SmartFusion2 SoC FPGA - Cache Controller Configuration - Libero SoC v11.7

Table of Contents

Purpose

This application note explains the cache controller features and how to configure the cache controller for various cacheable memories in the SmartFusion®2 system-on-chip (SoC) FPGA devices.

References

The following are the references:

- *AC390: SmartFusion2 SoC FPGA – Remapping eNVM, eSRAM, and DDR/SDR SDRAM Memories Application Note*
- *SmartFusion2 MSS ARM Cortex-M3 Configuration Guide*
- *UG0451: SmartFusion2 and IGLOO2 Programming User Guide*
- *UG0450: SmartFusion2 SoC and IGLOO2 FPGA System Controller User Guide*
- *UG0331: SmartFusion2 Microcontroller Subsystem User Guide*
- *Configuring Serial Terminal Emulation Programs*

Introduction

The SmartFusion2 devices integrate an 8 kb instruction cache. The following memories are cacheable:

- Embedded non-volatile memory (eNVM)
- Low power double data rate (LPDDR)

To aid in system reliability, the instruction cache is constructed of single event upset (SEU) tolerant latches. This application note describes the configuration of cache controller for various cacheable memories.

Design Requirements

Table 1 shows the design requirements.

Table 1 • **Design Requirements**

Design Requirements	Description
Hardware Requirements	
SmartFusion2 Security Evaluation Kit • 12 V adapter • FlashPro4 programmer • USB A to Mini-B USB cable *Note:* Refer the *UG0594: SmartFusion2 Security Evaluation Kit User Guide* for more information	Rev D or later
Host PC or Laptop	• Windows XP SP2 Operating System - 32-bit or 64-bit • Windows 7 Operating System - 32-bit or 64-bit
Software Requirements	
Libero® System-on-Chip (SoC)	v11.7
SoftConsole	v3.4 SP1*
Host PC Drivers	USB to UART drivers

*Note: *For this application note, SoftConsole v3.4 SP1 is used. For using SoftConsole v4.0, see the TU0546: SoftConsole v4.0 and Libero SoC v11.7 Tutorial.*

SmartFusion2 Cache Controller Overview

Figure 1 shows the system-level view of the cache controller in the SmartFusion2 device.

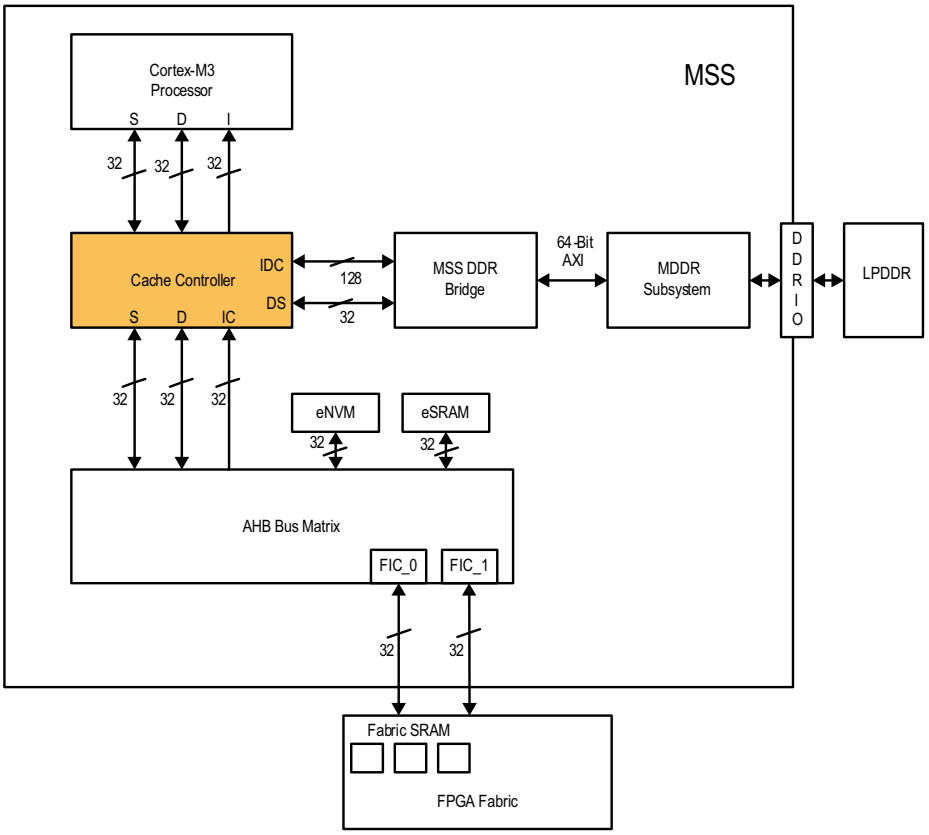

***Figure 1* • System-Level View of Cache Controller in the SmartFusion2 Device**

Figure 2 shows the block diagram of the SmartFusion2 cache controller. Refer to the *UG0331: SmartFusion2 Microcontroller Subsystem User Guide* for more information on cache controller.

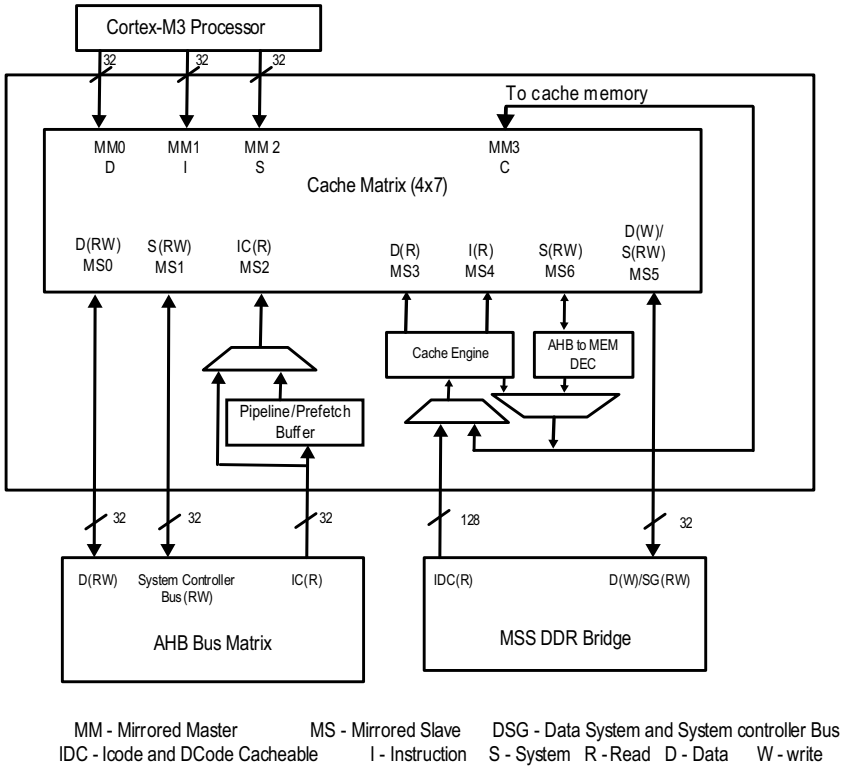

***Figure 2* • SmartFusion2 Device Cache Controller Block Diagram**

Cacheable Memory Regions

The following sections explain memory mapping of eNVM and LPDDR address space to cacheable memory regions. The code space of the ARM® Cortex®-M3 processor ranges from 0x00000000 to 0x1FFFFFFF (0.5 GB). The address space of eNVM or LPDDR can be mapped to code space of the Cortex-M3 processor to make the memory region cacheable. Design examples are provided in "Appendix: Design and Programming Files" on page 16.

Remapping eNVM as Cacheable Region

The address range of eNVM_0 is from 0×60000000 to 0×6003FFFF and the address range of eNVM_1 is from 0×60040000 to 0×6007FFFF. By default, the full eNVM memory from 0×60000000 to 0×6007FFFF is mapped as a cacheable region. The eNVM base address 0×60000000 is remapped to Corte×-M3 processor address space 0×00000000. You can remap any offset of eNVM address to the Corte×-M3 processor address space 0×00000000 by using the ENVM_CR, ENVM_REMAPSIZE, and ENVM_REMAP_BASR_CR system registers.

Refer to "Appendix: Design and Programming Files" on page 16 for eNVM as cacheable region design files and follow "Running the Design" on page 11 for executing the reference design.

Table 2 • **Memory Map of eNVM to Cortex-M3 Processor Code Region**

Data/Code Region	Space	Address Range
M3 Data Region	**RESERVED**	0×E000_0000 to 0×FFFF_FFFF
	DDR _SPACE 3 (256 MB)	0×D000_0000 to 0×DFFF_FFFF
	DDR _SPACE 2 (256 MB)	0×C000_0000 to 0×CFFF_FFFF
	DDR_ SPACE 1 (256 MB)	0×B000_0000 to 0×BFFF_FFFF
	DDR _SPACE 0 (256 MB)	0×A000_0000 to 0×AFFF_FFFF
	eNVM SFR, Remap Area etc (1 GB)	0×6000_0000 to 0×9FFF_FFFF
	Peripheral [SPI, UART, CAN, Fabric etc] (0.5 GB)	0×4000_0000 to 0×5FFF_FFFF
	RESERVED	0×2001_0000 to 0×3FFF_FFFF
	eSRAM-1 (32 KB)	0×2000_8000 to 0×2000_FFFF
	eSRAM-0 (32 KB)	0×2000_0000 to 0×2000_7FFF
M3 Code Region	**RESERVED**	0×0008_0000 to 0×1FFF_FFFF
	eNVM (Virtual View) [512 KB]	0×0000_0000 to 0×0007_FFFF

Remapping of External RAM as Cacheable Region

Remap the LPDDR memory address to the bottom (0×0000_0000) of the Cortex-M3 processor code region by using the DDR_CR system register. Any portion of the mapped memory can be made cacheable. The cacheable region can be configured to 128 MB, 256 MB or 512 MB dynamically by using the CC_REGION_CR system register. Ensure that the stack and data/heap sections of the application are out of the cacheable memory region. Refer to the *AC390: Remapping eNVM, eSRAM, and LPDDR Memories Application Note,* for more information on remapping techniques and linker script file generation.

Table 3 • **Memory Map of External RAM to Cortex-M3 Processor Code Region**

Data/Code Region	Space	Address Range
M3 Data Region	**RESERVED**	0×E000_0000 to 0×FFFF_FFFF
	DDR _SPACE 3 (256 MB)	0×D000_0000 to 0×DFFF_FFFF
	DDR _SPACE 2 (256 MB)	0×C000_0000 to 0×CFFF_FFFF
	DDR_ SPACE 1 (256 MB)	0×B000_0000 to 0×BFFF_FFFF
	DDR _SPACE 0 (256 MB)	0×A000_0000 to 0×AFFF_FF×FF
	eNVM SFR, Remap Area etc (1 GB)	0×6000_0000 to 0×9FFF_FFFF
	Peripheral [SPI, UART, CAN, Fabric etc] (0.5 GB)	0×4000_0000 to 0×5FFF_FFFF
	RESERVED	0×2001_0000 to 0×3FFF_FFFF
	eSRAM-1 (32 KB)	0×2000_8000 to 0×2000_FFFF
	eSRAM-0 (32 KB)	0×2000_0000 to 0×2000_7FFF
M3 Code Region	**DDR _SPACE 1 (256 MB)**	0×1000_0000 to 0×1FFF_FFFF
	DDR _SPACE 0 (256 MB)	0×0000_0000 to 0×0FFF_FFFF

SmartFusion2 Cache Controller Features

The following sections explain the various user configurable features of the cache controller in the SmartFusion2 device:

- Cache Memory Enable or Disable
- Cache Flush
- Cache Locked Mode

Cache Memory Enable or Disable

Cache memory can be enabled or disabled dynamically by using the CC_CR system register. Instructions are cached when the cache memory is enabled. In Cache Disabled mode, all transactions are treated as non-cacheable.

Use the following steps to enable or disable the cache memory dynamically using the application code:

1. Set the cacheable region.
2. Enable cache memory.
3. Run the task.
4. Get the cache status information.
5. Disable cache memory.

Refer to Table 4 on page 11 for APIs to enable or disable cache memory and to get cache status information.

Cache Flush

Cache memory can be flushed in the following two ways:

- Complete cache memory flush: When you flush the full cache memory, all the cached instructions are deleted.
- Index based cache memory flush: When you flush one index in the cache memory, it invalidates all tags of four sets at one index only.

The following steps describe how to flush the cache memory:

1. Enable cache memory.
2. Run the task (the instructions are cached).
3. Disable cache memory.
4. Flush the cache memory (the cached instructions are deleted).

Refer to Table 4 on page 11 for APIs to flush the cache memory.

Cache Locked Mode

The Cache Locked mode is a special mode that provides predictable execution, which is a requirement for some specific applications. Before enabling the Cache Locked mode, the software ensures that the code is copied to cache memory by simulating a sequential location cache miss through I-code. After copying the complete 8 KB of data, the Cache Locked mode is enabled. After the Cache Locked mode is enabled, any access from 0 to 8 KB is directly read from the cache and the cache is not invalidated or refilled for normal operations. Memory region beyond 8 KB is treated as non-cacheable and accessed as per the memory map.

The Cache Locked mode can only be used with either DDR or eNVM memory and the lock base address must be in the Cortex-M3 processor code region. The code image that is copied to cache memory is also present in eNVM or DDR memory. After executing the code from the cache, the execution control comes to the main memory to execute the remaining code image. You can enable or disable the Cache Locked mode dynamically.

Figure 3 shows a simple program execution flow with Cache Locked mode.

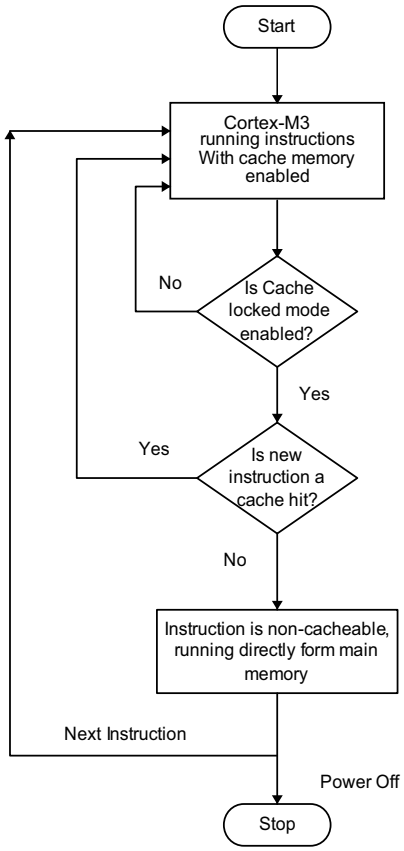

***Figure 3* • Simple Program Execution Flow with Cache Locked Mode**

Refer to Table 4 on page 11 for APIs to enable or disable the Cache Locked mode.

Design Description

The example designs use MMUART_1, eSRAM, DDR, and eNVM memory controllers. In the design example, microcontroller subsystem clock conditioning circuitry (MSS CCC) is configured to run M3_CLK at 80 MHz, which drives the clock to Cortex-M3 processor. The cache controller can be configured either using cache controller block in System Builder configurator or through APIs (Table 4 on page 11). The software application calculates the n^{th} Fibonacci number with and without cache controller and compares the execution time. It also gets the cache status information such as, cache hit, cache miss, cache hit rate, and cache miss rate. This application also supports cache memory flushing.

Hardware Implementation

The hardware implementation involves configuring MDDR, MMUART_1, and clocks using System Builder. Figure 4 shows the top-level SmartDesign of the cache controller configuration.

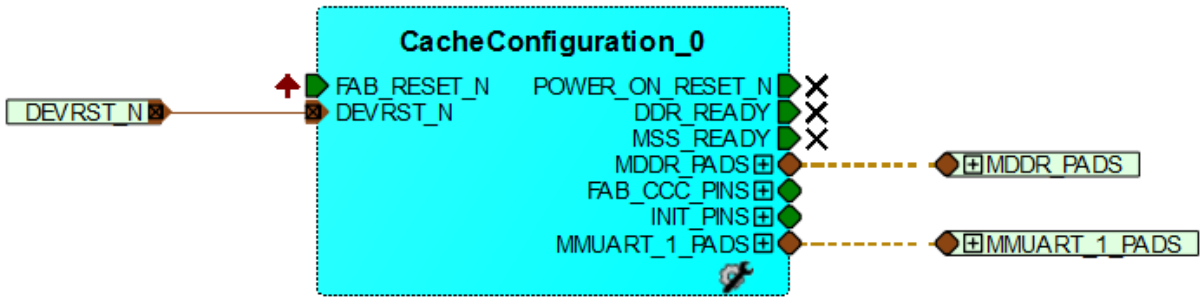

***Figure 4* • Top-Level SmartDesign**

The MSS_CCC clock is sourced from Fabric CCC. Fabric CCC is configured to provide the 80 MHz clock using GL0. Figure 5 shows the system clocks configurations for M3_CLK, MDDR_CLK, and APB_0_CLK/APB_1_CLK.

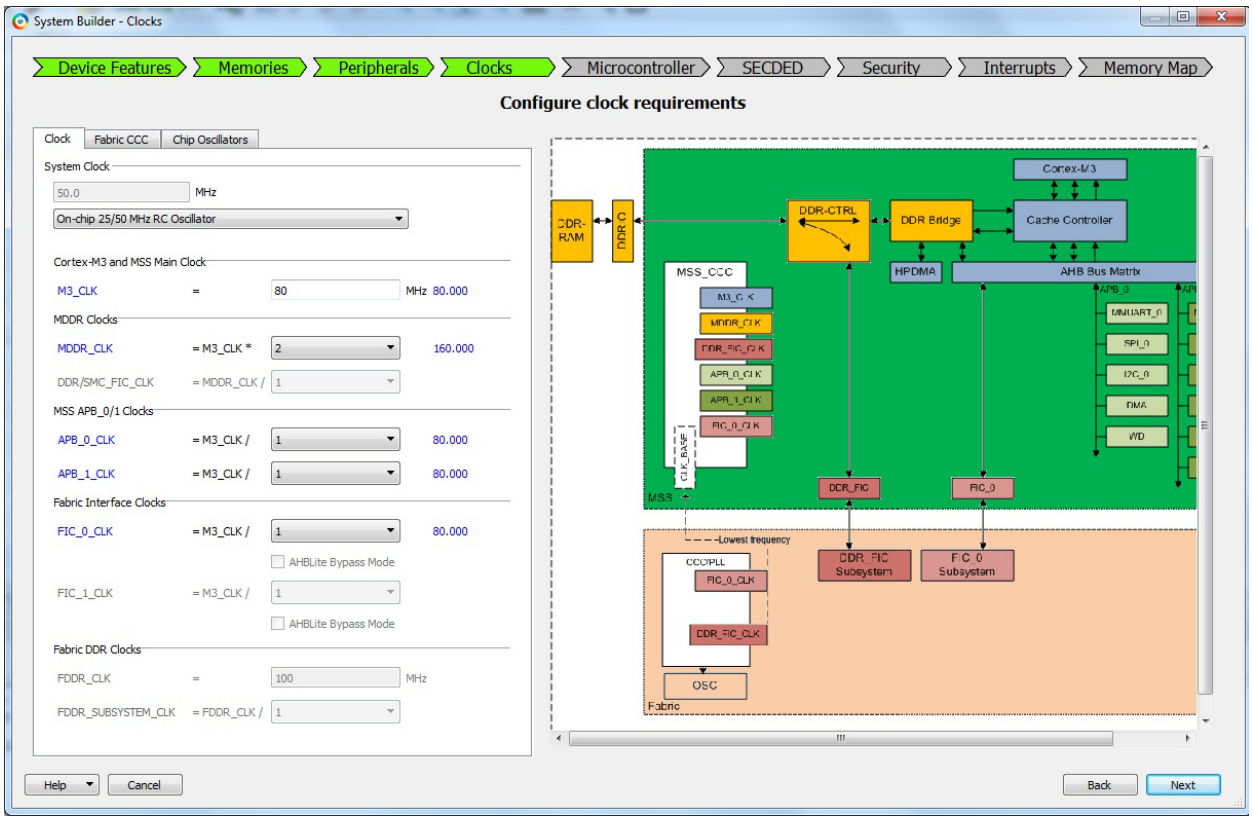

***Figure 5* • Clock Configurations**

MMUART_1 is routed through the FPGA fabric for communicating with the serial terminal program. MDDR is configured for LPDDR at 80 MHz speed. Figure 6 shows MSS MDDR configuration settings. Click **Import Configuration** to import the register configuration for LPDDR (refer to "Appendix: Design and Programming Files" on page 16 for DDR configuration file).

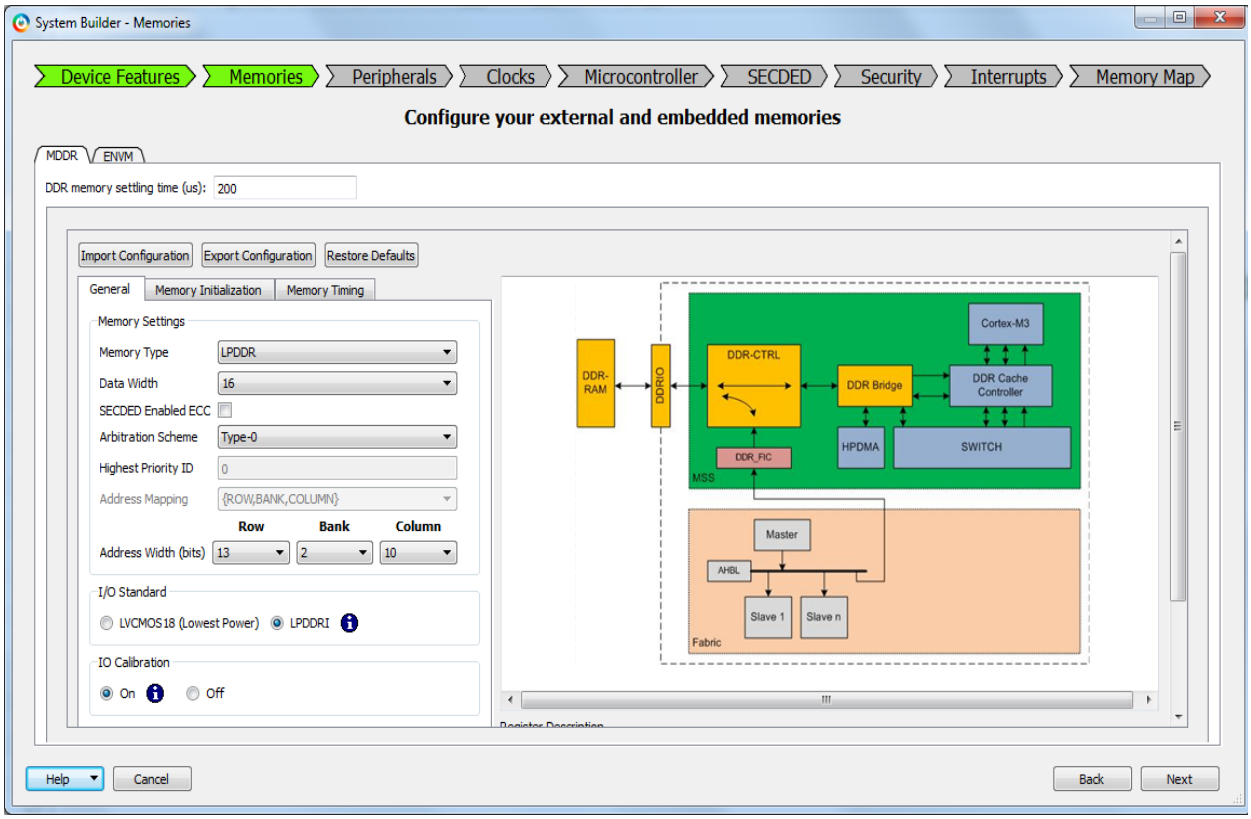

***Figure 6* • MSS External Memory Configurator**

Figure 7 shows the cache controller configuration through **System Builder** in the Libero SoC software.

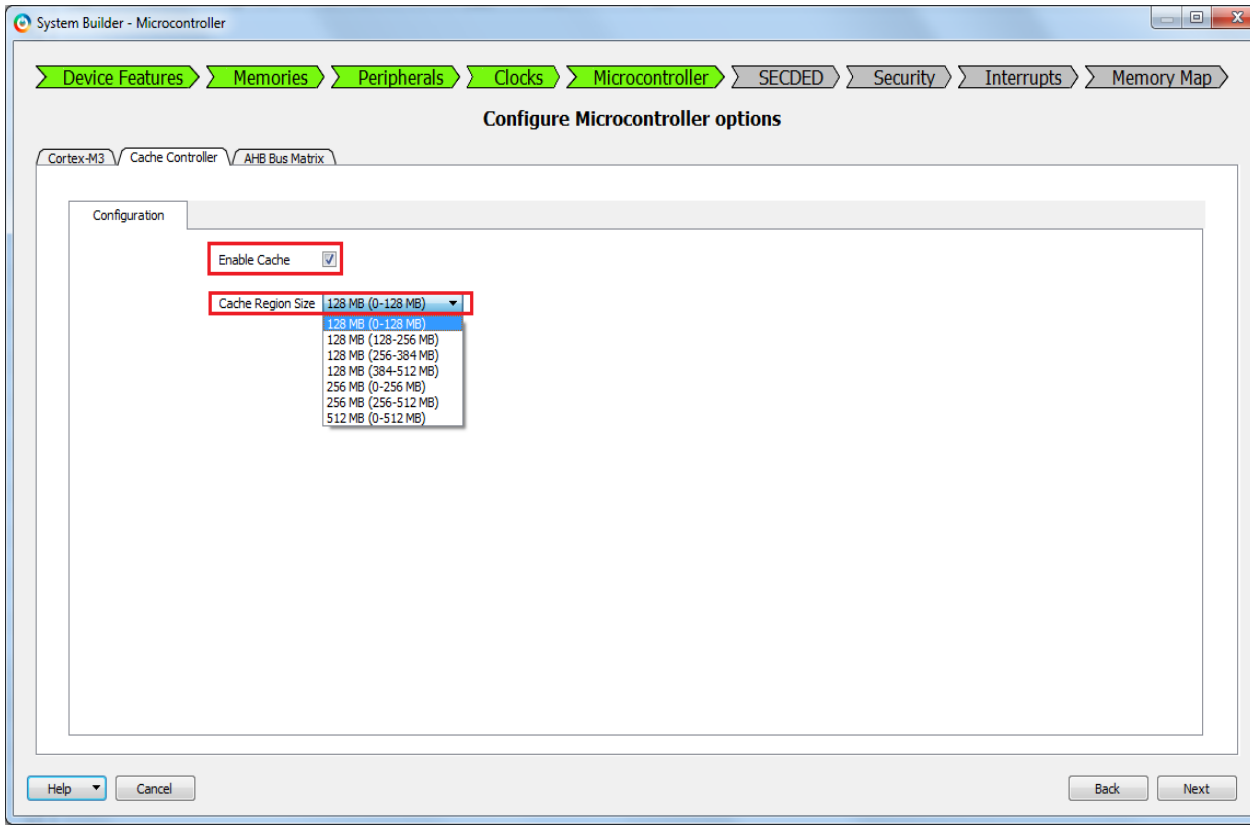

***Figure 7* • Cache Controller Configuration Through System Builder in Libero**

Note: Enabling cache through System Builder is not required in this application note design.

Software Implementation

The software design example performs the following operations:

- Enabling or disabling the cache controller
- Selecting cacheable region in case of DDR memory
- Cache memory flushing
- Initialization of timer to measure execution time
- Calculating cache hit rate, cache miss rate, and task execution time
- Displaying results on serial terminal program (for example, HyperTerminal) using MMUART_1

The example software design uses UART based serial communication to communicate with serial terminal program on the host PC. This example takes finding the n^{th} Fibonacci number as the task and calculates the result with and without cache controller. The Fibonacci number is randomly selected. It displays the cache hit rate, cache miss rate, and execution time on serial terminal program.

In this design example, the following application images are created, which can be remapped to the bottom (0×0000_0000) of the Cortex-M3 processor code region to execute the image:

1. eNVM as a cacheable region.
2. LPDDR as a cacheable region with stack and data segment in the non-cacheable LPDDR memory.
   128 MB of LPDDR memory is selected as a cacheable region and the following 32 KB of LPDDR is spared for stack and data/heap sections. The stack and data/heap sections of the application must be allocated in the non-cacheable LPDDR memory region.
3. LPDDR as a cacheable region with stack and data segment in eSRAM.
   128 MB of LPDDR memory is selected as a cacheable region, and 32 KB of eSRAM is spared for stack and data/heap sections.

Firmware Drivers:

The following firmware drivers are used in this application:

- MSS MMUART driver
  - To communicate with serial terminal program on the host PC
- MSS Timer driver
  - To measure the task execution time

List of APIs:

The following APIs in Table 4 are implemented in the software design to configure the cache controller.

Table 4 • **APIs to Configure the Cache Controller**

API	Description	Input Parameters
MSS_CC_enable	Enables the cache memory	Void
MSS_CC_disable	Disables the cache memory	Void
MSS_CC_enable_lock	Enables Cache locked mode	Void
MSS_CC_disable_lock	Disables Cache locked mode	Void
MSS_CC_flush_index	Flushes one index in the cache memory, which is used to invalidate all tags of four sets at one index only	Index value
MSS_CC_flush	Flushes the cache memory, which is used to invalidate all tags of four sets at the same time	Void
MSS_CC_set_region	Sets the cacheable region size to 128 MB, 256 MB, or 512 MB	Cacheable region value
MSS_CC_get_miss_cnt	Returns the total number of cache misses that occur in the cacheable region through ICode bus	Void
MSS_CC_get_hits_cnt	Returns the total number of cache hits occur in the cacheable region through ICode bus	Void
MSS_CC_get_trans_cnt	Returns the total number of transaction counts processed by cache engine	Void

Running the Design

This application note provides the following design files and describes the hardware and software requirements, board settings, and steps to run the design.

- eNVM as cacheable region
- LPDDR as cacheable region with stack and data segment in non cacheable LPDDR memory
- LPDDR as cacheable region with stack and data segment in eSRAM

Board Settings

Connect the following jumpers on the SmartFusion2 Security Evaluation Kit, as described in Table 5. Switch OFF the power supply switch SW7 on the board, while making the jumper connections.

Table 5 • **SmartFusion2 Security Evaluation Kit Jumper Settings**

Jumper	Pin (From)	Pin (To)	Comments
J22, J23, J24, J8, J3	1	2	These are the default jumper settings of the SmartFusion2 Security Evaluation Kit board. Make sure these jumpers are set accordingly.

Steps to Run the Design

The following steps describe how to run the design:

1. Connect the FlashPro4 programmer to the J5 connector of the SmartFusion2 Security Evaluation Kit. Connect the J18 connector provided on the SmartFusion2 Security Evaluation Kit to the host PC using the USB mini-B cable. Ensure that the USB to UART bridge drivers are automatically detected by verifying the Device Manager, as shown in Figure 8.

Note: Copy the COM port number for serial port configuration. Ensure that the COM port location is specified as **on USB Serial Converter D**, as shown in Figure 8.

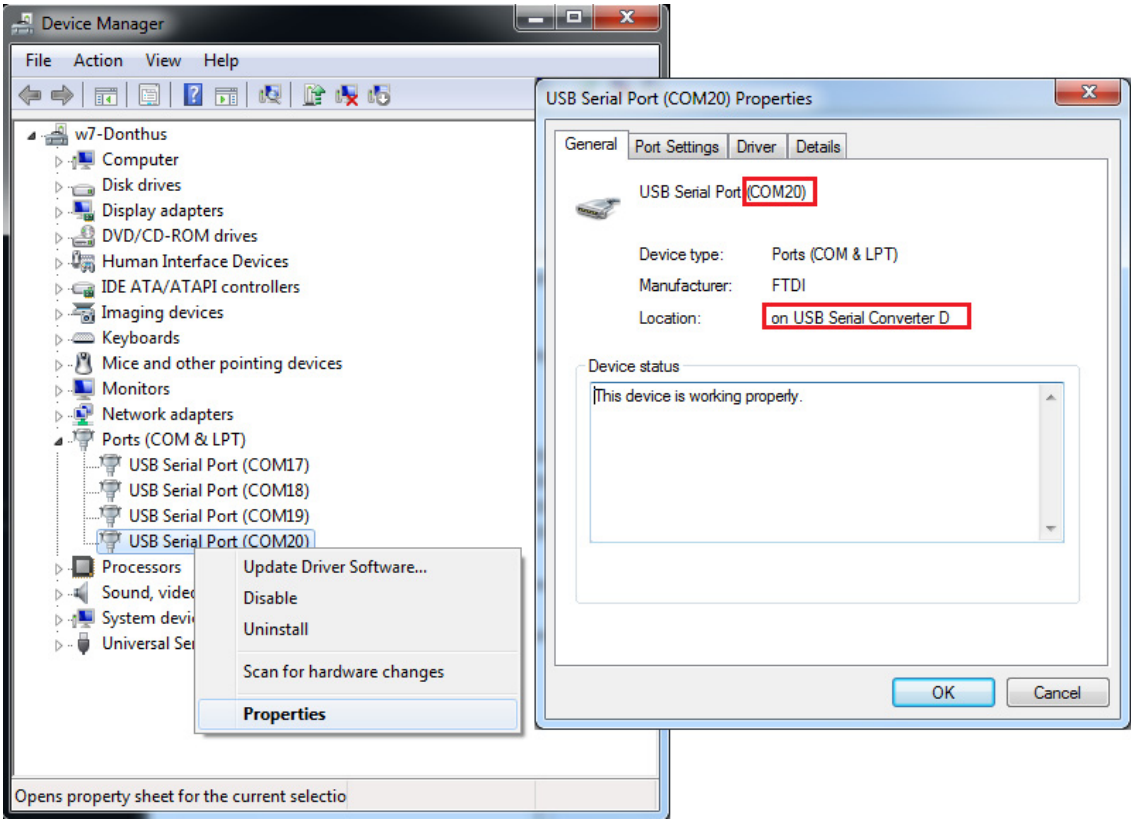

Figure 8 • **USB to UART Bridge Drivers**

2. If USB to UART bridge drivers are not installed, download and install the drivers from www.microsemi.com/soc/documents/CDM_2.08.24_WHQL_Certified.zip.

3. Connect the power supply to the J6 connector and switch ON the power supply switch, SW7. Start the HyperTerminal program with a baud rate of 115200, 8 data bits, 1 stop bit, no parity, and no flow control. If your computer does not have HyperTerminal program, use any free serial terminal emulation program such as PuTTY or Tera Term. Refer to the *Configuring Serial Terminal Emulation Programs* tutorial for configuring the HyperTerminal, Tera Term, and PuTTY.
4. Program the SmartFusion2 Security Evaluation Kit board with the provided programming file *<download_folder>\m2s_ac389_liberov11p6_df\programming_file\CacheConfiguration.stp*, refer to "Appendix: Design and Programming Files" on page 16 using the FlashPro software.
5. Press **SW6** switch to reset the board after successful programming.

   The serial terminal program displays the user options, as shown in Figure 9.

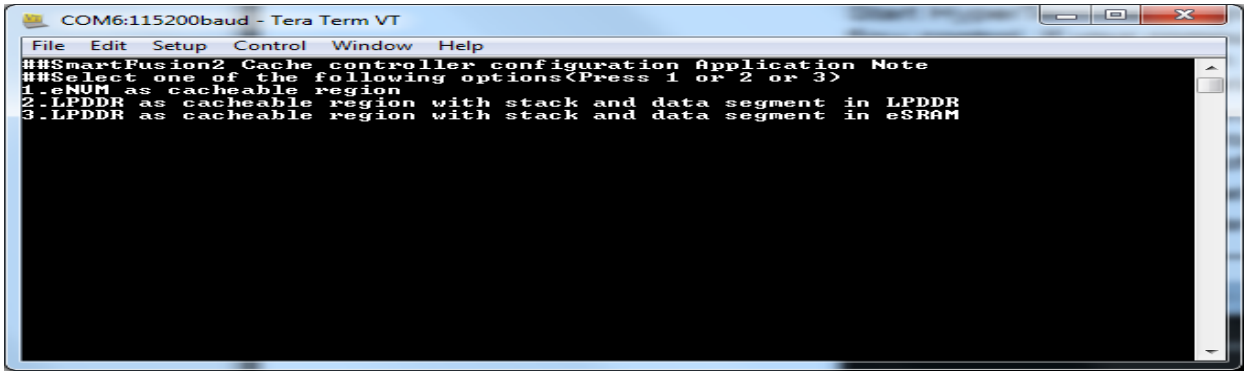

***Figure 9* • User Options**

6. Select the option to remap the image to the bottom (0×0000_0000) of the Cortex-M3 processor code region and to execute the code. Select option 1 to execute eNVM as cacheable memory application image as shown in Figure 10. Select 2 or 3 to execute LPDDR as cacheable memory application image as shown in Figure 11 on page 14 and Figure 12 on page 14.

Note: Reset the SmartFusion2 Security Evaluation Kit board to switch among the application images.

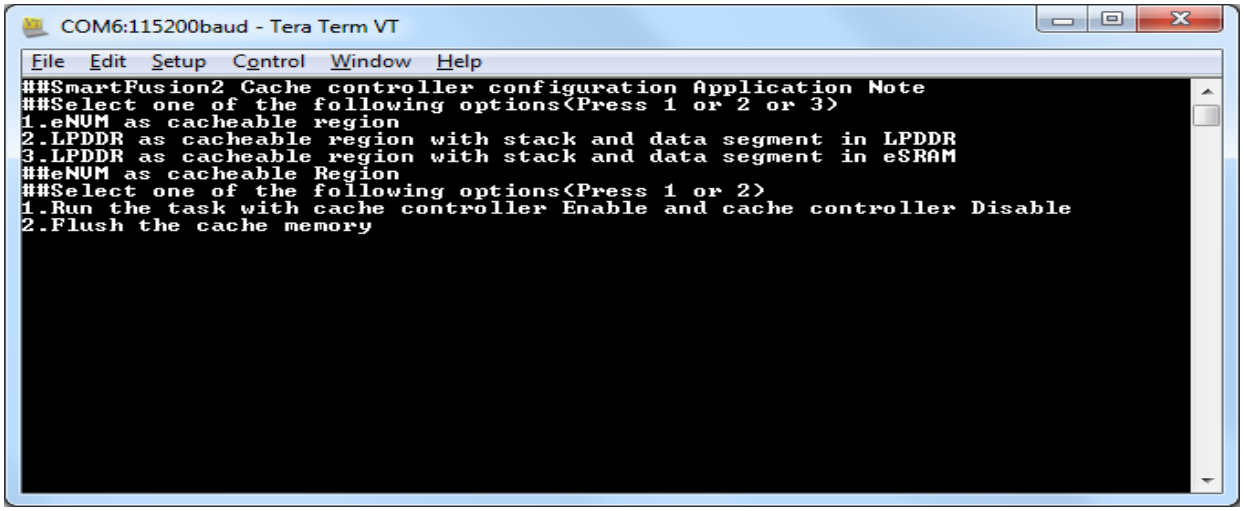

***Figure 10* • Executing eNVM as Cacheable Region Application**

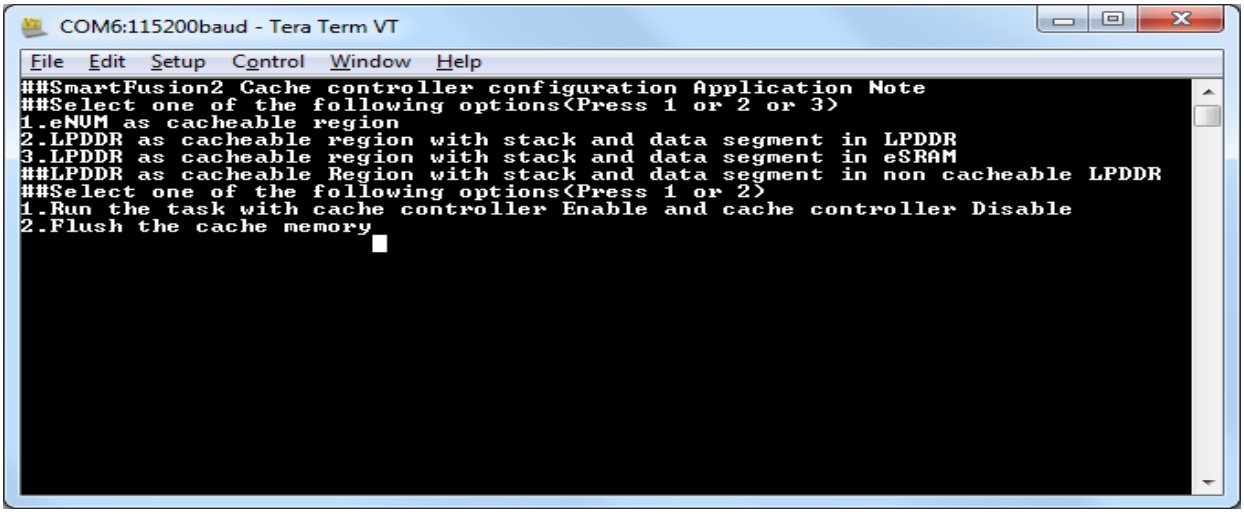

Figure 11 **• Executing LPDDR as Cacheable Region Application**

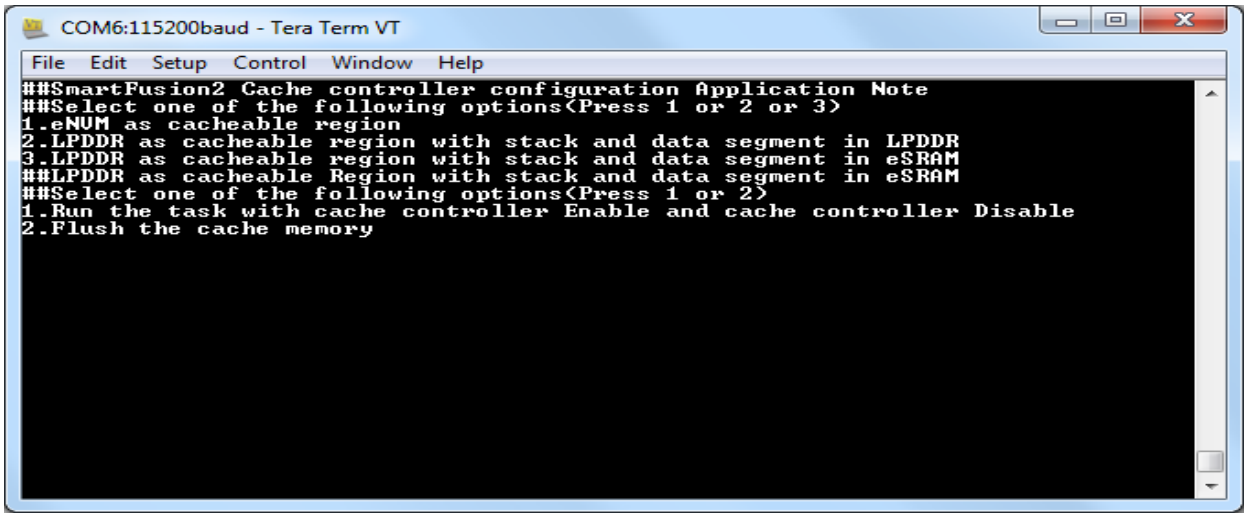

Figure 12 **• Executing LPDDR as Cacheable Region Application**

Selecting option 1 executes the task with the cache controller enabled. The application program calculates the task execution time with cache memory and without cache memory and also displays the cache status information, as shown in Figure 13.

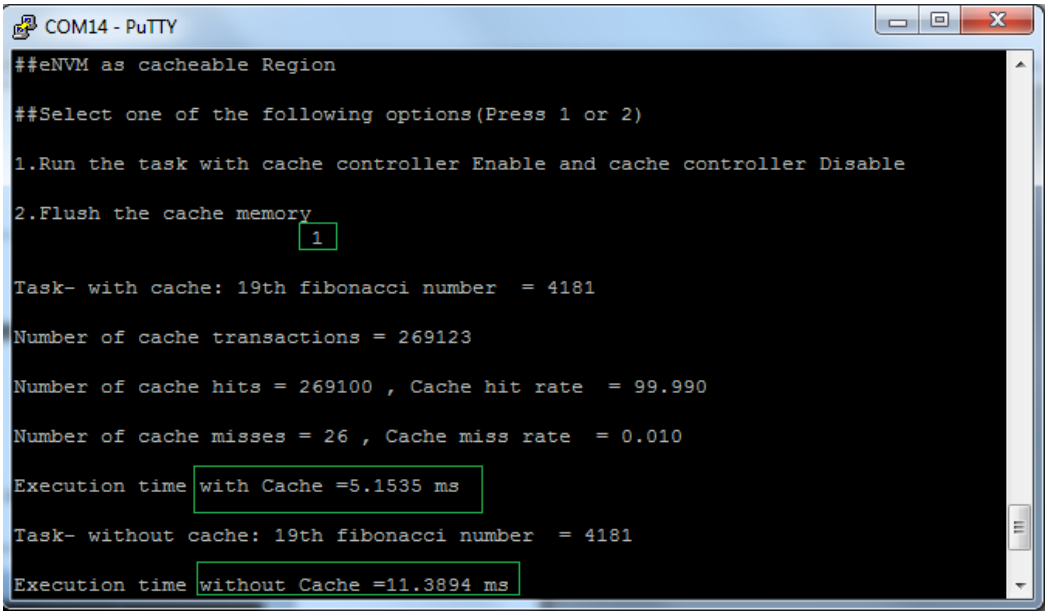

***Figure 13* • Cache Status Information With Cache Memory Enabled**

Selecting option 2 flushes the cache memory completely.

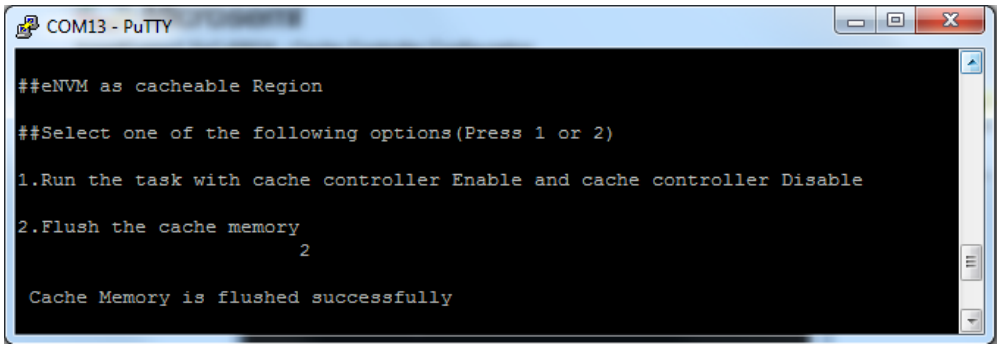

***Figure 14* • Cache Memory Flush**

Conclusion

This application note explains the cache controller configuration in eNVM and DDR memory remap modes supported by the SmartFusion2 devices.

Appendix: Design and Programming Files

Download the design files from the Microsemi SoC Products Group website:

http://soc.microsemi.com/download/rsc/?f=m2s_ac389_liberov11p7_df

The design file consists Libero SoC Verilog project, SoftConsole software project, and programming files (*.stp) for the SmartFusion2 Security Evaluation Kit board. Refer to the `Readme.txt` file included in the design file for the directory structure and description.

Download the programming files from the Microsemi SoC Products Group website:

http://soc.microsemi.com/download/rsc/?f=m2s_ac389_liberov11p7_pf

The programming file consists STAPL programming file (*.stp) for the SmartFusion2 Security Evaluation Kit board.

List of Changes

The following table shows the important changes made in this document for each revision.

Revision	Changes	Page
Revision 10 (March 2016)	Updated the document for Libero SoC v11.7 software release (SAR 76876).	N/A
Revision 9 (October 2015)	Updated the document for Libero SoC v11.6 software release (SAR 71810).	N/A
Revision 8 (February 2015)	Updated the document for Libero SoC v11.5 software release (SAR 64751).	N/A
Revision 7 (October 2014)	Updated the document for Libero SoC v11.4 software release (SAR 61633).	N/A
Revision 6 (May 2014)	Updated the document for Libero SoC v11.3 software release (SAR 57102).	N/A
Revision 5 (November 2013)	Updated the document for Libero SoC v11.2 software release (SAR 52966).	N/A
Revision 4 (November 2013)	Updated note (SAR 51331).	10
	Updated Figure 5 and Figure 6 (SAR 51331).	8, 9
	Updated Table 5 and Table 6 with latest version of s/w v11.1 SP2 and the latest silicon Rev D (SAR 51331).	12
	Deleted Jumper J2 from the Table 5 (SAR 51331).	12
Revision 3 (May 2013)	Updated the document for Libero SoC v11.0 software release (SAR 47616).	N/A
Revision 2 (March 2013)	Updated for Libero SoC v11.0 beta SP1 release (SAR 45274).	N/A
Revision 1 (November 2012)	Modified "Remapping eNVM as Cacheable Region" section (SAR 42936).	4
	Modified "Remapping of External RAM as Cacheable Region" section (SAR 42936).	5
	Modified "Cache Flush" section (SAR 42936).	6
	Modified "Software Implementation" section (SAR 42936).	10
	Modified "Running the Design" section (SAR 42936).	11
	Updated Figure 9, Figure 13 and Figure 14 (SAR 42936).	13, 15
	Modified "Conclusion" section (SAR 42936).	15
	Modified "Appendix: Design and Programming Files" section (SAR 42936).	16

Microsemi Corporate 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

E-mail: sales.support@microsemi.com

Microsemi Corporation (Nasdaq: MSCC) offers a comprehensive portfolio of semiconductor and system solutions for communications, defense & security, aerospace 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; Power-over-Ethernet ICs and midspans; as well as custom design capabilities and services. Microsemi is headquartered in Aliso Viejo, Calif., and has approximately 4,800 employees globally. Learn more at **www.microsemi.com**.

51900257-10/03.16