HB0115
Handbook
CoreMemCtrl v2.2

Microsemi
a Microchip company

Microsemi Headquarters
One Enterprise, Aliso Viejo,
CA 92656 USA
Within the USA: +1 (800) 713-4113
Outside the USA: +1 (949) 380-6100
Sales: +1 (949) 380-6136
Fax: +1 (949) 215-4996
Email: sales.support@microsemi.com
www.microsemi.com

About Microsemi

Microsemi, a wholly owned subsidiary of Microchip Technology Inc. (Nasdaq: MCHP), offers a comprehensive portfolio of semiconductor and system solutions for aerospace & defense, communications, data center and industrial markets. Products include high-performance and radiation-hardened analog mixed-signal integrated circuits, FPGAs, SoCs and ASICs; power management products; timing and synchronization devices and precise time solutions, setting the world's standard for time; voice processing devices; RF solutions; discrete components; enterprise storage and communication solutions, security technologies and scalable anti-tamper products; Ethernet solutions; Power-over-Ethernet ICs and midspans; as well as custom design capabilities and services. Learn more at www.microsemi.com.

Contents

Figures

Tables

1 Revision History

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

1.1 Revision 3.0

Updated for CoreMemCtrl v2.2 release.

1.2 Revision 2.0

Updated for CoreMemCtrl v2.1 release.

1.3 Revision 1.0

The first publication of this document. Created for CoreMemCtrl v2.0.

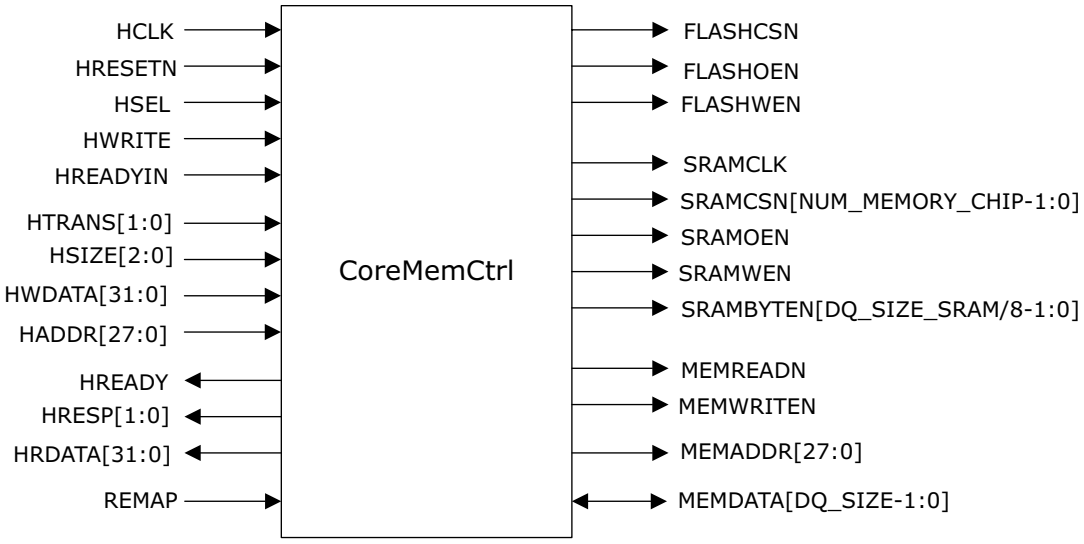
2 Introduction

2.1 Overview

The CoreMemCtrl is an advanced high-performance bus (AHBL) slave component that interfaces to external flash and SRAM memory devices. Both synchronous and asynchronous SRAM are supported. The core provides an AHBL Slave Interface that is suitable for connection to the CoreAHBLite bus.

Various configuration options exist on the core to allow a variety of different memory devices to be supported. Figure 1 shows a block diagram of CoreMemCtrl.

Figure 1 • **CoreMemCtrl Block Diagram**



2.2 Key Features

CoreMemCtrl has the following features:

- Provides an AHB interface to external memory devices.
- Configurable external memory interface, up to 4 chip select for synchronous/asynchronous SRAM.
- Interfaces to external flash and either synchronous or asynchronous external SRAM.
- Supports 32-bit word, 16-bit halfword, and 8-bit byte accesses to SRAM.
- Supports 32-bit word, 16-bit halfword, and 8-bit byte accesses to flash.
- The locations of flash and SRAM in the address space can be swapped by asserting the REMAP input.

2.3 Core Version

This handbook applies to CoreMemCtrl version 2.2. The release notes provided with the core list the known discrepancies between this handbook and the core release associated with the release notes.

2.4 Supported Device Families

- PolarFire® SoC
- PolarFire®
- RTG4™
- IGLOO®2
- SmartFusion®2
- IGLOO®
- IGLOOe
- IGLOO PLUS
- ProASIC®3
- ProASIC3E
- ProASIC3L
- SmartFusion
- ProASICPLUS®
- Fusion
- RTAX-S
- RTSX-S
- EX
- AX

2.5 Device Utilization and Performance

Table 1 gives resource usage and performance figures for various configurations of CoreMemCtrl. Table 1 does not cover every possible configuration, but instead lists a range of configurations which should give a good indication of the expected resource usage and performance of the core.

Table 1 • **CoreMemCtrl Device Utilization and Performance**

Parameter																			
NUM_MEMORY_CHIP	SYNC_SRAM	FLOW_THROUGH	FLASH_DQ_SIZE	MEM_0_DQ_SIZE	MEM_1_DQ_SIZE	MEM_2_DQ_SIZE	MEM_3_DQ_SIZE	NUM_WS_FLASH_READ	NUM_WS_FLASH_WRITE	NUM_WS_SRAM_WRITE_CH0	NUM_WS_SRAM_READ_CH0	NUM_WS_SRAM_WRITE_CH1	NUM_WS_SRAM_READ_CH1	NUM_WS_SRAM_WRITE_CH2	NUM_WS_SRAM_READ_CH2	NUM_WS_SRAM_WRITE_CH3	NUM_WS_SRAM_READ_CH3	Logical Elements	Frequency (MHz)
4	0	0	32	32	32	32	32	7	7	7	7	7	7	7	7	7	7	595	213
4	0	0	16	16	16	16	16	7	7	7	7	7	7	7	7	7	7	505	199
4	1	1	32	32	32	32	32	7	7	-	-	-	-	-	-	-	-	508	213
4	1	1	16	16	16	16	16	7	7	-	-	-	-	-	-	-	-	555	201
4	1	0	32	32	32	32	32	7	7	-	-	-	-	-	-	-	-	499	205
4	1	0	16	16	16	16	16	7	7	-	-	-	-	-	-	-	-	563	217

Note: The data in this table was achieved using typical synthesis and layout settings. Frequency was set to 200 MHz on the PolarFire MPF500T-1FCG1152E device.

3 Functional Description

3.1 Functional Overview

The CoreMemCtrl core provides a standard AHBL interface to translate AHB bus reads and writes into read and write commands with the signaling and timing of standard synchronous/asynchronous SRAM and Flash device.

The Core has two mode of Memory address mapping that is, FIXED and USER. If the MEMORY_ADDRESS_CONFIG_MODE = 0 then FIXED address map configuration enabled else USER address map configuration enabled.

3.1.1 FIXED Mode

Always one Flash and one SRAM interface enabled. The 256MB of address space is evenly split between flash and SRAM memory so that up to 128 MB of each type of memory may be accessed. The REMAP input which may be used to swap the positions of flash and SRAM in the memory map.

3.1.2 USER Mode

One Flash and up to four SRAM interface can be enable. The 256MB of address space is evenly split between flash and SRAM memory so that up to 128 MB of each type of memory may be accessed. The 128 MB of SRAM memory space can be split up to 4 SRAM chips with individual chip select. The core has a REMAP input, which can be used to swap the positions of flash and SRAM in the memory map.

The Flash interface can be disabled with the help of the ENABLE_FLASH_IF parameter. If this parameter is set to 0, then a total of 256 MB of memory space is available for SRAM devices, and 256 MB of SRAM memory space can be split up to 4 SRAM chips with individual chip select. In this case, REMAP logic for swapping the position of flash and SRAM is invalid.

The SRAM interface can be enabled/disabled by the ENABLE_SRAM_IF parameter. If this parameter is set to 0, then a total of 256 MB of memory space is available for the flash device. In this case, REMAP the logic for swapping the position of flash and SRAM is invalid.

The SRAM memory space can be split up to 4 SRAM chips, with the individual chip select with the help of NUM_MEMORY_CHIP, MEM_0_BASEADDR, MEM_0_ENDADDR, MEM_1_BASEADDR, MEM_1_ENDADDR, MEM_2_BASEADDR, MEM_2_ENDADDR, MEM_3_BASEADDR, MEM_3_ENDADDR parameters.

The CoreMemCtrl core performs address decoding functionalities for enabled Flash and SRAM devices. The address decoding logic utilizes the AHBL address to determine the chip select and read/write address in the enabled memory chip.

The Memory interface timing for the asynchronous SRAM and flash memory interfaces are parameterized and can be configured independently for each chip.

Note: The Start and End address for different enabled SRAM chips should not overlap during IP configuration.

Note: USER MODE (MEMORY_ADDRESS_CONFIG_MODE = 1) can be enabled only with libero version 12 and later series.

3.2 Supported Interfaces

CoreMemCtrl has an advanced microcontroller bus architecture (AMBA) AHBL slave interface through which an AHB master can initiate read and write accesses to external memory.

The core also has a group of ports for interfacing to external flash and SRAM memory devices. The design of the core assumes that the address bus and bidirectional data bus are common to both flash and SRAM devices. Aside from these buses, chip select, output enable, and write enable signals are provided for connection to the memory devices.

It is possible, via the configuration window for the core, to choose to use common read and write enable signals for flash and SRAM. When this option is selected, the MEMREADN and MEMWRITEN signals can be connected to both flash and SRAM. This option can be useful when the number of pins available for interfacing to external memories is limited.

4 Register Map and Descriptions

CoreMemCtrl does not contain any memory-mapped registers such as control or status registers.

5 Memory Map

CoreMemCtrl uses 256 MB of address space on the AHB-Lite bus to which it is connected. This address space is evenly divided between flash memory and SRAM memory, so that, each memory type can be up to 128 MB in size. The REMAP input is used to control which memory type appears in which half of the 256 MB slot. Table 2 shows the relationship between the REMAP input and address offsets of flash and SRAM.

Table 2 • **REMAP and Address Offset Relationship**

REMAP	Address Offset	Memory Type
0	0x00000000 to 0x07FFFFFF	Flash
	0x08000000 to 0x0FFFFFFF	SRAM
1	0x00000000 to 0x07FFFFFF	SRAM
	0x08000000 to 0x0FFFFFFF	Flash

6 Interface

6.1 Core Parameters

Table 3 • **CoreMemCtrl Core Parameters**

Parameter	Values	Default	Description
FAMILY	9, 10, 11, 12, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27	17	Must be set to match the supported FPGA family. 9 – RTSX-S 10 - EX 11 – AX 12 – RTAX-S 14 - ProASICPLUS® 15 – ProASIC3 16 – ProASIC3E 17 – Fusion 18 - SmartFusion® 19 – SmartFusion2 20 – IGLOO 21 – IGLOOe 22 – ProASIC3L 23 – IGLOO PLUS 24 – IGLOO2 25 - RTG4 26 – PolarFire 27 – PolarFireSoc
MEMORY_ADDRESS_CONFIG_MODE	0,1	0	Memory address space configuration mode. 0= FIXED, Fixed address mapping configuration enabled (128 MB of address space for SRAM and 128 MB of address space for Flash device). 1= USER , User configurable address mapping for Flash and maximum of 4 SRAM Devices enabled . **Note :** MEMORY_ADDRESS_CONFIG_MODE = 1 valid only for Libero 12 and later series.
ENABLE_FLASH_IF	0, 1	1	0 = Disabled Flash Interface 1 = Enabled Flash Interface **Note:** Valid only when MEMORY_ADDRESS_CONFIG_MODE = 1.

Table 3 • **CoreMemCtrl Core Parameters**

Parameter	Values	Default	Description
ENABLE_SRAM_IF	0, 1	1	0 = Disabled SRAM Interface 1 = Enabled SRAM Interface **Note:** Valid only when MEMORY_ADDRESS_CONFIG_MODE = 1.
SYNC_SRAM	0, 1	1	Selects the type of external SRAM. 0 = Interfacing to asynchronous SRAM 1 = Interfacing to synchronous SRAM
FLASH_TYPE	0,1	0	0 = Types of Flash devices connect where A0 address used during Byte mode configuration. (MEMADDR[0] is the LSB address of device) 1 = Types of Flash devices connect where A0 address not used during Byte mode configuration. (MEMADDR[1] is the LSB address of device) **Note:** Valid only when ENABLE_FLASH_IF = 1 and FLASH_DQ_SIZE = 16.
NUM_MEMORY_CHIP	1 to 4	1	This parameter configures number of SRAM Interface. **Note:** The parameter is only valid when ENABLE_SRAM_IF = 1 and MEMORY_ADDRESS_CONFIG_MODE = 1. 1 = SRAMCSN[0] is valid 2 = SRAMCSN[0] and SRAMCSN[1] are valid 3 = SRAMCSN[0], SRAMCSN[1], and SRAMCSN[2] are valid 4 = SRAMCSN[0], SRAMCSN[1], SRAMCSN[2], and SRAMCSN[3] are valid

Table 3 • **CoreMemCtrl Core Parameters**

Parameter	Values	Default	Description
MEM_0_DQ_SIZE	8,16,32	8	Width of SRAM chip 0 data bus(DQ). This determines the byte mapping of AHBL to SRAM. **Note:** Valid only when NUM_MEMORY_CHIP = 1 or 2 or 3 or 4.
MEM_1_DQ_SIZE	8,16,32	8	Width of SRAM chip 1 data bus(DQ). This determines the byte mapping of AHBL to SRAM. **Note:** Valid only when NUM_MEMORY_CHIP = 2 or 3 or 4.
MEM_2_DQ_SIZE	8,16,32	8	Width of SRAM chip 2 data bus(DQ). This determines the byte mapping of AHBL to SRAM. **Note:** Valid only when NUM_MEMORY_CHIP = 3 or 4.
MEM_3_DQ_SIZE	8,16,32	8	Width of SRAM chip 3 data bus(DQ). This determines the byte mapping of AHBL to SRAM. **Note:** Valid only when NUM_MEMORY_CHIP = 4.
FLASH_DQ_SIZE	8,16,32	8	Width of flash data bus (DQ), This determines the byte mapping of AHBL to SRAM.
FLOW_THROUGH	0, 1	0	Synchronous SRAM mode selection. 0 = External synchronous SRAM device is operating in pipeline mode. 1 = External synchronous SRAM device is operating in flow-through mode. **Note:** This parameter is only valid when SYNC_SRAM = 1 and should be set to match the operation of the external synchronous SRAM device. Some synchronous SRAM devices can be configured to operate in either pipeline or flow-through mode.

Table 3 • **CoreMemCtrl Core Parameters**

Parameter	Values	Default	Description
NUM_WS_FLASH_READ	1 to 31	1	Number of wait states inserted during a read from flash. **Note:** Valid only when ENABLE_FLASH_IF = 1
NUM_WS_FLASH_WRITE	1 to 31	1	Number of wait states inserted during a write to flash. **Note:** Note: Valid only when ENABLE_FLASH_IF = 1
NUM_WS_SRAM_READ_CH0	1 to 31	1	Number of wait states inserted during a read from asynchronous SRAM corresponding to SRAMCSN[0] chip. **Note:** This parameter is only valid when SYNC_SRAM = 0 and NUM_MEMORY_CHIP = 1 or 2 or 3 or 4
NUM_WS_SRAM_READ_CH1	1 to 31	1	Number of wait states inserted during a read from asynchronous SRAM corresponding to SRAMCSN[1] chip. **Note:** This parameter is only valid when SYNC_SRAM = 0 and NUM_MEMORY_CHIP = 2 or 3 or 4.
NUM_WS_SRAM_READ_CH2	1 to 31	1	Number of wait states inserted during a read from asynchronous SRAM corresponding to SRAMCSN[2] chip. **Note:** This parameter is only valid when SYNC_SRAM = 0 and NUM_MEMORY_CHIP = 3 or 4.
NUM_WS_SRAM_READ_CH3	1 to 31	1	Number of wait states inserted during a read from asynchronous SRAM corresponding to SRAMCSN[3] chip. **Note:** This parameter is only valid when SYNC_SRAM = 0 and NUM_MEMORY_CHIP = 4.

Table 3 • **CoreMemCtrl Core Parameters**

Parameter	Values	Default	Description
NUM_WS_SRAM_WRITE_CH0	1 to 31	1	Number of wait states inserted during a write to asynchronous SRAM corresponding to SRAMCSN[0] chip. **Note:** This parameter is only valid when SYNC_SRAM = 0 and NUM_MEMORY_CHIP = 1 or 2 or 3 or 4.
NUM_WS_SRAM_WRITE_CH1	1 to 31	1	Number of wait states inserted during a write to asynchronous SRAM corresponding to SRAMCSN[1] chip. **Note:** This parameter is only valid when SYNC_SRAM = 0 and NUM_MEMORY_CHIP = 2 or 3 or 4.
NUM_WS_SRAM_WRITE_CH2	1 to 31	1	Number of wait states inserted during a write to asynchronous SRAM corresponding to SRAMCSN[2] chip. **Note:** This parameter is only valid when SYNC_SRAM = 0 and NUM_MEMORY_CHIP =3 or 4.
NUM_WS_SRAM_WRITE_CH3	1 to 31	1	Number of wait states inserted during a write to asynchronous SRAM corresponding to SRAMCSN[3] chip. **Note:** This parameter is only valid when SYNC_SRAM = 0 and NUM_MEMORY_CHIP = 4.

Table 3 • **CoreMemCtrl Core Parameters**

Parameter	Values	Default	Description
SHARED_RW	0, 1	0	0 = Separate read and write enable signals (that is, FLASHOEN, FLASHWEN, SRAMOEN, and SRAMWEN) are used for flash and SRAM. 1 = Common read and write enable signals (that is, MEMREADN and MEMWRITEN) are used for both flash and SRAM. **Note:** This setting is intended for use in cases where a limited number of pins are available for interfacing to external memory.
MEM_0_BASEADDR	0 when REMAP = 1 or ENABLE_FLASH_IF = 0, 8000000 when REMAP = 0	8000000	Memory Chip 0 Base Address. **Note:** Valid for NUM_MEMORY_CHIP equals to 1 or 2 or 3 or 4 and MEMORY_ADDRESS_CONFIG_MODE = 1.
MEM_0_ENDADDR	1 to 7FFFFFF when REMAP =1, 8000001 to 7FFFFFF when REMAP =0 1 to FFFFFFF when ENABLE_FLASH_IF =0	9FFFFFF	Memory Chip 0 End Address. **Note:** Valid for NUM_MEMORY_CHIP equals to 1 or 2 or 3 or 4 and MEMORY_ADDRESS_CONFIG_MODE = 1.
MEM_1_BASEADDR	Must be greater than MEM_0_ENDADDR	A000000	Memory Chip 1 Base Address. **Note:** Valid for NUM_MEMORY_CHIP equals to 2 or 3 or 4 and MEMORY_ADDRESS_CONFIG_MODE = 1.
MEM_1_ENDADDR	(MEM_1_BASEADDR +1) to FFFFFFF when REMAP=0 or ENABLE_FLASH_IF =0, MEM_1_BASEADDR +1 to 7FFFFFF when REMAP=1	BFFFFFF	Memory Chip 1 End Address. **Note:** Valid for NUM_MEMORY_CHIP equals to 2 or 3 or 4 and MEMORY_ADDRESS_CONFIG_MODE = 1.

Table 3 • **CoreMemCtrl Core Parameters**

Parameter	Values	Default	Description
MEM_2_BASEADDR	Must be greater than MEM_1_ENDADDR	C000000	Memory Chip 2 Base Address. **Note:** Valid for NUM_MEMORY_CHIP equals to 3 or 4 and MEMORY_ADDRESS_CONFIG_MODE = 1.
MEM_2_ENDADDR	(MEM_2_BASEADDR +1) to FFFFFFF when REMAP=0 or ENABLE_FLASH_IF =0, (MEM_2_BASEADDR +1) to 7FFFFFF when REMAP=1	DFFFFFF	Memory Chip 2 End Address. **Note:** Valid for NUM_MEMORY_CHIP equals to 3 or 4 and MEMORY_ADDRESS_CONFIG_MODE = 1.
MEM_3_BASEADDR	Must be greater than MEM_2_ENDADDR	E000000	Memory Chip 3 Base Address. **Note:** Valid for NUM_MEMORY_CHIP equals to 4 and MEMORY_ADDRESS_CONFIG_MODE = 1.
MEM_3_ENDADDR	(MEM_3_BASEADDR +1) to FFFFFFF when REMAP=0 or ENABLE_FLASH_IF =0, (MEM_3_BASEADDR +1) to 7FFFFFF when REMAP=1	FFFFFFF	Memory Chip 3 End Address **Note:** Valid for NUM_MEMORY_CHIP equals to 4 and MEMORY_ADDRESS_CONFIG_MODE = 1.

6.2 Ports

Table 4 outlines the top-level signals for CoreMemCtrl.

Table 4 • **CoreMemCtrl Ports**

Name	Type	Description
AHB-Lite Bus Signals		
HCLK	Input	AHB system clock. Reference clock for all internal logic
HRESETN	Input	AHB active Low asynchronous reset
HSEL	Input	AHB slave select
HWRITE	Input	AHB write/read indication
HREADYIN	Input	AHB ready indication from bus
HTRANS[1:0]	Input	AHB transfer type
HSIZE[2:0]	Input	AHB size of transfer
HWDATA[31:0]	Input	AHB write data
HADDR[27:0]	Input	AHB address bus
HRDATA[31:0]	Output	AHB read data
HREADY	Output	AHB ready output to bus

Table 4 • **CoreMemCtrl Ports *(continued)***

Name	Type	Description
HRESP[1:0]	Output	AHB response
HRDATA[31:0]	Output	AHB read data
Remap Control		
REMAP	Input	Remap control. When asserted, flash and SRAM locations are swapped in the address map. **Note:** Valid when both SRAM and FLASH interface enabled.
External Memory Interface		
FLASHCSN	Output	Flash chip select, active low. Valid when parameter ENABLE_FLASH_IF = 1
FLASHOEN	Output	Flash output enable, active low. Valid when parameter ENABLE_FLASH_IF = 1
FLASHWEN	Output	Flash write enable, active low. Valid when parameter ENABLE_FLASH_IF = 1
SRAMCLK	Output	SRAM clock. This clock signal is the inverse of the AHB clock input, HCLK. Valid when parameter SYNC_SRAM = 1
SRAMCSN[NUM_MEMORY_CHIP-1:0]	Output	SRAM chip select, active low. Valid when parameter ENABLE_SRAM_IF = 1
SRAMOEN	Output	SRAM output enable, active low. Valid when parameter ENABLE_SRAM_IF = 1
SRAMWEN	Output	SRAM write enable, active low. Valid when parameter ENABLE_SRAM_IF = 1
SRAMBYTEN[DQ_SIZE_SRAM/8-1:0][2]	Output	SRAM byte enables, active low. Valid when parameter ENABLE_SRAM_IF = 1 and DQ_SIZE_SRAM not equal to 8.
MEMREADN	Output	Common read enable, active low. Can be connected to both flash and SRAM and visible at port when the SHARED_RW parameter is set.
MEMWRITEN	Output	Common write enable, active low. Can be connected to both flash and SRAM and visible at port when the SHARED_RW parameter is set.
MEMADDR[27:0]	Output	Common flash/SRAM address bus
MEMDATA[DQ_SIZE-1:0][1]	Inout	Common flash/SRAM data bus

Note: Unless otherwise noted, all of the above signals are active High.

Note: 1. The DQ_SIZE parameter is not visible in Configurator user interface, it is internally calculated in the packager depending on the ENABLE_FLASH_IF, ENABLE_SRAM_IF, MEM_0_DQ_SIZE, MEM_1_DQ_SIZE, MEM_2_DQ_SIZE, MEM_3_DQ_SIZE, and FLASH_DQ_SIZE parameters.

- If the ENABLE_FLASH_IF = 1 and ENABLE_SRAM_IF = 1, DQ_SIZE is equal to the maximum between all enabled SRAM and FLASH memory width.
- If the parameter ENABLE_FLASH_IF = 0, ENABLE_SRAM_IF = 1, DQ_SIZE equal to the maximum between all enabled SRAM Memory.
- If the parameter ENABLE_FLASH_IF = 1, ENABLE_SRAM_IF = 0, DQ_SIZE equal to the FLASH_DQ_SIZE.

For example: If ENABLE_FLASH_IF = 1, ENABLE_SRAM_IF = 1, NUM_MEMORY_CHIP = 4, MEM_0_DQ_SIZE = 32, MEM_1_DQ_SIZE = 16, MEM_2_DQ_SIZE = 8, MEM_3_DQ_SIZE = 8, FLASH_DQ_SIZE = 16, then the generated DQ_SIZE parameter will be equal to 32.

If ENABLE_FLASH_IF = 0, ENABLE_SRAM_IF = 1, NUM_MEMORY_CHIP = 3, MEM_0_DQ_SIZE = 16, MEM_1_DQ_SIZE = 16, MEM_2_DQ_SIZE = 8, MEM_3_DQ_SIZE = 32, FLASH_DQ_SIZE = 32, then the generated DQ_SIZE parameter will be equal to 16.

Note: 2. The DQ_SIZE_SRAM parameter is not visible in the Configurator user interface, then it is internally calculated in the packager depending on ENABLE_SRAM_IF, MEM_0_DQ_SIZE, MEM_1_DQ_SIZE, MEM_2_DQ_SIZE, and MEM_3_DQ_SIZE parameters.

- If ENABLE_SRAM_IF = 1, DQ_SIZE_SRAM equal to the maximum between all enabled SRAM memory width.

For example: If ENABLE_SRAM_IF = 1, MEM_0_DQ_SIZE = 32, MEM_1_DQ_SIZE = 16, MEM_2_DQ_SIZE = 8, MEM_3_DQ_SIZE = 8, FLASH_DQ_SIZE = 16, then the generated DQ_SIZE_SRAM parameter will be equal to 32.

7 Timing Diagrams

The waveforms in this section show the timing of the CoreMemCtrl signals.

Figure 2 • **Flash Access: Three Writes and Three Reads Transfer**

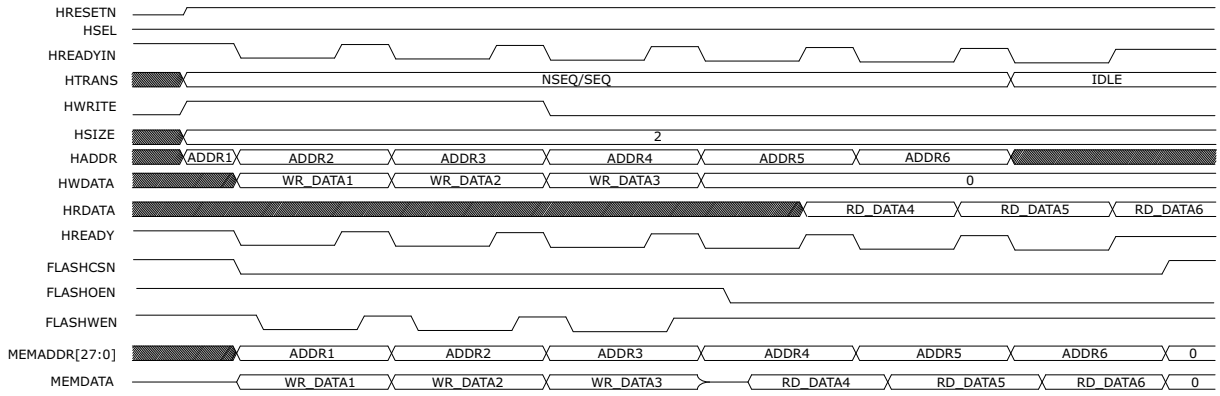


Figure 3 • **Asynchronous SRAM Access: Three Writes and Three Reads Transfer**

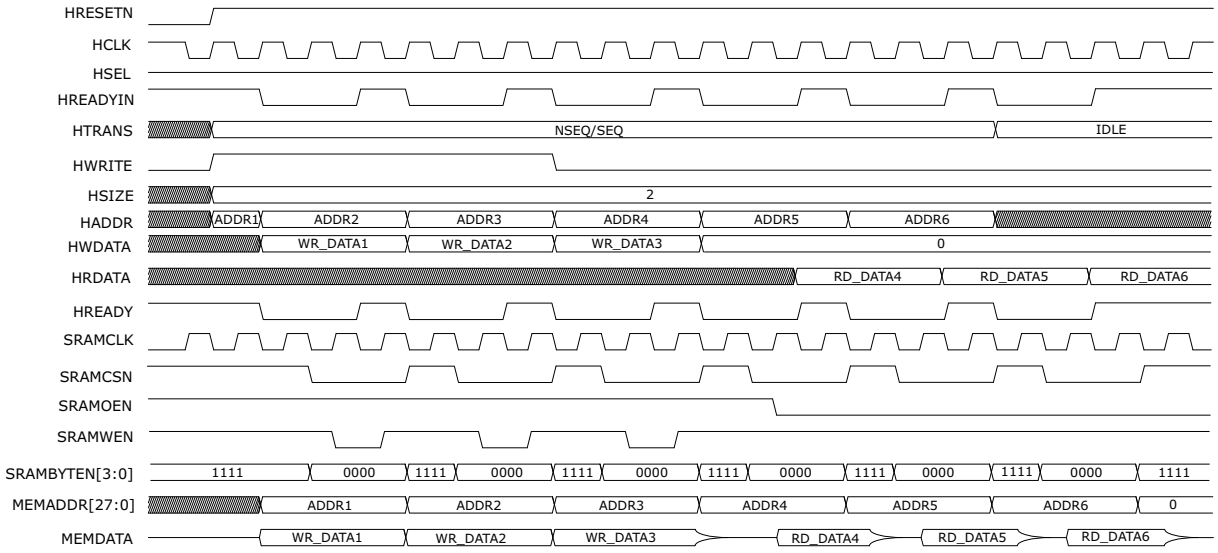
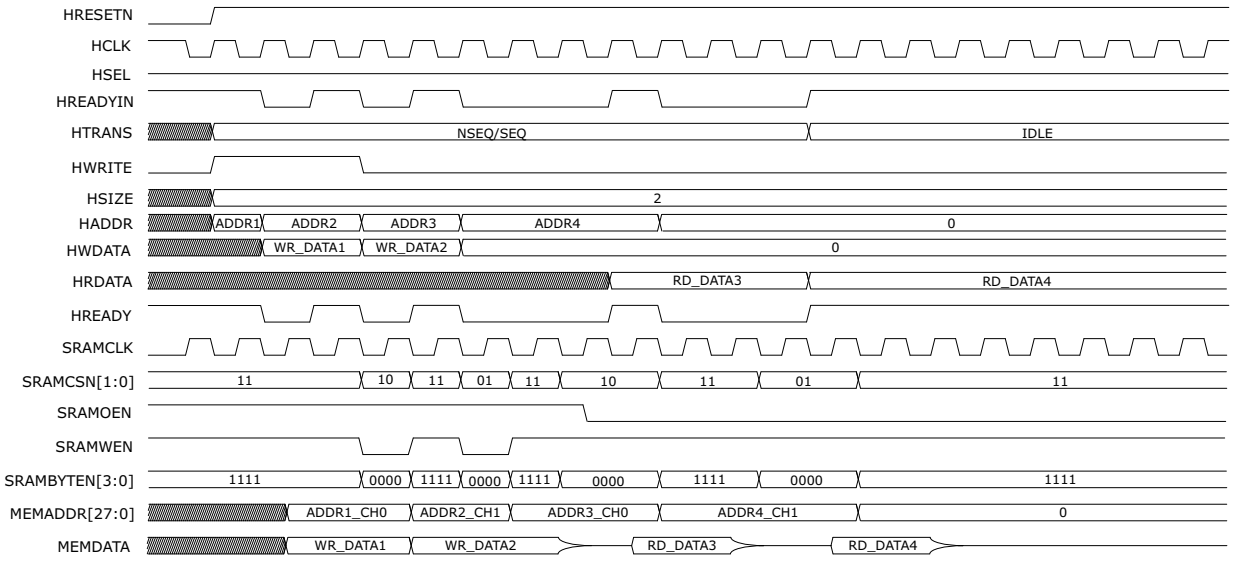


Figure 4 • **Synchronous SRAM Access: Two Writes, Two Reads, and Two SRAM chip enabled**



8 Tool Flow

8.1 License

The CoreMemCtrl does not require any license.

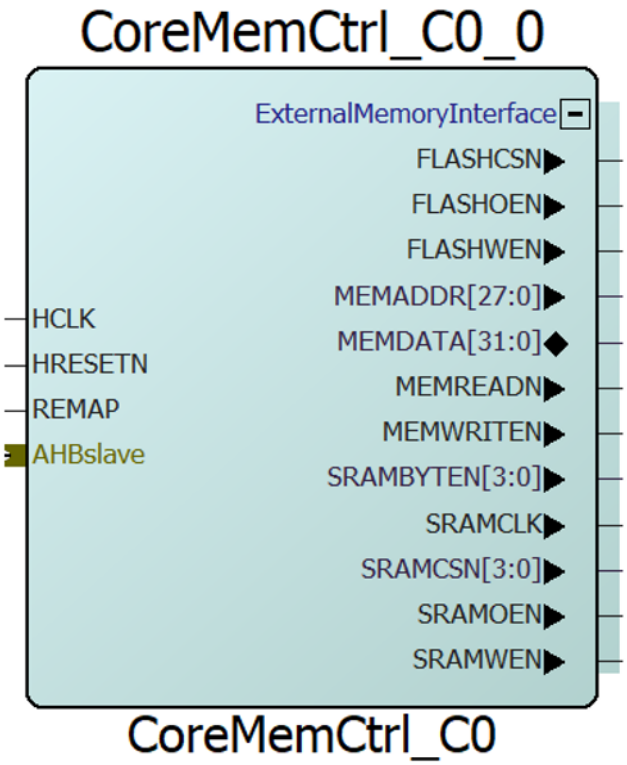
8.2 Register Transfer Level (RTL)

Complete RTL source code is provided for the core and testbenches.

8.3 SmartDesign

CoreMemCtrl is preinstalled in the SmartDesign IP deployment design environment. An example instantiated view is shown in the following figure. The core can be configured using the configuration GUI within SmartDesign, as shown in Figure 6. For more information on using SmartDesign to instantiate and generate cores, refer to *Using DirectCore in Libero SoC User Guide* or consult the *Libero SoC online help*.

Figure 5 • **CoreMemCtrl Instance View**



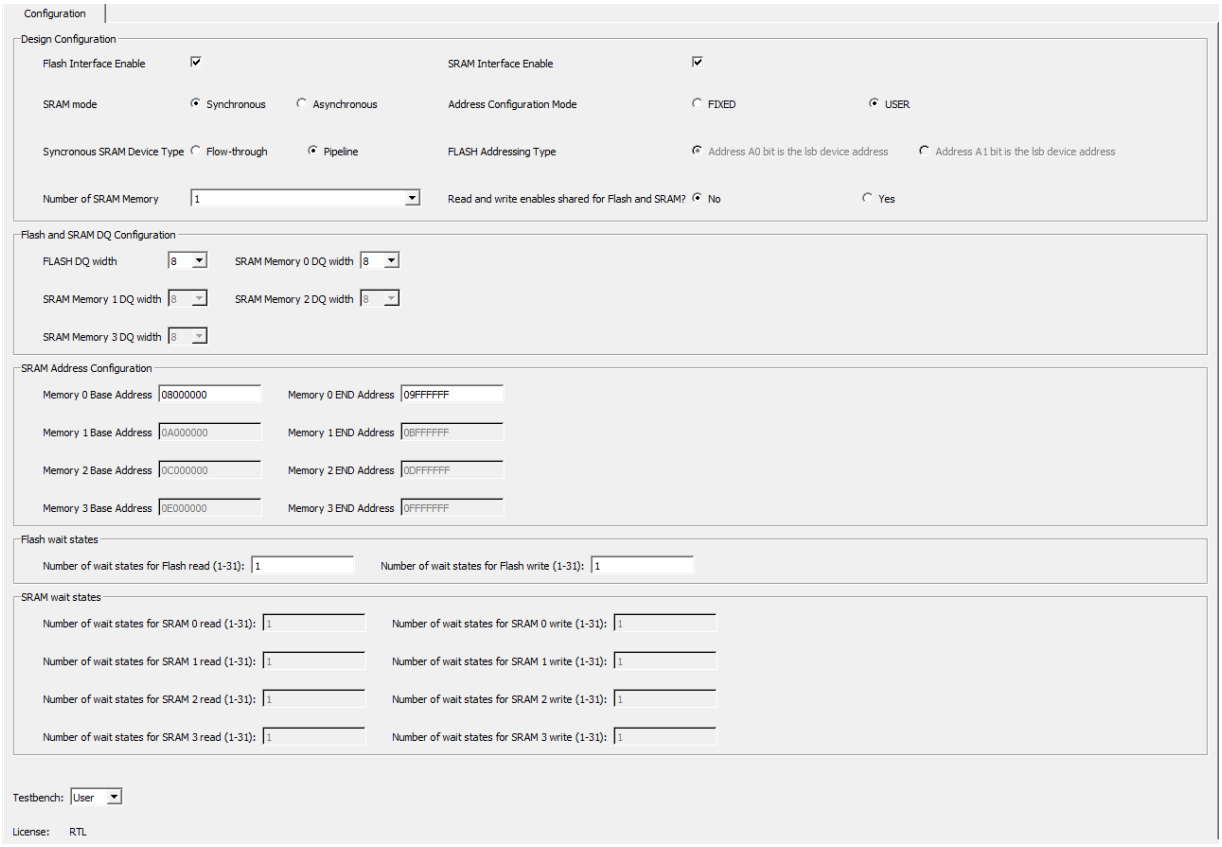
8.4 Configuring CoreMemCtrl in SmartDesign

The core can be configured using the configuration GUI within SmartDesign, as shown in Figure 6. In a typical system, the AHB slave interface of CoreMemCtrl is connected to one of the slave slots on CoreAHBLite. CoreMemCtrl is often connected to slot 0 on the bus so that the external memory appears at address 0x00000000 in the address map of a processor connected as a master in the system. The ports for connection to external memory devices should be routed to the top level of the design and assigned to appropriate pins for connection to the external memories.

CoreMemCtrl is suitable for interfacing to a variety of flash and SRAM memories, but the core must be configured to suit the particular devices being used. Figure 6 shows the CoreMemCtrl configuration

window, along with cross-references to the corresponding top-level parameters. The parameters of the core are fully described in the Core Parameters.

Figure 6 • **CoreMemCtrl Configuration Window**



The configuration options for CoreMemCtrl are described in the following paragraphs. The CoreMemCtrl configuration window is used to adjust the values of the underlying parameters in the RTL code for the core. Each configuration option presented in the configuration window corresponds directly to an actual parameter in the RTL code for CoreMemCtrl.

8.4.1 SRAM and Flash Interface Enable

The core has ENABLE_FLASH_IF and ENABLE_SRAM_IF which allows to enable or disable flash and SRAM memory interface respectively.

8.4.2 SRAM Type

You can use the SYNC_SRAM parameter to set the interfaced SRAM to either synchronous or asynchronous. If this parameter is set to synchronous SRAM, the FLOW_THROUGH parameter must be set to match the mode of operation of the synchronous SRAM device. Some devices only operate in a flow-through manner (where the data appears in the clock cycle after the address), whereas others can be configured to operate in pipeline mode (with the data appearing in two cycles after the address).

8.4.3 Flash and SRAM DQ Configuration

The core has a FLASH_DQ_SIZE, MEM_0_DQ_SIZE, MEM_1_DQ_SIZE, MEM_2_DQ_SIZE, and MEM_3_DQ_SIZE parameters, which allows to connect different DQ width of flash and SRAM devices.

8.4.4 SRAM Address Configuration

The core has MEM_0_BASEADDR, MEM_0_ENDADDR, MEM_1_BASEADDR, MEM_1_ENDADDR, MEM_2_BASEADDR, MEM_2_ENDADDR, MEM_3_BASEADDR, and MEM_3_ENDADDR parameters, which are used to generate an individual chip select signals for the SRAM devices during read and write transactions.

For Example: If NUM_MEMORY_CHIP = 4 and REMAP = 0, then the upper 128 MB of total memory space of 256 MB will be split to 4 address spaces for the SRAM chips with an individual chip select. And the size of each address space is calculated by using the MEM_0_BASEADDR, MEM_0_ENDADDR, MEM_1_BASEADDR, MEM_1_ENDADDR, MEM_2_BASEADDR, MEM_2_ENDADDR, MEM_3_BASEADDR and MEM_3_ENDADDR parameters.

- If user need to integrate 4 SRAM chips of size MEM_0 = 32MB, MEM_1 = 16MB, MEM_2 = 16MB, MEM_3 = 64MB, then the required parameter values are as following:
  - (MEM_0_ENDADDR - MEM_0_BASEADDR) +1 = 32*1024*1024 (32MB)
  - (MEM_1_ENDADDR - MEM_1_BASEADDR) +1 = 16*1024*1024 (16MB)
  - (MEM_2_ENDADDR - MEM_2_BASEADDR) +1 = 16*1024*1024 (16MB)
  - (MEM_3_ENDADDR - MEM_3_BASEADDR) +1 = 64*1024*1024 (64MB)

8.4.5 Number of Wait States

It is possible to adjust the number of wait states inserted during read and write access to flash and asynchronous SRAM. Up to thirty-one wait states can be configured for each type of access. Increasing the number of wait states enables more clock cycles for completion of read and write transactions. In systems where a relatively slow flash or asynchronous SRAM device is used, increasing the number of wait states enables the system clock speed to be increased while still meeting the timing requirements of the memory devices. However, there is a performance cost to increasing the number of wait states because several cycles will be required for each access to the external memory. The optimum balance between system clock frequency and the number of cycles required to access memory will depend on the system design and the performance required.

8.4.6 Shared Read and Write Enables

CoreMemCtrl generates read and write enable signals for both flash and SRAM. These signals are named FLASHOEN, FLASHWEN, SRAMOEN, and SRAMWEN. The core also generates two common read and write enable signals, MEMREADN and MEMWRITEN, which can be connected to both flash and SRAM devices when the SHARED_RW parameter is set. This option is intended for use in situations where the number of FPGA pins available for interfacing to external memory is limited.

8.5 Simulation

To run simulations, select the user testbench within the SmartDesign CoreMemCtrl configuration GUI, right-click, and select **Generate Design** (Figure 6). When SmartDesign generates the design files, it will install the appropriate testbench files. To run the simulation, set the design root to the CoreMemCtrl instantiation in the Libero design hierarchy pane, and click **Simulation** in the Libero **Design Flow** window. This invokes QuestaSim® and automatically runs the simulation.

8.6 Synthesis in Libero SoC

To run synthesis on the core with the parameter settings selected in SmartDesign, set the design root appropriately, and click the Synthesis icon in the Libero. The Synthesis window appears, displaying the Synplicity® project. To perform synthesis, click **Run**.

8.7 Place-and-Route in Libero SoC

After setting the design root appropriately and running a synthesis, click **Layout** in Libero to invoke Designer. CoreMemCtrl requires no special place-and-route settings.

9 Testbench

9.1 Testbench Operation

The following testbench is provided with CoreMemCtrl:

- Verilog testbench
- VHDL testbench

9.2 Verilog/VHDL Testbench

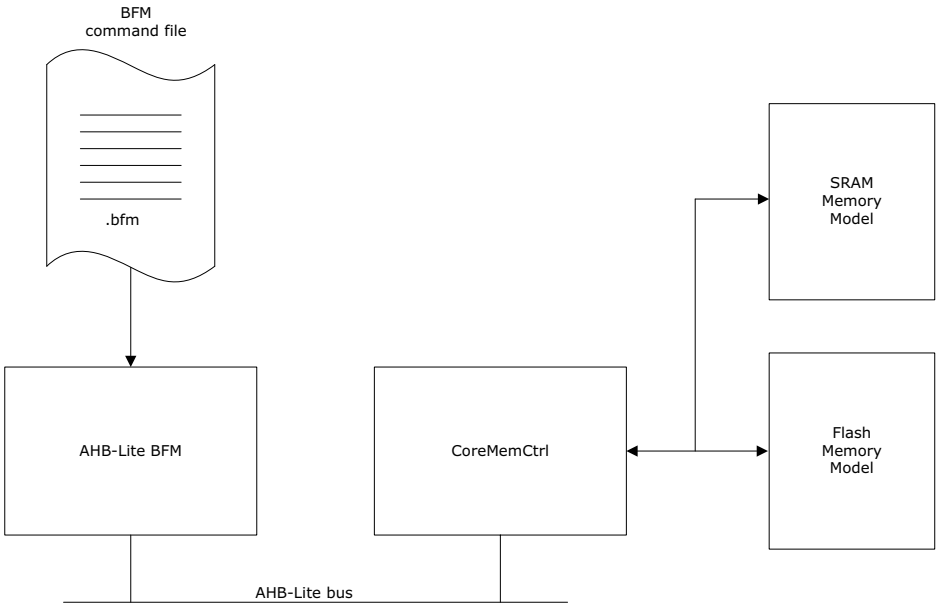
The Verilog/VHDL user testbench is provided as a reference and can be modified to suit the requirements. The source code for the Verilog/VHDL user testbench is provided to ease the process of integrating the CoreMemCtrl macro into the design and verifying its functionality.

9.3 Testbench Description

A user testbench is included with the RTL release of CoreMemCtrl. A simplified block diagram of the testbench is shown in Figure 7. The testbench instantiates the Design Under Test (DUT), which is the CoreMemCtrl, the Memory models (Flash, Synchronous, and Asynchronous SRAM model), as well as the test vector modules that provide stimuli sources for the DUT. A procedural testbench controls each module and applies the sequential stimuli to the DUT.

An overview of the CoreMemCtrl user testbench is shown in Figure 7.

Figure 7 • **Overview of CoreMemCtrl Testbench**



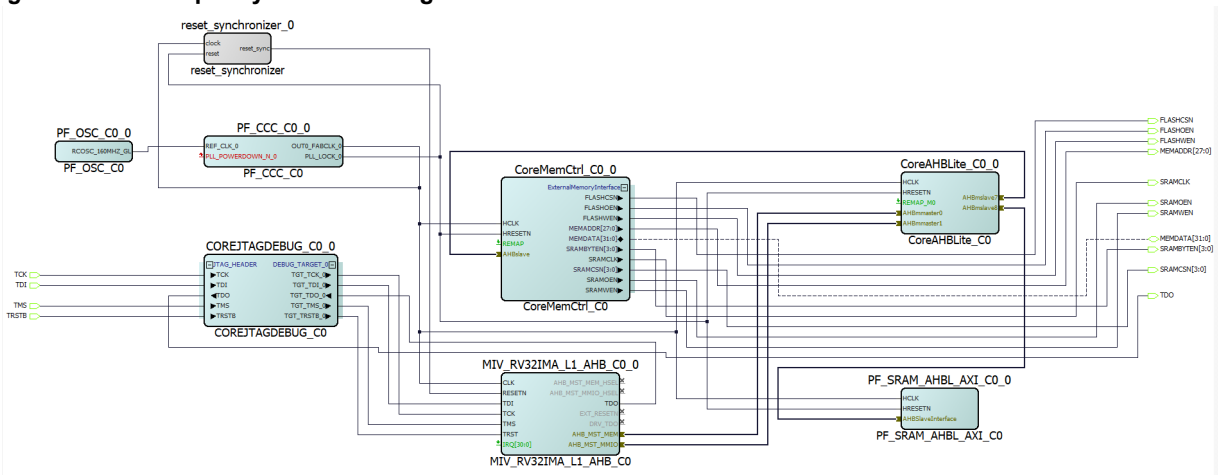
The testbench is based around a bus functional model (BFM) which acts as an AHB-Lite master and is controlled by a command file, named corememctrl_usertb.bfm. Simple memory models are used to model external flash and SRAM devices. The BFM writes to and reads from the memory models through CoreMemCtrl.

Each time the testbench is run, the corememctrl_usertb.bfm command file is processed by an executable to create corememctrl_usertb.vec. This instruction vectors file is read in by the BFM module in the testbench. The corememctrl_usertb.bfm file is a text file and can be edited to suit any specific needs. After generating a CoreMemCtrl design from SmartDesign, the file can be found in the <project>/simulation directory. For more information, refer to *DirectCore AMBA BFM User Guide*.

10 System Integration

This section provides an example, that shows the integration of CoreMemCtrl.

Figure 8 • **Example System Including CoreMemCtrl**



The example design described in this section contains CoreMemCtrl which is configured for both Flash and SRAM interface enabled and four synchronous SRAM chip enabled. The following explains the connection of components in the design.

- The output pin **reset_sync** of **Reset_synchronizer_0** is connected to the input pin **RESETN** of **MIV_RV32IMA_L1_AHB_C0_0**.
- The output pin **PLL_LOCK_0** of **PF_CCC_C0_0** is connected to the **HRESETN** of multiple modules including CoreMemCtrl.
- REF_CLK_0 of the PF_CCC_C0_0 instance is a 160MHz clock, driven from the on-chip oscillator.
- The Output pin **OUT0_FABCLK_0** of **PF_CCC_C0_0** (100 MHz) is connected to all the input clock signals in the design.
- All the External Memory Interface signals of **CoreMemCtrl_C0_0** are connected to either one of on board Flash Memory, SRAM memory or both of them.

The example design can be obtained from the Microsemi technical support team.

11 Connecting to External Memories

This section includes several diagrams, which illustrate how to connect the external memory interface ports of CoreMemCtrl to SRAM and flash devices. In the diagrams, generic representations of SRAM and flash devices are used. These representations are intended to cover devices from a range of manufacturers. Some manufacturers label the pins of their devices differently from others. In the following diagrams, the pin names on the SRAM and flash memories should be interpreted as described in Table 5. Table 5 Lists the pin descriptions as used in the diagrams.

Table 5 • **Pin Descriptions for Connection Diagrams**

Pin Name in Diagrams	Description
E#	Chip enable/chip select
G#	Read or output enable
W#	Write enable
BW#	Byte write enable on SRAM
LB#	Lower byte enable on SRAM
UB#	Upper byte enable on SRAM
BYTE#	Byte (8 bit) mode operation on flash
A0, A1, …, An	Address bus pins. An is the most significant address bit.
DQ[n:m]	Bidirectional data bus
CLK	Clock input on synchronous SRAM
ADV#	Burst address advance on synchronous SRAM
ADSP#	Address status processor on synchronous SRAM
ADSC#	Address status controller on synchronous SRAM
FT#	Flow-through/pipeline mode operation on synchronous SRAM

16-bit flash devices are available from several manufacturers and normally have a BYTE# input which, when asserted Low, causes the device to operate in 8-bit mode. Only the lower half of the data bus (that is, DQ[7:0]) is used to carry data in 8-bit mode. Manufacturers of flash devices generally use one of two approaches to addressing:

- The A0 pin is only used (for the least significant address bit) when in 8-bit mode. When operating in 16-bit mode, the A0 pin is not used; typically the A0 input buffer is turned off in this type of flash device when the BYTE# pin is High. The remaining address pins (A[n:1]) are always used for addressing.
- All of the address pins, including A0, are used for addressing in both 8-bit and 16-bit mode. On this type of flash device, the upper data pin, DQ[15], is typically reused as the least significant address bit input when operating in 8-bit mode.

Refer to the datasheet of the external flash device being used with CoreMemCtrl to check how the device handles addressing. The flash addressing configurable option should be set to suit the flash device(s) in use. Flash addressing Type must also be correctly set when interfacing to a single 16-bit flash device, that is when the flash data bus width is set to 16 bit.

Snippets from the CoreMemCtrl configuration window are included in each of the following connection diagrams to show the correspondence between the configuration options and the associated memory connections.

Figure 9, page 28, and Figure 10, page 28 shows two different external memory systems. Figure 9, page 28 illustrate flash devices which use addressing as described in the first addressing approach. The flash devices in Figure 10, page 28 use the approach to addressing described in the second addressing

approach. Figure 11, page 29 and Figure 12, page 30 illustrate additional pin connections relevant to synchronous SRAM devices, which are not shown in previous diagrams.

Figure 9 • **Connecting to SRAM and Single Flash Device with 16-bit Flash Data Bus and Flash of Type 1**

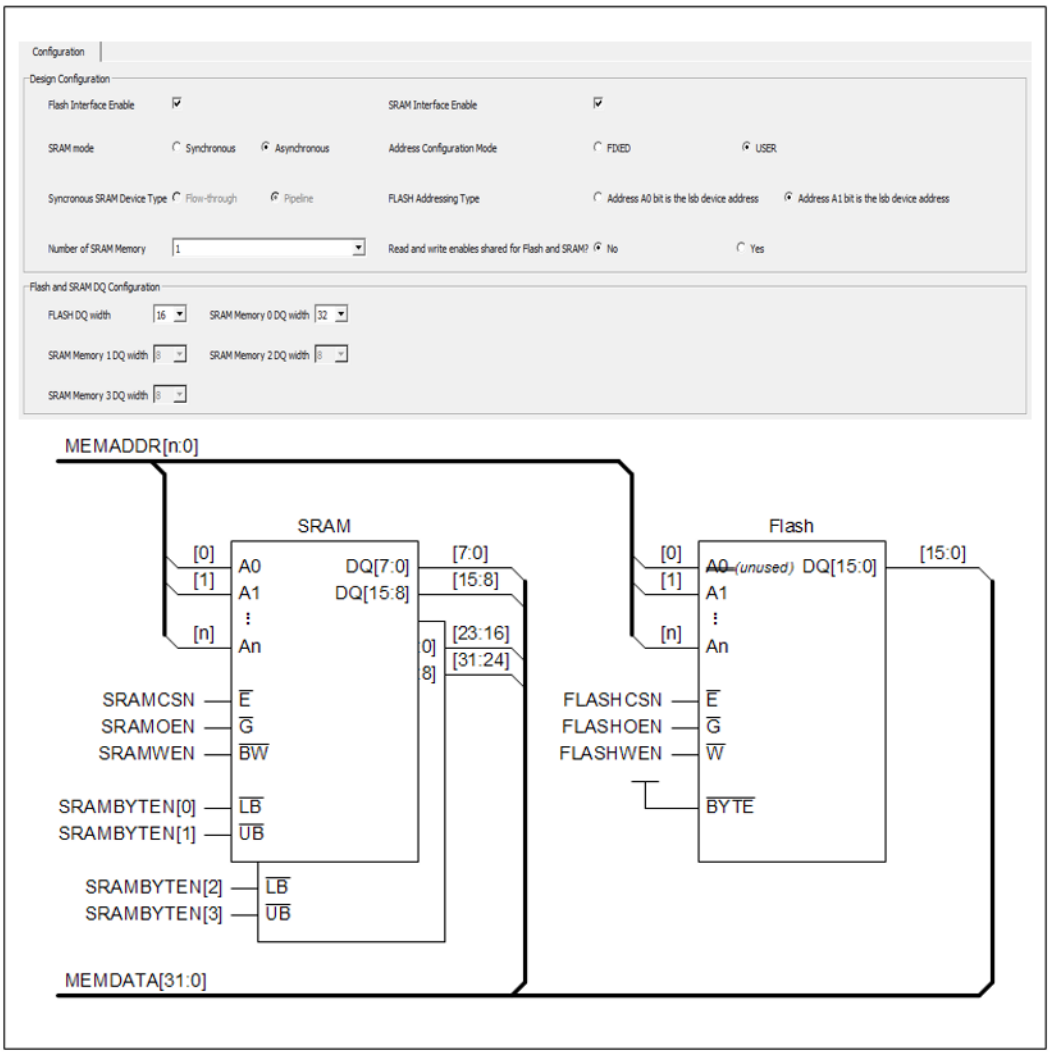


Figure 10 • **Connecting to SRAM and Single Flash Device with 16-bit Flash Data Bus and**

Flash of Type 2

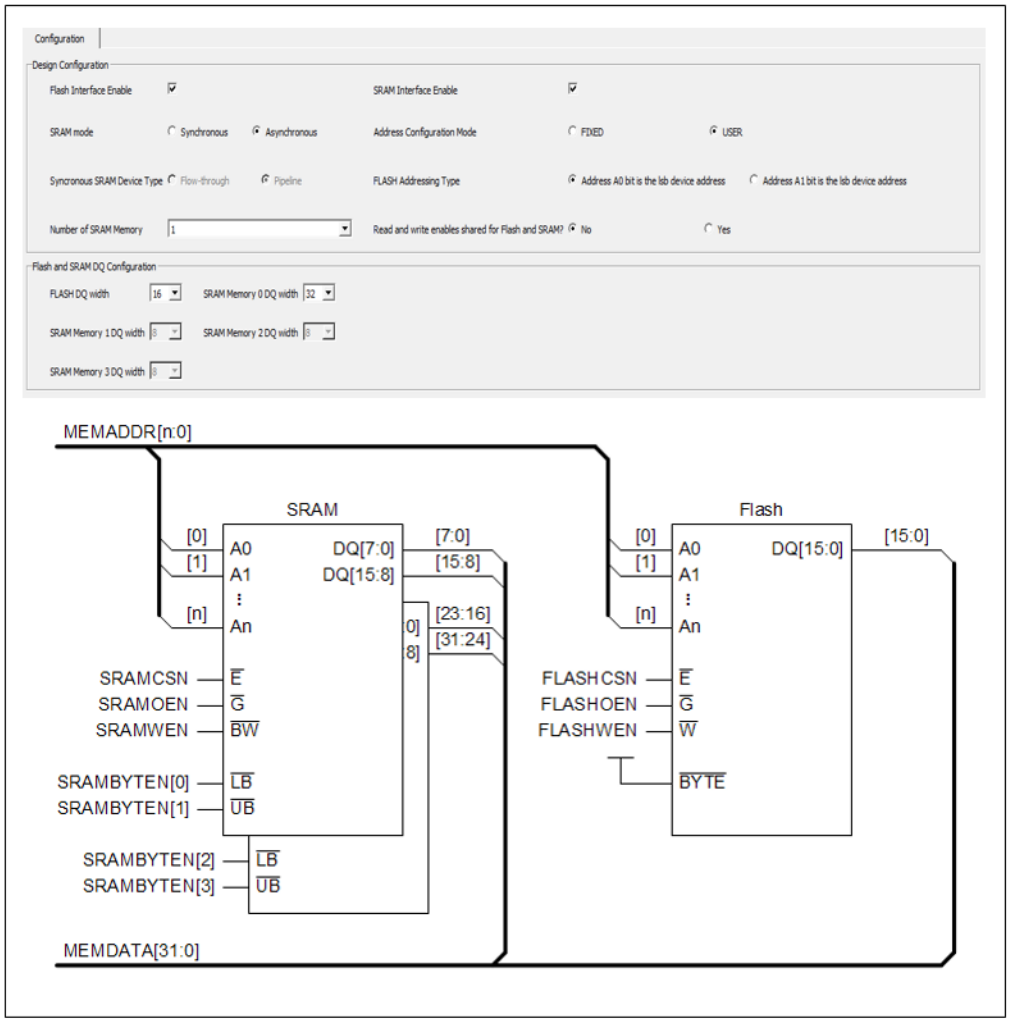


Figure 11 • **Matching CoreMemCtrl Configuration to FT# Pin Level on Synchronous SRAM**

Devices with FT# Input

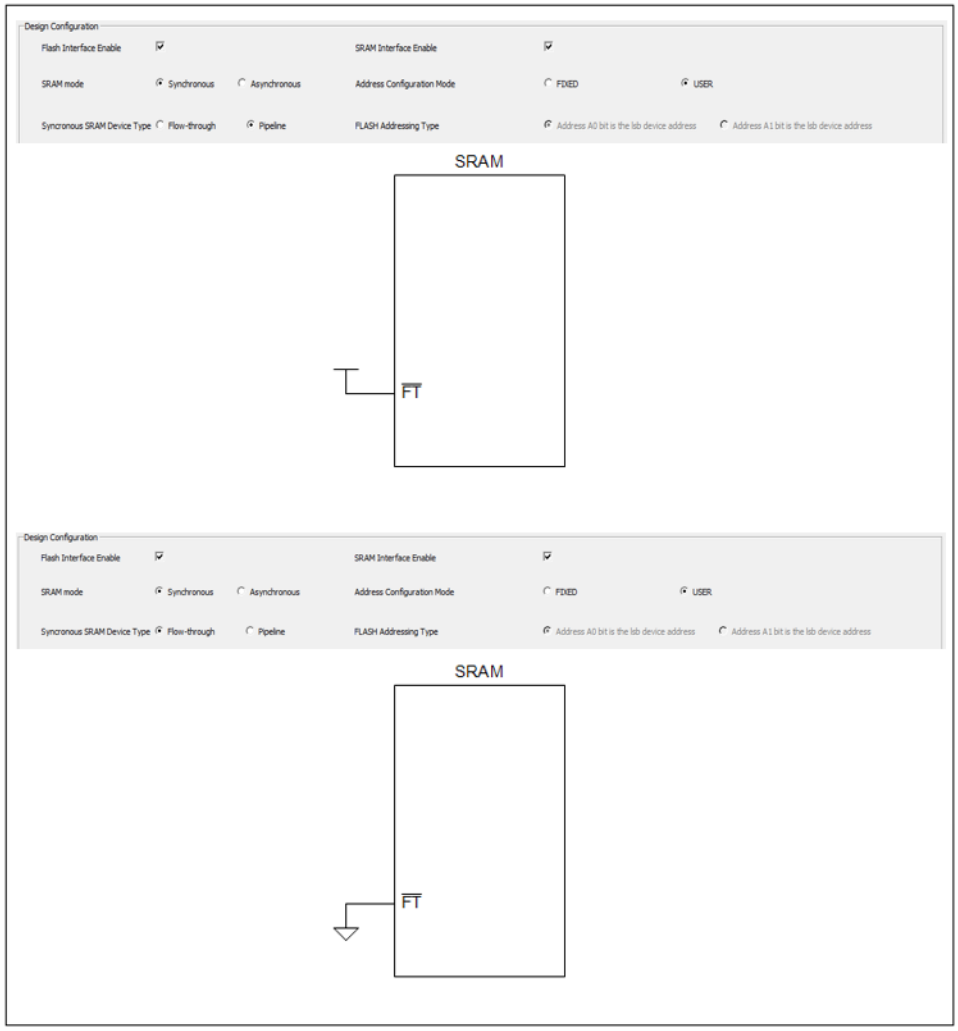


Figure 12 • **Additional Connections for Synchronous SRAM Devices**