BFM commands to initialize the MDDR registers.

8. Simulate the design to verify the read/write transactions to DDR memory.

9. Open **I/O Attribute Editor** to configure the ODT and drive strengths.

10. Program the device.

11. Use the generated firmware project to access the DDR memory from the Cortex-M3 processor through MDDR. The firmware project initializes the MDDR subsystem before executing the instructions in main() with the register settings provided in the above step 2.

See MDDR Tutorial, which describes the steps to create the design for accessing the MDDR from the Cortex-M3 processor. The tutorial also explains the steps for simulating the design in Libero SoC.

1.12.6 Use Model 4: Accessing MDDR from the HPDMA (Ask a Question)

The HPDMA controller can access DDR SDRAM connected to the MDDR subsystem through the MSS DDR bridge, as shown in the following illustration.

Figure 1-73. Accessing MDDR from HPDMA
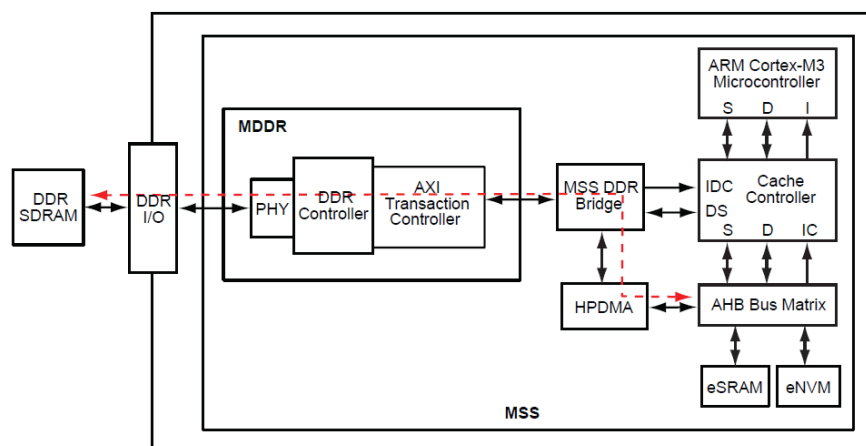
The steps for accessing the MDDR from the HPDMA are the same as in 1.12.5. Use Model 3: Accessing MDDR from Cortex-M3 Processor. Use the generated firmware project to access DDR memory from the HPDMA through the MDDR. The HPDMA driver has the MSS_HPDMA_start() API to initiate memory transfers and DDR memory from and to other memory locations. This API requires the parameter's source address, destination address, and number of bytes to transfer. For more Information on how to use HPDMA, refer to the HPDMA chapter in UG0331: SmartFusion2 Microcontroller Subsystem User Guide.

For information on timing diagrams, refer to the 1.7. Timing Diagrams.

2.   Fabric DDR Subsystem (Ask a Question)

The FDDR is a hardened ASIC block for interfacing the DDR2, DDR3, and LPDDR1 memories. The FDDR subsystem is used to access DDR memories for high-speed data transfers. The FDDR subsystem includes the DDR memory controller, DDR PHY, and arbitration logic to support multiple masters.

FPGA fabric masters communicate with the DDR memories interfaced to the FDDR subsystem through AXI or AHB interfaces.

2.1   Features (Ask a Question)

- Integrated on-chip DDR memory controller and PHY
- Configurable to support LPDDR1, DDR2, and DDR3 memory devices
- Up to 667 Mbps (333 MHz DDR) performance
- Supports memory densities up to 2 GB
- Supports 8/16/32-bit data bus width modes
- Supports a maximum of 8 memory banks
- Supports single rank memory
- Single Error Correction and Double Error Detection (SECDED) enable or disable feature
- Supports DRAM burst lengths of 4, 8, or 16, depending on the Bus-width mode and DDR type configuration
- Support for sequential and interleaved burst ordering
- Programs internal control for ZQ short calibration cycles for DDR3 configurations
- Supports dynamic scheduling to optimize bandwidth and latency
- Supports self refresh entry and exit on command
- Supports deep power-down entry and exit on command
- Flexible address mapper logic to allow application specific mapping of row, column, bank, and rank bits
- Configurable support for 1T or 2T timing on the DDR SDRAM control signals
- Supports autonomous DRAM power-down entry and exit caused by lack of transaction arrival for programmable time
- Advanced power-saving design includes toggling of command, address, and data pins

The system level block diagram of the FDDR subsystem is shown in the following illustration.

Figure 2-1. System Level FDDR Block Diagram

Note: Blocks and arrows marked in grey, only in SmartFusion2 MSS. Rest are all similar in MSSHPMS.
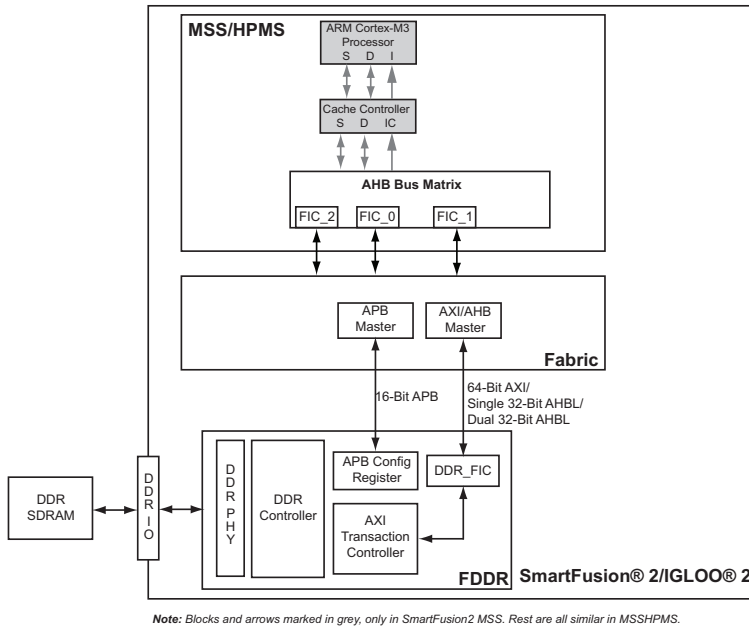
The FDDR subsystem accepts data transfer requests from AXI or AHB interfaces. Any read or write transactions to the DDR memories can occur through the AXI or AHBL masters in the FPGA fabric through DDR_FIC interface.

> **Important:** The maximum DDR3 data rate supported by FDDR is 333 MHz/667 Mbps. Therefore, Write Leveling is not mandatory and the interface works if the board layout includes length matching and follows AN4153: Board and Layout Design Guidelines for SmartFusion2 SoC and IGLOO2 FPGAs. For Read Leveling, Libero SOC auto-generates pre-defined static delay ratios for FDDR initialization. These delay values are sufficient if the board layout follows the SmartFusion 2/IGLOO 2 board-level guidelines.

2.2 Memory Configurations (Ask a Question)

The SmartFusion 2 and IGLOO 2 FDDR subsystem supports a wide range of common memory types, configurations, and densities, as shown in the following table. If SECDED mode is enabled in the FDDR controller, the external memory module must be connected to the following:

- Data lines FDDR_DQ_ECC[3:0] when data width is x32
- Data lines FDDR_DQ_ECC[1:0] when data width is x16
- Data line FDDR_DQ_ECC[0] when data width is x8

Table 2-1. Supported Memory (DDR2, DDR3, and LPDDR1) Configurations

Memory Depth	Width	Width (in SECDED Mode)	SmartFusion® 2/IGLOO® 2 Devices		
			M2S150 (FCV484)	M2S050/M2GL050 (FG896)	M2S150/M2GL150 (FC1152)
128M or Less	x32	x36	—	√	√
	x16	x18	√	√	√
	x8	x9	√	—	√

...........continued					
Memory Depth	**Width**	**Width (in SECDED Mode)**	**SmartFusion® 2/IGLOO® 2 Devices**		
256M	x32	x36	—	√	√
	x16	x18	√	√	√
	x8	x9	√	—	√
512M	x32	x36	—	√	√
	x16	x18	√	√	√
	x8	x9	√	—	√
1G	x32	x36	—	√	√
	x16	x18	√	√	√
	x8	x9	√	—	√

2.3 Performance (Ask a Question)

The following table shows the maximum data rates supported by the FDDR subsystem for supported memory types. For more information on DDR Speeds, see the DDR Memory Interface Characteristics section in IGLOO2 and SmartFusion2 Datasheet.

Table 2-2. DDR Speeds

Memory Type	Maximum Data Rate (Mbps)
LPDDR1	400 Mbps (200 MHz)
DDR2	667 Mbps (333 MHz)
DDR3	667 Mbps (333 MHz)

2.4 I/O Utilization (Ask a Question)

The following table shows the I/O utilization for SmartFusion 2 and IGLOO 2 devices corresponding to supported bus widths. The remaining I/Os in bank 0 can be used for general purposes.

Table 2-3. I/O Utilization for SmartFusion® 2 and IGLOO® 2 Devices

FDDR Bus Width	M2S050/M2GL050 (FG896)	M2S150/M2GL150 (FC1152)
36-bit	Bank5 (85 pins)	Bank1 (85 pins)
32-bit	Bank5 (76 pins)	Bank1 (76 pins)
18-bit	Bank5 (59 pins)	Bank1 (59 pins)
16-bit	Bank5 (53 pins)	Bank1 (53 pins)
9-bit	—	Bank1 (47 pins)
8-bit	—	Bank1 (41 pins)

Important: If FDDR is configured for LPDDR, one more I/O also available for every 8-bit as the LPDDR does not have DQS_N.

For general purpose use of the unused I/Os in the FDDR bank, select one of the I/O standards with the same voltage level as the DDR I/Os.

Self refresh must be disabled if the FDDR banks contain a mixed of I/Os used for DDR and for general purpose fabric I/Os. For more information, see "Self Refresh" in the 2.6.3.4.5. Power Saving Modes.

2.5 Functional Description (Ask a Question)

This section provides a detailed description of the FDDR subsystem with the following sub-sections:

2.5.1 Architecture Overview (Ask a Question)

A functional block diagram of the FDDR subsystem is shown in the following illustration. The main components include the DDR fabric interface controller (DDR_FIC), AXI transaction handler, DDR memory controller, and DDR PHY.

Figure 2-2. FDDR Subsystem Functional Block Diagram
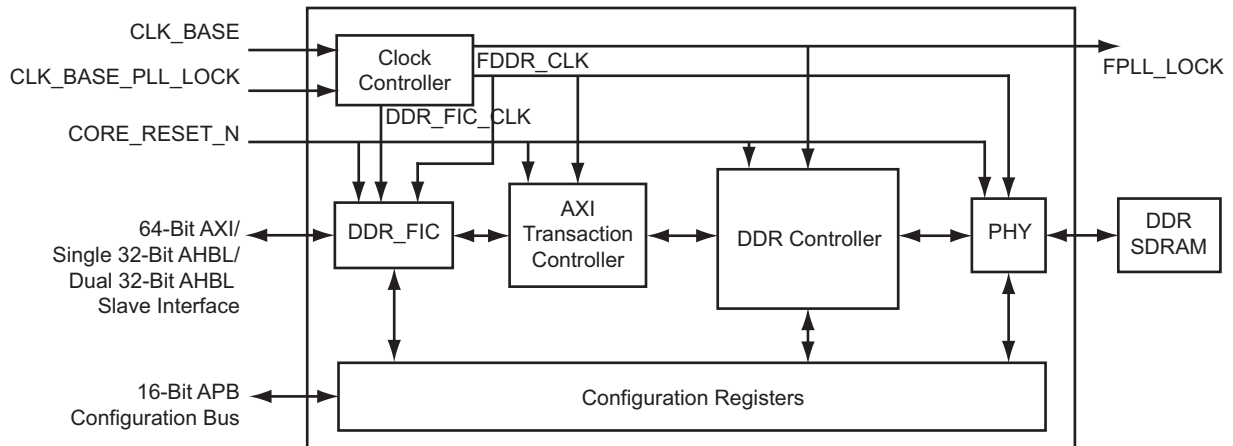
The FDDR subsystem has a dedicated clock controller for generating clocks to the components of FDDR from the base clock (CLK_BASE). The CLK_BASE for the FDDR originates from a fabric CCC or an external source through the FPGA fabric.

The DDR_FIC facilitates communication between the FPGA fabric masters and AXI transaction controller. The DDR_FIC can be configured to provide either one 64-bit AXI slave interface or two independent 32-bit AHB-Lite (AHBL) slave interfaces to the FPGA fabric masters.

The AXI transaction controller receives read and write requests from AXI masters (DDR_FIC) and schedules for the DDR controller by translating them into DDR controller commands.

The DDR controller receives the commands from the AXI transaction controller. These commands are queued internally and scheduled for access to the DDR SDRAM while satisfying DDR SDRAM constraints, transaction priorities, and dependencies between the transactions. The DDR controller in turn issues commands to the PHY module, which launches and captures data to and from the DDR SDRAM.

DDR PHY receives commands from the DDR controller and generates DDR memory signals required to access the external DDR memory.

The 16-bit APB configuration bus provides an interface for configuring the FDDR subsystem registers.

2.5.2 Port List (Ask a Question)

Table 2-4. FDDR Subsystem Interface Signals

Signal Name	Type	Polarity	Description
APB_S_PCLK	In	—	APB clock. This clock drives all the registers of the APB interface.

...........continued

Signal Name	Type	Polarity	Description
APB_S_PRESET_N	In	Low	APB reset signal. This is an active-low signal. This drives the APB interface and is used to generate the soft reset for the DDR controller as well.
CORE_RESET_N	In	Low	Global reset. This resets the DDR_FIC/DDRC/PHY/DDRAXI logic.
FDDR_SUBSYSTEM_CLK	In	—	Base clock to the FDDR clock controller. This clock is used as the reference clock to the Fabric Phase-Locked Loop (FPLL). The user sets the multiplier of the FPLL based on the rate of the AXI/AHB interface. This is done by setting the DDR_FIC divider setting in the System Builder Clocks tab.
AXI_S_RMW	In	High	AXI mode only Indicates whether all bytes of a 64-bit lane are valid for all beats of an AXI transfer. • 0: Indicates that all bytes in all beats are valid in the burst and the controller should default to write commands • 1: Indicates that some bytes are invalid and the controller should default to RMW commands This is classed as an AXI write address channel sideband signal and is valid with the AWVALID signal. Only used when SECDED is enabled.
HPMS_DDR_FIC_SUBSYSTEM_CLK[3]	Out	—	This output clock is derived from the FDDR_CLK and is based on the DDR_FIC divider ratio. This is the clock that should be used for the AXI or AHB slave interfaces to move data in and out of the FDDR.
HPMS_DDR_FIC_SUBSYSTEM_LOCK[3]	Out	—	HPMS_DDR_FIC_SUBSYSTEM_LOCK indicates the lock from FCCC which generates HPMS_DDR_FIC_SUBSYSTEM_CLK
FDDR_SUBSYTEM_CLK_PLL_LOCK	In	High	Fabric PLL lock input
Slave Interfaces			
AXI_SLAVE[1]	Bus	—	AXI slave interface 1.0 bus
AHB0_SLAVE[1]	Bus	—	AHB0 slave interface 3.0 bus
AHB1_SLAVE[1]	Bus	—	AHB1 slave interface 3.0 bus
APB_SLAVE	Bus	—	APB slave interface 3.0 bus
DRAM Interface			
FDDR_CAS_N	Out	Low	DRAM CASN
FDDR_CKE	Out	High	DRAM CKE
FDDR_CLK	Out	—	DRAM single-ended clock – for differential pads
FDDR_CLK_N	Out	—	DRAM single-ended clock – for differential pads
FDDR_CS_N	Out	Low	DRAM CSN
FDDR_ODT	Out	High	DRAM ODT. 0: Termination Off 1: Termination On
FDDR_RAS_N	Out	Low	DRAM RASN
FDDR_ RESET_N	Out	Low	DRAM reset for DDR3
FDDR_WE_N	Out	Low	DRAM WEN
FDDR_ADDR[15:0]	Out	—	Dram address bits
FDDR_BA[2:0]	Out	—	Dram bank address
FDDR_DM_RDQS[3:0]	In/out	—	DRAM data mask – from bidirectional pads
FDDR_DQS[3:0]	In/out	—	DRAM single-ended data strobe output – for bidirectional pads

...........continued

Signal Name	Type	Polarity	Description
FDDR_DQS_N[3:0]	In/out	—	DRAM single-ended data strobe output – for bidirectional pads
FDDR_DQ[31:0]	In/out	—	DRAM data input or output – for bidirectional pads
FDDR_DQ_ECC[3:0]	In/out	—	DRAM data input or output for SECDED
FDDR_DM_RDQS_ECC	In/out	High	DRAM single-ended data strobe output – for bidirectional pads
FDDR_DQS_ECC	In/out	High	DRAM single-ended data strobe output – for bidirectional pads
FDDR_DQS_ECC_N	In/out	Low	DRAM data input or output – for bidirectional pads
FDDR_DQS_TMATCH_0_IN	In	High	DQS enables input for timing match between DQS and system clock. For simulations, tie to FDDR_DQS_TMATCH_0_OUT.
FDDR_DQS_TMATCH_1_IN	In	High	DQS enables input for timing match between DQS and system clock. For simulations, tie to FDDR_DQS_TMATCH_1_OUT.
FDDR_DQS_TMATCH_0_OUT	Out	High	DQS enables output for timing match between DQS and system clock. For simulations, tie to FDDR_DQS_TMATCH_0_IN.
FDDR_DQS_TMATCH_1_OUT	Out	High	DQS enables output for timing match between DQS and system clock. For simulations, tie to FDDR_DQS_TMATCH_1_IN.
FDDR_DQS_TMATCH_ECC_IN	In	High	DQS enables input for timing match between DQS and system clock. For simulations, tie to FDDR_DQS_TMATCH_ECC_OUT.
FDDR_DQS_TMATCH_ECC_OUT	Out	High	DQS enables output for timing match between DQS and system clock. For simulations, tie to FDDR_DQS_TMATCH_ECC_IN.

Notes:

1. AXI or AHB interface, depending on configuration.
2. FDDR_DQS_N[3:0] signals are not available for LPDDR.
3. Only in IGLOO 2 Devices.
4. TMATCH_IN and TMATCH_OUT pins are required to be connected together outside the device. They are used for gate training as part of the read data capture operation. The two pins create an internal DQS Enable signal that is used to calibrate the flight path. DQS needs to be gated to prevent false triggering of the FIFO write clock. This DQS Enable signal is derived from the system clock and physically matches the clock output buffer and DQS input buffer to compensate for I/O buffer uncertainty due to Process-Voltage-Temperature (PVT) changes. Without this connection, the circuit is not operable.

2.5.2.1 AXI Slave Interface (Ask a Question)

The following table describes the FDDR AXI slave interface signals. These signals are available only if FDDR interface is configured for AXI mode. For more information on AXI protocol, see AMBA AXI v1.0 protocol specification.

Table 2-5. FDDR AXI Slave Interface Signals

Signal Name	Direction	Polarity	Description
AXI_S_ARREADY	Output	High	Indicates whether the slave is ready to accept an address and associated control signals. • 1: Slave ready • 0: Slave not ready

Microchip

..........continued

Signal Name	Direction	Polarity	Description
AXI_S_AWREADY	Output	High	Indicates that the slave is ready to accept an address and associated control signals. • 1: Slave ready • 0: Slave not ready
AXI_S_BID[3:0]	Output	—	Indicates response ID. The identification tag of the write response.
AXI_S_BRESP[1:0]	Output	—	Indicates write response. This signal indicates the status of the write transaction. • 00: Normal access okay • 01: Exclusive access okay • 10: Slave error • 11: Decode error
AXI_S_BVALID	Output	High	Indicates whether a valid write response is available. • 1: Write response available • 0: Write response not available
AXI_S_RDATA[63:0]	Output	—	Indicates read data.
AXI_S_RID[3:0]	Output	—	Read ID tag. This signal is the ID tag of the read data group of signals.
AXI_S_RLAST	Output	High	Indicates the last transfer in a read burst.
AXI_S_RRESP[1:0]	Output	—	Indicates read response. This signal indicates the status of the read transfer. • 00: Normal access okay • 01: Exclusive access okay • 10: Slave error • 11: Decode error
AXI_S_RVALID	Output	—	Indicates whether the required read data is available and the read transfer can complete. • 1: Read data available • 0: Read data not available
AXI_S_WREADY	Output	High	Indicates whether the slave can accept the write data. • 1: Slave ready • 0: Slave not ready
AXI_S_ARADDR[31:0]	Input	—	Indicates initial address of a read burst transaction.
AXI_S_ARBURST[1:0]	Input	—	Indicates burst type. The burst type, coupled with the size information, details how the address for each transfer within the burst is calculated. • 00: FIXED - Fixed-address burst FIFO type • 01: INCR - Incrementing-address burst normal sequential memory • 10: WRAP - Incrementing-address burst that wraps to a lower address at the wrap boundary • 11: Reserved
AXI_S_ARID[3:0]	Input	—	Indicates identification tag for the read address group of signals.

...........continued

Signal Name	Direction	Polarity	Description
AXI_S_ARLEN[3:0]	Input	—	Indicates burst length. The burst length gives the exact number of transfers in a burst. • 0000: 1 • 0001: 2 • 0010: 3 • 0011: 4 • 0100: 5 • 0101: 6 • 0110: 7 • 0111: 8 • 1000: 9 • 1001: 10 • 1010: 11 • 1011: 12 • 1100: 13 • 1101: 14 • 1110: 15 • 1111: 16
AXI_S_ARLOCK[1:0]	Input	—	Indicates lock type. This signal provides additional information about the atomic characteristics of the read transfer. • 00: Normal access • 01: Exclusive access • 10: Locked access • 11: Reserved
AXI_S_ARSIZE[1:0]	Input	—	Indicates the maximum number of data bytes to transfer in each data transfer, within a burst. • 00: 1 • 01: 2 • 10: 4 • 11: 8
AXI_S_ARVALID	Input	High	Indicates the validity of read address and control information. • 1: Address and control information valid • 0: Address and control information not valid
AXI_S_AWADDR[31:0]	Input	—	Indicates write address. The write address bus gives the address of the first transfer in a write burst transaction.
AXI_S_AWBURST[1:0]	Input	—	Indicates burst type. The burst type, coupled with the size information, details how the address for each transfer within the burst is calculated. • 00: FIXED - Fixed-address burst FIFO-type • 01: INCR - Incrementing-address burst normal sequential memory • 10: WRAP - Incrementing-address burst that wraps to a lower address at the wrap boundary • 11: Reserved
AXI_S_AWID[3:0]	Input	—	Indicates identification tag for the write address group of signals.

..........continued

Signal Name	Direction	Polarity	Description
AXI_S_AWLEN[3:0]	Input	—	Indicates burst length. The burst length gives the exact number of transfers in a burst. This information determines the number of data transfers associated with the address. • 0000: 1 • 0001: 2 • 0010: 3 • 0011: 4 • 0100: 5 • 0101: 6 • 0110: 7 • 0111: 8 • 1000: 9 • 1001: 10 • 1010: 11 • 1011: 12 • 1100: 13 • 1101: 14 • 1110: 15 • 1111: 16
AXI_S_AWLOCK[1:0]	Input	—	Indicates lock type. This signal provides additional information about the atomic characteristics of the write transfer. • 00: Normal access • 01: Exclusive access • 10: Locked access • 11: Reserved
AXI_S_AWSIZE[1:0]	Input	—	Indicates the maximum number of data bytes to transfer in each data transfer, within a burst. • 00: 1 • 01: 2 • 10: 4 • 11: 8
AXI_S_AWVALID	Input	High	Indicates whether valid write address and control information are available. • 1: Address and control information available • 0: Address and control information not available
AXI_S_BREADY	Input	High	Indicates whether the master can accept the response information. • 1: Master ready • 0: Master not ready
AXI_S_RREADY	Input	High	Indicates whether the master can accept the read data and response information. • 1: Master ready • 0: Master not ready
AXI_S_WDATA[63:0]	Input	—	Indicates write data.
AXI_S_WID[3:0]	Input	—	Indicates response ID. The identification tag of the write response.
AXI_S_WLAST	Input	High	Indicates the last transfer in a write burst.

...........continued

Signal Name	Direction	Polarity	Description
AXI_S_WSTRB[7:0]	Input	—	Indicates which byte lanes to update in memory.
AXI_S_WVALID	Input	High	Indicates whether valid write data and strobes are available. • 1: Write data and strobes available • 0: Write data and strobes not available

2.5.2.2 AHB Slave (Ask a Question)

The following table describes the FDDR AHB slave interface signals. These signals are available only if FDDR interface is configured for single or dual AHB mode. For more information on AHB protocol, see AMBA AHB v3.0 protocol specification.

Table 2-6. FDDR AHB Slave Interface Signals

Signal Name	Direction	Polarity	Description
AHBx_S_HREADYOUT	Output	High	Indicates that a transfer has finished on the bus. The signal is asserted LOW to extend a transfer. Input to Fabric master.
AHBx_S_HRESP	Output	High	Indicates AHB transfer response to Fabric master.
AHBx_S_HRDATA[31:0]	Output	—	Indicates AHB read data to Fabric master.
AHBx_S_HSEL	Input	High	Indicates AHB slave select signal from Fabric master.
AHBx_S_HADDR[31:0]	Input	—	Indicates AHB address initiated by Fabric master.
AHBx_S_HBURST[2:0]	Input	—	Indicates AHB burst type from Fabric master. • 000: Single burst • 001: Incrementing burst of undefined length • 010: 4-beat wrapping burst • 011: 4-beat incrementing burst • 100: 8-beat wrapping burst • 101: 8-beat incrementing burst • 110: 16-beat wrapping burst • 111: 16-beat incrementing burst
AHBx_S_HSIZE[1:0]	Input	—	Indicates AHB transfer size from Fabric master. • 00: 8 Byte • 01: 16 Halfword • 10: 32 Word
AHBx_S_HTRANS[1:0]	Input	—	Indicates AHB transfer type from Fabric master. • 00: IDLE • 01: BUSY • 10: NONSEQUENTIAL • 11: SEQUENTIAL
AHBx_S_HMASTLOCK	Input	High	Indicates AHB master lock signal from Fabric master.
AHBx_S_HWRITE	Input	High	Indicates AHB write-control signal from Fabric master.
AHBx_S_HREADY	Input	High	Indicates that a transfer has finished on the bus. Fabric master can drive this signal LOW to extend a transfer.
AHBx_S_HWDATA[31:0]	Input	—	Indicates AHB write data from Fabric master.

Important: AHBx indicates AHB0 or AHB1.

2.5.2.3 APB Slave (Ask a Question)

The following table describes the FDDR APB slave interface signals. For information on APB protocol, see AMBA APB v3.0 protocol specification.

Table 2-7. FDDR APB Slave Interface Signals

Signal Name	Direction	Polarity	Description
APB_S_PREADY	Output	High	Indicates APB Ready signal to Fabric master.
APB_S_PSLVERR	Output	High	Indicates error condition on an APB transfer to Fabric master.
APB_S_PRDATA[15:0]	Output	—	Indicates APB read data to Fabric master.
APB_S_PENABLE	Input	High	Indicates APB enable from Fabric master. The enable signal is used to indicate the second cycle of an APB transfer.
APB_S_PSEL	Input	High	Indicates APB slave select signal from Fabric master
APB_S_PWRITE	Input	High	Indicates APB write-control signal form Fabric master
APB_S_PADDR[10:2]	Input	—	Indicates APB address initiated by Fabric master.
APB_S_PWDATA[15:0]	Input	—	Indicates APB write data from Fabric master.

2.6 Initialization (Ask a Question)

After power-up, the FDDR needs to have all of the configuration registers written to establish the operating modes of the blocks. When using the **System Builder** design flow through Libero SoC, this is all handled for the user through the use of the **System Builder** module. All of the configuration register values are selected by the user and stored in a special portion of the eNVM. Before the FDDR subsystem is active, it goes through an initialization phase and this process starts with a reset sequence. For DDR3 memories, the initialization phase also includes ZQ calibration and DRAM training.

2.6.1 Reset Sequence (Ask a Question)

The following illustration shows the required reset sequence for FDDR subsystem from the power-on-reset stage. Once the CORE_RESET_N signal of the FDDR subsystem is asserted, the MSS_RESET_N_M2F, HPMS_RESET_N_M2F, and FPLL_LOCK signals are asserted and the APB register configuration is complete (see the following figure). Assertion of CORE_RESET_N signifies the end of the reset sequence. The DDR controller performs external DRAM memory reset and initialization as per the JEDEC specification, including reset, refresh, and mode registers.

The following figure shows the FDDR reset sequence.

Figure 2-3. FDDR Reset Sequence

2.6.1.1 DDRIO Calibration (Ask a Question)

Each DDRIO has an ODT feature, which is calibrated depending on the DDR I/O standard. DDR I/O calibration occurs after the DDR I/Os are enabled. If the impedance feature is enabled, impedance can be programmed to the desired value in three ways:

- Calibrate the ODT/driver impedance with a calibration block (recommended)
- Calibrate the ODT/driver impedance with fixed calibration codes
- Configure the ODT/driver impedance to the desired value directly

The I/O calibration is always enabled when the DDR subsystem is configured for DDR2 and DDR3 memories.

The I/O calibration can be disabled or enabled using the DDR configurator when the DDR subsystem is configured for LPDDR memories.

> **Important:** If I/O calibration is enabled, all I/Os in the DDR bank are calibrated even though the DDR controller is not using all I/Os in the bank.

The FDDR_IO_CALIB_CR register can be configured for changing the ODT value to the desired value. For more information on DDR I/O calibration, see Configurable ODT and Driver Impedance section of the I/Os chapter in the UG0445: IGLOO2 FPGA and SmartFusion2 SoC FPGA Fabric User Guide.

Figure 2-4. Reset Sequence
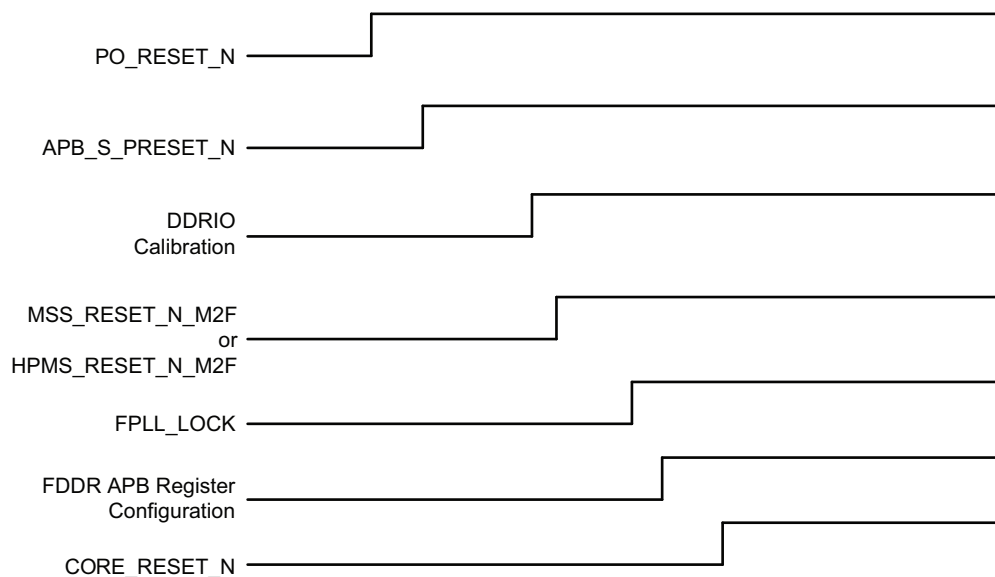
2.6.2 ZQ Calibration (Ask a Question)

ZQ calibration is applicable for DDR3 only. This is used to calibrate DRAM output drivers (R_{ON}) and On-Die Termination (ODT) values. DDR3 SDRAM needs a longer time to calibrate R_{ON} and ODT at initialization and a relatively smaller time to perform periodic calibrations.

The DDR controller performs ZQ calibration by issuing a ZQ calibration long (ZQCL) command and ZQ calibration short (ZQCS) command.

ZQCL is used to perform initial calibration during the power-up initialization sequence. This command is allowed for a period of t_{ZQinit}, as specified by memory vendor. The value of t_{ZQinit} can be modified through register bits REG_DDRC_T_ZQ_LONG_NOP, Table 1-59.

The ZQCS command is used to perform periodic calibration to account for voltage and temperature variations. A shorter timing window is provided to perform calibration and transfer of values as defined by timing parameter t_{ZQCS}. The t_{ZQCS} parameter can be modified through register bits REG_DDRC_T_ZQ_SHORT_NOP, Table 1-60.

Other activities are not performed by the controller for the duration of t_{ZQinit} and t_{ZQCS}. All DRAM banks are precharged and t_{RP} met before ZQCL or ZQCS commands are issued by the DDR controller.

2.6.2.1 DRAM Training (Ask a Question)

High-Speed DDR3 memories typically require the DDR controller to implement Write-Leveling, Read DQS Gate Training, and Read Data Eye Training. However, FDDR only supports a maximum data rate of 333 MHz/667 Mbps, which means the clock period and data window are relatively large compared to high-speed DDR3 memory interfaces. Therefore dynamic write-leveling and read training are not performed. The following sections describe how write-leveling and read training are addressed by the FDDR.

2.6.2.1.1 Write Leveling (Ask a Question)

Dynamic write-leveling is not required for the FDDR controller. The board-layout needs to follow AN4153: Board and Layout Design Guidelines for SmartFusion2 and IGLOO2 FPGAs to keep the skew between DQS and CK within the JEDEC DDR3 tDQSS limit of ±750 ps at each memory device.

2.6.2.1.2 Read Leveling (Ask a Question)

FDDR does not perform dynamic Read DQS Gate Training and Data Eye Training. Instead, these functions are achieved by using built-in static delay values automatically generated by Libero SoC for the FDDR automatic register initialization.

Read Gate

The DQS gate is aligned by the Libero SoC auto-generated FDDR initialization code containing fixed delay ratios to account for board round-trip time between FPGA and the DDR3 memory. The TMATCH_OUT and TMATCH_IN signals are shorted close to the FPGA balls to remove the FPGA output and input delays from the round trip delay time. Therefore, the fixed delay ratios represent only the board delay.

The fixed delay ratios work in combination with board layouts which follow the SmartFusion2/IGLOO2 Board Design Guidelines (refer AN4153: Board and Layout Design Guidelines for SmartFusion2 SoC and IGLOO2 FPGAs).

2.6.2.1.3 DQS Alignment within Data Eye (Ask a Question)

The incoming read DQS is internally centered within the read DQ data window using a static delay ratio. This static delay is applied by the Libero SoC auto-generated FDDR initialization code. The fixed delay ratios work in combination with board layouts which follow the SmartFusion2/IGLOO2 Board Design Guidelines (refer AN4153: Board and Layout Design Guidelines for SmartFusion2 SoC and IGLOO2 FPGAs).

2.6.2.2 DDR Memory Initialization Time (Ask a Question)

The time to initialize the DDR memory depends on the following factors:

- Power-up and register initialization by system controller. It depends on the Power-on Reset delay configuration in the Libero project (**Project** > **Project Settings** > **Device settings**).
- DDR controller and PHY configuration registers initialization. In SmartFusion 2 devices, the Cortex-M3 initializes these registers. In IGLOO 2 devices, the ConfigMaster in the FPGA fabric initializes these registers.
- DDR memory initialization by the DDR Controller according to the JEDEC standard (mode register configuration and training).
- DDR memory settling time configured in the System Builder memory configuration window.

Microchip

2.6.3 Details of Operation (Ask a Question)

This section provides a functional description of each block in the FDDR subsystem, as shown in Figure 2-5.

2.6.3.1 Clock Controller (Ask a Question)

The FDDR subsystem has a dedicated clock controller for generating aligned clocks to all the FDDR sub-blocks for correct operation and synchronous communication with user logic in the FPGA fabric. The base clock (FDDR_SUBSYSTEM_CLK) for the FDDR comes from a fabric CCC or an external source through the FPGA fabric. The FDDR clock controller is associated with a dedicated PLL (FPLL) for clock synthesis and de-skewing the internal DDR_FIC clock from the base clock.

The FDDR clock controller consists of an FPLL and fabric alignment clock controller (FACC).

2.6.3.1.1 FPLL (Ask a Question)

The FDDR_SUBSYSTEM_CLK from the FPGA fabric is used as a reference clock to the FPLL, and is multiplied to generate a clock frequency of up to 333 MHz. The FDDR_SUBSYSTEM_CLK can be generated from a fabric CCC/PLL, one of the on-chip oscillators, or directly from multi-standard user I/Os (MSIO) through FPGA fabric.

The supplies required to power the FPLL are the device core supply (VDD) for the digital section and the analog supply (FDDR_PLL_VDDA) for analog section. The required voltage for the FDDR_PLL_VDDA is 2.5V or 3.3V, based on the power supply availability on the board. The analog power supply voltage (2.5V or 3.3V) does not impact the FPLL frequency range. See DS0128: IGLOO2 and SmartFusion2 Datasheet or the FPLL operational range and characteristics.

The FPLL generates a lock signal (FPLL_LOCK) to indicate that the FPLL is locked onto the FDDR_SUBSYSTEM_CLK signal. The precision of the FPLL_LOCK discrimination can be adjusted using the lock window controls. The lock window represents the phase error window for lock assertion. The lock window can be adjusted between 500 parts per million (ppm) and 32,000 ppm in powers of 2. The integration of the lock period can be adjusted using a built-in lock counter. The lock counter or lock delay indicates the number of reference clock cycles to wait after the FPLL is locked for asserting the FPLL_LOCK signal. The lock delay is useful for avoiding false toggling of the FPLL lock signal. The lock counter can be configured between 32 and 32,768 cycles in multiples of 2.

There are two interrupts to indicate FPLL lock assertion and deassertion.

2.6.3.1.2 FACC (Ask a Question)

Within the FDDR clock controller, the FACC is responsible for interfacing with the FPLL, generating the aligned clocks required by the FDDR subsystem, and controlling the alignment of FPGA fabric interface clocks.

The clocks generated by the FACC are as follows:

- FDDR_CLK clocks the FDDR subsystem. FDDR_CLK can be operated up to 333 MHz, depending on the type of DDR present in the system.
- FDDR_SUBSYSTEM_CLK clocks the DDR_FIC, and defines the frequency at which the connected FPGA fabric subsystem is intended to operate.
- The possible FDDR_CLK:DDR_ FIC_CLK ratios are 1:1, 2:1, 3:1, 4:1, 6:1, 8:1, 12:1, and 16:1.

The FACC includes no-glitch multiplexers (NGMUXs) to feed the DDR_FIC clock with a standby clock (CK_STANDBY) during the FPLL initialization. During initialization, the FDDR is not operational until after FPLL lock is achieved. However, the glitch-free multiplexers are still used to ensure that the clock being driven to DDR_FIC during this time comes from the RC oscillator, avoiding the potentially high frequency output of the FPLL, which may be outside of the supported range of operation of DDR_FIC.

2.6.3.1.3 FPLL Initialization (Ask a Question)

In order to attain clock alignment between the FPGA fabric and the FDDR subsystem, it is necessary to use the FPLL to perform de-skewing of the FDDR clocks. After the FPLL is initialized, it typically

takes over 500 divided reference clock cycles for lock to be achieved. The FPLL lock assertion time is also dependent on the FPLL lock parameters (lock window and lock delay). There is no provision made for operation of the FDDR subsystem before FPLL lock is achieved.

2.6.3.1.4 PLL Lock Monitoring (Ask a Question)

The FDDR has an input, CLK_BASE_PLL_LOCK, to monitor the fabric PLL lock. It must be connected to the lock signal generated by the fabric PLL which is being used to generate the base clock to the FDDR.

Within the FDDR subsystem, there are two interrupts related to the PLL lock. A lock interrupt, indicating FPLL lock achieved, and an FPLL lock lost interrupt. Each of these two interrupts has a corresponding interrupt enable bit in the FDDR subsystem registers. It is also possible to read the state of the two PLL lock signals through the FDDR registers.

In the event of loss of FPLL lock, even though its output is not exactly in phase lock with the reference, the FPLL still generates a clock. User logic in the FPGA fabric can use the FPLL_LOCK signal to prevent communication with the FDDR subsystem during this time.

2.6.3.2   DDR_FIC (Ask a Question)

The following illustration shows the DDR_FIC block diagram.

Figure 2-5. DDR_FIC Block Diagram
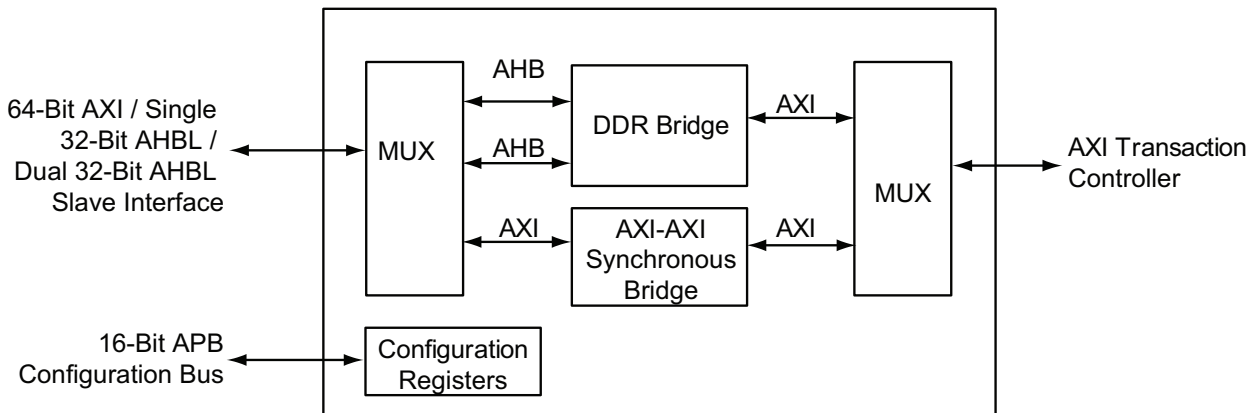
Fabric masters can access the FDDR subsystem in the following ways:

- Single AXI-64 interface
- Single AHB-32 interface
- Dual AHB-32 bit interfaces

If the AXI-64 interface is selected, the DDR_FIC acts as an AXI to AXI synchronous bridge and also supports locked transactions. During locked transactions a user configurable 20-bit down counter keeps track of the duration of the locked transfer. If the transfer is not completed before the down counter reaches zero, a single clock cycle pulse interrupt is generated to the fabric interface.

If single or dual AHB-32 interfaces are selected, the DDR_FIC converts the single or dual 32-bit AHBL master transactions from the FPGA fabric to 64-bit AXI transactions. The DDR bridge, which is embedded as part of the DDR_FIC, is enabled in this case. The DDR bridge has an arbiter that uses a round robin priority scheme on read and write requests from the two AHB masters. Refer to the 3. DDR Bridge for a detailed description.

The DDR_FIC input interface is clocked by the FPGA fabric clock and the AXI transaction controller is clocked by FDDR_CLK from the FDDR clock controller. Clock ratios between FDDR_CLK and DDR_FIC clock can vary. Supported ratios are shown in the following table. Clock ratios can be configured through Libero System-on-Chip (SoC) software or through the FDDR_FACC_DIVISOR_RATIO register.

Table 2-8. FDDR_CLK to FPGA Fabric Clock Ratios

DIVISOR_A[1:0]	DDR_FIC DIVISOR[2:0]	FDDR_CLK: FPGA FABRIC Clock Ratio
00	000	1:1
00	001	2:1
00	010	4:1
00	100	8:1
00	101	16:1
01	000	2:1
01	001	4:1
01	010	8:1
01	100	16:1
11	000	3:1
11	001	6:1
11	010	12:1

2.6.3.3 AXI Transaction Controller (Ask a Question)

The AXI transaction controller receives 64-bit AXI transactions from DDR_FIC and translates them into DDR controller transactions. The following illustration shows the block diagram of the AXI transaction controller interfaced with the DDR controller.

Figure 2-6. AXI Transaction Controller Block Diagram
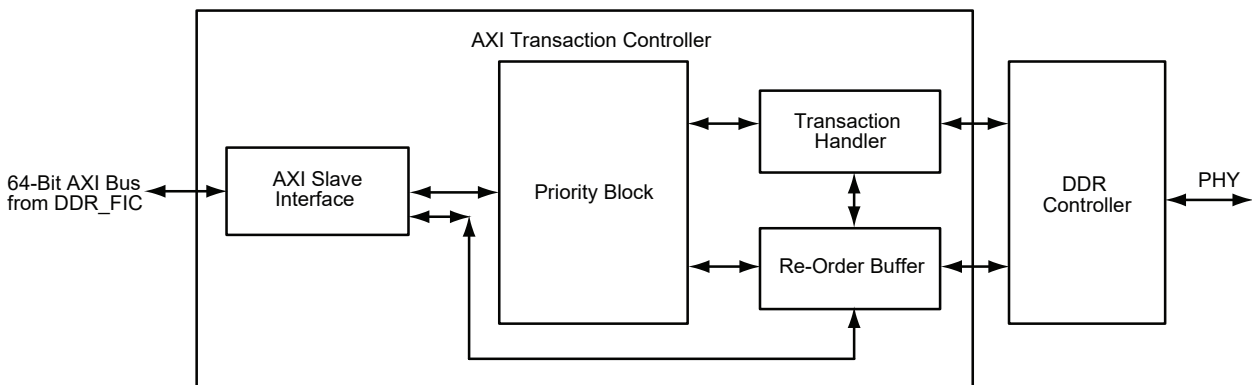
The AXI transaction controller comprises four major blocks:

- AXI slave interface
- Priority block
- Transaction handler
- Reorder buffer

2.6.3.3.1 AXI Slave Interfaces (Ask a Question)

The AXI transaction controller has a 64-bit AXI slave interface from DDR_FIC. The AXI slave port is 64 bits wide and is in compliance with the standard AXI protocol. Each transaction has an ID related to the master interface. Transactions with the same ID are completed in order, while the transactions with different read IDs can be completed in any order, depending on when the instruction is executed by the DDR controller. If a master requires ordering between the transactions, the same ID should be used.

The AXI slave interface has individual read and write ports. The read port queues read AXI transactions and it can hold up to four read transactions. The write port handles only one write transaction at a time and generates the handshaking signals on the AXI interface.

2.6.3.3.2 Priority Block (Ask a Question)

The priority block prioritizes AXI read/write transactions and provides control to the transaction handler. AXI read transactions have higher priority. The fabric master through DDR_FIC can be programmed to have a higher priority by configuring the PRIORITY_ID and PRIORITY_ENABLE_BIT bit fields in the DDRC_AXI_FABRIC_PRI_ID_CR register, Table 1-75.

2.6.3.3.3 Transaction Handler (Ask a Question)

The transaction handler converts AXI transactions into DDR controller commands. The transaction handler works on one transaction at a time from the read/write port queue that is selected by the priority block. The transaction handler has a write command controller and read command controller for write and read transactions.

The write command controller fetches the command from the AXI slave write port and sends a pure write instruction to the DDR controller. If SECDED is enabled, a read modified write (RMW) instruction is sent to the DDR controller. The read command controller generates read transactions to the DDR controller.

2.6.3.3.4 Reorder Buffer (Ask a Question)

The reorder buffer receives data from the DDR controller and orders the data as requested by the AXI master when a single AXI transaction is split into multiple DDR controller transactions, depending on the transfer size.

2.6.3.4   DDR Controller (Ask a Question)

The DDR controller receives requests from the AXI transaction controller, performs the address mapping from system addresses to DRAM addresses (rank, bank, row, and column) and prioritizes requests to minimize the latency of reads (especially high priority reads) and maximize page hits. It also ensures that DRAM is properly initialized, all requests are made to DRAM legally (accounting for associated DRAM constraints), refreshes are inserted as required, and the DRAM enters and exits various power-saving modes appropriately. The following illustration shows the DDR controller connections in the FDDR subsystem.

Figure 2-7. DDR Controller Block Diagram
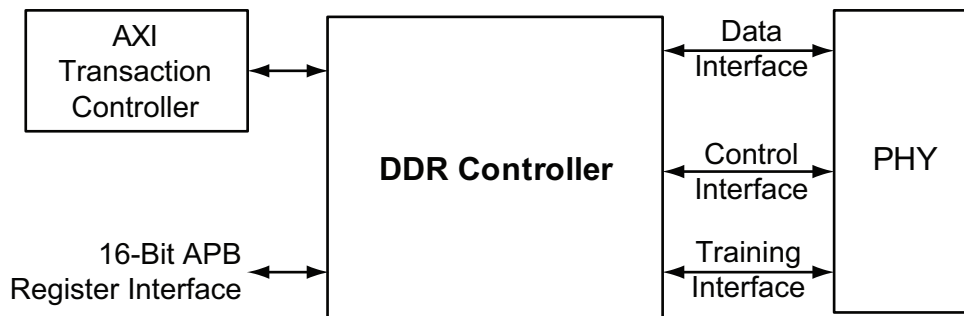
The following sections describe key functions of the DDR controller.

2.6.3.4.1 Address Mapping (Ask a Question)

Read and write requests to the DDR controller requires a system address. The controller is responsible for mapping this system address with rank, bank, row, and column address to DRAM.

The address mapper maps linear request addresses to DDR memory addresses by selecting the source bit that maps to each and every applicable DDR memory address bit. The address map interface registers can be configured to map source address bits to DRAM address (for more information, refer to 2.6.10. Address Mapping on configuring the FDDR features).

2.6.3.4.2 Transaction Scheduling (Ask a Question)

The DDR controller schedules the read and write transactions to DDR memory. The DDR controller classifies the transactions into three types, based on the commands from the AXI transaction controller:

![Microchip logo]

- Low priority reads (LPR)

- High priority reads (HPR)

- Writes (WR)

Each type of transaction has a queue and the queued transactions can be in normal state or in critical state. The transactions in a queue moves from normal state to critical state when that transaction is not serviced for a count of MAX_STARVE_X32 clocks. The MAX_STARVE_X32 values for each queue can be configured using the DDR controller performance registers (refer to the 1.5.5.10. Performance). The DDR controller completes the critical transactions with high priority.

2.6.3.4.3 Write Combine (Ask a Question)

The DDR controller combines multiple writes to the same address into a single write to DDR memory. When a new write collides with the queued write, the DDR controller overwrites the data for the queued write with that from the new write and only performs one write transaction. The write combine functionality can be disabled by setting the register bit REG_DDRC_DIS_WC to 1 (Table 1-54).

2.6.3.4.4 SECDED (Ask a Question)

The DDR controller supports built-in SECDED capability for correcting single-bit errors and detecting dual-bit errors. The SECDED feature can be enabled. When SECDED is enabled, the DDR controller adds 8 bits of SECDED data to every 64 bits of data.

When SECDED is enabled, a write operation computes and stores a SECDED code along with the data, and a read operation reads and checks the data against the stored SECDED code.

The SECDED bits are interlaced with the data bits, as shown in the following table.

Table 2-9. SECDED DQ Lines at DDR

Mode	SECDED Data Pins	
	M2S050/M2GL050 (FG896)	M2S150/M2GL150 (FC1152)
Full bus width mode	FDDR_DQ_ECC[3:0]	FDDR_DQ_ECC[3:0]
Half bus width mode	FDDR_DQ_ECC[1:0]	FDDR_DQ_ECC[1:0]
Quarter bus width mode	—	FDDR_DQ_ECC[0]

When the controller detects a correctable SECDED error, it does the following:

- Generates an interrupt signal which can be monitored by reading the interrupt status register, DDRC_ECC_INT_SR (Table 1-98). The FDDR also generates ECCINT interrupt signal, which can be monitored from FPGA fabric.

- Sends the corrected data to the read requested MSS/HPMS and FPGA fabric master as part of the read data.

- Sends the SECDED error information to the DDRC_LCE_SYNDROME_1_SR register, Table 1-86.

- Performs a read-modify-write operation to correct the data present in the DRAM.

When the controller detects an uncorrectable error, it does the following:

- Generates an interrupt signal that can be monitored by reading the interrupt status register DDRC_ECC_INT_SR, Table 1-98. The FDDR also generates an ECC_INT interrupt signal, which can be monitored from FPGA fabric.

- Sends the data with error to the read requested MSS/HPMS and FPGA fabric master as part of the read data.

- Sends the SECDED error information to the DDRC_LUE_SYNDROME_1_SR register, Table 1-79.

The following SECDED Registers in Table 1-26 can be monitored for identifying the exact location of an error in the DDR SDRAM.

- DDRC_LUE_ADDRESS_1_SR and DDRC_LUE_ADDRESS_2_SR gives the row/bank/column information of the SECDED unrecoverable error.
- DDRC_LCE_ADDRESS_1_SR and DDRC_LCE_ADDRESS_2_SR gives the row/bank/column information of the SECDED error correction.
- DDRC_LCB_NUMBER_SR indicates the location of the bit that caused the single-bit error in the SECDED case (encoded value).
- DDRC_ECC_INT_SR indicates whether the SECDED interrupt is because of a single-bit error or double-bit error. The interrupt can be cleared by writing zeros to DDRC_ECC_INT_CLR_REG, Table 1-99.

2.6.3.4.5 Power Saving Modes (Ask a Question)

The DDR controller can operate DDR memories in three power saving modes:

- Precharge power-down (DDR2, DDR3, LPDDR1)

If power-down is enabled in the System Builder FDDR configuration or REG_DDRC_POWERDOWN_EN = 1 (Table 1-30), the DDR controller automatically keeps DDR memory in Precharge power-down mode when the period specified by the power down entry time or REG_DDRC_POWERDOWN_TO_X32 register (Table 1-57) has passed, while the controller is idle (except for issuing refreshes).

The controller automatically performs the precharge power-down exit on any of the following conditions:

- A refresh cycle is required to any rank in the system.
- The controller receives a new request from the core logic.
- REG_DDRC_POWERDOWN_EN is set to 0.
- Self refresh (DDR2, DDR3, LPDDR1)

The DDR controller keeps the DDR memory devices in Self-refresh mode whenever the REG_DDRC_SELFREF_EN register bit (Table 1-28) is set and no reads or writes are pending in the controller.

The DDR controller can be programmed to issue single refreshes at a time (REG_DDRC_REFRESH_BURST = 0, see Table 1-29) to minimize the worst-case impact of a forced refresh cycle. It can be programmed to burst the maximum number of refreshes allowed for DDR (REFRESH_BURST = 7, for performing 8 refreshes at a time) to minimize the bandwidth lost when refreshing the pages.

The controller takes the DDR memory out of Self-refresh mode whenever the REG_DDRC_SELFREF_EN input is deasserted or new commands are received by the controller.

When the DDR self-refresh is enabled, the DDR I/O bank may go into recalibration and a glitch may occur in the MDDR bank I/Os, which are being used for general purpose rather than for the DDR memory. The DDR I/Os ODT is periodically calibrated and will be effected only when the I/Os are in tri-state (DDR I/Os are tri-stated only in self-refresh mode.

- Deep power-down (LPDDR1)

This is supported only for LPDDR1. The DDR controller puts the DDR SDRAM devices in Deep Power-down mode whenever the REG_DDRC_DEEPPOWERDOWN_EN bit (Table 1-30) is set and no reads or writes are pending in the DDR controller.

The DDR controller automatically exits Deep Power-down mode and reruns the initialization sequence when the REG_DDRC_DEEPPOWERDOWN_EN bit is reset to 0. The contents of DDR memory may be lost upon entry into Deep Power-down mode.

2.6.3.4.6 DRAM Initialization (Ask a Question)

After Reset, the DDR controller initializes DDR memories through an initialization sequence, depending on the type of DDR memory used. For more information on the initialization process, refer to the JEDEC specification.

2.6.4   FDDR Subsystem Features Configuration (Ask a Question)

The FDDR subsystem registers must be initialized before accessing DDR memory through the FDDR subsystem. When using the System Builder flow through Libero SoC all of the necessary registers are initialized automatically by the resulting module. This section provides the registers features of the FDDR. All registers are listed with their bit definitions in the 1.11. MDDR Configuration Registers section.

2.6.5   Memory Type (Ask a Question)

DDRC_MODE_CR (Table 1-31) must be configured to select the memory type (DDR2, DDR3, or LPDDR1) to access memory from the FDDR subsystem.

2.6.6   Bus Width Configurations (Ask a Question)

The FDDR supports various bus widths, as listed in the following table. The FDDR can be programmed to work in full, half, or quarter Bus width mode by configuring the DDRC_MODE_CR (Table 1-31) and PHY_DATA_SLICE_IN_USE_CR registers (Table 1-100) when the controller is in soft reset.

Table 2-10. Supported Bus Widths

Bus Width	M2S050/M2GL050 (FG896)	M2S150/M2GL150 (FC1152)
Full bus width	√	√
Half bus width	√	√
Quarter bus width	—	√

2.6.7   Burst Mode (Ask a Question)

The DDR controller performs burst write operations to DDR memory, depending on the Burst mode selection. Burst mode is selected as sequential or interleaving by configuring REG_DDRC_BURST_MODE to 1 or 0 (Table 1-69).

Burst length can be selected as 4, 8, or 16 by configuring REG_DDRC_BURST_RDWR (Table 1-63).

Supported burst modes for DDR SDRAM types and PHY widths are given in the following table. For M2GL050, only sequential Burst mode and a burst length of 8 is supported.

Table 2-11. Supported Burst Modes for M2S150 and M2GL150

Bus Width	Memory Type	Sequential/Interleaving		
		4	8	16
32	LPDDR1	√	√	—
	DDR2	√	√	—
	DDR3	—	√	—
16	LPDDR1	—	√	√
	DDR2	—	√	—
	DDR3	—	√	—
8	LPDDR1	—	√	—
	DDR3	—	√	—
	DDR2	—	√	—

MICROCHIP

2.6.8 Configuring Dynamic DRAM Constraints (Ask a Question)

Timing parameters for DDR memories must be configured according to the DDR memory specification. Dynamic DRAM constraints are subdivided into three basic categories:

- Bank constraints affect the transactions that are scheduled to a given bank
- Rank constraints affect the transactions that are scheduled to a given rank
- Global constraints affect all transactions

2.6.9 Dynamic DRAM Bank Constraints (Ask a Question)

The timing constraints which affect the transactions to a bank are listed in the following table. The control bit field must be configured as per the DDR memory vendor specification.

Table 2-12. Dynamically Enforced Bank Constraints

Timing Constraint of DDR Memory	Control Bit	Description
Row cycle time (tRC)	REG_DDRC_T_RC, Table 1-44	Minimum time between two successive activates to a given bank.
Row precharge command period (tRP)	REG_DDRC_T_RP, Table 1-51	Minimum time from a precharge command to the next command affecting that bank.
Minimum bank active time (tRAS(min))	REG_DDRC_T_RAS_MIN, Table 1-48	Minimum time from an activate command to a precharge command to the same bank.
Maximum bank active time (tRAS(max))	REG_DDRC_T_RAS_MAX, Table 1-48	Maximum time from an activate command to a precharge command to the same bank.
RAS-to-CAS delay (tRCD)	REG_DDRC_T_RCD, Table 1-51	Minimum time from an activate command to a Read or Write command to the same bank.
Write command period (tWR)	REG_DDRC_WR2PRE, Table 1-46	Minimum time from a Write command to a precharge command to the same bank.
Read-to-precharge delay (tRTP)	REG_DDRC_RD2PRE, Table 1-46	Minimum time from a Read command to a precharge command to the same bank. Set this to the current value of additive latency plus half of the burst length.

2.6.9.1 Dynamic DRAM Rank Constraints (Ask a Question)

The timing constraints which affect the transactions to a rank are listed in the following table. The control bit field must be configured as per the DDR memory vendor specification.

Table 2-13. Dynamically Enforced Bank Constraints

Timing Constraints of DDR Memory	Control Bit	Description
Nominal refresh cycle time (tRFC(nom) or tREFI)	REG_DDRC_T_RFC_NOM_X32, Table 1-29	Average time between refreshes for a given rank. The actual time between any two refresh commands may be larger or smaller than this; this represents the maximum time allowed between refresh commands to a given rank when averaged over a large period of time.
Minimum refresh cycle time tRFC(min)	REG_DDRC_T_RFC_MIN, Table 1-28	Minimum time from refresh to refresh or activate.
RAS-to-RAS delay (tRRD)	REG_DDRC_T_RRD, Table 1-51	Minimum time between activates from bank A to bank B.
RAS-to-CAS delay (tCCD)	REG_DDRC_T_CCD, Table 1-51	Minimum time between two reads or two writes (from bank A to bank B).
Four active window (tFAW)	REG_DDRC_T_FAW, Table 1-44	Sliding time window in which a maximum of 4 bank activates are allowed in an 8-bank design. In a 4-bank design, set this register to 0x1.

2.6.9.2 Dynamic DRAM Global Constraints (Ask a Question)

The timing constraints which affect global transactions are listed in the following table. The control bit field must be configured as per the DDR memory vendor specification.

Table 2-14. Dynamic DRAM Global Constraints

Timing Constraint	Control Bit	Description
Read-to-write turnaround time	REG_DDRC_RD2WR, Table 1-49	Minimum time to allow between issuing any Read command and issuing any WRITE command
Write-to-read turnaround time	REG_DDRC_WR2RD, Table 1-49	Minimum time to allow between issuing any Write command and issuing any Read command
Write latency	REG_DDRC_WRITE_LATENCY, Table 1-45	Time after a Write command that write data should be driven to DRAM.

The DDR memories require delays after initializing the mode registers. The following registers must be configured for delay requirements for the DDR memories. The DDR controller uses these delay values while initializing the DDR memories.

- DDRC_CKE_RSTN_CYCLES_1_CR (Table 1-38), recommended value is 0x4242
- DDRC_ CKE_RSTN_CYCLES_2_CR (Table 1-39), recommended value is 0x8

2.6.10 Address Mapping (Ask a Question)

The DDR controller maps linear request addresses to DDR memory addresses by selecting the source bit that maps to each and every applicable DDR memory address bit.

Each DDR memory address bit has an associated register vector to determine its source. The source address bit number is determined by adding the internal base of a given register to the programmed value for that register, as described in the following equation.

[Internal base] + [register value] = [source address bit number]

For example, reading the description for REG_DDRC_ADDRMAP_COL_B3, the internal base is 3; so when the full data bus is in use, the column bit 4 is determined by 3 + [register value].

If this register is programmed to 2, then the source address bit is: 3 + 2 = 5.

The DDR configurator assigns values to the address mapping registers depending on the selected number of Columns, Rows and Banks. The following illustration provides the default mapping of the memory row, bank, and column address to the user interface address domain.

Figure 2-8. Address Mapping

The following are the address mapping registers:

- DDRC_ADDR_MAP_BANK_CR, Table 1-32
- DDRC_ADDR_MAP_COL_1_CR, Table 1-33
- DDRC_ADDR_MAP_COL_2_CR, Table 1-34
- DDRC_ADDR_MAP_COL_3_CR, Table 1-54
- DDRC_ADDR_MAP_ROW_1_CR, Table 1-35
- DDRC_ADDR_MAP_ROW_2_CR, Table 1-36

While configuring the registers, ensure that two DDR memory address bits are not determined by the same source address bit.

> **Important:** Some registers map multiple source address bits (REG_DDRC_ADDRMAP_ROW_B0_11)

To arrive at the right address for the DDR controller, the system address or AXI address bits [4:0] are mapped by the FDDR.

- In Full Bus Width mode, the system address bits [4:0] are used to map the lower column address bits (C0, C1, C2).
- In Half Bus Width mode, the system address bits [4:0] are used to map the lower column address bits (C0, C1, C2, C3).
- In quarter bus width mode, the system address bits [4:0] are used to map the lower column address bits (C0, C1, C2, C3, C4).

The FDDR configurator uses {Row, Bank, Column} address mapping as shown in the following example.

2.6.10.1 Example (Ask a Question)

In this example, the Address map registers are configured to access a 512 MB DDR3 SDRAM memory (MT41J512M8RA) from the FDDR subsystem as shown in 2.8.2. Example 2: Connecting 32-Bit DDR3 to FDDR_PADs with SECDED. The 512M x 8-bit DDR3 memory module has 3 bank address lines, 16 rows, and 10 columns.

- The column address bits 3 to 9 are mapped for system address bit[5] to system address bit[11]. To map the column 3-bit (C3) to address [5], the field is configured to 3, as the base value is 2. Similarly, the other column address bits are configured:
    - DDRC_ADDR_MAP_COL_1_CR = 0x3333
    - DDRC_ADDR_MAP_COL_2_CR = 0x3FFF
    - DDRC_ADDR_MAP_COL_3_CR = 0x3300
- The bank address bits 0 to 2 are mapped for system address bit[12] to system address bit[14]. To map the bank bit0 to address [12], the field is configured to A, as the base value is 2. Similarly, the other bank address bits are configured:
    - DDRC_ADDR_MAP_BANK_CR = 0xAAA
- The row address bits 0 to 15 are mapped for system address bit[15] to system address bit[27]. To map the bank bit0 to address [15], the field is configured to 9, as the base value is 6. Similarly, the other bank address bits are configured:
    - DDRC_ADDR_MAP_ROW_1_CR = 0x9999
    - DDRC_ADDR_MAP_ROW_2_CR = 0x9FF

> **Important:** The FDDR can access the 2 GB address space (0x00000000 - 0x7FFFFFFF). But in this example, 512 MB (0x00000000 - 0x1FFFFFFF) DDR3 SDRAM is connected to the 16 address lines of FDDR. The memory visible in the other memory space is mirrored of this 512 MB memory.

2.6.10.2 DDR Mode Registers (Ask a Question)

After reset, the DDR controller initializes the mode registers of DDR memory with the values in the following registers. The mode registers must be configured according to the specification of the external DDR memory when the controller is in soft reset.

- DDRC_INIT_MR_CR, Table 1-40
- DDRC_INIT_EMR_CR, Table 1-41
- DDRC_INIT_EMR2_CR, Table 1-42
- DDRC_INIT_EMR3_CR, Table 1-43

The T_MOD and T_MRD bits in DDRC_DRAM_MR_TIMING_PARAM_CR (Table 1-47) must be configured to the required delay values. T_MOD and T_MRD are delays between loading the mode registers.

2.6.10.3 SECDED (Ask a Question)

To enable SECDED mode, set the REG_DDRC_MODE bits to 101 in DDRC_MODE_CR, Table 1-31. The PHY_DATA_SLICE_IN_USE_CR register (Table 1-113) must be configured to enable data slice 4 of the PHY.

The register value REG_DDRC_LPR_NUM_ENTRIES in the performance register, DDRC_PERF_PARAM_1_CR (Table 1-63), must be increased by 1 to the value used in Normal mode (without SECDED).

2.6.10.4 Read Write Latencies (Ask a Question)

The read and write latencies between DDR controller and DDR PHY can be configured. Configure the PHY_DATA_SLICE_IN_USE_CR register for adding latencies for read and writes (Table 1-113).

2.6.10.5 Performance (Ask a Question)

The DDR controller has several performance registers which can be used to increase the speed of the read and write transactions to DDR memory.

The DDR controller has a transaction store, shared for low and high priority transactions. The DDRC_PERF_PARAM_1_CR register (Table 1-63) can be configured for allocating the transaction store between the low and high priority transactions. For example, if the REG_DDRC_LPR_NUM_ENTRIES field (Table 1-63) is configured to 0, the controller allocates more time to high priority transactions. The ratio for LPR: HPR is 1:7 (as the transaction store depth is 8).

The DDRC_HPR_QUEUE_PARAM_1_CR (Table 1-64), DDRC_LPR_QUEUE_PARAM_1_CR (Table 1-66), and DDRC_WR_QUEUE_PARAM_CR (Table 1-68) registers can be configured for the minimum clock values for treating the transactions in the HPR, LPR, and WR queue as critical and non-critical.

To force all incoming transactions to low priority, configure the DDRC_PERF_PARAM_2_CR register (Table 1-69). By default, it is configured to force all the incoming transactions to low priority.

2.6.10.6 Refresh Controls (Ask a Question)

The DDR controller automatically issues refresh commands to DDR memory for every tRFC (min). The DDR controller can be programmed to issue single refreshes at a time (REG_DDRC_REFRESH_BURST = 0) TO MINIMIZE THE WORST-CASE IMPACT OF A FORCED REFRESH CYCLE. It can be programmed to burst the maximum number of refreshes allowed for DDR (REFRESH_BURST = 7, for performing 8 refreshes at a time) to minimize the bandwidth lost when refreshing the pages.

2.6.10.7 1T or 2T Timing (Ask a Question)

The DRAM can be used in 1T or 2T Timing mode by configuring the DDRC_PERF_PARAM_3_CR register (Table 1-70). The address bus can be clocked using 1T or 2T clocking. With 1T, the DDR controller can issue a new command on every clock cycle. In 2T timing the DDR controller will hold the address and command bus valid for two clock cycles. This reduces the efficiency of the bus to one command per two clocks, but it doubles the amount of setup and hold time. The data bus remains the same for all of the variations in the address bus, Default configuration is 1T timing mode.

2.6.10.8 ODT Controls (Ask a Question)

The ODT for a specific rank of memory can be enabled or disabled by configuring the DDRC_ODT_PARAM_1_CR (Table 1-52) and DDRC_ODT_PARAM_2_CR (Table 1-53) registers. These must be configured before taking the controller out of soft reset. They are applied to every read or write issued by the controller.

2.6.10.9 Soft Resets (Ask a Question)

Set the REG_DDRC_SOFT_RSTB bit of DDRC_DYN_SOFT_RESET_CR (Table 1-27) to 0 to reset the DDR controller. To release the DDR controller from reset, set the REG_DDRC_SOFT_RSTB bit of DDRC_DYN_SOFT_RESET_ALIAS_CR (Table 1-74) to 1.

2.7 How to Use FDDR in IGLOO 2 Devices (Ask a Question)

This section describes how to use FDDR in the IGLOO 2 devices. To configure the IGLOO 2 device features and then build a complete system, use the **System Builder** graphical design wizard in the Libero Software.

The following image shows the initial **System Builder** window where you can select the features that you require. For information on how to launch the **System Builder** wizard and how to use it, see the IGLOO2 System Builder User's Guide. You can also use CoreABC based initialization as described in the *IGLOO2 Standalone Peripheral Initialization User Guide*.

Figure 2-9. System Builder - Device Features Window
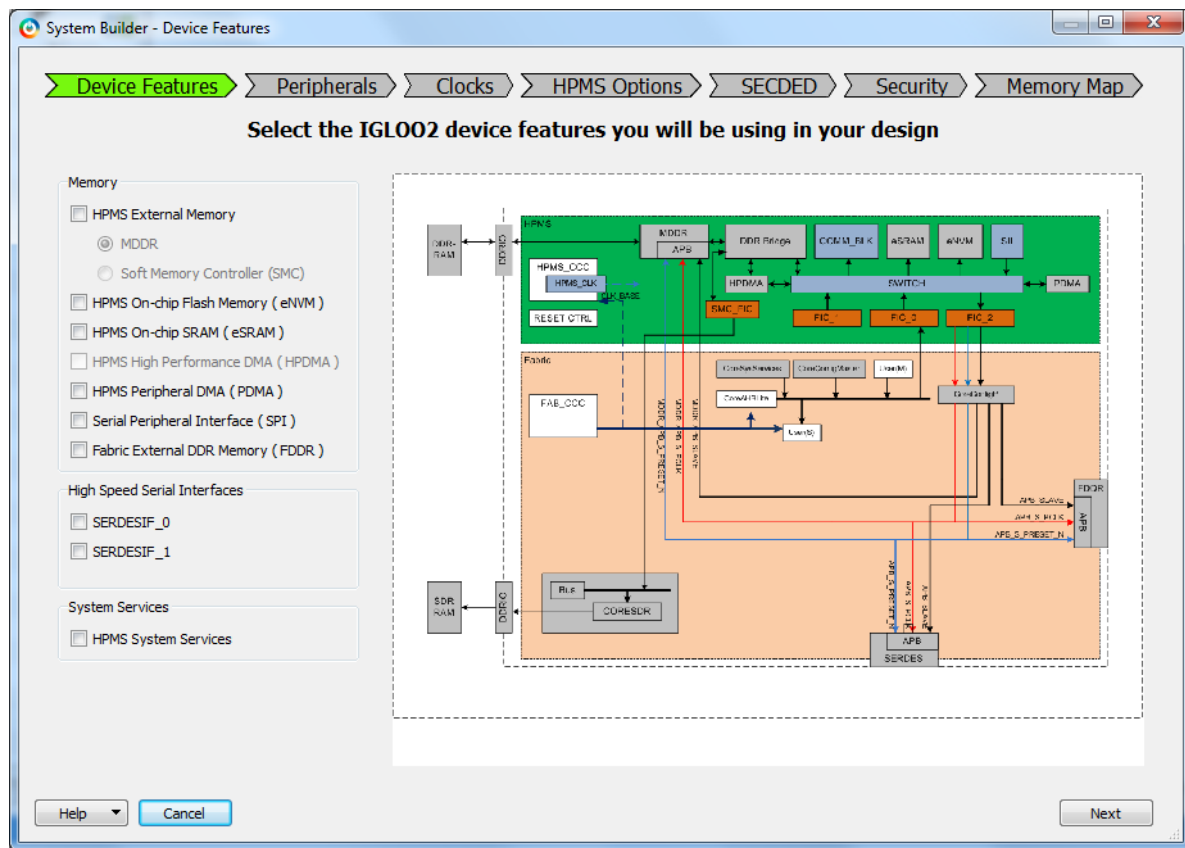
For more information about how to use MDDR in the SmartFusion 2 devices, see 2.10. Appendix A: How to Use the FDDR in SmartFusion 2 Devices.

2.7.1   Configuring FDDR (Ask a Question)

The following steps describe how to configure the FDDR:

1. Check the **Fabric External DDR Memory (FDDR)** check box under the **Device Features** tab and leave the other check boxes unchecked. The following image shows the **System Builder** > **Device Features**.

Figure 2-10. System Builder - Device Features Tab
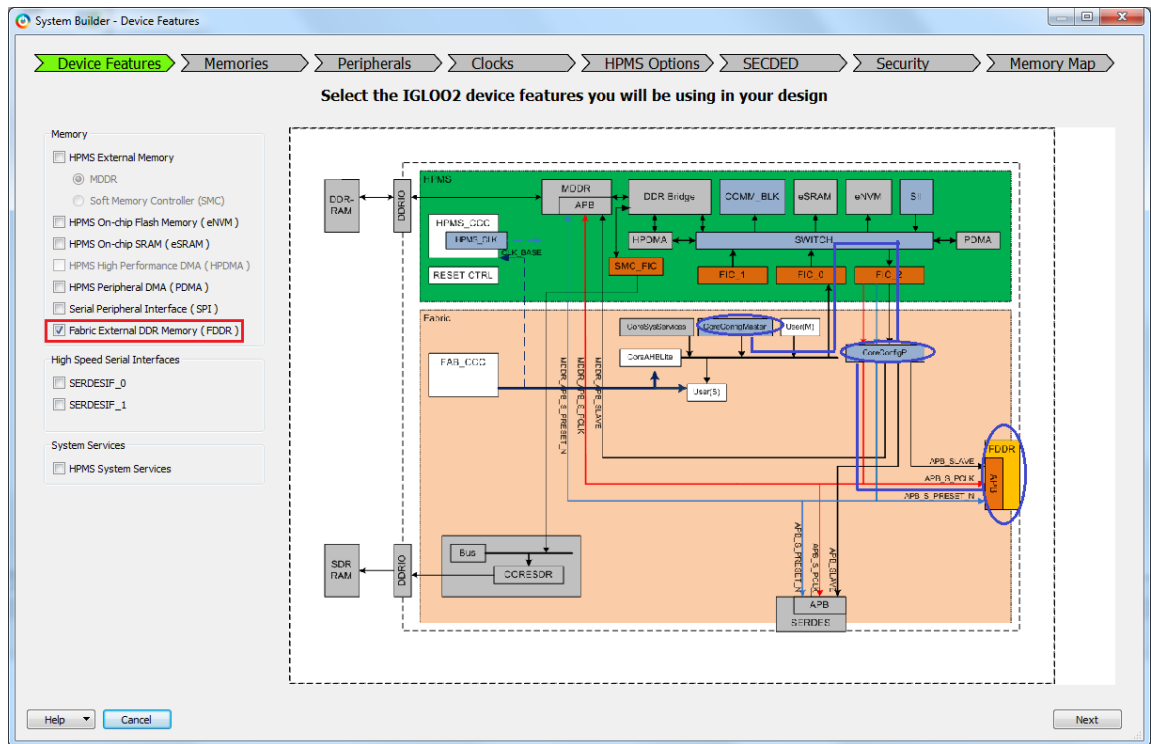
2. Selecting the **Fabric External DDR Memory (FDDR)** check box in the **System Builder** performs the following actions:

   – Instantiates the required IPs like CoreConfigMaster, CoreConfigP that initializes the FDDR Controller.

   – Establishes the initialization path:
   CoreConfigMaster → FIC_0 → eNVM → FIC_2 → CoreConfigP → APB bus of the FDDR subsystem.

   – CoreConfigMaster (AHB Master) accesses the DDR configuration data stored in eNVM through FIC_0.

   – The configuration data is sent to CoreConfigIP through the FIC_2 master port.

   – CoreConfigP sends the configuration data to APB bus of the FDDR subsystem.

3. Navigate to the **Memories** tab. Depending on the application requirement, select the memory settings under the **General** tab as shown in the following image.

   – Memory Type can be selected as DDR2, DDR3 or LPDDR.

   – The Data width can be selected as 32- bit, 16-bit, or 8-bit. See Table 1-11 for supported data widths for various IGLOO2 device packages.

   – The SECDED (ECC) can be enabled or disabled.

   – Address Mapping - The register settings to perform mapping to system address bits for various Row, Bank and Column combinations are automatically computed by the configurator using address mapping option. The following table shows the supported range for Row, Bank and Column.

Table 2-15. Supported Address Width Range for Row, Bank and Column

Width	DDR2	DDR3	LPDDR
Row Address	12-16	12-16	12-16
Bank Address	2-3	2-3	2-3
Column Address	9-12	9-12	9-12

- For more information, see 2.6.10. Address Mapping.
- Select the I/O Drive Strength as Half Drive Strength or Full Drive Strength, as shown in the following table. The DDR I/O standard is configured as listed in the following table based on this setting.

Table 2-16. DDR I/O Standard is Configured based on I/O Drive Strength Setting

I/O Drive Strength	Memory Type	
	DDR2	DDR3
Half Drive Strength	SSTL18I	SSTL15I
Full Drive Strength	SSTL18II	SSTL15II

Figure 2-11. Fabric DDR Memory Configuration
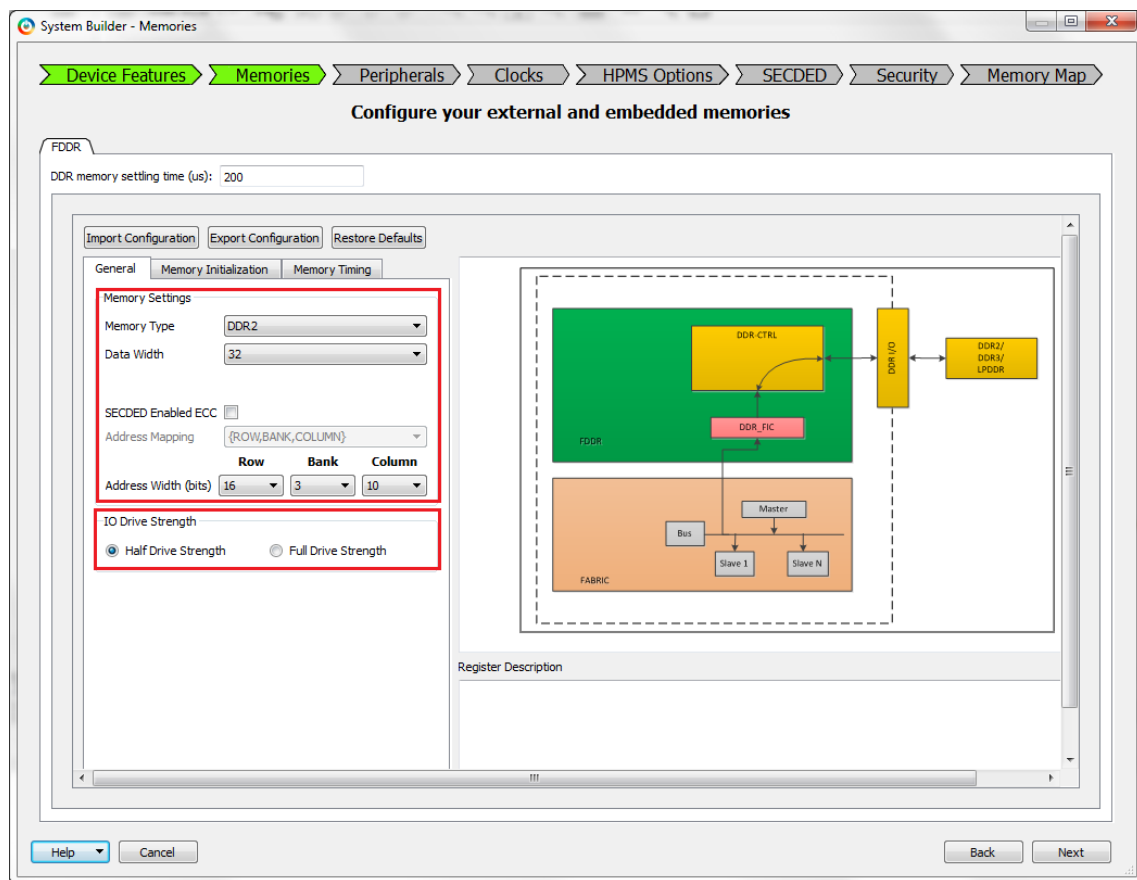
4. For only LPDDR memory, the I/O standard and I/O calibration settings are available as shown in the following image.
   - Select I/O standard as LVCMOS18 or LPDDRI.

> **Important:** If LVCMOS18 is selected, all I/Os are configured to LVCMOS1.8 except CLK/CLK_N.CLK and CLK_N are configured to LPDDRI standard as they are differential signals.

 – Select I/O calibration as ON or OFF. If I/O calibration is selected as ON, then the IGLOO2 FDDR_IMP_CALIB pin must be pulled down with a resistor. For resistor values, see the Impedance Calibration section in DS0124: IGLOO2 Pin Descriptions Datasheet.

Figure 2-12. Selecting I/O Standard as LVCMOS18 or LPDDRI
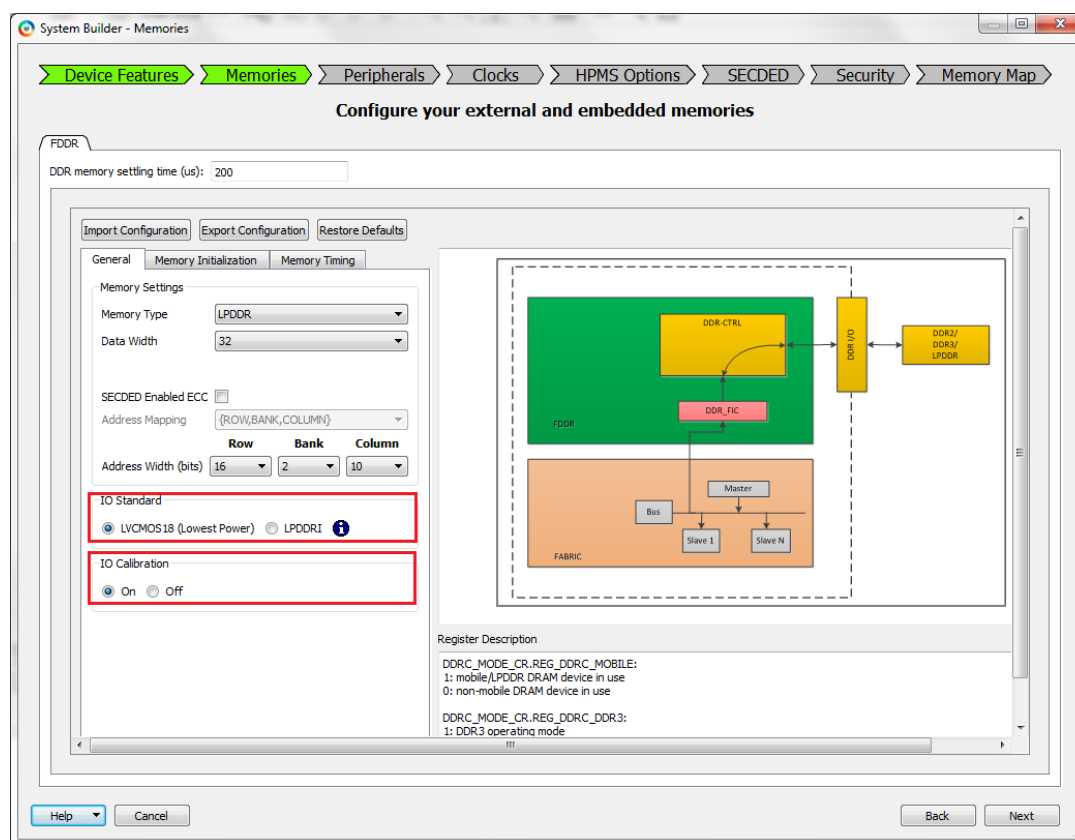
5. Depending on the application requirement, select the Memory Initialization settings under the **Memory Initialization** tab as shown in Figure 2-13.
    – Select the below performance related settings
        • Burst Length can be selected as 4, 8, or 16. See Table 1-12 for supported burst lengths.
        • Burst order can be selected as sequential or interleaved. See Table 1-12 for supported burst orders.
        • Timing mode can be selected as 1T or 2T. For more details see 2.6.10.7. 1T or 2T Timing.
        • CAS latency is the delay, in clock cycles, between the internal READ command and the availability of the first bit of output data. Select the CAS latency according to the DDR memory (Mode register) datasheet.
    – Select the below power saving mode settings. See 2.6.3.4.5. Power Saving Modes for more details.
        • Self-Refresh Enabled

- Auto Refresh Burst Count
- Power down Enabled
- Stop the clock: supported only for LPDDR
- Deep Power down Enabled: supported only for LPDDR
- Power down entry time

– Select the additional performance settings.

- Additive CAS Latency is defined by EMR[5:3] register of DDR2 memory and by MR1[4:3] register of DDR3 memory. It enables the DDR2 or DDR3 SDRAM to allow a READ or WRITE command from DDR Controller after the ACTIVATE command for the same bank prior to tRCD (MIN). This configuration is part of DDR2 Extended Mode Register and DDR3 Mode Register1.

- CAS Write Latency (CWL) is defined by DDR3 MR2[5:3] and is the delay, in clock cycles, from the releasing of the internal write to the latching of the first data in. The overall WRITE latency (WL) is equal to CWL + AL, where CWL is set to 5 clock cycles by default.

– Select the below ZQ Calibration settings for DDR3 memory. For more details refer 2.6.2. ZQ Calibration.

- Zqinit
- ZQCS
- ZQCS Interval

– Select other settings.

- Local ODT setting is not supported for LPDDR memory. For DDR2/DDR3 memory type, user can choose any option for "Local ODT". User can enable or disable "LOCAL ODT" during read transcation.

- Drive strength setting is defined by EMR[7:5] register bits of LPDDR memory with drop down options of **Full**, **Half**, **Quarter** and **One-eighth** drive strength, it is defined by EMR[1] register bit of DDR2 memory with drop down options of **Full** and **Weak** drive strength and it is defined by MR1 register bits M5 and M1 of DDR3 memory with drop down options of **RZQ/6** and **RZQ/7**.

- Partial array self-refresh coverage setting is defined by EMR[2:0] register bits of LPDDR memory with drop down options of **Full**, **Quarter**, **One-eighth**, and **One-sixteenth**. This feature helps in improving power savings during self-refresh by selecting the amount of memory to be refreshed during self-refresh.

- RTT (Nominal) setting is defined by EMR[6] and EMR[2] register bits of DDR2 memory which determines what ODT resistance is enabled with drop down options of `RTT disabled', '50 ohms', '75 Ω' and `150 Ω' and it is defined by MR1[9], MR1[6] and MR1[2] register bits of DDR3 memory. In DDR3 memory RTT nominal termination is allowed during standby conditions and WRITE operations and NOT during READ operations with drop down options of **RZQ/2**, **RZQ/4** and **RZQ/6**.

- RTT_WR (Dynamic ODT) setting is defined by MR2[10:9] register bits of DDR3 memory. This is applicable only during WRITE operations. If dynamic ODT (Rtt_WR) is enabled, DRAM switches from normal ODT (RTT_nom) to dynamic ODT (Rtt_WR) when beginning WRITE burst and subsequently switches back to normal ODT at the end of WRITE burst. The drop down options provided to the user are **off**, **RZQ/4**, and **RZQ/2**.

- Auto self-refresh setting is defined by MR2[6] register bit of DDR3 memory with drop down option of **Manual** and **Auto**. Self-refresh temperature setting is defined by MR2[7] register bit of DDR2 memory with drop down options of **Normal** and **Extended**.

Figure 2-13. Memory Initialization Configuration
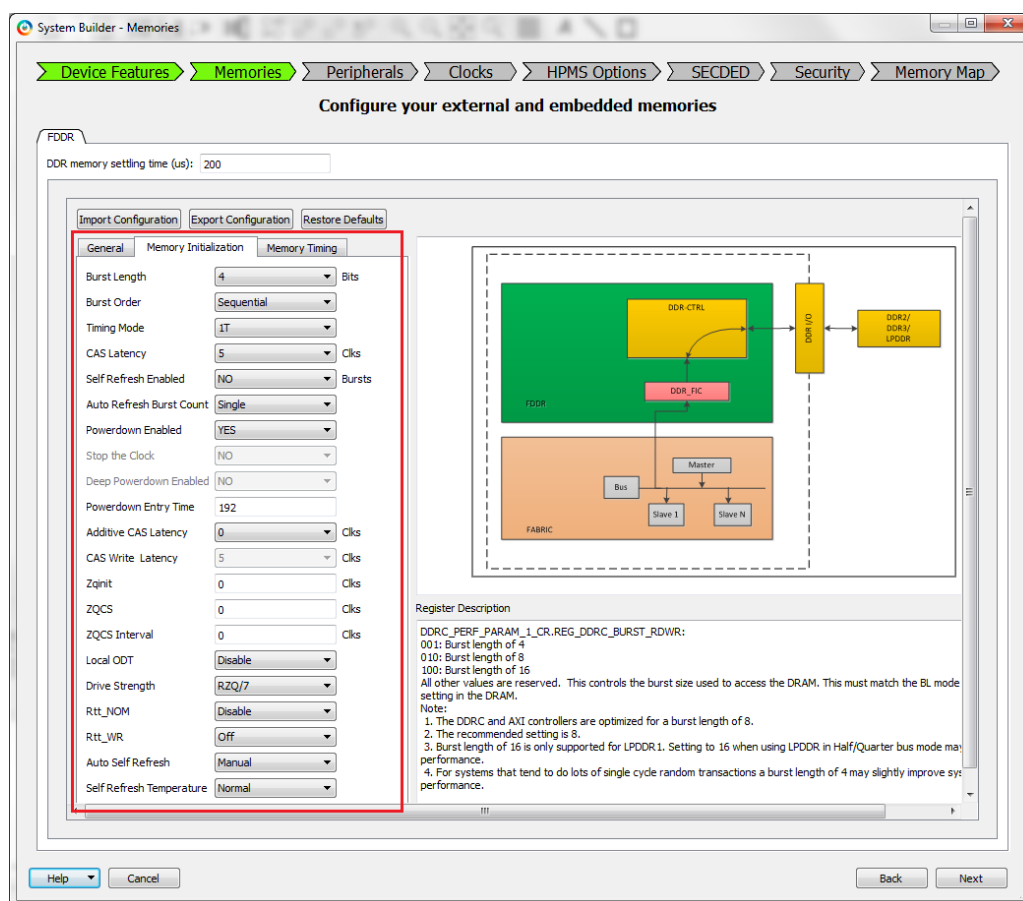
6. Select the memory timing settings under the **Memory Timing** tab according to the DDR memory vendor data sheet as shown in the following image. For more information, see 2.6.8. Configuring Dynamic DRAM Constraints section.

Figure 2-14. Memory Timing Configuration
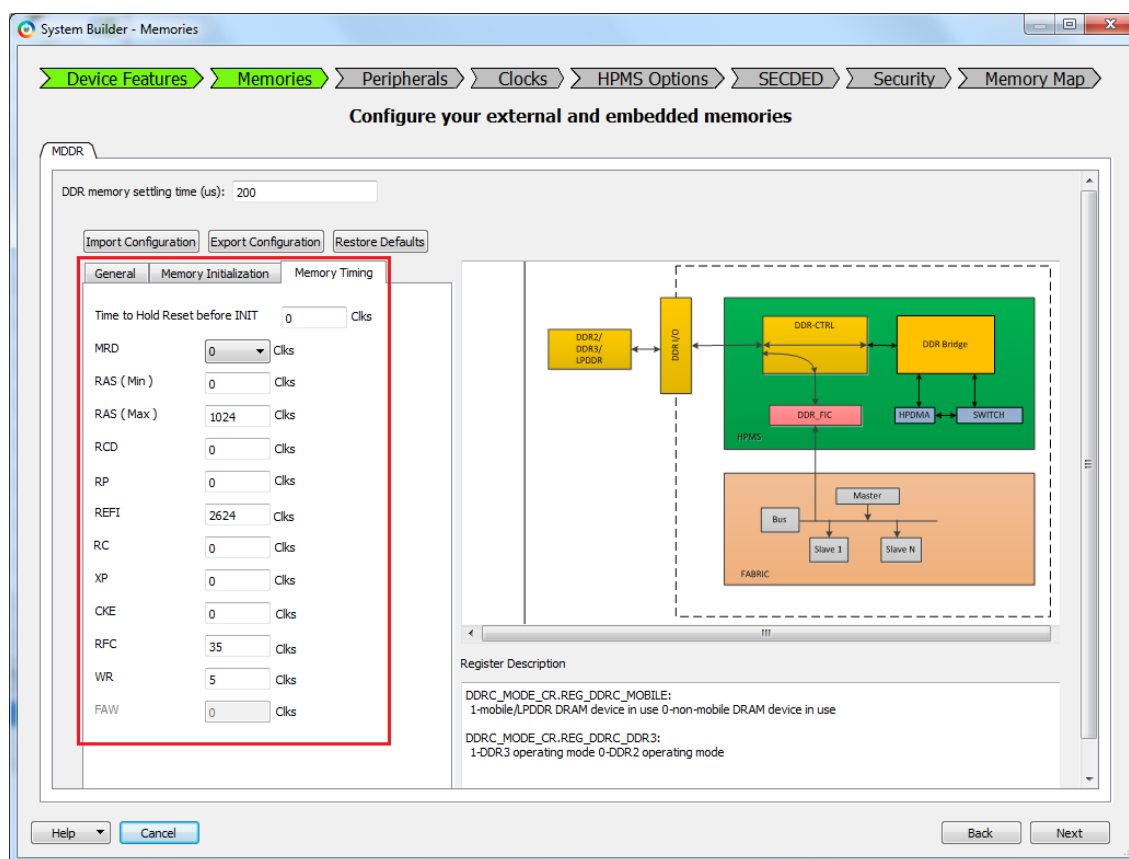
The configurator also provides the option to import and export the register configurations. The configuration settings are stored in eNVM. Configuration files for accessing LPDDR memory on IGLOO2 Evaluation kit can be downloaded from: www.microsemi.com/soc/documents/LPDDR_Emcraft_Config.zip.

An example of FDDR register configurations for operating the LPDDR memory (MT46H64M16LF) with clock 166 MHz is given below.

 – Device Memory Settling Time (us): 200

The DDR memories require settling time for the memory to initialize before accessing it. the LPDDR memory model MT46H64M16LF needs 200us settling time.

General

 – Memory Type: Select LPDDR

 – Data Width: 16

Memory Initialization

 – Burst length: 8

 – Burst Order: Interleaved

 – Timing Mode: 1T

 – CAS Latency: 3

 – Self Refresh Enabled: No

 – Auto Refresh Burst Count: 8

- PowerDown Enabled: Yes

- Stop the clock: No

- Deep PowerDown enabled: No

- No Activity clocks for Entry: 320

Memory Timing

- Time To Hold Reset Before INIT – 67584 clks

- MRD: 4 clks

- RAS (Min): 8 clks

- RAS (Max): 8192 clks

- RCD: 6 clks

- RP: 7 clks

- REFI: 3104 clks

- RC: 3 clks

- XP: 3 clks

- CKE: 3 clks

- RFC: 79 clks

- FAW: 0 clks
   Navigate to the **Peripherals** tab. The **Peripherals** tab allows to configure the Fabric AMBA
   Master and Fabric AMBA Slave required for the design. Drag and drop the required master/
   slave to the corresponding subsystem. The following image shows the Peripherals tab. Drag
   and drop the Fabric Master core to the Fabric DDR Subsystem. This allows to configure
   the type of interface as AXI, single AHB-Lite. On completing the configuration, the selected
   interface is enabled. The user logic in the FPGA fabric can access the DDR memory through
   the FDDR using these interfaces.

Figure 2-15. System Builder - Peripherals Tab
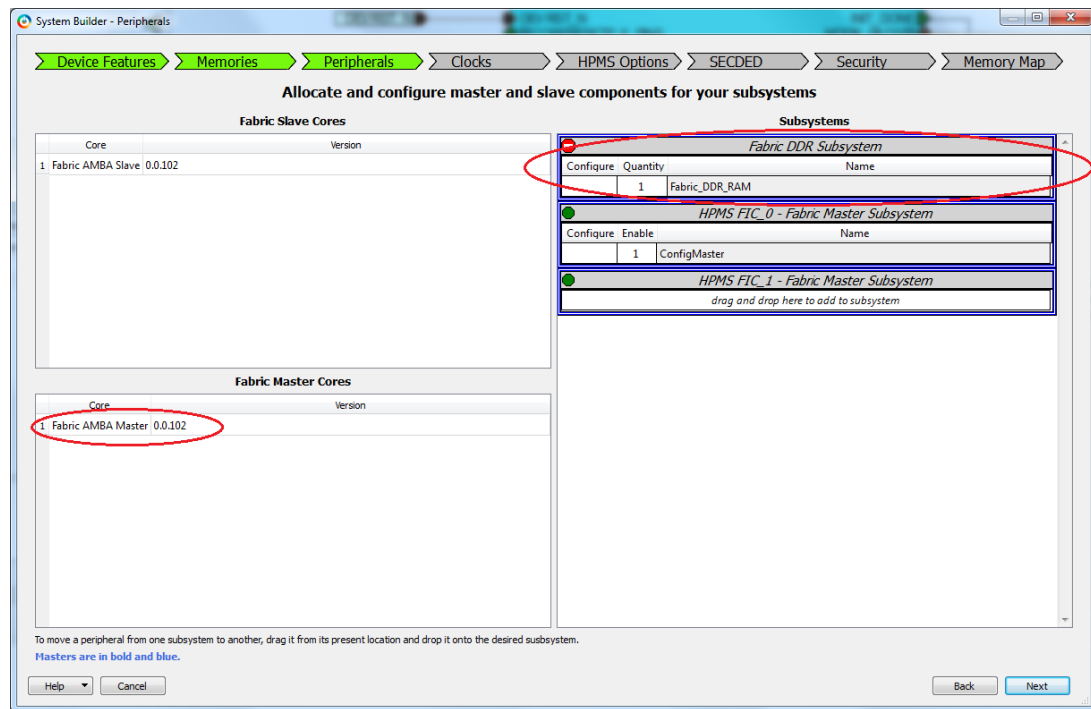
7. Click **Next** to navigate to the **Clocks** tab. The **Clocks** tab allows to configure the System Clock and subsystem clocks. The FDDR subsystem operates on FDDR_CLK frequency, which can be configured up to 333 MHz. The FDDR subsystem clock (CLK_BASE) can be configured as a ratio-1, 2, 3, 4, 6, 8, 12, or 16 of FDDR_CLK. The maximum frequency of FDDR subsystem clock is 200 MHz. FDDR subsystem clock has to be driven from the FPGA fabric. The following image shows the **System Builder** > **Clocks tab**.

Figure 2-16. FDDR Clock Configuration
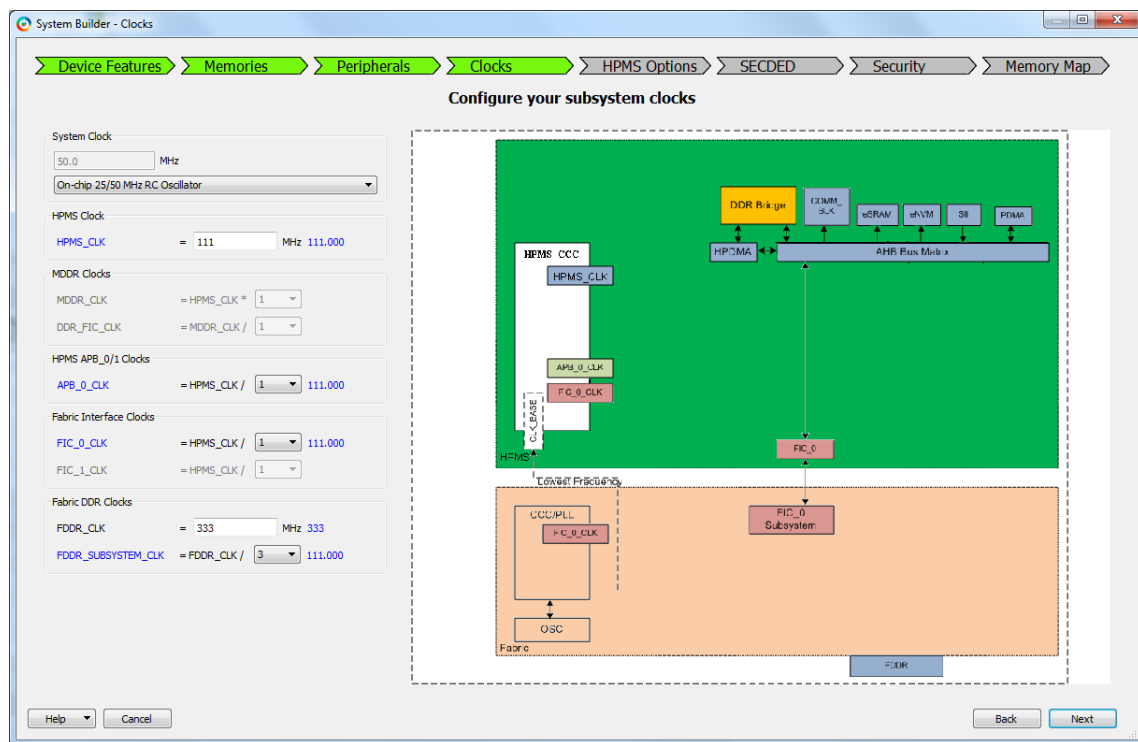
2.7.1.1 I/O Configuration (Ask a Question)

In the **I/O Editor** window, configure the I/O settings such as ODT and drive strength. The following image shows the **I/O Editor** window.

Figure 2-17. I/O Editor Window

2.7.2 Simulating the FDDR Subsystem (Ask a Question)

For simulating the FDDR subsystem, connect the FDDR DRAM interface to the DDR memory simulation model in a testbench. Use a vendor-specific memory simulation model that is JEDEC compliant.

Note: Libero SoC does not support the in-house memory simulation model.

2.7.3 Accessing FDDR from FPGA Fabric through the AXI Interface (Ask a Question)

The AXI master in the FPGA fabric can access the DDR memory through the FDDR subsystem. The following illustration shows the FDDR with the AXI interface. The FDDR registers are configured from the FPGA fabric using the CoreConfigMaster IP through the CoreConfigP IP APB interface.

Figure 2-18. FDDR Subsystem with AXI Interface
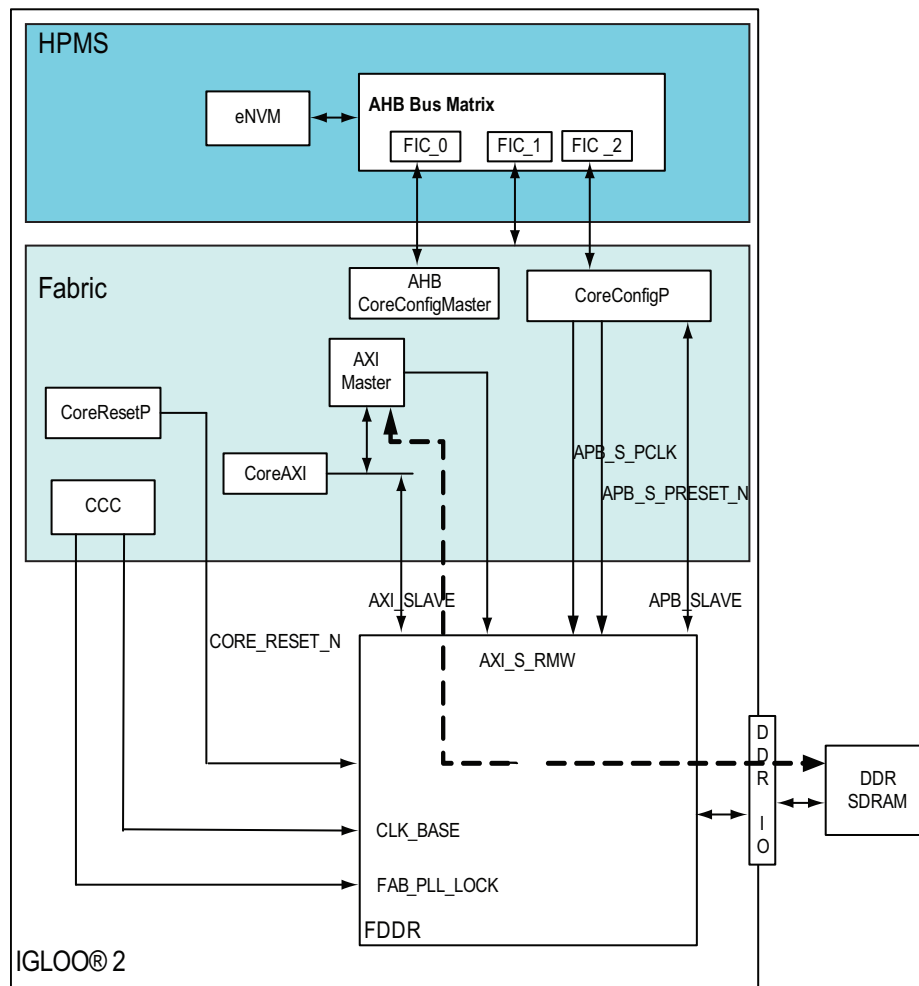
Read, write, and read-modify-write transactions are initiated by the AXI master to read from or write the data to the DDR memory after initializing the FDDR registers.

The following steps access the FDDR from the AXI master in the FPGA fabric:

1. Go to the **System Builder** - **Device Features** tab and check the **Fabric External DDR Memory (FDDR)** check box and leave the rest of the check boxes unchecked. The following image shows the **System Builder** - **Device Features** tab.

Figure 2-19. System Builder - Device Features Tab
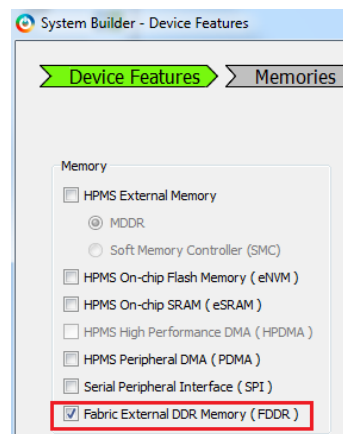
2. Configure the **HPMS External Memory** in **Memories** tab (shown in the following image). In this example, the design is created to access the DDR3 memory with a 32-bit data width and no ECC.

3. Set the **DDR memory settling time** to 200 µs and then click **Import Register Configuration**.

Figure 2-20. Memory Configuration
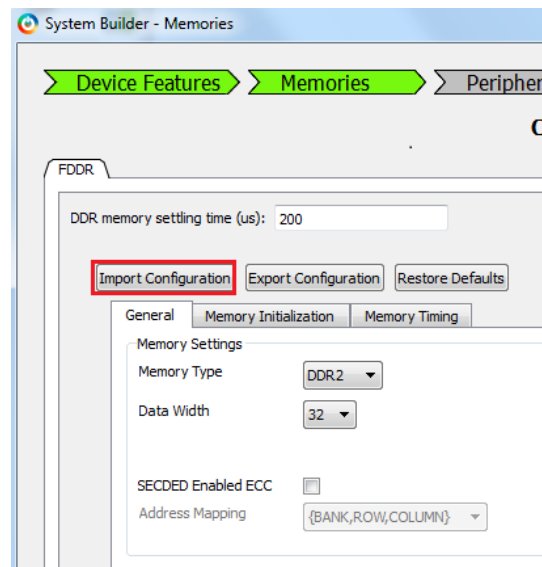
4. Navigate to the **Peripherals** tab. In the **Peripherals** tab, drag the **Fabric Master** Core and drop on to the **Fabric DDR Subsystem**. You can see that the master is added to the subsystem. The following image shows the **Peripherals** tab with the AMBA_MASTER_0 added.

5. Click the **Configure** icon to open the **AMBA_MASTER_0** dialog. The following image shows the **Peripherals** tab with the **Configure** icon highlighted.

Figure 2-21. Fabric DDR Subsystem Configuration Dialog
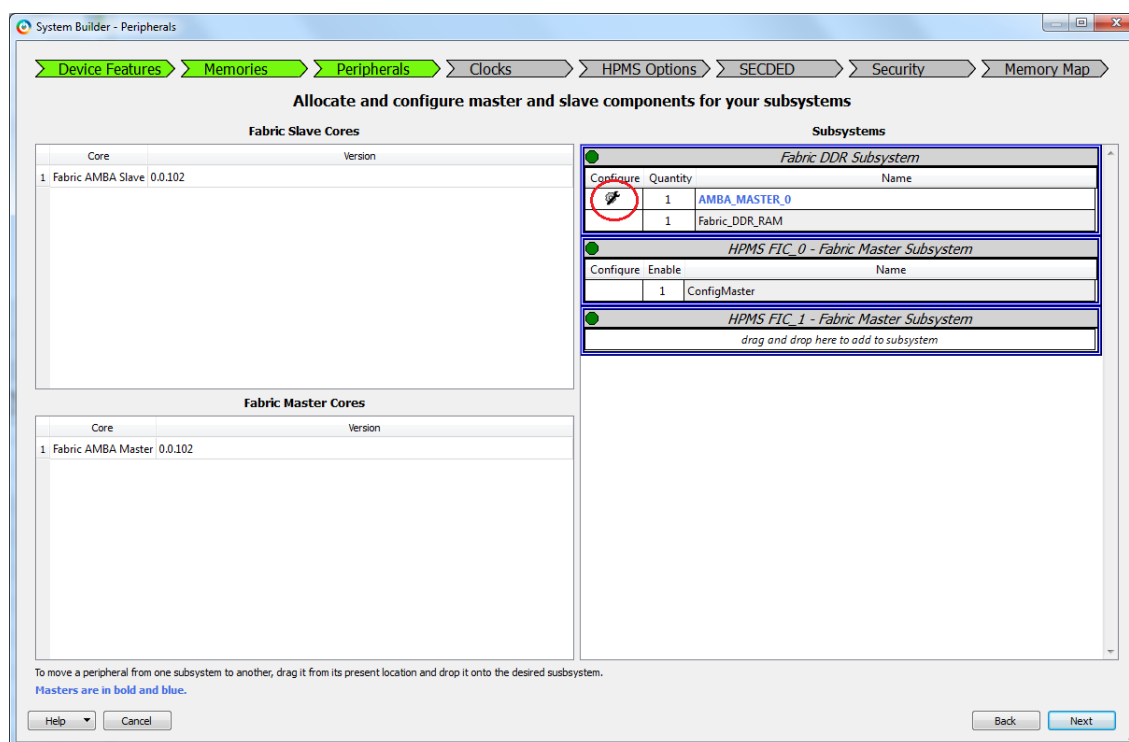
6. In the **Configuring AMBA_MASTER_0** dialog, select the **Interface Type** as **AXI** and then click **OK**.
    The following image shows the AMBA Master - Configuration dialog.

    Figure 2-22. AMBA Master Configuration
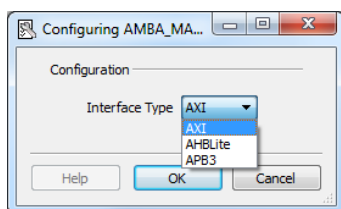
7. Configure the **System Clock** and **Subsystem** clocks in **Clocks** tab. The following image shows
    the **Clocks configuration** dialog.

Figure 2-23. Clocks Configuration
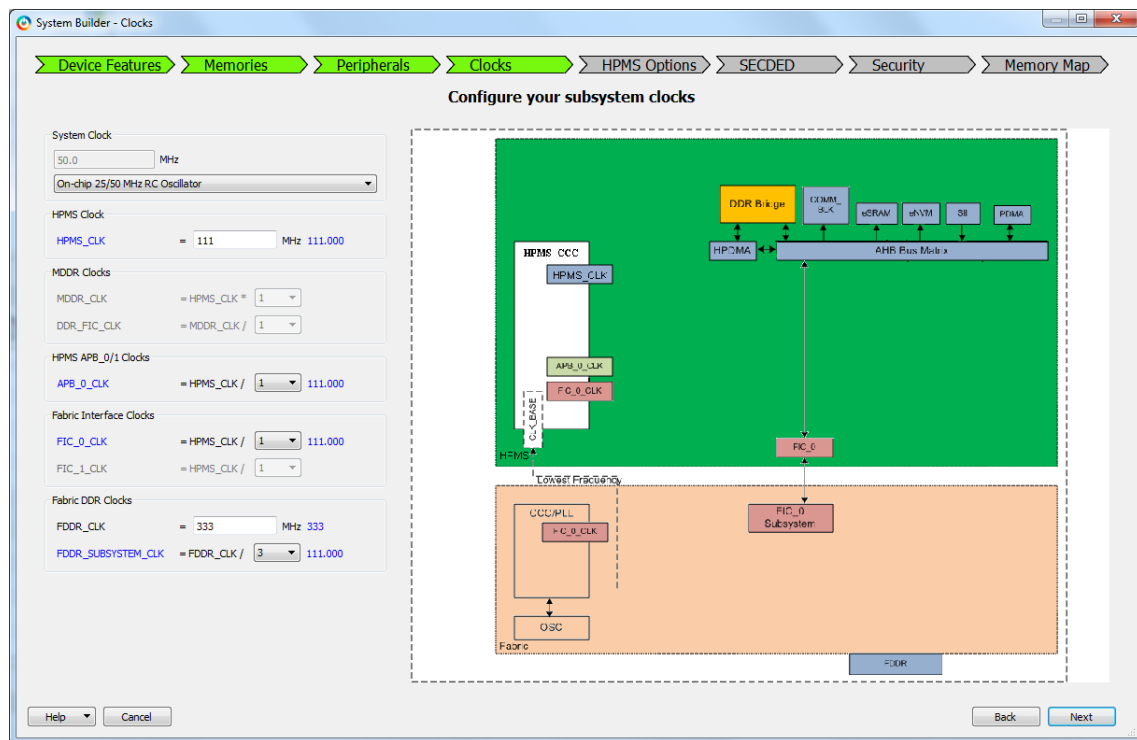
- – Select the On-chip 25/50 MHz RC Oscillator
- – Configure HPMS_CCC for HPMS_CLK, APB_0_CLK, and FIC_0_CLK.
  Configure FDDR_CLK as 333 MHz and FDDR_SUBSYSTEM_CLK as FDDR_CLK/3, that is, 111 MHz (need to drive this from the Fabric clock).
- – CLK to GL0 of FCCC_0 and FDDR_SUBSYSTEM_CLK
- – LOCK to LOCK of FCCC_0 and FDDR_SUBSYSTEM_LOCK
- – RESET_N to INIT_DONE of FDDR_AXI_0
- – AXI_S_RMW to FDDR_AXI_S_RMW of FDDR_AXI_0 block
8. Navigate to the **Memory Map** tab giving the required data in the rest of the **System Builder** tabs.
9. Instantiate the user AXI master logic in the SmartDesign canvas to access the FDDR through the AXI interface. Make sure that the AXI master logic accesses the FDDR after configuring the FDDR registers.
10. Instantiate the CCC block in the SmartDesign canvas and configure it to generate 111 MHz clock.
11. Connect the AXI_Master logic signals as follows:
- – CLK to GL0 of FCCC_0 and FDDR_SUBSYSTEM_CLK
- – LOCK to LOCK of FCCC_0 and FDDR_SUBSYSTEM_LOCK
- – RESET_N to INIT_DONE of FDDR_AXI_0
- – AXI_S_RMW to FDDR_AXI_S_RMW of FDDR_AXI_0 block

The following illustration shows the rest of the connections in the top level design.

Figure 2-24. SmartDesign Connections (Top-Level View)
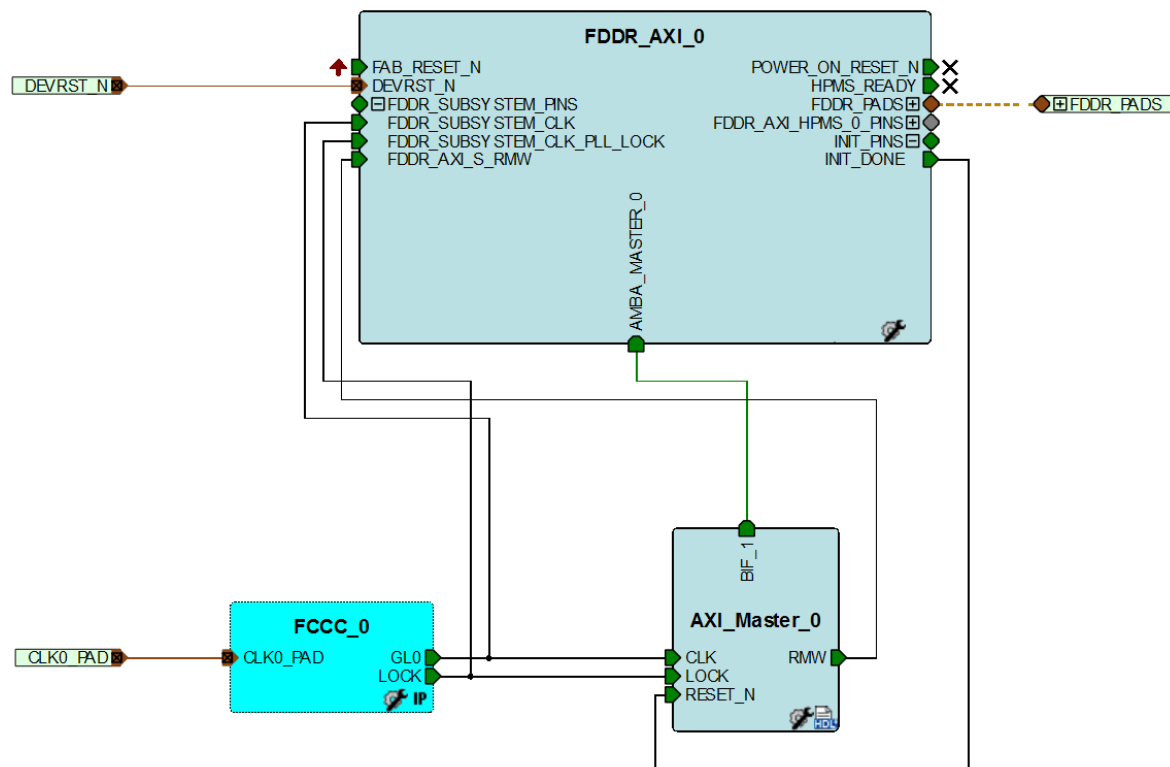
For FDDR AXI throughput, see AC422: SmartFusion2 - Optimizing DDR Controller for Improved Efficiency - Libero v11.7 Application Note.

2.7.4   Accessing FDDR from FPGA Fabric through the AHB Interface (Ask a Question)

The FDDR subsystem can be used to access the DDR memory using the AHB-Lite interface. The following illustration shows the FDDR with AHB-Lite interface.

Figure 2-25. FDDR with AHB-Lite interface
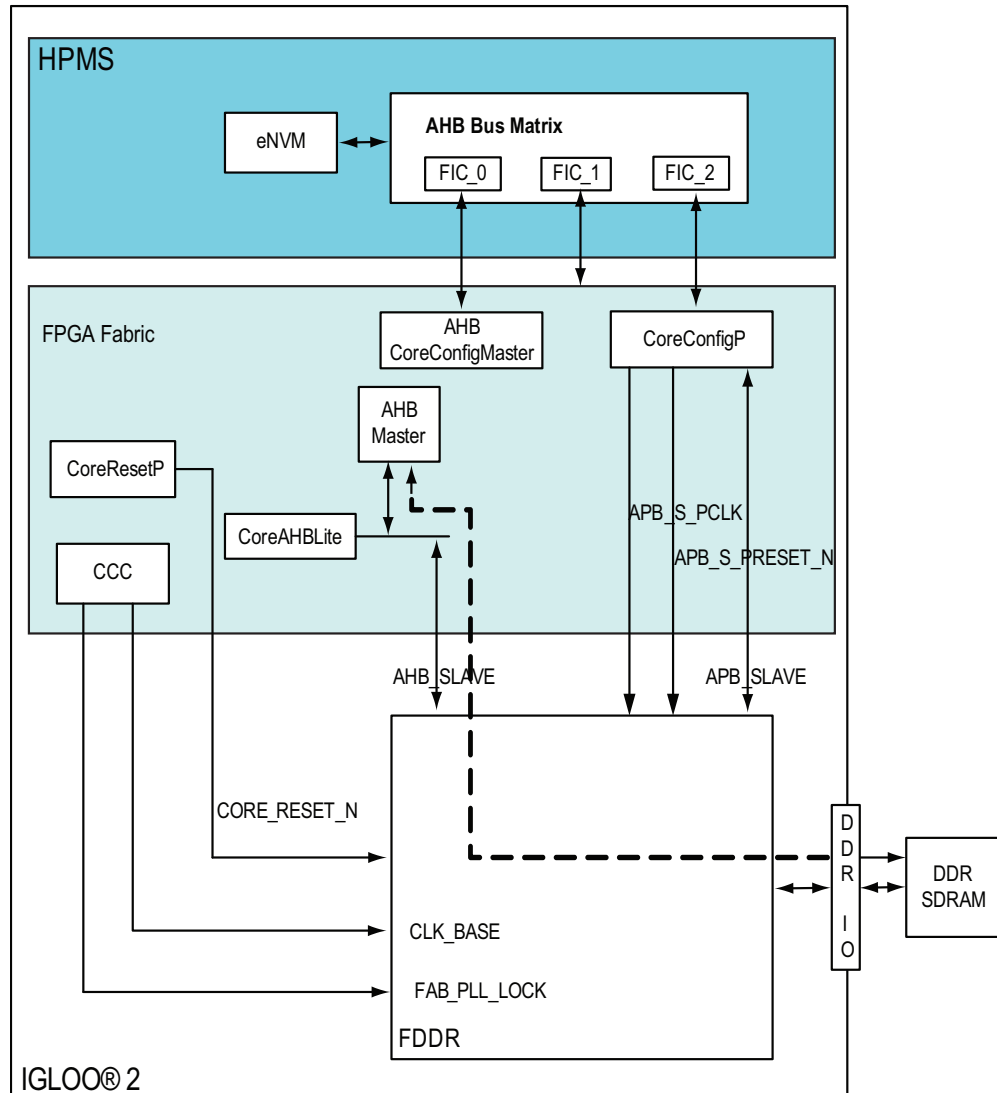
The procedure for accessing the FDDR from the AHB master in the FPGA fabric is the same as 2.7.3. Accessing FDDR from FPGA Fabric through the AXI Interface, except for the following:

- Configure the AMBA Master Interface Type as AHB-Lite in the Fabric DDR Subsystem in the Peripherals tab of the System Builder wizard.

Table 1-21 lists the FDDR throughput for the following configuration:

- Fabric Interface: AHB
- FDDR Mode: DDR3
- Fabric Clock to FDDR Clock Ratio: 1:4
- PHY Width: 16 and 32
- Clock Frequency: 80 MHz

The other parameters are configured similar to the FDDR configuration in AC422: SmartFusion2 - Optimizing DDR Controller for Improved Efficiency - Libero v11.7 Application Note.

Table 2-17. FDDR Throughput (for AHB)

FDDR-Fabric Interface-Memory	Frequency Ratio (CLK_BASE:FDDR_CLK)	PHY Width	Write Transaction BW (MB/sec)	Read Transaction BW (MB/sec)
FDDR_AHB-DDR3	1:4 80 MHz:320 MHz	PHY_16	80 MB	79 MB
		PHY_32	80 MB	79 MB

2.8 DDR Memory Device Examples (Ask a Question)

This section describes how to connect DDR memories to IGLOO2 FDDR_PADs with examples.

2.8.1 Example 1: Connecting 32-Bit DDR2 to FDDR_PADs (Ask a Question)

The following illustration shows DDR2 SDRAM connected to the FDDR of a IGLOO 2 device. Micron's MT47H64M16 is a 128 MB density device with x16 data width. The FDDR is configured in Full Bus Width mode and without SECDED. The total amount of DDR2 memory connected to the FDDR is 256 MB.

Figure 2-26. x16 DDR2 SDRAM Connected to FDDR
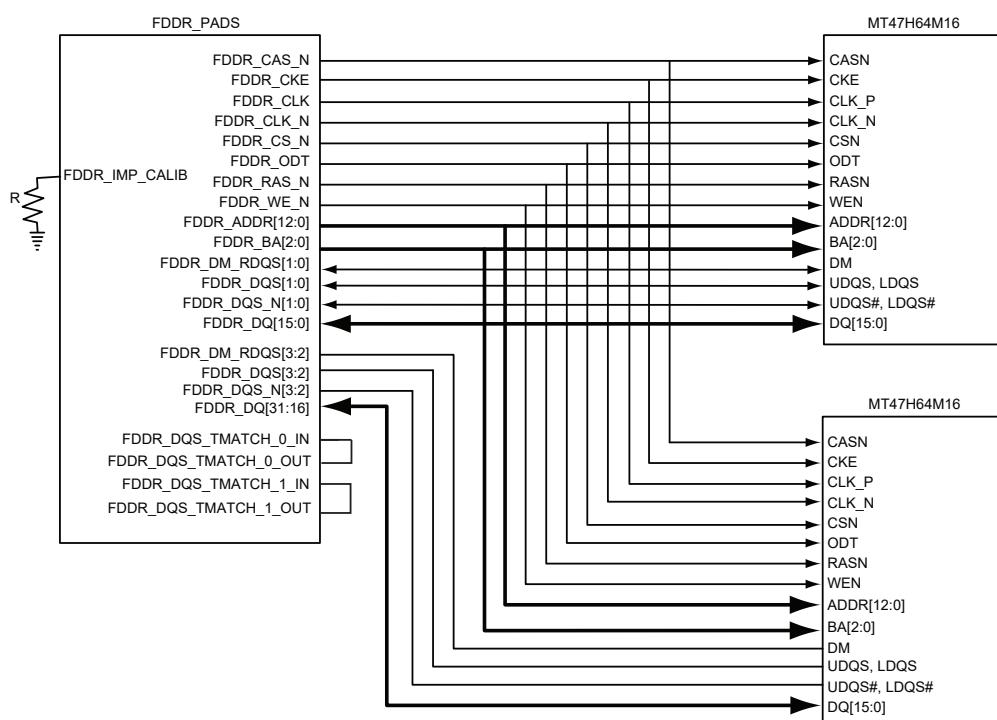
2.8.2 Example 2: Connecting 32-Bit DDR3 to FDDR_PADs with SECDED (Ask a Question)

The following illustration shows DDR3 SDRAM connected to the FDDR of a IGLOO 2 device. Micron's MT41J512M8RA is a 512 MB density device with x8 data width. The FDDR is configured in Full Bus Width mode with SECDED enabled. The SDRAM connected to FDDR_DQ_ECC[3:0] is used to store SECDED bits. The total amount of DDR3 memory (excluding memory for SECDED) connected to FDDR is 2 GB.

Figure 2-27. x8 DDR3 SDRAM Connection to FDDR
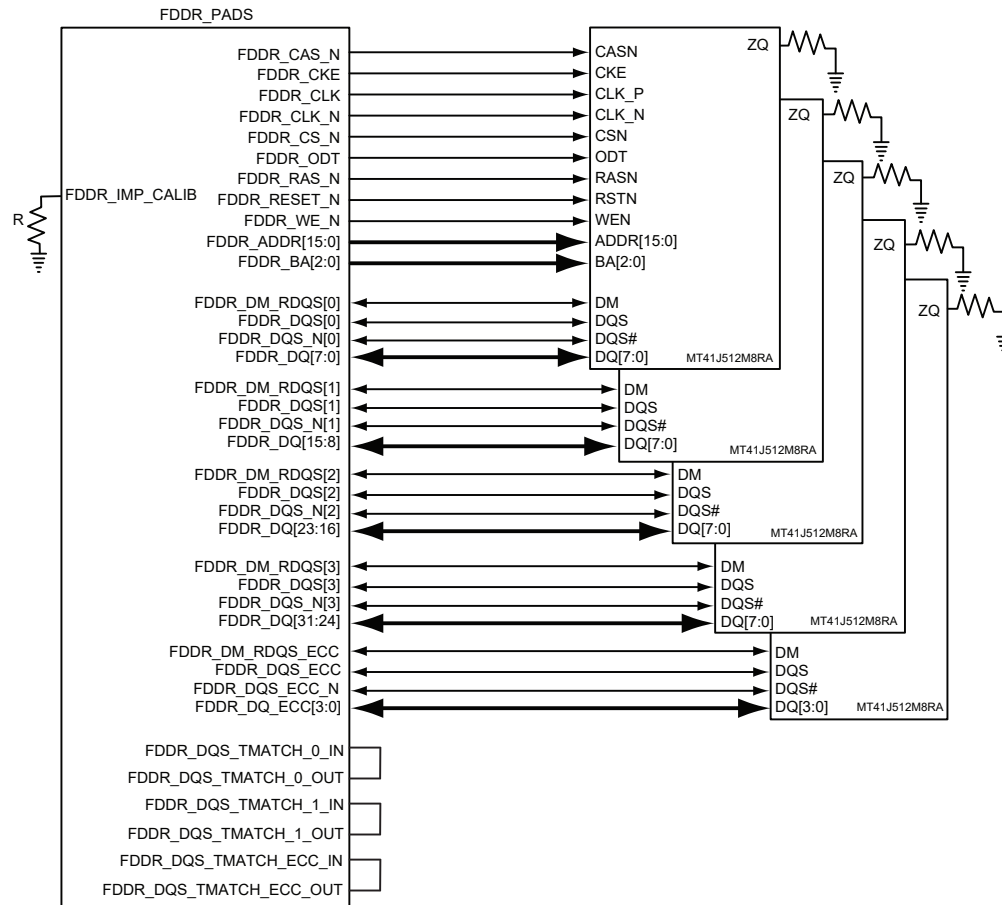
2.8.3 Example 3: Connecting 16-Bit LPDDR to FDDR_PADs with SECDED (Ask a Question)

The following ilustration shows LPDDR1 SDRAM connected to the FDDR of a IGLOO 2 device. The Micron's MT46H32M16LF is a 64 MB density device with x16 data width. The FDDR is configured in Full Bus Width mode with SECDED enabled. The SDRAM connected to FDDR_DQ_ECC[1:0] is used to store SECDED bits. The total amount of LPDDR1 memory (excluding memory for SECDED) connected to FDDR is 64 MB.

Figure 2-28. x16 LPDDR1 SDRAM Connection to FDDR
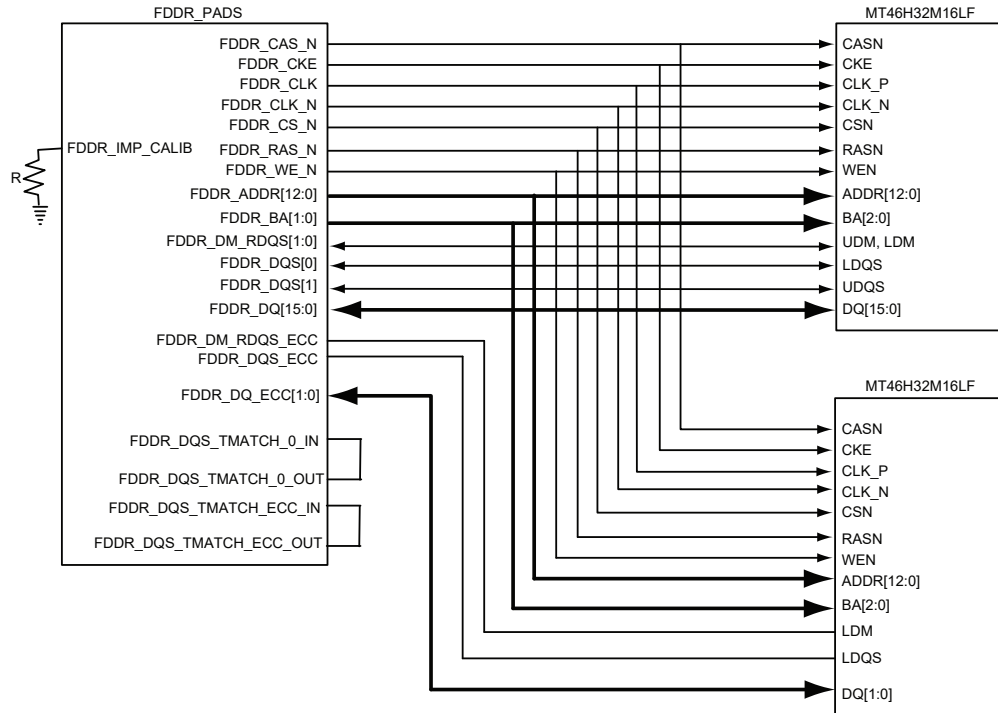
2.9 FDDR Configuration Registers (Ask a Question)

This section provides FDDR subsystem registers along with the address offset, functionality, and bit definitions. The registers are categorized based on the controller blocks in the FDDR subsystem.

The following table lists the categories of registers and their offset addresses.

Table 2-18. Address Table for Register Interfaces

Registers	Address Offset Space
DDR Controller Configuration Register, Table 1-26	0x000:0x1FC
PHY Configuration Register Summary, Table 1-100	0x200:0x3FC
DDR_FIC Configuration Register Summary, Table 1-215	0x400:0x4FC
FDDR SYSREG Configuration Register Summary, Table 2-19	0x500:0x5FC
Reserved	0x600:0x7FC

> **Important:** The FDDR SYSREG configuration registers can be locked to prevent them from being overwritten by the masters that have access to these registers. For information on how to lock/unlock these registers, see 2.11. Appendix B: Register Lock Bits Configuration.

2.9.1 FDDR SYSREG Configuration Register Summary (Ask a Question)

Table 2-19. FDDR SYSREG

Register Name	Address Offset	Register Type	Flash	Reset Source	Description
Table 2-20	0x500	RW	P	PRESETN	Comes from SYSREG. Controls the corresponding configuration input of the FPLL.

..........continued

Register Name	Address Offset	Register Type	Flash	Reset Source	Description
Table 2-21	0x504	RW	P	PRESETN	Comes from SYSREG. Controls the corresponding configuration input of the FPLL.
Table 2-22	0x508	RW	P	PRESETN	Comes from SYSREG. Controls the corresponding configuration input of the FPLL.
Table 2-23	0x50C	RW	P	PRESETN	Enables the clock to the DDR memory controller.
Table 2-24	0x510	RW	P	PRESETN	Selects the standby glitch-free multiplexers within the fabric alignment clock controller (FACC).
Table 2-25	0x514	RW	P	PRESETN	Selects the ratio between CLK_A and CLK_DDR_FIC.
Table 2-26	0x518	RW	P	PRESETN	Selects the delay values to be added to the FPLL.
Table 2-27	0x51C	RW	P	PRESETN	Soft reset register for FDDR
Table 2-28_CR	0x520	RW	P	PRESETN	Configurations register for DDRIO calibration block
Table 2-29	0x524	RW	P	PRESETN	Interrupt enable register
Table 2-30	0x528	RW	P	PRESETN	Selects AXI/AHB interface in the fabric.
Table 2-31	0x52C	RW	P	PRESETN	Automatic calibration lock is enabled.
Table 2-32	0x530	RO	—	PRESETN	Indicates the lock status of the fabric PLL.
Table 2-33	0x534	RO	—	PRESETN	Indicates the lock status of the fabric PLL.
Table 2-34	0x53C	RO	—	PRESETN	Interrupt status register
Table 2-35	0x544	RO	—	PRESETN	I/O calibration status register
Table 2-36	0x548	RW	P	PRESETN	Reset to fabric portion of the fabric alignment test circuit

2.9.2 FDDR SYSREG Configuration Register Bit Definitions (Ask a Question)

Table 2-20. PLL_CONFIG_LOW_1

Bit Number	Name	Reset Value	Description
[31:16]	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
[15:6]	PLL_FEEDBACK_DIVISOR	0×2	Can be configured to control the corresponding configuration input of the FPLL. Feedback divider value (SSE = 0) (binary value + 1: 00000000 = ÷1, ... 1111111111 = ÷ 1,024) Feedback divider value (SSE = 1) (binary value + 1: 0000000 = ÷1, ... 1111111 = ÷ 128)
[5:0]	PLL_REF_DIVISOR	0×1	Can be configured to control the corresponding configuration input of the FPLL. Reference divider value (binary value + 1: 000000 = ÷ 1)

Table 2-21. PLL_CONFIG_LOW_2

Bit Number	Name	Reset Value	Description
[31:4]	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
3	FDDR_PLL_RESET	0×1	This is only for FDDR in M2S/M2GL 150 device. • 1: FDDR PLL held in reset • 0: FDDR PLL is not in reset
[2:0]	PLL_OUTPUT_DIVISOR	0×2	Configures the amount of division to be performed on the internal (multiplied) PLL clock, in order to generate the DDR clock. Output divider value 000: ÷1 001: ÷2 010: ÷4 011: ÷8 100: ÷16 101: ÷32 It is possible to configure the PLL output divider as ÷1; this setting must not be used when the DDR is operational.

Table 2-22. PLL_CONFIG_HIGH

Bit Number	Name	Reset Value	Description
[31:16]	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
15	PLL_PD	0×0	When PD is asserted, the PLL will power down and outputs will be Low. PD has precedence over all other functions.
14	PLL_FSE	0×0	Chooses between internal and external input paths. 0: FB pin input 1: Internal feedback FB should be tied off (High or Low) and not left floating when FSE is High. FB should connect directly or through the clock tree to PLLOUT when FSE is Low. SSE is ineffective when FSE = 0.
13	PLL_MODE_3V3	0×1	Analog voltage selection • 1: 3.3V • 0: 2.5V
12	PLL_MODE_1V2	0×1	Core voltage selection • 1: 1.2V • 0: 1.0V The wrong selection (when operating at 1 V, the jitter is not within the required limit for operation of DDR) may cause the PLL not to function, but will not damage the PLL.
11	PLL_BYPASS	0×1	If 1, powers down the PLL core and bypasses it such that PLLOUT tracks REFCK. BYPASS has precedence over RESET. Microchip recommends that either BYPASS or RESET are asserted until all configuration controls are set in the desired working value, and the power supply and reference clock are stable within operating range.
[10:7]	PLL_LOCKCNT	0×F	Configured to control the corresponding configuration input of the FPLL. LOCK counter Value $2^{(binary\ value\ +\ 5)}$ • 0000: 32 • 1111: 1048576 For the number of reference cycles before LOCK is asserted from LOCK being detected.

..........continued

Bit Number	Name	Reset Value	Description
[6:4]	PLL_LOCKWIN	0×0	• 000: 500 ppm • 100: 8000 ppm • 001: 1000 ppm • 101: 16000 ppm • 010: 2000 ppm • 110: 32000 ppm • 011: 4000 ppm • 111: 64000 ppm Phase error window for Lock assertion as a fraction of divided reference period. Values are at typical PVT only and are not PVT compensated.
[3:0]	PLL_FILTER_RANGE	0×9	PLL filter range • 0000: BYPASS • 0111: 18–29 MHz • 0001: 1–1.6 MHz • 1000: 29–46 MHz • 0010: 1.6–2.6 MHz • 1001: 46–75 MHz • 0011: 2.6–4.2 MHz • 1010: 75–120 MHz • 0100: 4.2–6.8 MHz • 1011: 120–200 MHz • 0101: 6.8–11 MHz • 0110: 11–18 MHz

Table 2-23. FDDR_FACC_CLK_EN

Bit Number	Name	Reset Value	Description
[31:1]	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
0	DDR_CLK_EN	0×1	Enables the clock to the DDR memory controller.

Table 2-24. FDDR_FACC_MUX_CONFIG

Bit Number	Name	Reset Value	Description
[31:9]	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
8	FACC_FAB_REF_SEL	0×0	Selects the source of the reference clock to be supplied to the FPLL. 0: 25/50 MHz RC oscillator selected as the reference clock for the FPLL. 1: Fabric clock (FDDR_SUBSYSTEM_CLK) selected as the reference clock for the FPLL

..........continued

Bit Number	Name	Reset Value	Description
7	FACC_GLMUX_SEL	0×1	Selects the four glitch-free multiplexers within the FACC, which are related to the aligned clocks. All four of these multiplexers are switched by one signal. Allowed values: • 0: HPMS_CLK, PCLK0, PCLK1, CLK_DDR_FIC, all driven from stage 2 dividers (from CLK_SRC) • 1: HPMS_CLK, PCLK0, PCLK1, CLK_DDR_FIC, all driven from CLK_STANDBY
6	FACC_PRE_SRC_SEL	0×0	Selects whether CLK_1MHZ or ccc2asic is to be fed into the source glitch-free multiplexer. • 0: CLK_1MHZ is fed into the source glitch-free multiplexer • 1: ccc2asic is fed into the source glitch-free multiplexer
[5:3]	FACC_SRC_SEL	0×0	Selects the source multiplexer within the FACC. This is used to allow one of four possible clocks to proceed through the FACC dividers, for generation of normal functional (run-time) FDDR subsystem clocks. There are three individual 2 to 1 glitch-free multiplexers in the 4 to 1 source glitch-free multiplexer. FACC_SRC_SEL[0] is used to select the lower source MUX. • 0: CLK_SRC driven from CLK_25_50 MHZ • 1: CLK_SRC driven from clk_xtal FACC_SRC_SEL[1] is used to select the upper source MUX. • 0: CLK_SRC driven from output of PRE_SRC_MUX (either clk_1mhz or ccc2asic) • 1: CLK_SRC driven from FDDR_PLL_OUT_CLK FACC_SRC_SEL[2] is used to select output source MUX • 0: CLK_SRC driven from output of lower source MUX • 1: CLK_SRC driven from output of upper source MUX
[2:0]	FACC_STANDBY_SEL	0×0	Selects the standby glitch-free multiplexers within the FACC. This is used to allow one of four possible clocks to proceed through to the FDDR subsystem during FACC PLL initialization time (before the FPLL comes into lock). FACC_STANDBY_SEL[0] is used to select the lower standby MUX. • 1: CLK_STANDBY driven from CLK_XTAL • 0: CLK_STANDBY driven from CLK_25_50 MHZ FACC_STANDBY_SEL[1] is used to select upper standby MUX. • 0: CLK_STANDBY driven from CLK_1 MHZ • 1: CLK_STANDBY driven from ccc2asic FACC_STANDBY_SEL[2] is used to select the output standby MUX. • 0: CLK_STANDBY driven from output of lower standby MUX • 1: CLK_STANDBY driven from output of upper standby MUX

Table 2-25. FDDR_FACC_DIVISOR_RATIO

Bit Number	Name	Reset Value	Description
[31:8]	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.

..........continued

Bit Number	Name	Reset Value	Description
[7:5]	BASE_DIVISOR	0×0	Selects the ratio between CLK_A and the regenerated version of CLK_BASE, called CLK_BASE_REGEN. Allowed values: • 000: CLK_A: CLK_BASE_REGEN ratio is 1:1 • 001: CLK_A: CLK_BASE_REGEN ratio is 2:1 • 010: CLK_A: CLK_BASE_REGEN ratio is 4:1 • 100: CLK_A: CLK_BASE_REGEN ratio is 8:1 • 101: CLK_A: CLK_BASE_REGEN ratio is 16:1 • 110: CLK_A: CLK_BASE_REGEN ratio is 32:1 • Other values: Reserved
[4:3]	DIVISOR_A	0×0	Selects the ratio between CLK_SRC and CLK_A, which is an intermediate clock within the FACC. • 00: CLK_SRC:CLK_A ratio is 1:1 • 01: CLK_SRC:CLK_A ratio is 2:1 • 10: CLK_SRC:CLK_A ratio is 3:1 • 11: Reserved
[2:0]	DDR_FIC_DIVISOR	0×0	Selects the ratio between CLK_A and CLK_DDR_FIC. • 000: CLK_A: CLK_DDR_FIC ratio is 1:1 • 001: CLK_A: CLK_DDR_FIC ratio is 2:1 • 010: CLK_A: CLK_DDR_FIC ratio is 4:1 • 100: CLK_A: CLK_DDR_FIC ratio is 8:1 • 101: CLK_A: CLK_DDR_FIC ratio is 16:1 • 110: CLK_A: CLK_DDR_FIC ratio is 32:1 • Other values: Reserved

Table 2-26. PLL_DELAY_LINE_SEL

Bit Number	Name	Reset Value	Description
[31:4]	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
[3:2]	PLL_FB_DEL_SEL	0×0	Selects the delay values that are added to the FPLL feedback clock before being output to the FPLL. • 00: No buffer delay • 01: One buffer delay • 10: Two buffers delay • 11: Three buffers delay
[1:0]	PLL_REF_DEL_SEL	0×0	Selects the delay values that are added to the FPLL reference clock before being output to the FPLL. • 00: No buffer delay • 01: One buffer delay • 10: Two buffers delay • 11: Three buffers delay

Table 2-27. FDDR_SOFT_RESET

Bit Number	Name	Reset Value	Description
[31:2]	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
1	FDDR_DDR_FIC_SOFTRESET	0×1	When 1, holds the DDR_FIC (AXI/AHB) interface controller in reset.
0	FDDR_CTLR_SOFTRESET	0×1	When 1, holds the FDDR subsystem in reset.

Table 2-28. FDDR_IO_CALIB

Bit Number	Name	Reset Value	Description
[31:15]	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
14	CALIB_TRIM	0×0	Indicates override of the calibration value from the pc code / programmed code values in the DDRIO calibration block.
13	CALIB_LOCK	0×0	Used in the DDRIO calibration block as an override to lock the codes during intermediate runs. When the firmware receives CALIB_INTRPT, it may choose to assert this signal by prior knowledge of the traffic without going through the process of putting the DDR into self refresh.
12	CALIB_START	0×0	Indicates that rerun of the calibration state machine is required in the DDRIO calibration block.
[11:6]	NCODE	0×0	Indicates the DPC override NCODE from flash in DDRIO calibration. This can also be overwritten from the firmware.
[5:0]	PCODE	0×0	Indicates the PC override PCODE from flash in the DDRIO calibration block. This is also be overwritten from the firmware.

Table 2-29. FDDR_INTERRUPT_ENABLE

Bit Number	Name	Reset Value	Description
[31:7]	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
6	DDR_FIC_INT_ENABLE	0×0	Masking bit to enable DDR_FIC interrupt
5	IO_CALIB_INT_ENABLE	0×0	Masking bit to enable DDR I/O calibration interrupt
4	FDDR_ECC_INT_ENABLE	0×0	Masking bit to enable ECC error interrupt
3	FABRIC_PLL_LOCKLOST_INT_ENABLE	0×0	Masking bit to enable FAB_PLL_LOCK_LOST interrupt
2	FABRIC_PLL_LOCK_INT_ENABLE	0×0	Masking bit to enable FAB_PLL_LOCK interrupt
1	FPLL_LOCKLOST_INT_ENABLE	0×0	Masking bit to enable FPLL_LOCK_LOST interrupt
0	FPLL_LOCK_INT_ENABLE	0×0	Masking bit to enable FPLL_LOCK interrupt

Table 2-30. F_AXI_AHB_MODE_SEL

Bit Number	Name	Reset Value	Description
[31:1]	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
0	F_AXI_AHB_MODE	0×0	• 1: AXI interface in the fabric will be selected • 0: AHB interface in the fabric will be selected

Table 2-31. PHY_SELF_REF_EN

Bit Number	Name	Reset Value	Description
[31:1]	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
0	PHY_SELF_REF_EN	0×0	If 1, automatic calibration lock is enabled.

Table 2-32. FDDR_FAB_PLL_CLK_SR

Bit Number	Name	Reset Value	Description
[31:1]	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
0	FAB_PLL_LOCK	0×0	Indicates the lock status of the Fabric PLL.

Table 2-33. FDDR_FPLL_CLK_SR

Bit Number	Name	Reset Value	Description
[31:1]	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
0	FPLL_LOCK	0×0	Indicates the lock status of the FPLL (PLL in FDDR).

Table 2-34. FDDR_INTERRUPT_SR

Bit Number	Name	Reset Value	Description
[31:5]	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
4	DDR_FIC_INT	0×0	Indicates interrupt from DDR_FIC.
3	IO_CALIB_INT	0×0	The interrupt is generated when the calibration is finished. For the calibration after reset, this typically would be followed by locking the codes directly. For in-between runs during functional operation of DDR, the assertion of an interrupt does not guarantee lock because the state machine would wait for the ideal time (DRAM self refresh) for locking. This can be used by firmware to insert the ideal time and provides an indication that locked codes are available.
2	FDDR_ECC_INT	0×0	Indicates when the ECC interrupt from the FDDR subsystem is asserted.
1	PLL_LOCKLOST_INT	0×0	This bit indicates that a falling edge event occurred on the FPLL_LOCK signal. This indicates that the FPLL lost lock.

..........continued

Bit Number	Name	Reset Value	Description
0	PLL_LOCK_INT	0×0	This bit indicates that a rising edge event occurred on the FPLL_LOCK signal. This indicates that the FPLL came into lock.

Table 2-35. FDDR_IO_CALIB_SR

Bit Number	Name	Reset Value	Description
31	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
14	CALIB_PCOMP	0×01	The state of the P analog comparator
13	CALIB_NCOMP	0×01	The state of the N analog comparator
[12:7]	CALIB_PCODE	0×3F	The current PCODE value set on the FDDR DDR I/O bank
[6:1]	CALIB_NCODE	0×3F	The current NCODE value set on the FDDR DDR I/O bank
0	CALIB_STATUS	0×0	This is 1 when the codes are actually locked. For the first run after reset, this would be asserted 1 cycle after CALIB_INTRPT. For in-between runs, this would be asserted only when the DRAM is put into self refresh or there is an override from the firmware (CALIB_LOCK).

Table 2-36. FDDR_FATC_RESET

Bit Number	Name	Reset Value	Description
[31:1]	Reserved	0×0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
0	FATC_RESET	0×1	Reset to the fabric portion of the fabric alignment test circuit. 1: Reset active

2.10 Appendix A: How to Use the FDDR in SmartFusion 2 Devices (Ask a Question)

This section describes how to use the FDDR subsystem in a design. It contains the following sections:

- 1.12.1. Design Flow Using System Builder
- 2.10.2. Design Flow Using SmartDesign
- 2.10.3. Use Model 1: Accessing FDDR from FPGA Fabric Through AXI Interface
- 2.10.4. Use Model 2: Accessing FDDR from FPGA Fabric Through AHB Interface

2.10.1 Design Flow Using System Builder (Ask a Question)

This section describes how to use FDDR in the SmartFusion 2 devices using the System Builder graphical design wizard in the Libero Software.

The following image shows the initial System Builder window where you can select the features that you require. For information on how to launch the System Builder wizard and how to use it, see SmartFusion2 System Builder User Guide. For more information on DDR initialization, see SmartFusion2 DDR Controller and Serial High Speed Controller Initialization Methodology.

Figure 2-29. System Builder - Device Features Window

The following steps describe how to configure the FDDR.

1. Check the **Fabric External Memory (FDDR)** check box under the **Device Features** tab and leave the other check boxes unchecked. The following image shows the **System Builder** > **Device Features tab**.

Figure 2-30. MSS External DDR Memory Selection
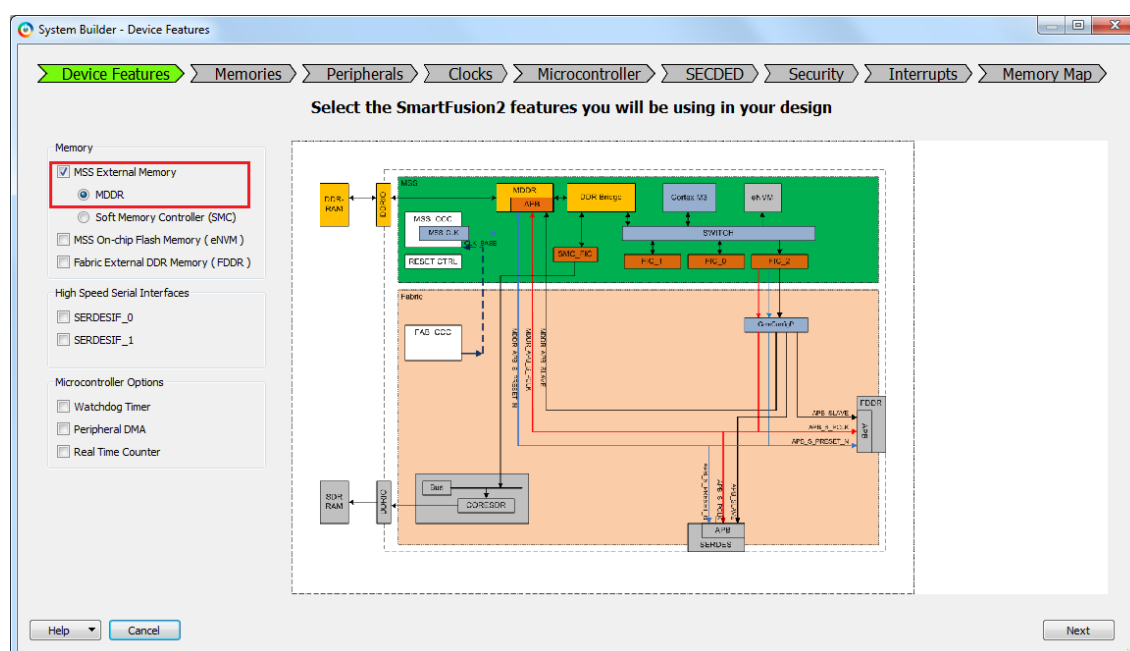
2. Navigate to the **Memories** tab. Depending on the application requirement, select the memory settings under the **General** tab as shown in the following image.

   – Memory Type can be selected as DDR2, DDR3 or LPDDR.

   – The Data width can be selected as 32- bit, 16-bit, or 8-bit. Refer Table 1-11 for supported data widths for various SmartFusion2 device packages.

   – The SECDED (ECC) can be enabled or disabled.

   – Address Mapping - The register settings to perform mapping to system address bits for various Row, Bank and Column combinations are automatically computed by the configurator using address mapping option. Table 1-19 shows the supported range for Row, Bank and Column.

Table 2-37. Supported Address Width Range for Row, Bank and Column Addressing in DDR/LPDDR

Width	DDR2	DDR3	LPDDR
Row Address	12-16	12-16	12-16
Bank Address	2-3	2-3	2-3
Column Address	9-12	9-12	9-12

   – Select the I/O Drive Strength as Half Drive Strength or Full Drive Strength as shown in Figure 1-3. The DDR I/O standard is configured as listed in Table 1-20 based on this setting.

Table 2-38. DDR I/O Standard is Configured Based on I/O Drive Strength Setting

I/O Drive Strength	Memory Type	
	DDR2	DDR3
Half Drive Strength	SSTL18I	SSTL15I
Full Drive Strength	SSTL18II	SSTL15II

Figure 2-31. Fabric DDR Memory Settings
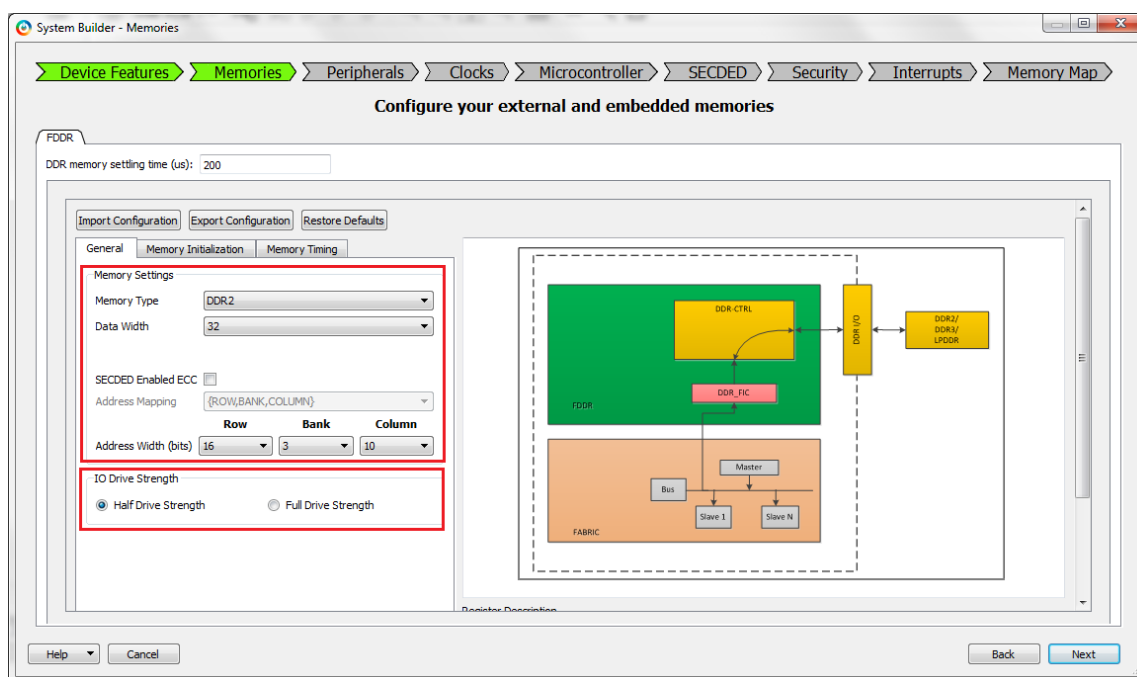
3. For only LPDDR memory, the I/O standard and I/O calibration settings are available as shown in Figure 2-12.
   – Select I/O standard as LVCMOS18 or LPDDRI.

> **Important:** If LVCMOS18 is selected, all I/Os are configured to LVCMOS1.8 except CLK/CLK_N.CLK and CLK_N are configured to LPDDRI standard as they are differential signals.

   – Select I/O calibration as ON or OFF. If I/O calibration is selected as ON, then the Smartfusion2 FDDR_IMP_CALIB pin must be pulled down with a resistor. For resistor values see Impedance Calibration section in DS0115: SmartFusion2 Pin Descriptions Datasheet.

Figure 2-32. Selecting I/O Standard as LVCMOS18 or LPDDRI
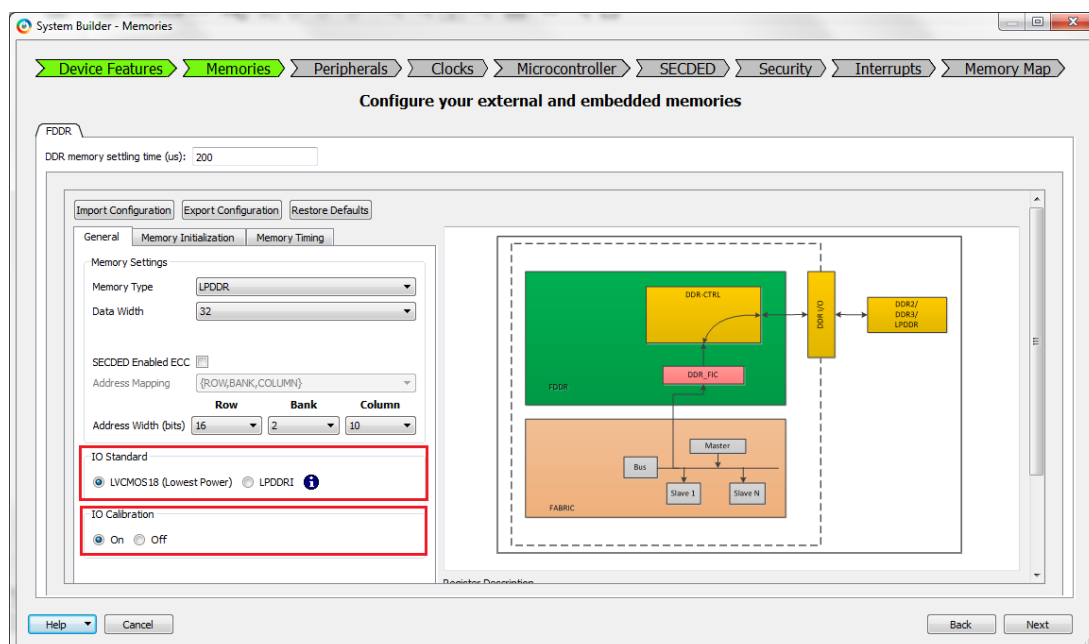
4. Depending on the application requirement, select the memory initialization settings under the **Memory Initialization** tab as shown in the following image.
   – Select the below performance related settings
     • Burst Length can be selected as 4, 8 or 16. Table 1-11 for supported burst lengths.
     • Burst order can be selected as sequential or interleaved. Refer Table 1-11 for supported burst orders.
     • Timing mode can be selected as 1T or 2T. For more details refer to 12.6.10.7. 1T or 2T Timing.
     • CAS latency is the delay, in clock cycles, between the internal READ command and the availability of the first bit of output data. Select the CAS latency according to the DDR memory (Mode register) datasheet.
   – Select the below power saving mode settings. Refer to 2.6.3.4.5. Power Saving Modes for more details.
     • Self-Refresh Enabled
     • Auto Refresh Burst Count
     • Power down Enabled
     • Stop the clock: supported only for LPDDR
     • Deep Power down Enabled: supported only for LPDDR
     • Power down entry time
   – Select the additional performance settings.
     • Additive CAS Latency is defined by EMR[5:3] register of DDR2 memory and by MR1[4:3] register of DDR3 memory. It enables the DDR2 or DDR3 SDRAM to allow a READ or WRITE command from DDR Controller after the ACTIVATE command for the same bank prior to tRCD (MIN). This configuration is part of DDR2 Extended Mode register and DDR3 Mode register1.

- CAS Write Latency (CWL) is defined by DDR3 MR2[5:3] and is the delay, in clock cycles, from the releasing of the internal write to the latching of the first data in. The overall WRITE latency (WL) is equal to CWL + AL, where CWL is set to 5 clock cycles by default.

– Select the below ZQ Calibration settings for DDR3 memory. For more details refer 2.6.2. ZQ Calibration.

- Zqinit
- ZQCS
- ZQCS Interval

– Select other settings.

- Local ODT setting is not supported for LPDDR memory. For DDR2/DDR3 memory type, user can choose any option for "Local ODT". User can enable or disable **LOCAL ODT** during read transaction.

- Drive strength setting is defined by EMR[7:5] register bits of LPDDR memory with drop down options of **Full**, **Half**, **Quarter**, and **One-eighth** drive strength, it is defined by EMR[1] register bit of DDR2 memory with drop down options of **Full** and **Weak** drive strength and it is defined by MR1 register bits M5 and M1 of DDR3 memory with drop down options of **RZQ/6** and **RZQ/7**.

- Partial array self-refresh coverage setting is defined by EMR[2:0] register bits of LPDDR memory with drop down options of **Full**, **Quarter**, **One-eighth** and **One-sixteenth**. This feature helps in improving power savings during self-refresh by selecting the amount of memory to be refreshed during self-refresh.

- RTT (Nominal) setting is defined by EMR[6] and EMR[2] register bits of DDR2 memory which determines what ODT resistance is enabled with drop down options of `RTT disabled', '50 ohms', '75 Ω' and `150 Ω' and it is defined by MR1[9], MR1[6] and MR1[2] register bits of DDR3 memory. In DDR3 memory RTT nominal termination is allowed during standby conditions and WRITE operations and NOT during READ operations with drop down options of **RZQ/2**, **RZQ/4**, and **RZQ/6**.

- RTT_WR (Dynamic ODT) setting is defined by MR2[10:9] register bits of DDR3 memory. This is applicable only during WRITE operations. If dynamic ODT (Rtt_WR) is enabled, DRAM switches from normal ODT (RTT_nom) to dynamic ODT (Rtt_WR) when beginning WRITE burst and subsequently switches back to normal ODT at the end of WRITE burst. The drop down options provided to the user are **off**, **RZQ/4**, and **RZQ/2**.

- Auto self-refresh setting is defined by MR2[6] register bit of DDR3 memory with drop down option of **Manual** and **Auto**. Self-refresh temperature setting is defined by MR2[7] register bit of DDR2 memory with drop down options of **Normal** and **Extended**.

Figure 2-33. DDR Memory initialization Settings
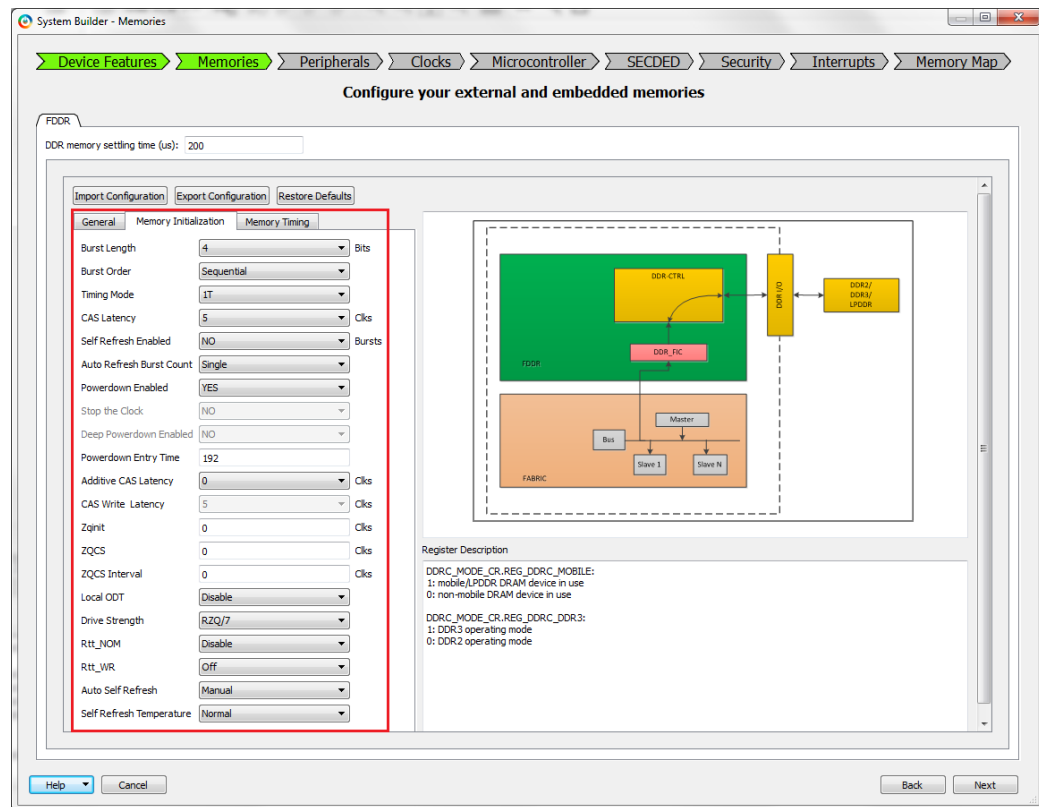
5. Select the memory timing settings under the **Memory Timing** tab according to the DDR memory vendor datasheet as shown in the following image. For more information, see 2.6.8. Configuring Dynamic DRAM Constraints.

Figure 2-34. DDR Memory Timing Settings
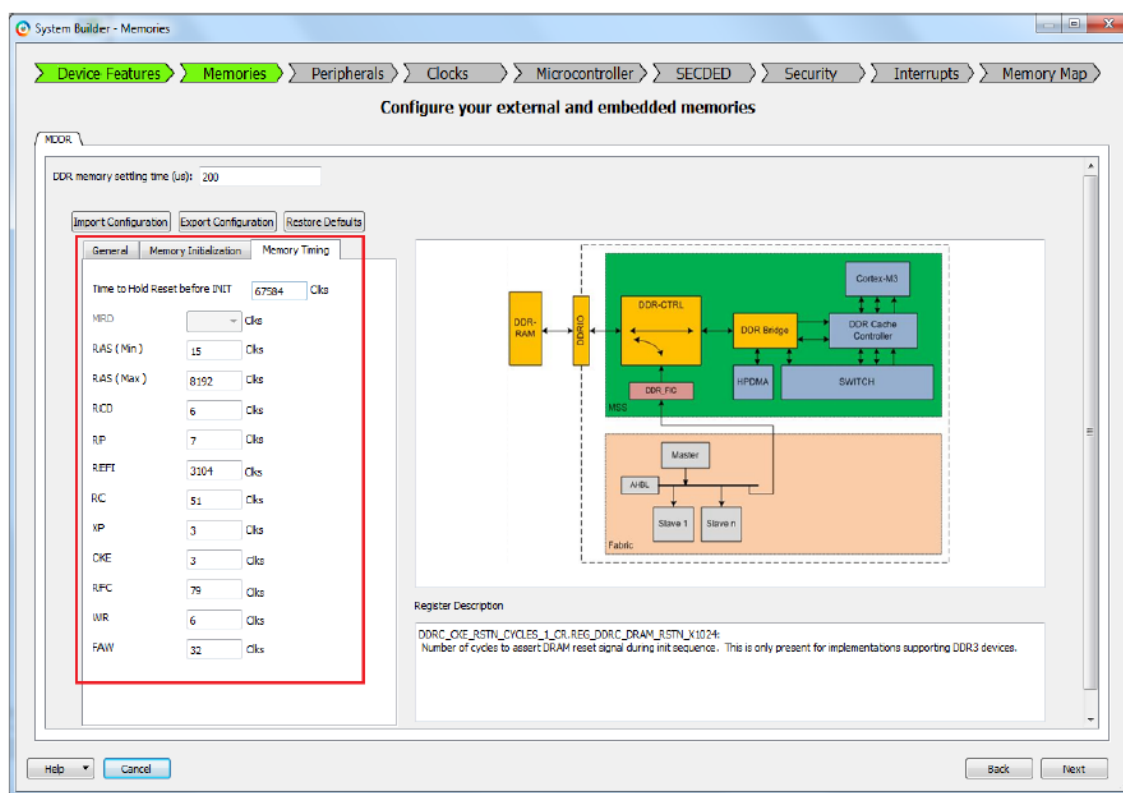
The configurator also provides the option to import and export the register configurations.

Configuration files for accessing DDR3 memory on SmartFusion2 Development kit can be downloaded from www.microsemi.com/soc/documents/FDDR3_16Bit_SB.zip.

Configuration files for accessing LPDDR memory on SmartFusion2 Starter kit can be downloaded from www.microsemi.com/soc/documents/LPDDR_Emcraft_Config.zip.

> **Important:** The firmware generated by Libero SoC stores these configurations and the FDDR subsystem registers are initialized by the Cortex-M3 processor during the system_init phase of the firmware projects (SoftConsole/IAR/Keil projects generated by Libero SoC).

An example of FDDR register configurations for operating the LPDDR memory (MT46H64M16LF) with clock 166 MHz is shown below.

Device Memory Settling Time (us): 200

The DDR memories require settling time for the memory to initialize before accessing it. the LPDDR memory model MT46H64M16LF needs 200 µs settling time.

General

– Memory Type: Select LPDDR

– Data Width: 16

Memory Initialization:

– Burst length: 8

- Burst Order: Interleaved
- Timing Mode: 1T
- CAS Latency: 3
- Self Refresh Enabled: No
- Auto Refresh Burst Count: 8
- PowerDown Enabled: Yes
- Stop the clock: No
- Deep PowerDown enabled: No
- No Activity clocks for Entry: 320

Memory Timing:

- Time To Hold Reset Before INIT - 67584 clks
- MRD: 4 clks
- RAS (Min): 8 clks
- RAS (Max): 8192 clks
- RCD: 6 clks
- RP: 7 clks
- REFI: 3104 clks
- RC: 3 clks
- XP: 3 clks
- CKE: 3 clks
- RFC: 79 clks
- FAW: 0 clks

6. Navigate to the **Peripherals** tab. To access the FDDR from the FPGA fabric, drag and drop the **Fabric AMBA Master to the MSS DDR FIC** Subsystem and click **Configure** to select the type of interface as AXI or single AHB-Lite. The user logic in the FPGA fabric can access the DDR memory through the FDDR using these interfaces. The following image shows the **Peripherals** tab.

MICROCHIP

Figure 2-35. MSS DDR FIC Subsystem Configuration
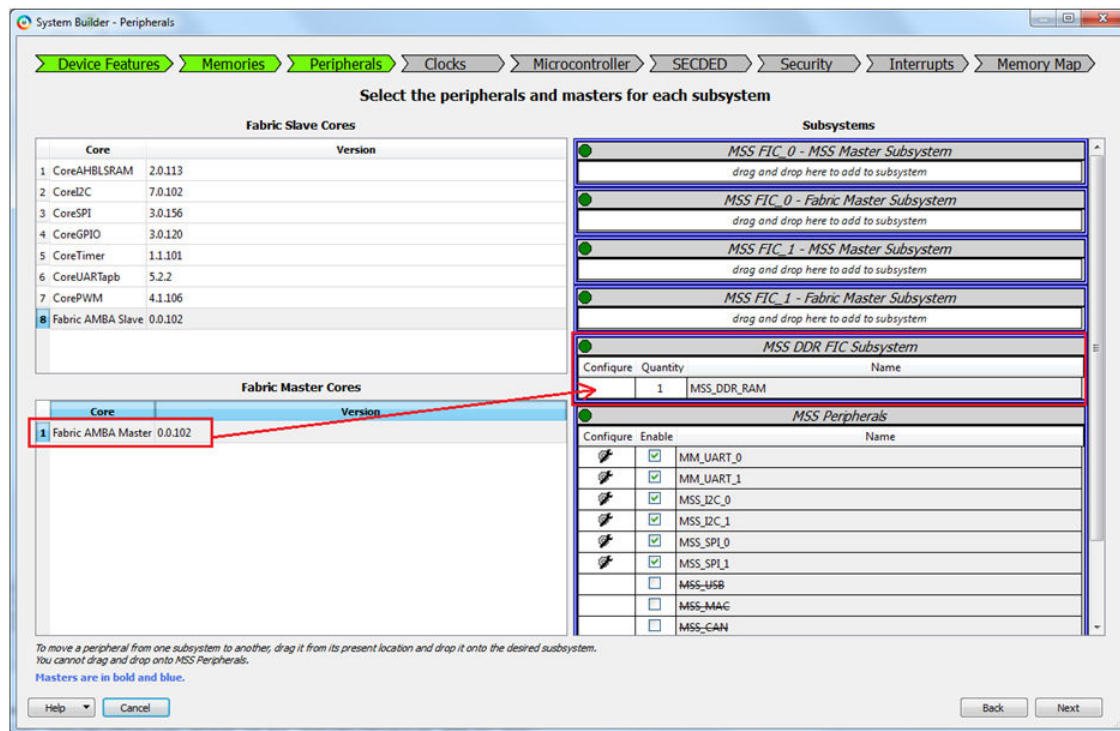
7. Navigate to the **Clocks** tab. The **Clocks** tab allows to configure the system clock and subsystem clocks. The FDDR subsystem operates on FDDR_CLK, which comes from MSS_CCC. The FDDR_CLK must be selected as multiples of 1, 2, 3, 4, 6 or 8-of M3_CLK. The maximum frequency of FDDR_CLK is 333.33 MHz.
    FDDR_SUBSYSTEM_CLK drives the DDR_FIC slave interface and defines the frequency at which the FPGA fabric subsystem connected to this interface is intended to run. DDR_FIC_CLK can be configured as a ratio of FDDR_CLK (1, 2, 3, 4, 6, 8, 12, or 16) using the Clocks configurator. The maximum frequency of FDDR_SUBSYSTEM_CLK is 200 MHz. The following image shows the FDDR_CLK configuration.

Figure 2-36. FDDR Clock Configuration

2.10.2 Design Flow Using SmartDesign (Ask a Question)

The following illustration shows the design flow for using the FDDR subsystem to access external DDR memory.

The design flow consists of two parts:

- **Libero flow**: This includes configuring the type of DDR memory, choosing fabric master interface type, clocking, and DDR I/O settings.
- **FDDR register initialization**: FDDR subsystem registers can be initialized using the Arm Cortex-M3 processor or FPGA fabric master. After MSS reset, the FDDR registers have to be configured according to application and DDR memory specification. The 1.5.5. MDDR Subsystem Features Configuration provides the details of required register configuration for FDDR features. While configuring the registers, the soft reset to the DDR controller must be asserted. After releasing the soft reset, the DDR controller performs DDR memory initialization and sets the status bits in DDRC_SR.

Figure 2-37. Design Flow
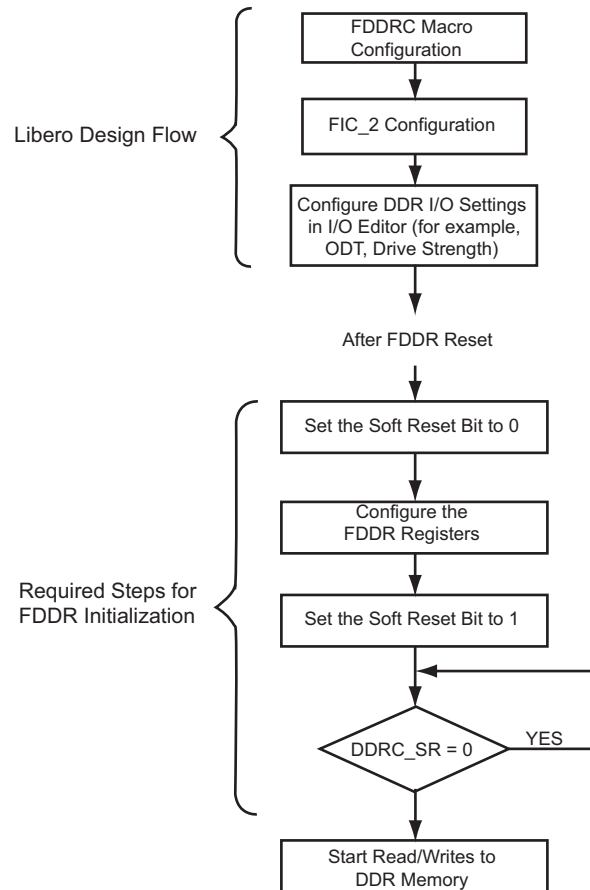
The configuration steps in the flowchart are explained in detail in the following sections.

2.10.2.1 DDR Memory Controller Macro Configuration (Ask a Question)

The DDR Memory Controller macro in the Libero IP Catalog has to be instantiated in SmartDesign to access the external DDR memory through the DDR Memory Controller subsystem. The FDDRC macro configurator shown in the following image enables configuration of the DDR Memory Controller subsystem.

Figure 2-38. Fabric External Memory DDR Controller Configurator
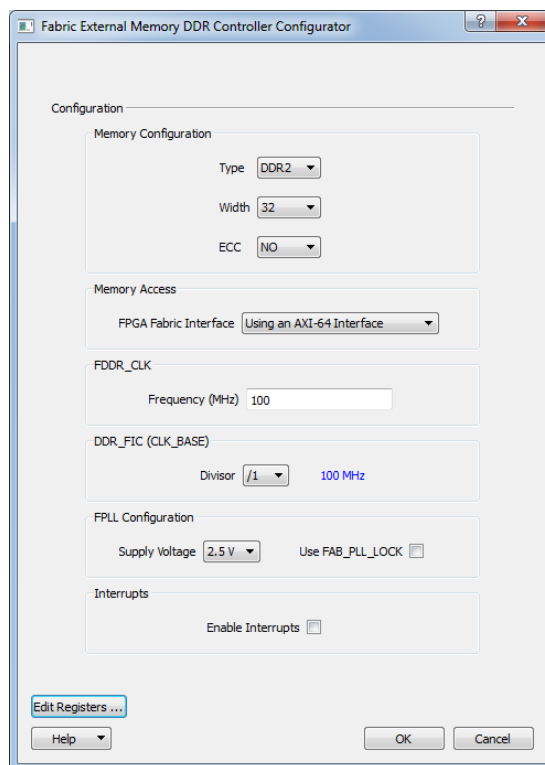
Depending on the application requirement, select the memory settings under the General tab as shown in the image.

- Memory Type can be selected as DDR2, DDR3 or LPDDR.
- The Data width can be selected as 32-bit, 16-bit, or 8-bit. See Table 1-13 for supported data widths for various SmartFusion 2 device packages.
- Clock Frequency can be selected between 20 MHz to 333 MHz. The FDDR subsystem operates on this clock (FDDR_CLK) frequency
- The SECDED (ECC) can be enabled or disabled.

Select FPGA Fabric Interface type as AXI, single AHBLite, or two AHBLite. On completion of the configuration, the selected interface is exposed in SmartDesign. User logic in the FPGA fabric can access DDR memory through the FDDR using these interfaces.

The DDR_FIC clock drives the DDR_FIC slave interface and defines the frequency at which the FPGA fabric subsystem connected to this interface is intended to run. DDR_FIC clock can be configured using FDDR CLOCK Divisor—1, 2, 3, 4, 6, 8, 12, or 16—of FDDR_CLK. The maximum frequency of DDR_FIC clock is 200 MHz. The DDR_FIC clock has to be driven from FPGA fabric.

The FPLL LOCK signal can be exposed to the FPGA fabric to monitor the health of the PLL (loss of lock requires special handling by the application).

The interrupts in the FDDR subsystem can be exposed in SmartDesign by selecting the Enable Interrupts check box.

Select the memory settings under Memory Initialization tab and Memory Timing tab as described in the steps 3 and 4 in the 1.12.1. Design Flow Using System Builder.

2.10.2.2 FIC_2 Configuration (Ask a Question)

This is required for initializing the FDDR registers from Cortex-M3 processor. Configure the FIC_2 (Peripheral Initialization) block as shown in the following image to expose the FIC_2_ APB_MASTER interface in Libero SmartDesign. CoreConfigP must be instantiated in SmartDesign and make the connections illustrated in the FIC_2 Configurator. The following image shows the connectivity between the APB configuration interface and FDDR subsystem.

Figure 2-39. FIC Configuration
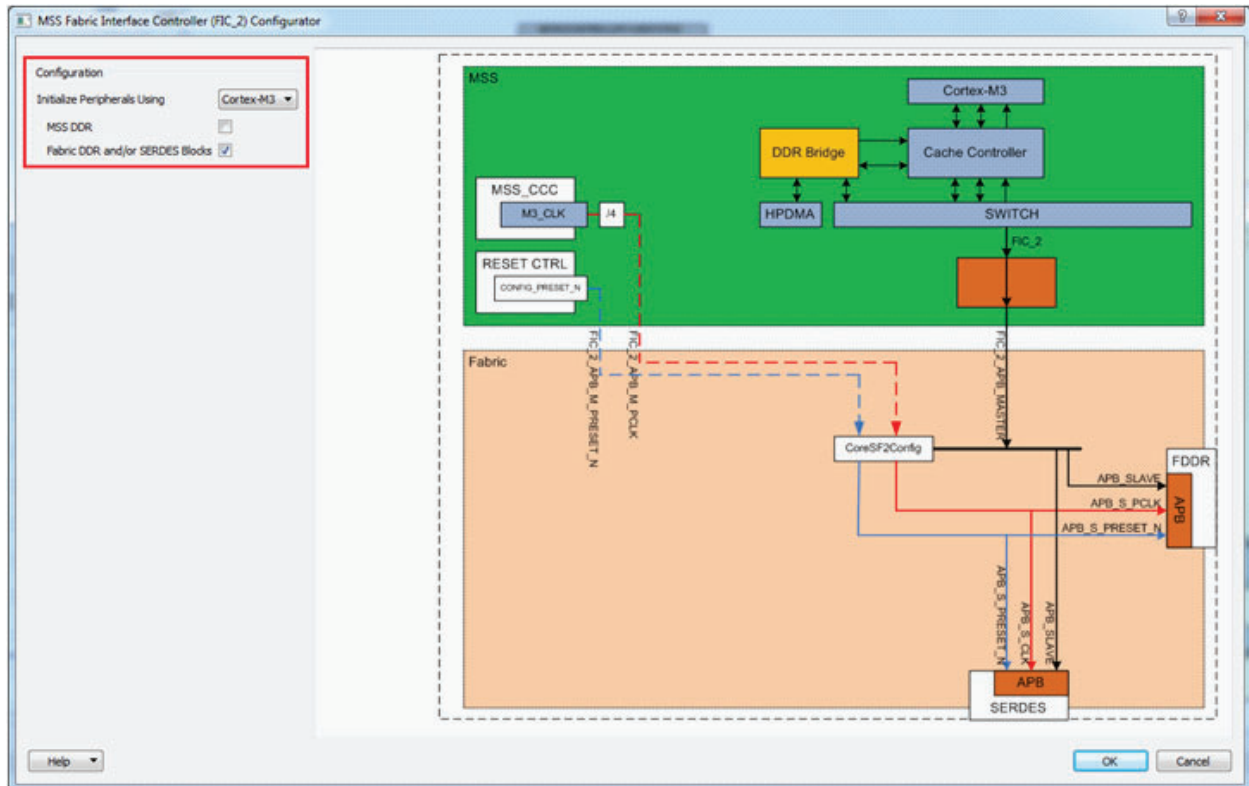
While enabling this option, the APB_S_PCLK and FIC_2_APB_M_PCLK signals are exposed in SmartDesign. The FDDR's APB_S_PCLK and APB_S_PRESET_N have to be connected to FIC_2_APB_M_PCLK and FIC_2_APB_M_PRESET_N. The FIC_2_APB_M_PCLK clock is generated from MSS_CCC and is identical to M3_CLK/4.

2.10.2.3 I/O Configuration (Ask a Question)

I/O settings such as ODT and drive strength can be configured as shown in the following image using the **I/O Editor** in Libero SoC.

Figure 2-40. I/O Configuration

For more information about FDDR Subsystem Features Configuration, see 1.5.5. MDDR Subsystem Features Configuration.

2.10.3 Use Model 1: Accessing FDDR from FPGA Fabric Through AXI Interface (Ask a Question)

The AXI master in the FPGA fabric accesses the DDR memory through the FDDR subsystem, as shown in the following illustration. The FDDR registers are configured from FPGA fabric through the APB interface. The APB master in the FPGA fabric asserts a ready signal to the AXI master, indicating successful initialization of the DDR memory.

Read, write, and read-modify-write transactions are initiated by the AXI master to read or write the data into the DDR memory after receiving a ready signal from the APB master.

Figure 2-41. FDDR with AXI Interface
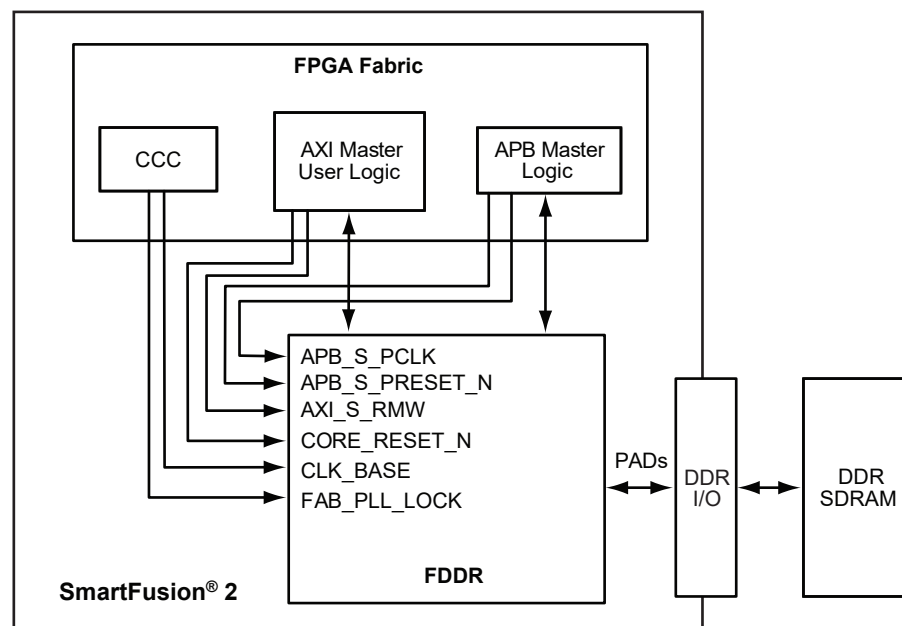
Use the following steps to access the FDDR from the AXI master in the FPGA fabric:

1. Instantiate the DDR Memory Controller macro in the SmartDesign canvas.
2. Configure the FDDR and select the AXI interface, as shown in the following image. In this example, the design is created to access DDR3 memory with a 32-bit data width. The FDDR clock is configured to 333 MHz and DDR_FIC is configured to 111 MHz.

Figure 2-42. FDDR Configuration
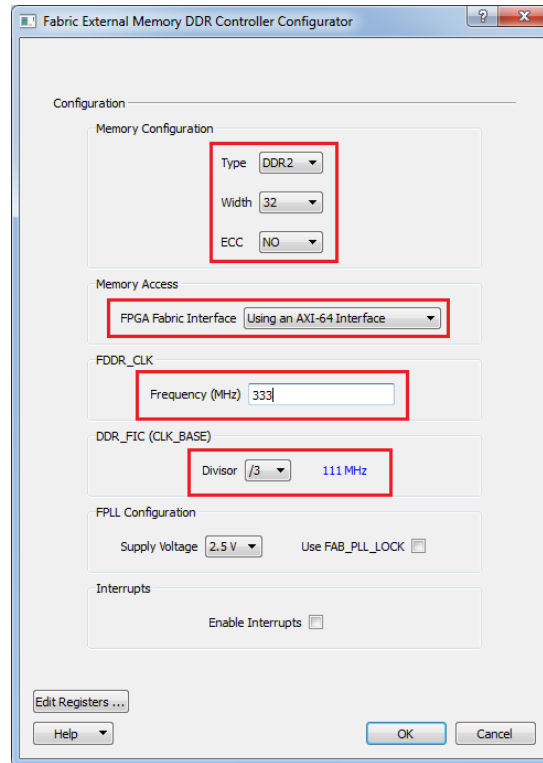
3. Instantiate the clock resources (FAB_CCC and chip oscillators) in the SmartDesign canvas and configure, as required. In this example, the fabric CCC is configured to generate 111 MHz, as shown in the following image.

Figure 2-43. Fabric CCC Configuration
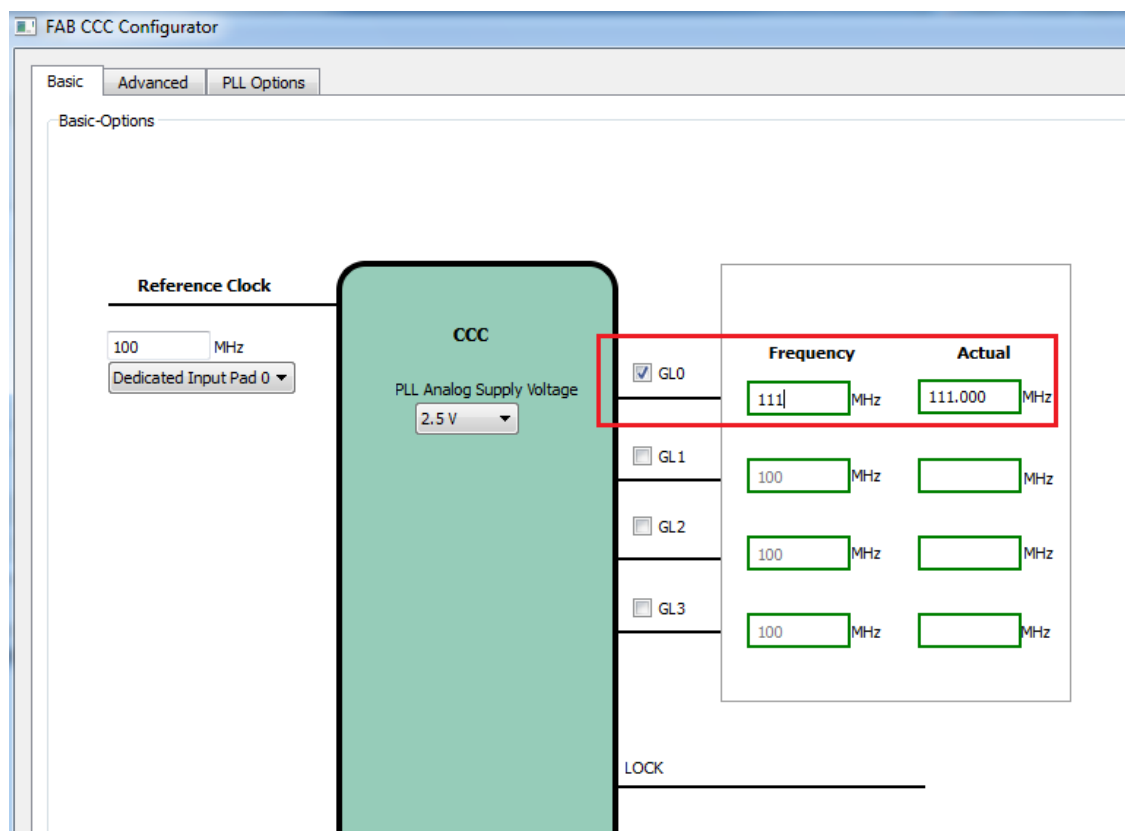
4. Instantiate user AXI master logic in the SmartDesign canvas to access the FDDR through the AXI interface. Ensure that the AXI master logic accesses the FDDR after configuring the FDDR registers from the APB master. The AXI master clock frequency should be same as FDDR DDR_FIC clock frequency.

5. Instantiate user APB master logic in the SmartDesign canvas to configure the FDDR registers through the APB interface.

6. Connect the AXI master to the FDDR AXI slave interface. Connect the APB master to the FDDR APB slave interface through CoreAPB.

7. Make the other connections in the SmartDesign canvas, as shown in the following image.

Figure 2-44. SmartDesign Canvas
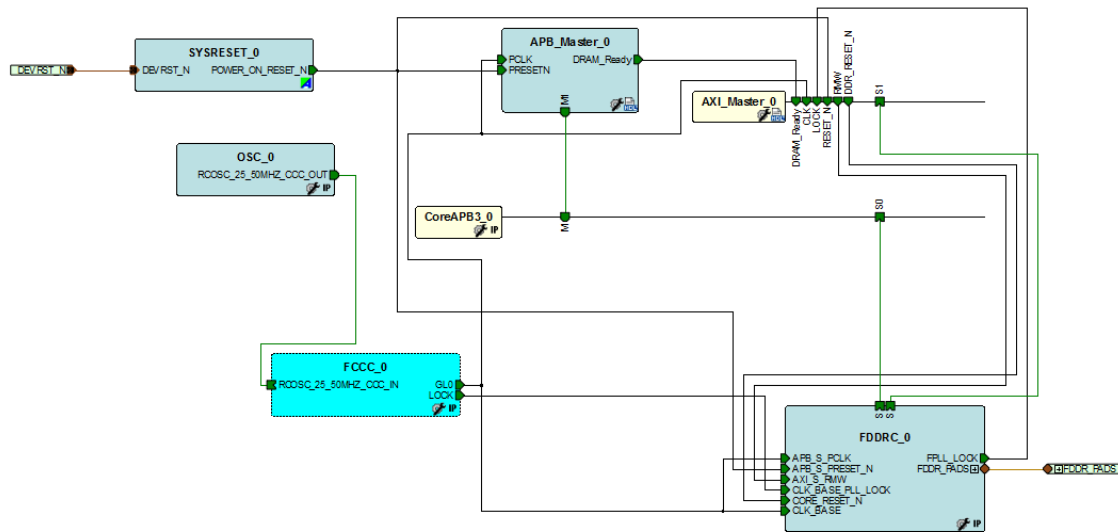
8. To verify the design in Libero SoC, create a SmartDesign testbench project and instantiate a DDR memory model provided by the DDR memory vendor. Simulate the design and observe the AXI read and write transactions.

> **Important:** The FDDR subsystem can be configured using the Cortex-M3 processor without having an APB master in the FPGA fabric. The System Builder can be used to create the design by following steps in 1.12.1. Design Flow Using System Builder. The System Builder provides "INIT_DONE" to indicate that the DDR memory has been successfully initialized.

2.10.4 Use Model 2: Accessing FDDR from FPGA Fabric Through AHB Interface (Ask a Question)

This use model shows an example of accessing DDR memory through the FDDR subsystem from two AHB masters (see the following illustration). FIC_0 is used as AHB master 0 and user logic in the fabric is used as AHB master 1. The FDDR registers are configured from the Cortex-M3 processor through CoreConfigP. The read, write, and read-modify-write transactions are initiated by the AXI master to read or write the data into the DDR memory after receiving the ready signal from the APB master.

**Figure 2-45.** Accessing FDDR Subsystem Through Dual AHB Interface
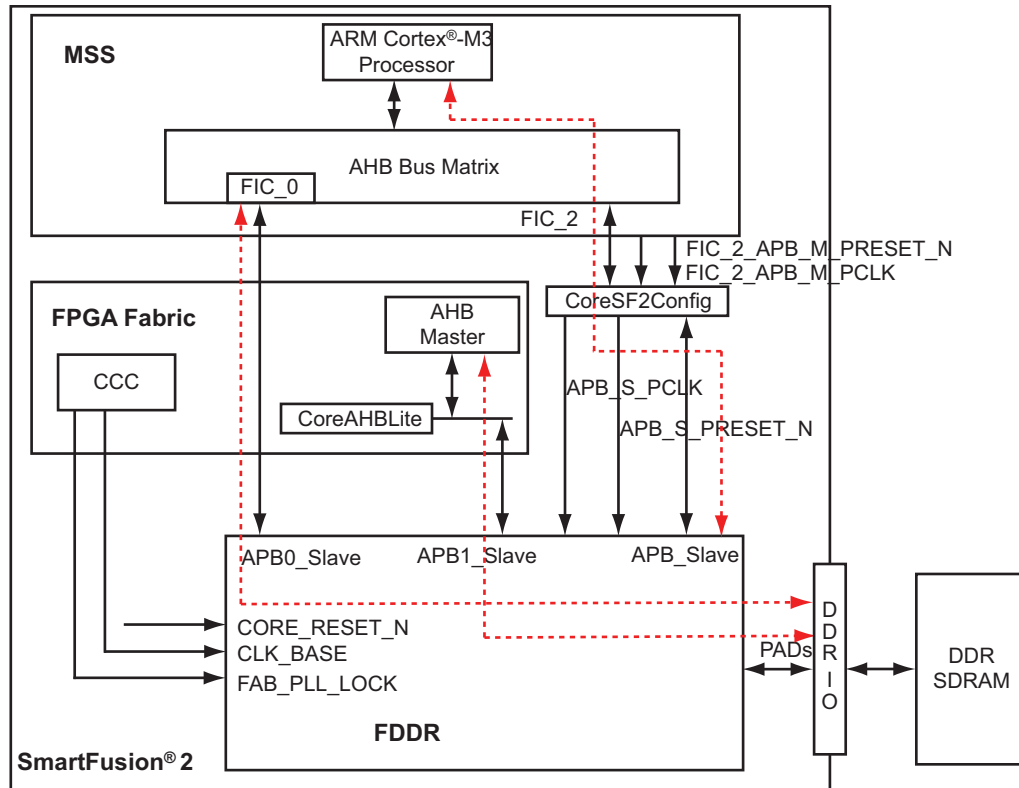
Use the following steps to access the FDDR from the AXI master in the FPGA fabric:

1. Instantiate the SmartFusion2 MSS component in the SmartDesign canvas.

2. Configure the SmartFusion2 MSS peripheral components as required using the MSS configurator. Configure FIC_0 as the AHB master.

3. Configure FIC_2 to enable the FIC_2 APB interface for configuring the FDDR subsystem registers from the Cortex-M3 processor, as shown in the following image.

Figure 2-46. FIC_2 Configuration
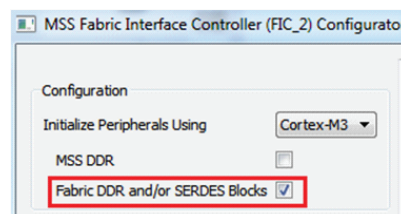
4. Configure MSSCCC for the FIC_0 clock, as shown in the following image. The FIC_0 clock is configured to 111 MHz.

**Figure 2-47.** MSS CCC Configuration
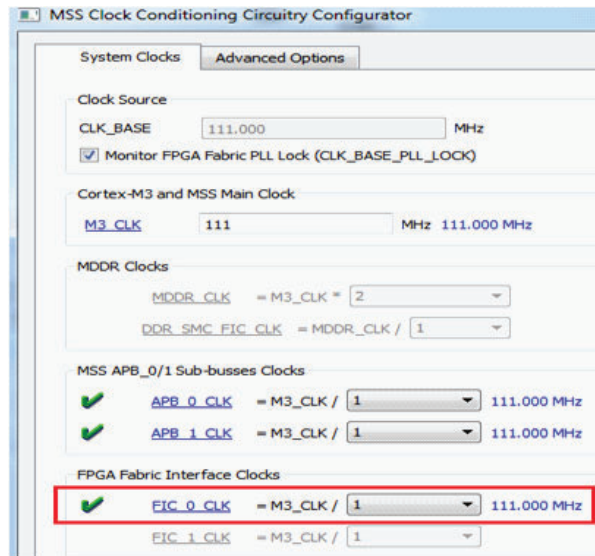
5. Instantiate the DDR Memory Controller macro in the SmartDesign canvas.

6. Configure the FDDR and select the dual AHB interface, as shown in the following image. In this example, the design is created to access DDR3 memory with a 32-bit data width. The FDDR clock is configured to 333 MHz and DDR_FIC is configured to 111 MHz.
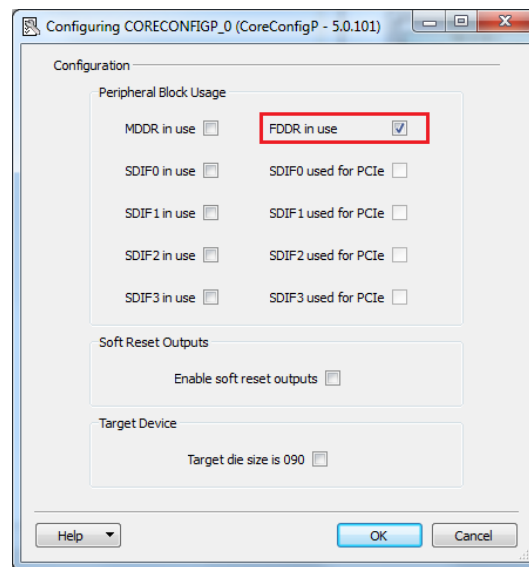
Figure 2-48. FDDR Configuration
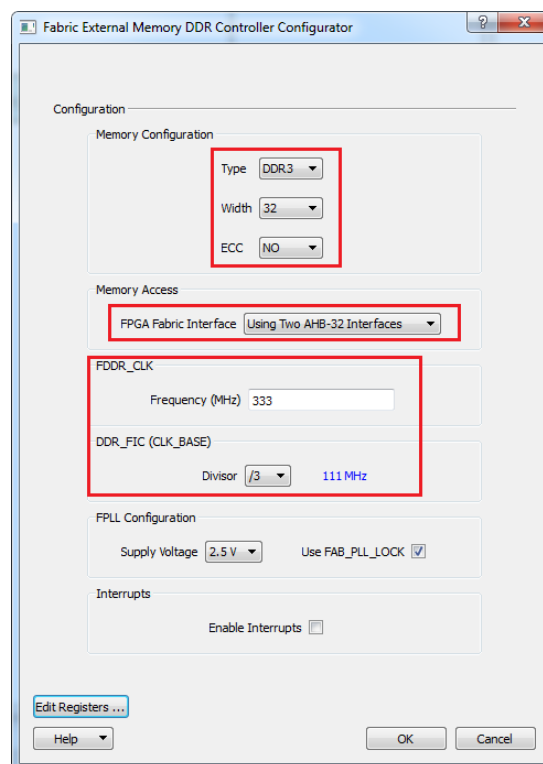
7. Depending on the application requirement, select the memory settings. For more information, see steps 3 and 4 in the 1.12.1. Design Flow Using System Builder.

8. Instantiate the clock resources (FCCC and chip oscillators) in the SmartDesign canvas and configure, as required. In this example, the fabric CCC is configured to generate 111 MHz, as shown in the following image.

    Figure 2-49. Fabric CCC Configuration
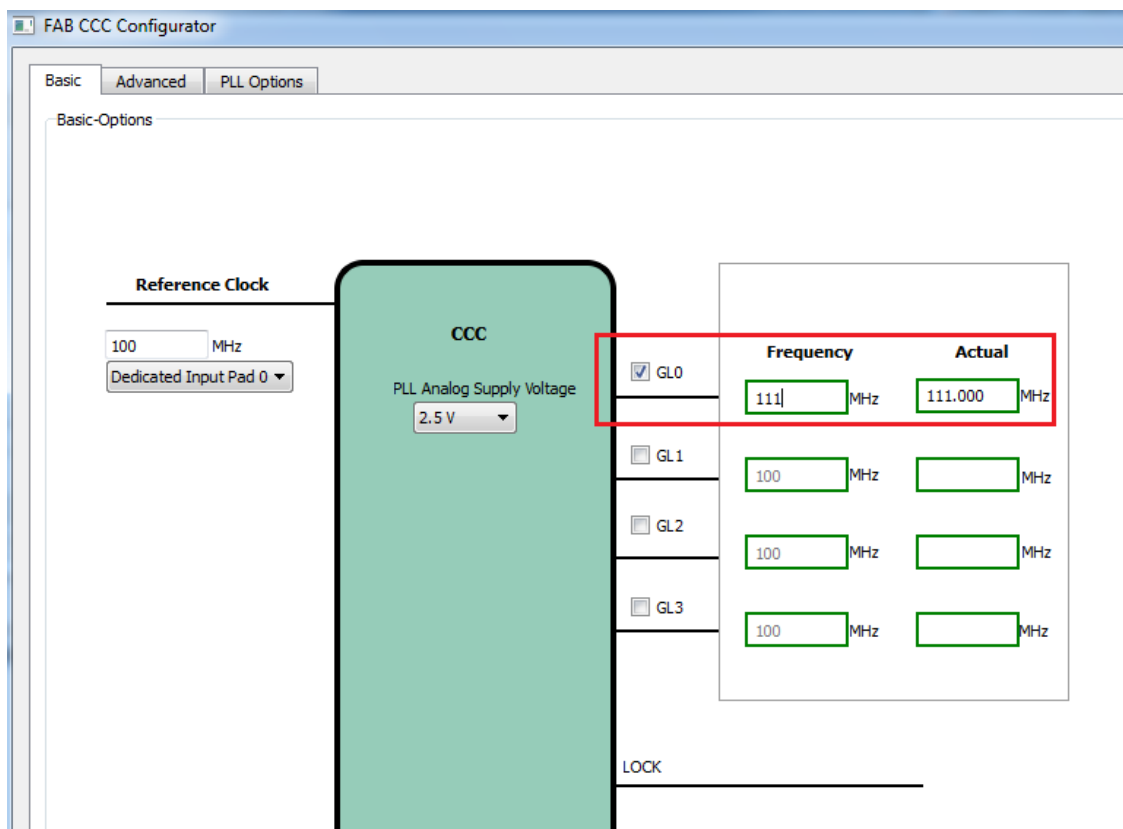
9. Instantiate CoreConfigP in the SmartDesign canvas and configure for FDDR, as shown in the following image. Make the FIC_2 and FDDR APB interface connections to CoreConfigP.
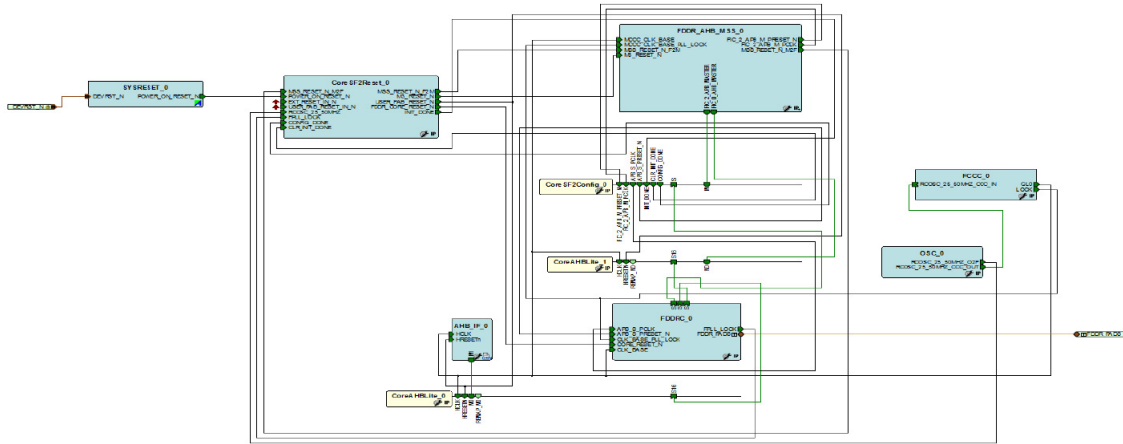
Figure 2-50. CoreConfigP IP Configuration

10. Instantiate CoreResetP in the SmartDesign canvas and configure for FDDR, as shown in the following image. Make the connections to CoreResetP and CoreConfigP accordingly.

Figure 2-51. CoreConfigP IP Configuration
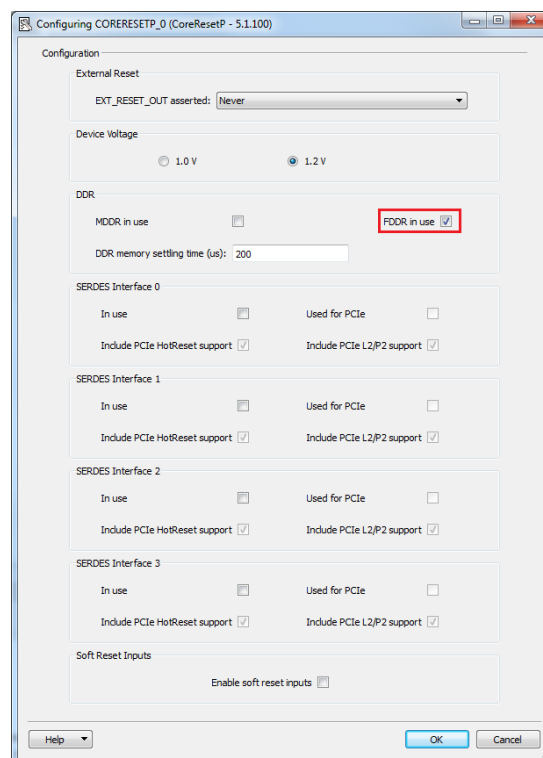
11. Instantiate user AHB master logic in the SmartDesign canvas to access the FDDR through the AHB interface. The AHB master clock frequency should be the same as the FDDR DDR_FIC clock frequency.

12. Connect the AHB master to the FDDR AHB slave0 interface through CoreAHBLite. Connect the FIC_0 master to the FDDR AHB slave1 interface through CoreAHBLite.

13. Make the other connections in the SmartDesign canvas, as shown in the following illustration.

Figure 2-52. SmartDesign Canvas

14. To verify the design in Libero SoC, create a SmartDesign testbench project and instantiate a DDR memory model provided by the DDR memory vendor. Simulate the design and observe the AHB read and write transactions.

> **Important:** Microchip provides the System Builder tool to simplify design creation. To use System Builder, select **Use System Builder** while creating a new project from the Design Templates and Creators panel in Libero SoC. Follow the steps in the **System builder - Device Features** GUI and generate the design.

2.11 Appendix B: Register Lock Bits Configuration (Ask a Question)

The Register Lock Bits Configuration tool is used to lock MSS, SerDes, and FDDR configuration registers of SmartFusion 2 devices to prevent them from being overwritten by masters that have access to these registers. Register lock bits are set in a text (`*.txt`) file, which is then imported into the SmartFusion 2 project.

2.11.1 Lock Bit File (Ask a Question)

An initial, default lock bit file can be generated by clicking **Generate FPGA Array Data** in the **Design Flow** window.

The default file located at `<proj_location>/designer/<root>/`
`<root>_init_config_lock_bits.txt` can be used to make the required changes.

> **Important:** Save the file using a different name if you modify the text file to set the lock bits.

2.11.2 Lock Bit File Syntax (Ask a Question)

A valid entry in the lock bit configuration file is defined as a <lock_parameters> < lock bit value> pair format.

The lock parameters are structured as follows:

- Lock bits syntax for a register: <Physical block name>_<register name>_LOCK
- Lock bits syntax for a specific field: <Physical block name>_<register name>_<field name>_LOCK

The following are the physical block names (varies with device family and die):

- MSS
- FDDR
- SERDES_IF_x (where x is 0,1,2,3 to indicate the physical SerDes location) for SmartFusion 2 and IGLOO 2 (010/025/050/150) devices
- SERDES_IF2 for SmartFusion 2 and IGLOO 2 (060/090) devices (only one SerDes block per device)

Set the lock bit value to 1 to indicate that the register can be written to (unlocked) and to 0 to indicate that the register cannot be written to (locked).

Lines starting with # or ; are comments. Empty lines are allowed in the lock bit configuration file.

The following figure shows the lock bit configuration file.

Figure 2-53. Lock Bit Configuration File

```
# Register Lock Bits Configuration File for MSS, SERDES(s) and Fabric DDR
# Microsemi Corporation - Microsemi Libero Software Release v11.7 SP1 (Version 11.7.1.2)
# Date: Tue Mar 29 13:24:54 2016

# sb_sb_0/sb_sb_MSS_0/MSS_ADLIB_INST/INST_MSS_050_IP
 MSS_ESRAM_CONFIG_LOCK                            0
 MSS_ESRAM_MAX_LAT_LOCK                           1
 MSS_DDR_CONFIG_LOCK                              1
 MSS_ENVM_CONFIG_LOCK                             0
 MSS_ENVM_REMAP_BASE_LOCK                         1
 MSS_ENVM_FAB_REMAP_LOCK                          1
 MSS_CC_CONFIG_LOCK                               0
 MSS_CC_CACHEREGION_LOCK                          1
 MSS_CC_LOCKBASEADDR_LOCK                         1
 MSS_CC_FLUSHINDX_LOCK                            0
 MSS_DDRB_BUF_TIMER_LOCK                          1
 MSS_DDRB_NB_ADR_LOCK                             1
 MSS_DDRB_NB_SIZE_LOCK                            0
 MSS_DDRB_CONFIG_LOCK                             1
 MSS_EDAC_ENABLE_LOCK                             1
 MSS_MASTER_WEIGHT_CONFIG0_LOCK                   1
 MSS_MASTER_WEIGHT_CONFIG1_LOCK                   1
 MSS_SOFT_INTERRUPT_LOCK                          1
 MSS_SOFTRESET_ENVM0_SOFTRESET_LOCK              1
 MSS_SOFTRESET_ENVM1_SOFTRESET_LOCK              1
 MSS_SOFTRESET_ESRAM0_SOFTRESET_LOCK             1
 MSS_SOFTRESET_ESRAM1_SOFTRESET_LOCK             1
 MSS_SOFTRESET_MAC_SOFTRESET_LOCK                1
 MSS_SOFTRESET_PDMA_SOFTRESET_LOCK               1
 MSS_SOFTRESET_TIMER_SOFTRESET_LOCK              1
 MSS_SOFTRESET_MMUART0_SOFTRESET_LOCK            1
 MSS_SOFTRESET_MMUART1_SOFTRESET_LOCK            1
 MSS_SOFTRESET_G4SPI0_SOFTRESET_LOCK             1
 MSS_SOFTRESET_G4SPI1_SOFTRESET_LOCK             1
 MSS_SOFTRESET_I2C0_SOFTRESET_LOCK               1
 MSS_SOFTRESET_I2C1_SOFTRESET_LOCK               1
 MSS_SOFTRESET_CAN_SOFTRESET_LOCK                1
 MSS_SOFTRESET_USB_SOFTRESET_LOCK                1
 MSS_SOFTRESET_COMBLK_SOFTRESET_LOCK             1
 MSS_SOFTRESET_FPGA_SOFTRESET_LOCK               1
 MSS_SOFTRESET_HPDMA_SOFTRESET_LOCK              1
 MSS_SOFTRESET_FIC32_0_SOFTRESET_LOCK            1
 MSS_SOFTRESET_FIC32_1_SOFTRESET_LOCK            1
 MSS_SOFTRESET_MSS_GPIO_SOFTRESET_LOCK           1
 MSS_SOFTRESET_MSS_GPOUT_7_0_SOFT_RESET_LOCK     1
 MSS_SOFTRESET_MSS_GPOUT_15_8_SOFT_RESET_LOCK    1
 MSS_SOFTRESET_MSS_GPOUT_23_16_SOFT_RESET_LOCK   1
 MSS_SOFTRESET_MSS_GPOUT_31_24_SOFT_RESET_LOCK   1
 MSS_SOFTRESET_MDDR_CTLR_SOFTRESET_LOCK          1
 MSS_SOFTRESET_MDDR_FIC64_SOFTRESET_LOCK         1
 MSS_M3_CONFIG_LOCK                              1
```

2.11.3 Locking and Unlocking a Register (Ask a Question)

A register can be locked or unlocked by setting the appropriate lock bit value in the lock bit configuration .txt file.

1. Browse to locate the lock bit configuration .txt file.

2. Do one or both of the following:
   - Set the lock bit value to 0 for the registers you want to lock.
   - Set the lock bit value to 1 for the registers you want to unlock.

3. Save the file, and import the file into the project **(Design Flow window > Configure Register Lock Bits)**.

**Figure 2-54.** Register Lock Bit Settings Window
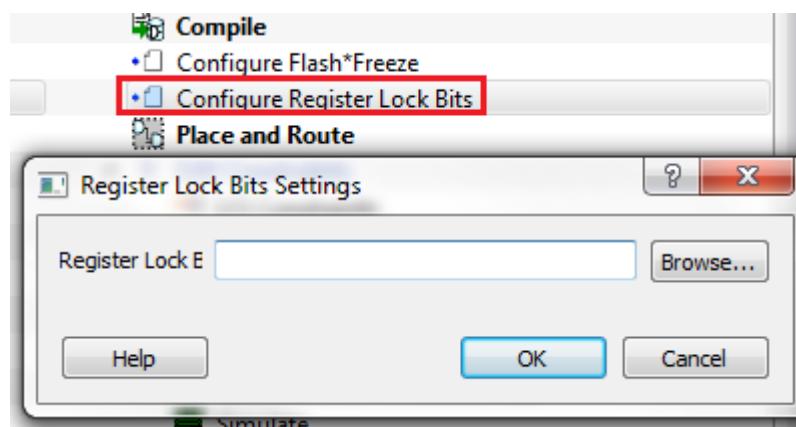
4.   Regenerate the bitstream.

3.   DDR Bridge (Ask a Question)

The DDR bridge facilitates multiple AHB bus masters to access a single AXI slave and optimizes read and write operations from multiple AHB masters to a single external DDR memory. The SmartFusion 2 and IGLOO 2 devices have three instances of the DDR bridge, one each for the MSS/HPMS, FDDR, and MDDR subsystems, as shown in the following image. The DDR bridge implemented in the MSS/HPMS (shown in red) provides an interface between AHB masters within the MSS/HPMS for accessing DDR memory. The DDR bridge implemented in the MDDR subsystem (shown in green) provides an interface between the user implemented AHB masters in the FPGA fabric for accessing DDR memory. Similarly, the DDR bridge in the FDDR (shown in blue) subsystem facilitates fabric masters to access DDR memory.

Figure 3-1. DDR Bridges in the SmartFusion 2/IGLOO 2 FPGA Device
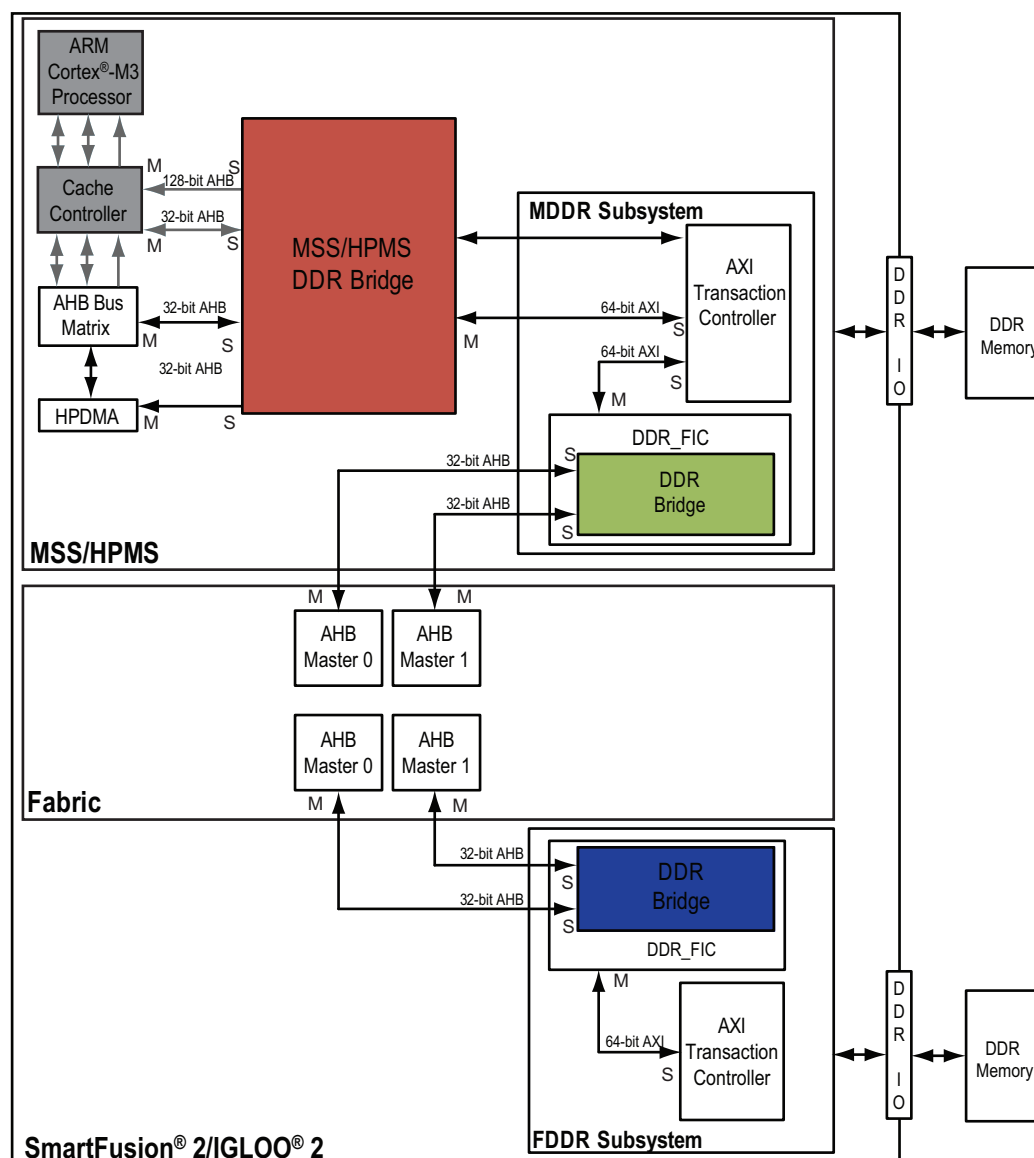
Note: Grey blocks and arrows indicate the steps happen only in MSS. Rest are same in SmartFusion2 and IGLOO2.

The DDR bridge supports a single 64-bit AXI and up to four 32-bit AHB interfaces. For SmartFusion 2 the four MSS AHB masters are fixed, as shown in the following table. For IGLOO 2, the two

HPMS AHB masters are fixed, as shown in the following table. The DDR bridges in the MDDR and FDDR subsystems support only two AHB interfaces out of four and these can be used for user implemented AHB masters.

Table 3-1. SmartFusion® 2 and IGLOO® 2 FPGA DDR Bridge Interface

Sub-System	DDR Bridge				
	AHB Interface 0 Read Only	AHB Interface 1 R/W	AHB Interface 2 R/W	AHB Interface 3 R/W	AXI Interface
HPMS	Not available	Not available	AHB bus matrix	HPDMA	MDDR subsystem
MSS	Cache Controller IDC	Cache Controller DS	AHB bus matrix	HPDMA	MDDR subsystem
MDDR	Not available	Not available	AHB master interface 0	AHB master interface 1	MDDR subsystem
FDDR	Not available	Not available	AHB master interface 0	AHB master interface 1	FDDR subsystem

Important: If the AXI bus is selected as the interface between the FPGA fabric and the MDDR/ FDDR subsystem, the DDR bridge in these subsystems is not used.

3.1 Functional Description (Ask a Question)

This section provides the detailed description of the DDR Bridge, which contains the following sections:

- 3.1.1. Architecture Overview
- Details of Operation

3.1.1 Architecture Overview (Ask a Question)

The DDR bridge consists of two main components: read and write combining buffers (WCB), and an arbiter, as shown in the following illustration. The DDR bridge buffers AHB write transactions into write combining buffers before bursting out to external DDR memory. It also includes read buffers for AHB masters to efficiently read data from the external DDR memory. All buffers within the DDR bridge are implemented with latches and hence are not subject to SEUs. The external DDR memory regions can be configured to be non-bufferable. If a master interface requests a write or read to a non-bufferable region, the DDR bridge is essentially bypassed. The size of the non-bufferable address space can also be configured.

Figure 3-2. DDR Bridge Functional Block Diagram
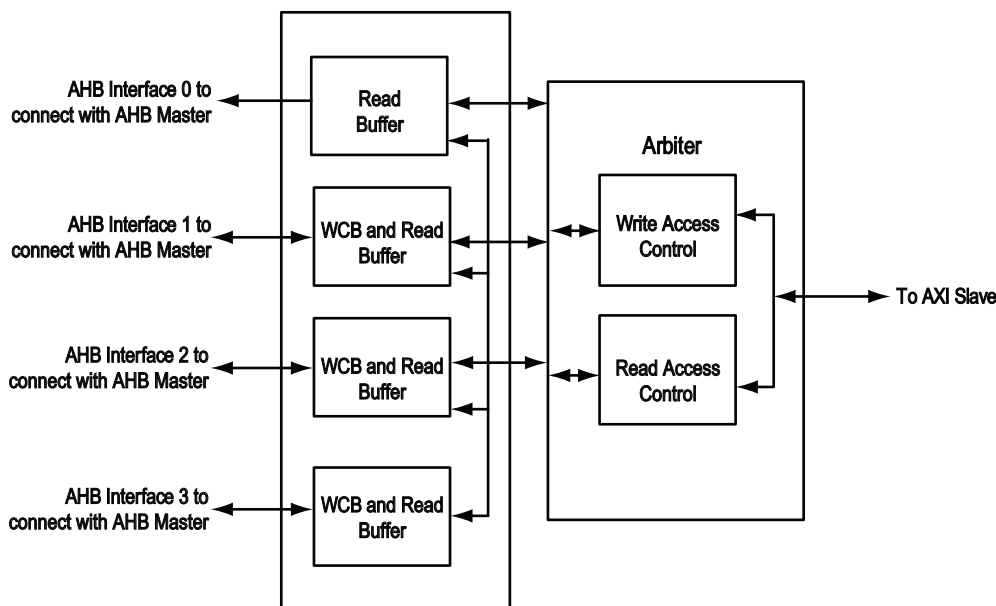
Arbitration between the four AHB interfaces is handled as follows:

- Fixed priority between AHB Interfaces 0 and 1, with 0 having the highest priority
- Round robin arbitration between interfaces 2 and 3

3.1.2 Details of Operation (Ask a Question)

This section provides a functional description of each block in the DDR Bridge, as shown in the previous illustration.

3.1.2.1 Write Combining Buffer (Ask a Question)

The Write Combining Buffer (WCB) combines multiple write transactions from the AHB master into AXI burst transactions. The WCB has a user configurable burst size of 16 or 32 bytes. Each WCB maintains a base address tag that stores the base address of the data to be combined in the buffer.

For each write transaction, the address is compared with the WCB tag. If the address matches the tag, data is combined into the buffer. The WCB writes to the correct byte location based on the offset address of the data. WCB can also be disabled, if buffering is not required.

The WCB has a 10-bit timer (down counter), which starts when the first bufferable write data is loaded into the WCB. The timer starts decrementing its value at every positive edge of the AHB clock and when it reaches zero, the data in the WCB is written to the AXI slave.

The WCB checks for any other master that has initiated a read to the same address for which data is already present in a write buffer or for which a write operation is ongoing. If the address for a read request matches the write buffer tag, the read request is held until the buffer is written completely to the AXI slave.

The following illustration shows the flowchart for WCB operation.

Figure 3-3. WCB Operation
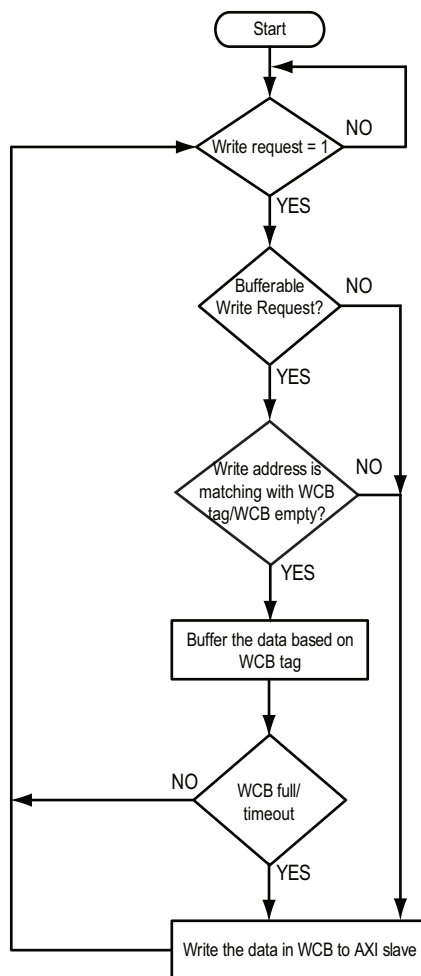
3.1.2.2 Read Buffer (Ask a Question)

The DDR bridge has a read buffer for each master to hold the fetched DDR burst data. Each read buffer has a configurable burst size of 16 or 32 bytes. The read buffer initiates a DDR burst size request for reads in the bufferable region, regardless of the size of request from the master. Each read buffer is associated with one specific master for reading; it does not check the read addresses of other masters to determine whether that data can be read from the read buffer-there is no cross buffer read access. The following illustration shows the flowchart for read operation.

Figure 3-4. Flow Chart for Read Operation
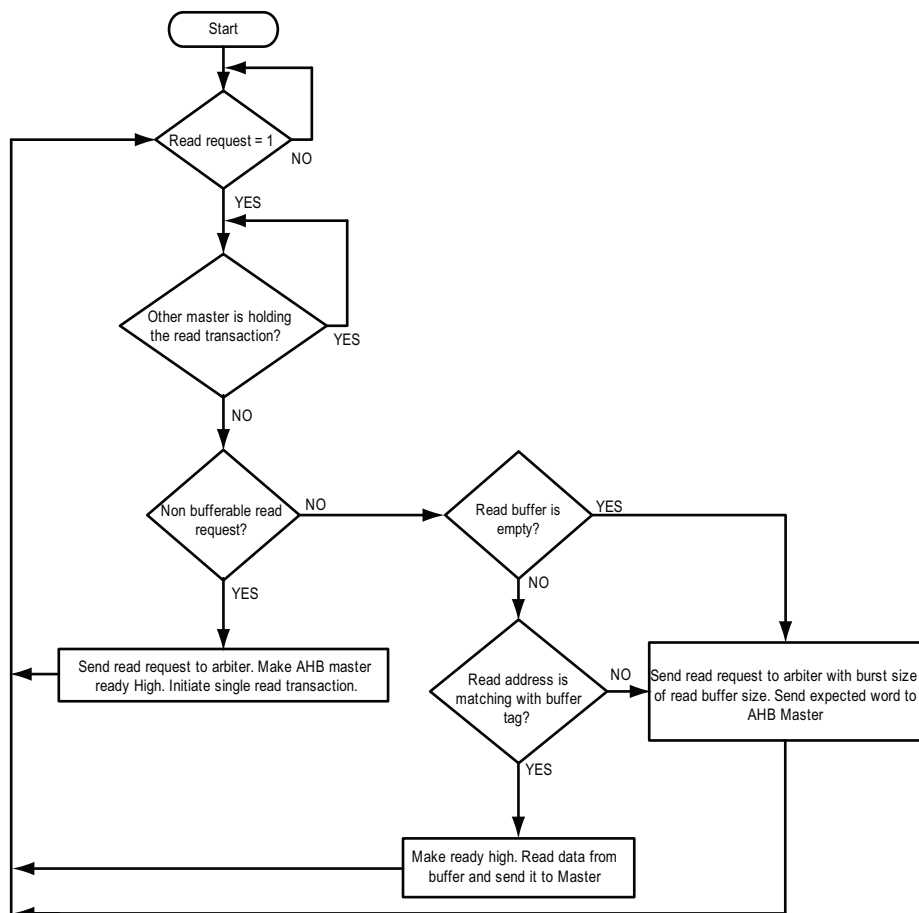
The read buffer is invalidated under the following conditions:

- If the address from the master is outside the TAG region, the current data in the read buffer is invalidated (TAG mismatch).
- To ensure proper data coherency, every master's write address is tracked. If an address matches that of the read buffer TAG, the read entry is invalidated.
- A non-bufferable or locked transaction is initiated by any master.
- An Invalidate command is issued.
- A buffer disable command is issued.
- An error response from DDR for the expected word read.

3.1.2.3   Arbiter (Ask a Question)
The DDR bridge arbiter includes two independent arbitration controllers for read and write requests.

3.1.2.3.1 Write Access Controller (Ask a Question)
The Write Access Controller (WAC) arbitrates write requests from the WCBs and grants access to one of the requesting masters based on its priority. All transactions from a single master have a dedicated master ID.

Combinations of fixed and round robin priorities are assigned to the following masters:

- Master Interface 1: Fixed first priority (Master Interface 0 is read only)
- Round robin between Master Interface 2 and Master Interface 3 for second and third priorities

MICROCHIP

Once a burst transaction is initiated to the external DDR memory, the transactions are completed without an interruption. No other master, even a high priority master, can interrupt this process. Subsequent write requests from the same master are held until the previous write transactions are completed to the external DDR memory. Subsequent write requests from other masters can be accepted and allowed to write into WCB, but the DDR bridge does not write this data until the previous write transactions are completed to the external DDR memory.

3.1.2.3.2 Read Access Controller (Ask a Question)

The Read Access Controller (RAC) arbitrates read requests from read buffers and grants access to one of the requesting masters depending on its priority.

Combinations of fixed and round robin priorities are assigned to the masters as below:

- Master Interface 0 and Master Interface 1 have fixed first and second priority
- Round robin between Master Interface 2 and Master Interface 3 for second and third priority

The RAC also routes the read data from the AXI slave (MDDR or FDDR) to the corresponding master based on the Read data ID.

3.1.2.3.3 Locked Transactions (Ask a Question)

The DDR bridge masters can initiate locked transfers by asserting the HMASTLOCK signal of the corresponding AHB interface. These locked transactions are initiated only after all the pending write and read transactions are completed.

The arbiter has a 20-bit up counter for detecting a lock timeout condition. The counter starts counting when a locked transaction is initiated on the bus. When the counter reaches its maximum value, an interrupt is generated. The interrupt can be cleared by setting the DDR_LOCKOUT bit in the MSS_EXTERNAL_SR from the SYSREG block. In SmartFusion 2, when the counter reaches its maximum value, an interrupt is generated to the Cortex-M3 processor. The error routine has to be stored in either eNVM or eSRAM for the Cortex-M3 processor to fetch the Interrupt Service Routine (ISR) without going through the DDR bridge. As part of the ISR, the Cortex-M3 processor reads the SYSREG registers to identify the master and take appropriate action to release the arbiter from dead lock. If the interrupt is cleared and the lock signal is still asserted, the counter will start counting again.

3.2 How to Use DDR Bridge in IGLOO 2 Device (Ask a Question)

This section describes how to use DDR bridge. To configure the IGLOO 2 device features and then build a complete system, use the System Builder graphical design wizard in the Libero SoC software.

The following illustration shows the initial System Builder window where you can select the features that you require. For information on how to launch the System Builder wizard and how to use it, see IGLOO2 System Builder User's Guide.

Figure 3-5. System Builder - Device Features Window

Navigate to the **HPMS Options** tab in the System Builder wizard.

For more information about how to use MDDR in SmartFusion 2 devices, see 3.5. How to Use DDR Bridge in SmartFusion 2 Device.

3.2.1 Configuring the DDR Bridge (Ask a Question)

The following sections describe the configuration options for the HPMS DDR Bridge.

3.2.1.1 HPMS DDR Bridge Configurations (Ask a Question)

The HPMS DDR bridge is statically configured through the DDR bridge configurator in the System Builder wizard. The following illustration shows the configuration options.

- **Write buffer time out counter**: This allows the user to configure the 10-bit timer of write buffer for time out value. By default this is configured for maximum wait time (0×3FF) to buffer the write transactions. For configuring to other values enter a 10-bit hexadecimal value in the provided field of DDR bridge configurator. Select timeout value to a non-zero value for buffering the write transactions.
- **Non-bufferable region size**: The size of non-bufferable memory region can be selected from a drop-down menu in the DDR bridge configurator. The menu has the options to select the region from 64 KB to 1 GB. It also has an option None to select the complete memory as bufferable. The default selection is 64 KB.
- **Non-bufferable region address**: The base address of the non-bufferable memory region can be selected by configuring this field. The value must be configured as a 16-bit hexadecimal address. The default address is 0×A000. If the non-bufferable region size and address is left

as default then the 64 KB memory from 0×A0000000 address to 0×A0010000 address will be non-bufferable.

Enable or disable respective buffers allocated for each master. The selection of disabling the write/read buffer makes all the transactions without buffering. By default, buffering is enabled. Select the DDR burst size for read/write buffers. The DDR bridge configurator allows to select the size of read/write buffers as 32 bytes or 16 bytes.

Figure 3-6. Configuring HPMS DDR Bridge for HPDMA

3.2.1.2   Configurations for the DDR Bridge in the MDDR or FDDR Subsystems (Ask a Question)

The DDR bridge in the MDDR or FDDR subsystem can be configured using the DDR_FIC registers listed in Table 3-3. The possible configurations and corresponding registers are:

- Enable or disable the write and read buffers of the DDR bridge using the DDR_FIC_HPD_SW_RW_EN_CR register
- Configure buffer size to 32 bytes or 16 bytes using the DDR_FIC_NBRWB_SIZE_CR register
- Configure the non-bufferable address using the DDR_FIC_NB_ADD register
- Configure the non-bufferable size using the DDR_FIC_NBRWB_SIZE_CR register
- Configure the timeout value for each write buffer using the DDR_FIC_LOCK_TIMEOUTVAL_1_CR and DDR_FIC_LOCK_TIMEOUTVAL_2_CR registers.

Set the timeout value to maximum or a non-zero value.

The configuration registers for the MDDR bridge and FDDR bridge are also listed under the 1.11.6. DDR_FIC Configuration Registers Summary section in the MDDR and FDDR chapters.

3.2.2 High-Speed Data Transactions from HPDMA (Ask a Question)

This section describes the use of the DDR bridge to increase the throughput from the HPDMA to the external DDR memories. The HPDMA performs only the single read and write transactions and not the burst transactions. The DDR bridge converts these single transactions into burst transactions and further increases the throughput. The HPDMA buffers are enabled for this, and the non-bufferable size is selected as **None**, as shown in the following image.

Figure 3-7. Configuring HPMS DDR Bridge For Non-Bufferable Region

3.2.3 Selecting Non-Bufferable Region (Ask a Question)

This section describes the use of the non-bufferable region selection in the DDR bridge. The buffering creates more latency in the applications which access non-continuous memory locations. In such cases non-bufferable region selection provides high throughput than bufferable. The application uses only 256 MB of memory segment (0xB000_0000 to 0XBFFF_FFFF) as non-bufferable and the other memory region as bufferable. The following image shows the selection of the non-bufferable region.

Figure 3-8. Configuring DDR Bridge

3.3 SYSREG Control Registers (Ask a Question)

The following table lists HPMS DDR bridge Control registers in the SYSREG block. Refer to the System Register Map chapter of the UG0448: IGLOO2 High Performance Memory Subsystem User Guide for a detailed description of each register and bit.

Table 3-2. SYSREG Control Registers

Register Name	Register Type	Flash Write Protect	Reset Source	Description
DDRB_BUF_TIMER_CR	RW-P	Register	SYSRESET_N	Uses a 10-bit timer interface to configure the timeout register in the write buffer module
DDRB_NB_ADDR_CR	RW-P	Register	SYSRESET_N	Indicates the base address of the non-bufferable address region
DDRB_NB_SIZE_CR	RW-P	Register	SYSRESET_N	Indicates the size of the non-bufferable address region
DDRB_CR	RW-P	Register	SYSRESET_N	HPMS DDR bridge configuration register
DDRB_HPD_ERR_ADR_SR	RO	—	SYSRESET_N	HPMS DDR bridge high performance DMA master error address status register
DDRB_SW_ERR_ADR_SR	RO	—	SYSRESET_N	HPMS DDR bridge switch error address status register
DDRB_BUF_EMPTY_SR	RO	—	SYSRESET_N	HPMS DDR bridge buffer empty status register
DDRB_DSBL_DN_SR	RO	—	SYSRESET_N	HPMS DDR bridge disable buffer status register
DDRB_STATUS	RO	—	SYSRESET_N	Indicates HPMS DDR bridge status
MSS_EXTERNAL_SR	SW1C	—	SYSRESET_N	HPMS external status register
MSSDDR_FACC1_CR	RW-P	Field	CC_RESET_N	HPMS DDR fabric alignment clock controller 1 configuration register

3.4 DDR Bridge Control Registers in MDDR and FDDR (Ask a Question)

The following table lists HPMS DDR bridge control registers in the MDDR and FDDR. Refer to the 1. MDDR Subsystem and the 2. Fabric DDR Subsystem for a detailed description of each register and bit.

Table 3-3. DDR Bridge Control Registers in MDDR and FDDR

Register Name	Address Offset	R/W	Reset Source	Description
DDR_FIC_NB_ADDR_CR, Table 1-216	0×400	RW	PRESET_N	Indicates the base address of the non-bufferable address region
DDR_FIC_NBRWB_SIZE_CR, Table 1-217	0×404	RW	PRESET_N	Indicates the size of the non-bufferable address region
DDR_FIC_BUF_TIMER_CR, Table 1-218	0×408	RW	PRESET_N	10-bit timer interface used to configure the timeout register
DDR_FIC_HPD_SW_RW_EN_CR, Table 1-219	0×40C	RW	PRESET_N	Enable write buffer and read buffer register for AHB-Lite (AHBL) master1 and master2
DDR_FIC_HPD_SW_RW_INVAL_CR, Table 1-220	0×410	RW	PRESET_N	Invalidates write buffer and read buffer for AHBL master1 and master2
DDR_LOCK_TIMEOUTVAL_1_CR, Table 1-231	0×440	RW	PRESET_N	Indicates maximum number of cycles a master can hold the bus for a locked transfer
DDR_LOCK_TIMEOUTVAL_2_CR, Table 1-232	0×444	RW	PRESET_N	Indicates maximum number of cycles a master can hold the bus for a locked transfer

3.5 How to Use DDR Bridge in SmartFusion 2 Device (Ask a Question)

This section describes how to use DDR Bridge in an application and contains the following sub-sections:

- 3.5.1. MSS DDR Bridge Configurations
- 3.5.3. Use Model 1: High Speed Data Transactions from Cortex-M3 Processor
- 3.5.4. Use Model 2: Selecting Non-Bufferable Region

3.5.1 MSS DDR Bridge Configurations (Ask a Question)

The MSS DDR bridge is statically configured through the DDR bridge configurator of the MSS configurator in Libero SoC, as shown in the following image. Configurable parameters are as follows:

- **Write buffer time out counter**: This allows to configure the 10-bit timer of write buffer for time out value. By default this is configured for maximum wait time (0×3FF) to buffer the write transactions. For configuring to other values enter a 10-bit hexadecimal value in the provided field of DDR bridge configurator. Select timeout value to a non zero value for buffering the write transactions.

- **Non-bufferable region size**: The size of non-bufferable memory region can be selected from a drop-down menu in the DDR bridge configurator. The menu has the options to select the region from 64 KB to 1 GB. It also has an option "none" to select the complete memory as bufferable. The default selection is 64 KB.

- **Non-bufferable region address**: The base address of the non-bufferable memory region can be selected by configuring this field. The value must be configured as a 16-bit hexadecimal address. The default address is 0×A000. If the non-bufferable region size and address is left as default then the 64 KB memory from 0×A0000000 address to 0×A0010000 address will be non-bufferable.

- **Enable or disable respective buffers allocated for each master**: The selection of disabling the write/read buffer makes all transactions without buffering. By default, buffering is enabled.

- **DDR burst size for read/write buffers**: The DDR bridge configurator allows to select the size of read/write buffers as 32 bytes or 16 bytes.

- **Figure 3-9.** Configuring MSS DDR Bridge
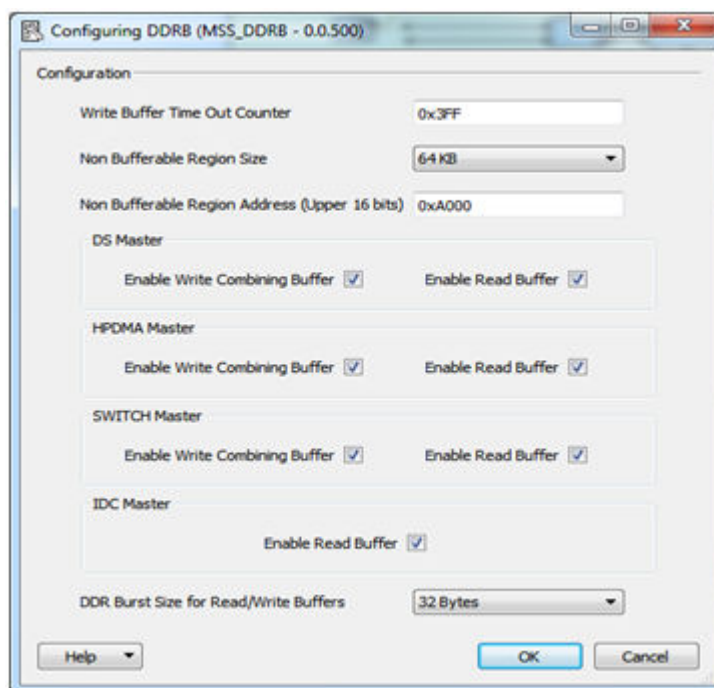

3.5.2 MDDR/FDDR DDR Bridge Configurations (Ask a Question)

The DDR bridge in the MDDR or FDDR subsystem can be configured through the DDR_FIC registers shown in Table 3-3. The possible configurations and corresponding registers are as follows:

- Enable or disable the write and read buffers of the DDR bridge using the DDR_FIC_HPD_SW_RW_EN_CR register.
- Configure buffer size to 32 bytes or 16 bytes using the DDR_FIC_NBRWB_SIZE_CR register.
- Configure the non-bufferable address using the DDR_FIC_NB_ADD register.
- Configure the non-bufferable size using the DDR_FIC_NBRWB_SIZE_CR register.
- Configure the timeout value for each write buffer using the DDR_FIC_LOCK_TIMEOUTVAL_1_CR and DDR_FIC_LOCK_TIMEOUTVAL_2_CR registers. Set the timeout value to maximum or a non- zero value.

The configuration registers for the MDDR DDR bridge and FDDR DDR bridge are also listed under the DDR FIC registers section in the MDDR and FDDR chapters.

3.5.3 Use Model 1: High Speed Data Transactions from Cortex®-M3 Processor (Ask a Question)

This use model shows the use of the DDR bridge for increasing throughput from the Cortex®-M3 processor to external DDR memories. The Cortex®-M3 processor performs only the single read and write transactions; not the burst transactions. The DDR bridge converts these single transactions into burst transactions and further increases the throughput. The buffers for DS and IDC masters are enabled for this, and the non-bufferable size is selected as **None**, as shown in the following image.

Figure 3-10. Configuring MSS DDR Bridge for Use Model 1
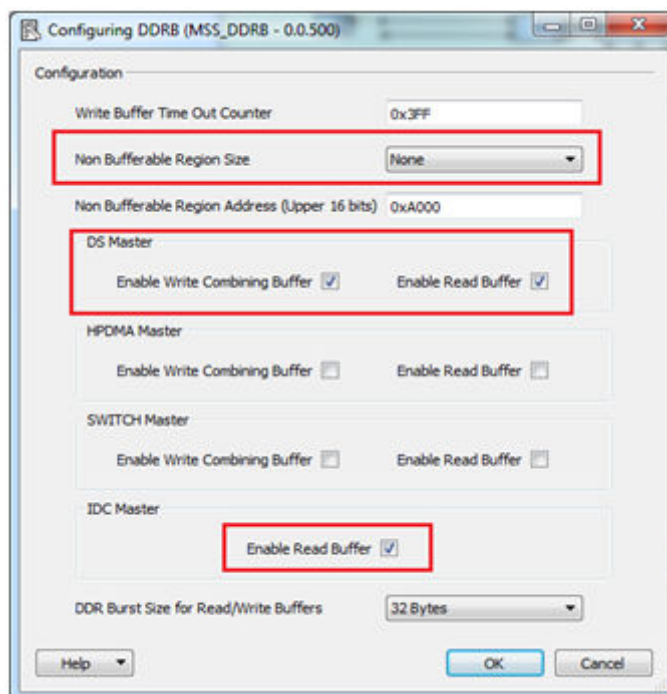
3.5.4   Use Model 2: Selecting Non-Bufferable Region (Ask a Question)

This use model shows the use of the non-bufferable region selection in the DDR bridge. The buffering creates more latency in the applications which access non-continuous memory locations. In such cases non-bufferable region selection provides high throughput than bufferable. For example, when Cortex-M3 processor fetches the data from data region that is, stack and the application has bulk data transactions then keeping the data region as bufferable and code region as non-bufferable is preferred.

In this use model, the application uses only 256 MB of memory segment (0xB000_0000 to 0XBFFF_FFFF) as non-bufferable and the other memory region as bufferable. The following image shows the selection of the non-bufferable region.

Figure 3-11. Configuring MSS DDR Bridge for Use Model 2

© 2023 Microchip Technology Inc. and its subsidiaries
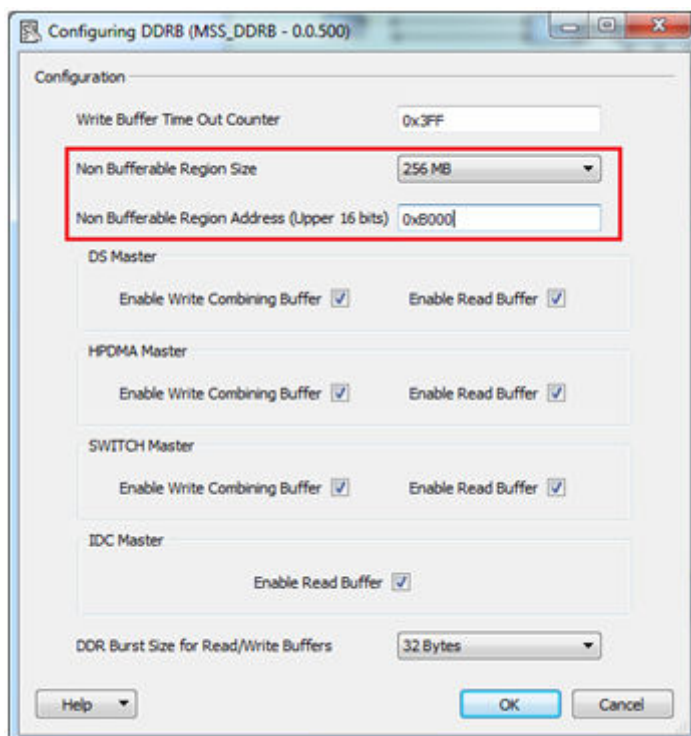
4.   Soft Memory Controller Fabric Interface Controller (Ask a Question)

The SmartFusion 2 and IGLOO 2 soft memory controller fabric interface controller (SMC_FIC) is used to access external bulk memories other than DDR through the FPGA fabric. The SMC_FIC can be used with a soft memory controller for the MSS/HPMS to access memories such as SDRAM, Flash, and SRAM. MSS/HPMS masters communicate with the SMC_FIC through an MSS/HPMS DDR bridge present in the MSS/HPMS.

If the SMC_FIC is enabled, the MDDR subsystem will not be available. In SMC_FIC mode, the DDRIOs associated with the MDDR subsystem are available for user applications.

The following illustration shows a soft memory controller instantiated in the FPGA fabric for interfacing with external memory.

Figure 4-1. System Level SMC_FIC Block Diagram
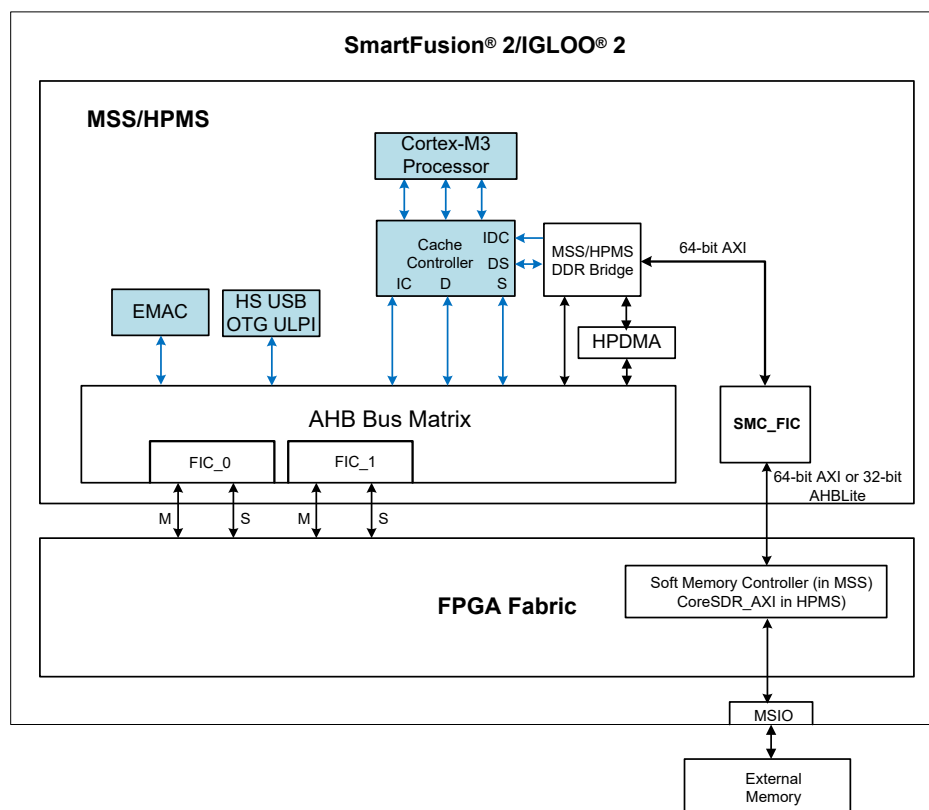
Note: Blue arrows and blocks refer to the flow only in MSS. Rest are similar in MSS and HPMS.

4.1   Functional Description (Ask a Question)

The SMC_FIC receives 64-bit AXI transactions from the MSS/HPMS DDR bridge and converts them into 64-bit AXI or 32-bit AHB-Lite transactions to the SMC in the FPGA fabric. The following illustration shows the block diagram of the SMC_FIC. The SMC_FIC has two bridges:

• The AXI-AHB bridge converts 64-bit AXI transactions into 32-bit AHB transactions. It implements the AXI master to AHBL master protocol translator. This bridge is enabled when the SMC_FIC is configured for a 32-bit AHB interface.

- The AXI-AXI bridge facilitates 64-bit AXI transactions from the MSS/HPMS DDR bridge to the 64-bit AXI FPGA fabric interface. This bridge is enabled when the SMC_FIC is configured for a 64-bit AXI interface.

The SMC_FIC receives a clock from the MSS/HPMS CCC that is identical to M3_CLK/HPMS_CLK. HPMS peripherals can access the external memory with the address space 0xA0000000 to 0xD0000000.

Figure 4-2. SMC_FIC Block Diagram
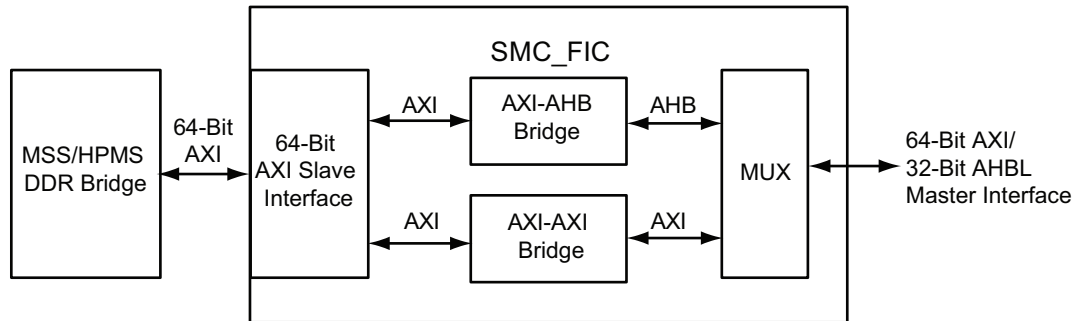
> **Important:** The Libero 11.2 System Builder configures the SMC_FIC in AHB mode for the devices M2GL005, M2GL010, and M2GL025. For other devices, it configures the SMC_FIC in AXI mode.

4.1.1 Port List (Ask a Question)

The following two tables show the 64-bit AXI and 32-bit AHBL port lists.

> **Important:** The SMC_FIC in M2S005, M2S010, M2S025, M2GL005, M2GL010, and M2GL025 devices provides only one 32-bit AHB-Lite interface.

The AXI interface has the following limitations:

- Supports only 64-bit read/write transactions on the AXI slave interface
- Exclusive access cycles are not supported

The SMC_AXI FIC AXI Read transactions can only be any of the following:

- Single transfer of 64 bit only aligned to 64-bit addresses
- Wrap Transactions of 64-bit size and Wrap burst length of 2 which are aligned to 128-bit (16 byte) addresses
- Wrap Transactions of 64-bit size and Wrap burst length of 4 which are aligned to 256-bit (32 byte) addresses

Table 4-1. SMC_FIC 64-bit AXI Port List

Signal	Direction	Polarity	Description
MDDR_SMC_AXI_M_WLAST	Output	High	Indicates the last transfer in a write burst.
MDDR_SMC_AXI_M_WVALID	Output	High	Indicates whether or not valid write data and strobes are available. • 1: Write data and strobes available • 0: Write data and strobes not available

..........continued

Signal	Direction	Polarity	Description
MDDR_SMC_AXI_M_BREADY	Output	High	Indicates whether or not the master can accept the response information. • 1: Master ready • 0: Master not ready
MDDR_SMC_AXI_M_AWVALID	Output	High	Indicates whether or not valid write address and control information are available. • 1: Address and control information available • 0: Address and control information not available
MDDR_SMC_AXI_M_ARVALID	Output	High	Indicates whether or not valid read address and control information are available. • 1: Address and control information valid • 0: Address and control information not valid
MDDR_SMC_AXI_M_RREADY	Output	High	Indicates whether or not the master can accept the read data and response information. • 1: Master ready • 0: Master not ready
MDDR_SMC_AXI_M_AWREADY	Input	High	Indicates that the slave is ready to accept an address and associated control signals. • 1: Slave ready • 0: Slave not ready
MDDR_SMC_AXI_M_WREADY	Input	High	Indicates whether or not the slave can accept the write data. • 1: Slave ready • 0: Slave not ready
MDDR_SMC_AXI_M_BVALID	Input	High	Indicates whether or not a valid write response is available. • 1: Write response available • 0: Write response not available
MDDR_SMC_AXI_M_ARREADY	Input	High	Indicates whether or not the slave is ready to accept an address and associated control signals. • 1: Slave ready • 0: Slave not ready
MDDR_SMC_AXI_M_RLAST	Input	High	Indicates the last transfer in a read burst.
MDDR_SMC_AXI_M_RVALID	Input	High	Indicates whether or not the required read data is available and the read transfer can complete. • 1: Read data available • 0: Read data not available

..........continued

Signal	Direction	Polarity	Description
MDDR_SMC_AXI_M_AWLEN[3:0]	Output	—	Indicates burst length. The burst length gives the exact number of transfers in a burst. This information determines the number of data transfers associated with the address. • 0000: 1 • 0001: 2 • 0010: 3 • 0011: 4 • 0100: 5 • 0101: 6 • 0110: 7 • 0111: 8 • 1000: 9 • 1001: 10 • 1010: 11 • 1011: 12 • 1100: 13 • 1101: 14 • 1110: 15 • 1111: 16
MDDR_SMC_AXI_M_AWBURST[1:0]	Output	—	Indicates burst type. The burst type, coupled with the size information, provides details on how the address for each transfer within the burst is calculated. • 00: FIXED – Fixed-address burst, FIFO-type • 01: INCR – Incrementing-address burst, normal sequential memory • 10: WRAP – Incrementing-address burst that wraps to a lower address at the wrap boundary • 11: Reserved
MDDR_SMC_AXI_M_AWID[3:0]	Output	—	Indicates identification tag for the write address group of signals.
MDDR_SMC_AXI_M_WDATA[63:0]	Output	—	Indicates write data.
MDDR_SMC_AXI_M_WID[3:0]	Output	—	Indicates ID tag of the write data transfer. The SMC_AXI64_WID value must match the SMC_AXI64_AWID value of the write transaction.
MDDR_SMC_AXI_M_WSTRB[7:0]	Output	—	Indicates which byte lanes to update in memory.
MDDR_SMC_AXI_M_ARID[3:0]	Output	—	Indicates identification tag for the read address group of signals.
MDDR_SMC_AXI_M_ARADDR[31:0]	Output	—	Indicates initial address of a read burst transaction.

..........continued

Signal	Direction	Polarity	Description
MDDR_SMC_AXI_M_ARLEN[3:0]	Output	—	Indicates burst length. The burst length gives the exact number of transfers in a burst. • 0000: 1 • 0001: 2 • 0010: 3 • 0011: 4 • 0100: 5 • 0101: 6 • 0110: 7 • 0111: 8 • 1000: 9 • 1001: 10 • 1010: 11 • 1011: 12 • 1100: 13 • 1101: 14 • 1110: 15 • 1111: 16
MDDR_SMC_AXI_M_ARSIZE[1:0]	Output	—	Indicates the maximum number of data bytes to transfer in each data transfer, within a burst. • 00: 1 • 01: 2 • 10: 4 • 11: 8
MDDR_SMC_AXI_M_ARBURST[1:0]	Output	—	Indicates burst type. The burst type, coupled with the size information, provides details on how the address for each transfer within the burst is calculated. • 00: FIXED – Fixed-address burst, FIFO type • 01: INCR – Incrementing-address burst, normal sequential memory • 10: WRAP – Incrementing-address burst that wraps to a lower address at the wrap boundary • 11: Reserved
MDDR_SMC_AXI_M_AWADDR[31:0]	Output	—	Indicates write address. The write address bus gives the address of the first transfer in a write burst transaction.
MDDR_SMC_AXI_M_AWSIZE[1:0]	Output	—	Indicates the maximum number of data bytes to transfer in each data transfer, within a burst. • 00: 1 • 01: 2 • 10: 4 • 11: 8

..........continued

Signal	Direction	Polarity	Description
MDDR_SMC_AXI_M_AWLOCK[1:0]	Output	—	Indicates lock type. This signal provides additional information about the atomic characteristics of the write transfer. • 00: Normal access • 01: Exclusive access • 10: Locked access • 11: Reserved
MDDR_SMC_AXI_M_ARLOCK[1:0]	Output	—	Indicates lock type. This signal provides additional information about the atomic characteristics of the read transfer. • 00: Normal access • 01: Exclusive access • 10: Locked access • 11: Reserved
MDDR_SMC_AXI_M_BID[3:0]	Input	—	Indicates response ID. The identification tag of the write response. The MDDR_SMC_AXI_M_BID value must match the MDDR_SMC_AXI_M_AWID value of the write transaction to which the slave is responding.
MDDR_SMC_AXI_M_RID[3:0]	Input	—	Read ID tag. This signal is the ID tag of the read data group of signals. The MDDR_SMC_AXI_M_RID value is generated by the slave and must match the MDDR_SMC_AXI_M_ARID value of the read transaction to which it is responding.
MDDR_SMC_AXI_M_RRESP[1:0]	Input	—	Indicates read response. This signal indicates the status of the read transfer. • 00: Normal access okay • 01: Exclusive access okay • 10: Slave error • 11: Decode error
MDDR_SMC_AXI_M_BRESP[1:0]	Input	—	Indicates write response. This signal indicates the status of the write transaction. • 00: Normal access okay • 01: Exclusive access okay • 10: Slave error • 11: Decode error
MDDR_SMC_AXI_M_RDATA[63:0]	Input	—	Indicates read data.

Table 4-2. SMC_FIC 32-bit AHB-Lite Port List

| Signal	Direction	Polarity	Description
MDDR_SMC_AHB_M_HMASTLOCK	Output	High	Indicates that the current master is performing a locked sequence of transfers.
MDDR_SMC_AHB_M_HWRITE	Output	High	Indicates write control signal. When High, this signal indicates a write transfer, and when Low, a read transfer.
MDDR_SMC_AHB_M_HRESP	Input	High	The transfer response indicates the status of transfer.
MDDR_SMC_AHB_M_HREADY	Input	High	When High, the signal indicates that a transfer has been completed on the bus. This signal may be driven Low to extend a transfer.
MDDR_SMC_AHB_M_HBURST[1:0]	Output		Indicates the burst type.

..........continued

Signal	Direction	Polarity	Description
MDDR_SMC_AHB_M_HTRANS[1:0]	Output		Indicates the type of the current transfer. • 00: Idle • 01: Busy • 10: Non-sequential • 11: Sequential
MDDR_SMC_AHB_M_HSIZE[1:0]	Output		Indicates the size of the transfer. • 00: Byte • 01: Half word • 10: Word
MDDR_SMC_AHB_M_HWDATA[31:0]	Output		The write data bus is used to transfer data during write operations.
MDDR_SMC_AHB_M_HADDR[31:0]	Output		Indicates address bus.
MDDR_SMC_AHB_M_HRDATA[31:0]	Input		The read data bus is used to transfer data from bus slaves to the bus master during read operations.

4.2 How to Use SMC_FIC in IGLOO2 Device (Ask a Question)

This section describes how to use SMC_FIC for accessing external SDR memory. The SMC_FIC can be enabled and configured using the System Builder in the Libero SoC design software. The System Builder uses the CoreSDR_AXI and connects to SMC_FIC interface. The CoreSDR_AXI IP is an AXI based SDR memory controller. The steps provided below are required to access the external SDR memory from CoreSDR_AXI.

1. Select the **HPMS External Memory, Soft Memory Controller (SMC) and HPDMA** in the **System Builder** > **Device Features** window as shown in the following image.
For information on how to launch the System Builder wizard and how to use it, see IGLOO2 System Builder User Guide.

Figure 4-3. HPMS External Memory Configurator
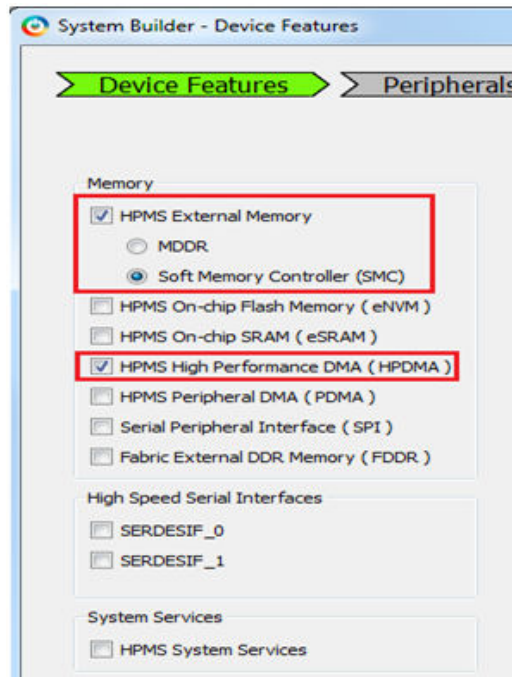
For more information on how to use SMC_FIC in SmartFusion 2 Device, see 4.4. How to Use SMC_FIC in SmartFusion 2 Devices.

2. Click **Next** to get the **Peripherals** window. Click **Configure** icon under **HPMS SMC_FIC** subsystem as shown in the following image.

Figure 4-4. HPMS SMC_FIC Subsystem Configuration
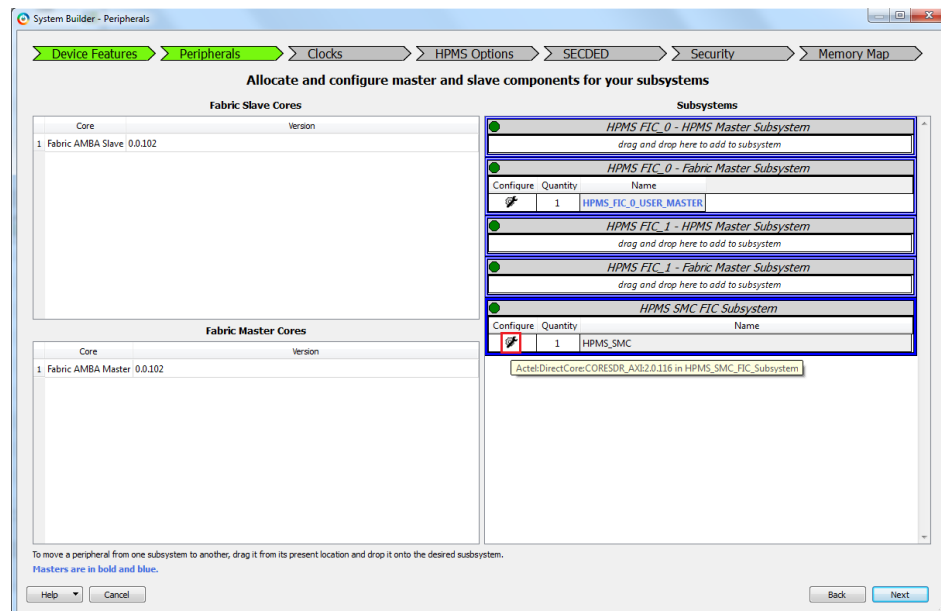
3. Configure CoreSDR_AXI to match the external memory parameters.

   Figure 4-5. CoreSDR_AXI Configuration

4. Navigate to the **Memory Map** tab giving the required data in the rest of the System Builder tabs. Click **Finish**.

5. The System Builder creates a SmartDesign with CoreSDR_AXI connected to SMC_FIC and exposes the AHB mirrored master interface which is connected to FIC_0 to access the HPDMA configuration registers.

6. Microchip provides CoreHPDMACtrl IP to configure the HPDMA. Connect the CoreHPDMACtrl IP to the AHB mirrored master interface of System Builder created design or connect user AHB master logic to configure the HPDMA to perform the DMA transactions from SDRAM.

4.3 SYSREG Control Register for SMC_FIC (Ask a Question)

Complete descriptions of each register and bit are located in the "System Register Map" chapter of the IGLOO2 High Performance Memory Subsystem User Guide and are listed as follows for clarity.

Table 4-3. MDDR_CR Register

Register Name	Register Type	Flash Write Protect	Reset Source	Description
MDDR_CR	RW-P	Register	PORESET_N	MDDR configuration register

4.4 How to Use SMC_FIC in SmartFusion 2 Devices (Ask a Question)

This section describes how to use SMC_FIC in an application and contains the following sections:

4.4.1   Design Flow (Ask a Question)

The SMC_FIC can be enabled and configured through the MSS external memory configurator, which is part of the MSS configurator in the Libero SoC design software. The following image shows the MSS external memory configurator. The external memory type interface must be selected as **Application Accesses Single Data Rate Memory from MSS** to enable the SMC_FIC.

Select the type of interface as AXI or AHB-32. After completing the configuration, the selected interface is exposed in SmartDesign. This interface must be connected to the SMC through CoreAXI or CoreAHB.

Microchip provides CoreSDR_AHB and CoreSDR_AXI SMC IPs for interfacing with external SDRAM. Any other custom soft memory controller can also be implemented in the FPGA fabric to access the external memories.

Figure 4-6. MSS External Memory Configurator
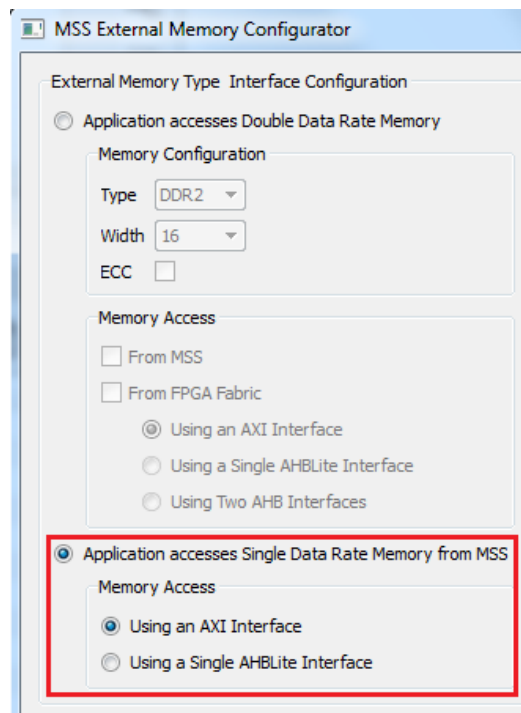
4.4.2   Use Model 1: Accessing SDRAM from MSS Through CoreSDR_AXI (Ask a Question)

This use model describes how to use the SMC_FIC to access external SDR memory from MSS. It uses the AXI interface of SMC_FIC to connect to CoreSDR_AXI. CoreSDR_AXI is an AXI-based SDR memory controller. The steps provided below are required to access the external SDR memory from CoreSDR_AXI.

1. Instantiate the SmartFusion 2 MSS component onto the SmartDesign canvas.
2. Configure the SmartFusion 2 MSS peripheral components to meet application needs using MSS configurator.
3. Configure the external memory interface type and select Using an AXI Interface, as shown in the previous image.

4. Instantiate and configure CoreAXI so that the master slot M0 is enabled for the slave slot S0, as shown in the following image. The slot size selection must be matched with the amount of external memory space.

Figure 4-7. Core_AXI Configuration
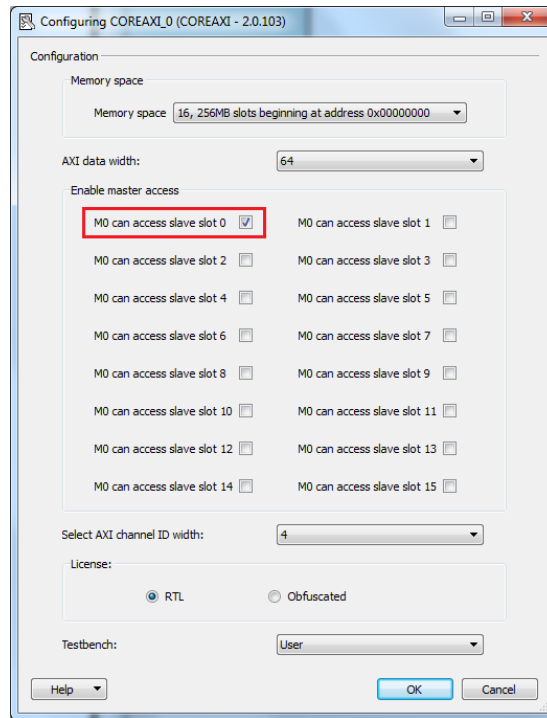
5. Instantiate and configure CoreSDR_AXI to match the external memory parameters.

6. Connect the subsystem together as shown in the following image. Connect the MSS SMC_FIC master interface port, MDDR_SMC_AXI_MASTER, to the CoreAXI bus mirrored-master M0. Connect the CoreAXI mirrored-slave bus interface (BIF) port S0 to the slave BIF port of the CoreSDR_AXI core instance.

Figure 4-8. Subsystem Connections in SmartDesign
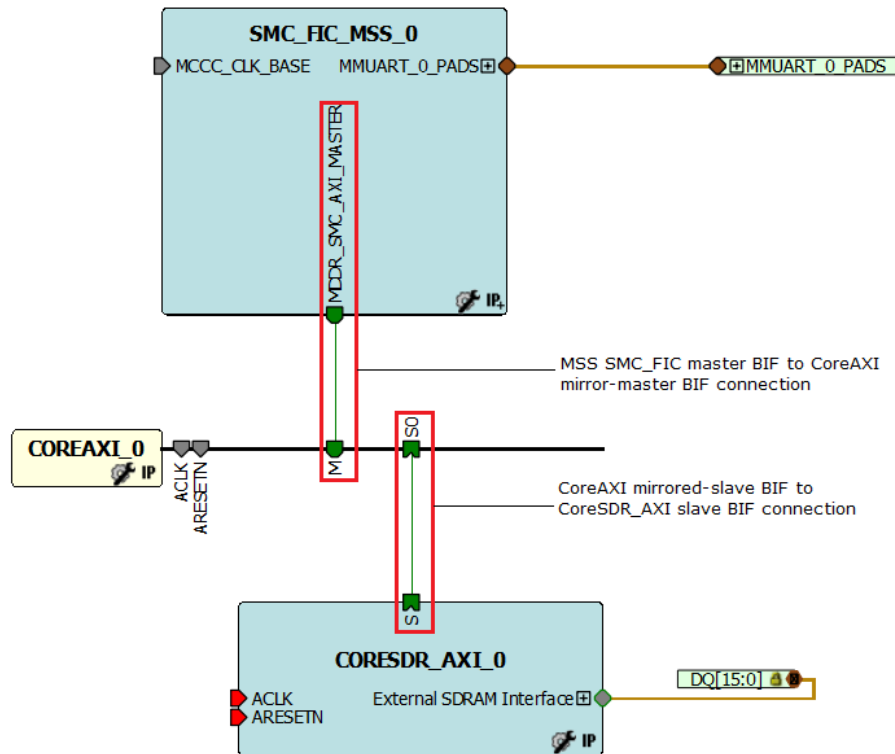
See Accessing External SDRAM through Fabric tutorial, which describes the steps for creating a design that accesses external SDR memory from the Cortex-M3 processor. The tutorial also explains the steps for simulating the design in Libero SoC.

5.   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.

Table 5-1. Revision History

Revision	Date	Description

B	11/2023	The following is a summary of the changes made in this revision:
		• Updated 2.7.2. Simulating the FDDR Subsystem as follows:
		  – Removed the names of vendors that provide DDR memory simulation models.
		  – Added a sentence to describe that Libero SoC does not support the in-house memory simulation model.

A	08/2023	The following is a summary of the changes made in this revision: • Converted the document to Microchip template. • The document number was changed to DS50003555 from UG0446. • Added all PHY registers, which are shown in the Libero SoC report, see 1.11.4. PHY Configuration Register Summary and 1.11.5. PHY Configuration Register Bit Definitions. • Added a note in 1.11.4. PHY Configuration Register Summary describing that the PHY register values must not be changed from Libero SoC defaults, as the write leveling is not performed.
Revision 10.0	—	The following is a summary of the changes made in this revision: • Added 2.7.2. Simulating the FDDR Subsystem. • Updated reset information in 2.6.1. Reset Sequence. • Added simulation waveform of the FDDR reset sequence in 2.6.1. Reset Sequence.
Revision 9.0	—	Modified the bit 5 (REG_DDRC_TEST_MODE bit) of the DDRC_MODE_CR as Reserved. See Table 1-31.
Revision 8.0	—	Decreased memory density supported by SmartFusion2 and IGLOO2 Fabric Double-Data Rate (FDDR) controller from 4 GB to 2 GB. Similarly, the memory density supported by the SmartFusion2 MSS Double-Data Rate (MDDR) controller is decreased from 4 GB to 2 GB. For more information, see Customer Notification (CN).
Revision 7.0	—	The following is a summary of the changes in this revision. • Read and Write leveling is not supported. Removed information about all the Read and Write leveling registers. • Most of the PHY registers have been reserved.
Revision 6.0	—	Updated 1.4. I/O Utilization, 2.4. I/O Utilization, 1.5.3.2. DDRIO Calibration, and 2.6.1.1. DDRIO Calibration.
Revision 5.0	—	The following is a summary of the changes in this revision. • Updated 1. MDDR Subsystem and 2. Fabric DDR Subsystem. • Updated Table 1-1, Table 1-3, Table 1-10. • Updated 1.5.3. Initialization and 1.5.4.3.5. Power Saving Modes. • Updated Table 1-84, Table 1-85, Table 1-91, Table 1-92. • Updated 1.5.1. Architecture Overview. • Added 1.5.3.5. DDR Memory Initialization Time. • Updated 2.11. Appendix B: Register Lock Bits Configuration.

..........continued

Revision	Date	Description
Revision 4.0	—	The following is a summary of the changes in this revision. • Merged SmartFusion 2 and IGLOO 2 User Guides. • Updated Additional Documentation. • Updated 1. MDDR Subsystem and 2. Fabric DDR Subsystem. • Updated 1. MDDR Subsystem. • Updated 2. Fabric DDR Subsystem. • Updated 3. DDR Bridge. • Updated 4. Soft Memory Controller Fabric Interface Controller.
Revision 3.0	—	The following is a summary of the changes in this revision. • Updated the Part Numbers (M2S075 to M2S090, M2S080 to M2S100, and M2S120 to M2S150) as required. • Updated 1. MDDR Subsystem. • Updated 2. Fabric DDR Subsystem. • Updated 4. Soft Memory Controller Fabric Interface Controller.
Revision 2.0	—	The following is a summary of the changes in this revision. • Restructured the user guide. • Updated 1. MDDR Subsystem. • Updated 2. Fabric DDR Subsystem. • Updated 4. Soft Memory Controller Fabric Interface Controller. • Updated 1.5.5.15. MDDR Memory Map. • Updated 2.6.10. Address Mapping.
Revision 1.0	—	The following is a summary of the changes in this revision. • Restructured the user guide. • Updated the user guide. • Updated 1. MDDR Subsystem, 2. Fabric DDR Subsystem, and 3. DDR Bridge. • Updated 1. MDDR Subsystem and 2. Fabric DDR Subsystem. • Updated 1. MDDR Subsystem. • Updated 2. Fabric DDR Subsystem.
Revision 0.0	—	The first publication of this document.

Microchip FPGA Support

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

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

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

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

Microchip Information

The Microchip Website

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

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

Product Change Notification Service

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

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

Customer Support

Users of Microchip products can receive assistance through several channels:

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

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

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

Microchip Devices Code Protection Feature

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

- Microchip products meet the specifications contained in their particular Microchip Data Sheet.

- Microchip believes that its family of products is secure when used in the intended manner, within operating specifications, and under normal conditions.

- Microchip values and aggressively protects its intellectual property rights. Attempts to breach the code protection features of Microchip product is strictly prohibited and may violate the Digital Millennium Copyright Act.

- Neither Microchip nor any other semiconductor manufacturer can guarantee the security of its code. Code protection does not mean that we are guaranteeing the product is "unbreakable". Code protection is constantly evolving. Microchip is committed to continuously improving the code protection features of our products.

Legal Notice

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

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

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

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

Trademarks

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

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

Adjacent Key Suppression, AKS, Analog-for-the-Digital Age, Any Capacitor, AnyIn, AnyOut, Augmented Switching, BlueSky, BodyCom, Clockstudio, CodeGuard, CryptoAuthentication, CryptoAutomotive, CryptoCompanion, CryptoController, dsPICDEM, dsPICDEM.net, Dynamic

Quality Management System

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

Worldwide Sales and Service

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