# Mobile SDRAM Interface Design Example

## Table of Contents



## Overview

This document describes the design example for interfacing Microsemi low-power FPGAs with Mobile SDRAM devices. When used alongside reprogrammable FPGAs such as Microsemi's IGLOO® and ProASIC®3 product families, this IP makes an ideal solution for high volume, portable applications, such as cell phones, smartphones, PDAs, MP3 players, digital still cameras, video cameras, GPS devices, and portable games.

Associated files for this design example can be downloaded from the Microsemi website:
*http://soc.microsemi.com/download/rsc/?f=Mobile_SDRAM_Interface_DF*

## Design Description

The Mobile SDRAM Interface design example explained in this document is targeted for an Microsemi IGLOO device.

Figure 1 shows a diagram of the mobile SDRAM interface. The design uses Microsemi's Core8051 embedded controller and generates the required control signal to interface with the mobile SDRAM.

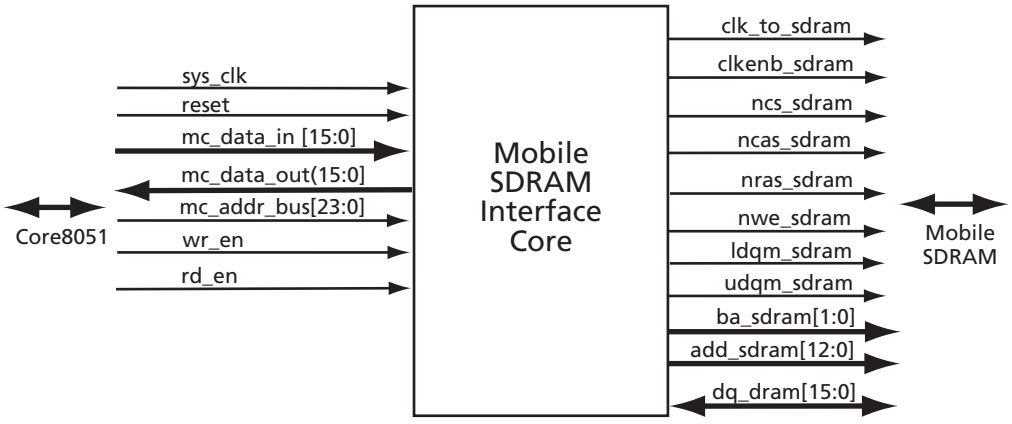


***Figure 1* • Mobile SDRAM Core Interface Diagram**

# Interface Description

The interface details of the IP are given in Table 1.

***Table 1 •*** **Interface Description**

| Port | Direction | Description |
|---|---|---|
| sys_clk | Input | System clock of 20 MHz |
| clk_to_sdram | Output | Mobile SDRAM clock 20 MHz |
| clkenb_sdram | Output | Clock enable (active High) |
| reset | Input | System reset (active Low) |
| rd_en_in | Input | Read enable (active High) |
| wr_en_in | Input | Write enable (active High) |
| mc_data_in[15:0] | Input | Write data for interface block |
| mc_data_out[15:0] | Output | Read back data from interface block |
| mc_addr_bus[23:0] | Input | Address bus |
| dq_sdram[15:0] | Bidirectional | Mobile SDRAM data bus |
| add_sdram[12:0] | Output | Mobile SDRAM address bus |
| ba_sdram[1:0] | Output | Bank address |
| ldqm_ sdram | Output | Lower data byte mask (active High) |
| udqm_ sdram | Output | Upper data byte mask (active High) |
| nwe_sdram | Output | Write enable for Mobile SDRAM (active Low) |
| nras_sdram | Output | Mobile SDRAM RAS (active Low) |
| ncas_sdram | Output | Mobile SDRAM CAS (active Low) |
| ncs_sdram | Output | Mobile SDRAM chip select (active Low) |

# Utilization Details

This design can be implemented in Microsemi AGL250 or A3P250 devices. However, for testing purposes, this design was verified using Microsemi's AGL600V2-484 FBGA IGLOO device. Table 2 gives the utilization details for AGL600V2-484 FBGA, which include mobile SDRAM interface, Core8051, and glue logic.

***Table 2 •*** **Utilization Details**

| Resource | Used | Total | Percentage |
|---|---|---|---|
| Core | 5,972 | 13,824 | 43.20% |
| I/O (with clocks) | 42 | 235 | 17.87% |
| Differential I/Os | 0 | 60 | 0.00% |
| Global (chip+quadrant) | 4 | 18 | 22.22% |
| PLL | 1 | 1 | 100.005 |
| RAM/FIFO | 11 | 24 | 45.83% |
| Low static $I_{CC}$ | 0 | 1 | 0.00% |
| FlashROM | 0 | 1 | 0.00% |
| User JTAG | 1 | 1 | 100.00% |

# Testing Scheme

For testing purposes, the mobile SDRAM interface design example is instantiated in a top level-file, as shown in Figure 2. The hardware verification of this design is done on Microsemi's IGLOO Development Kit, along with a customized daughter board, using Micron Mobile SDRAM (MT48LC8M16LFB4-75M). Mobile SDRAM was verified for erase, read, and write operation using the FS2 debugger and the software executable.

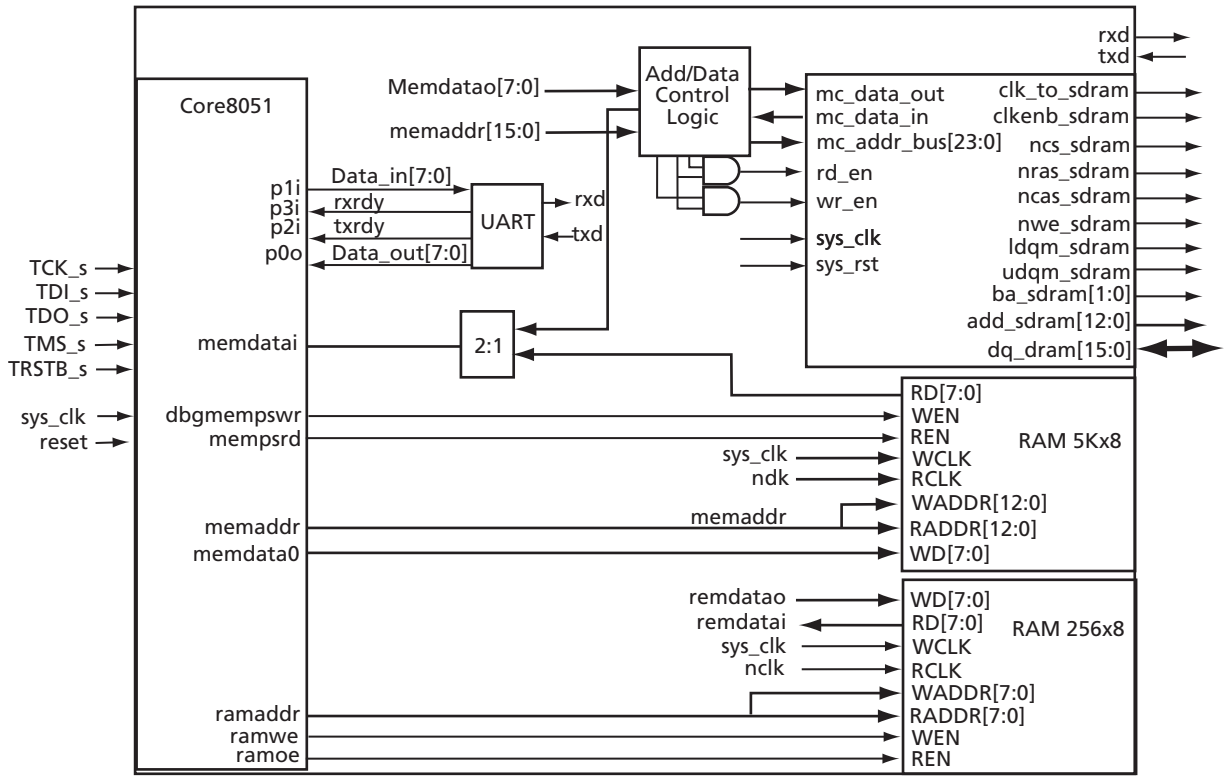


***Figure 2* • Test Setup for Mobile SDRAM Interface**

# Software Interface and Design details

The software modules consist of two applications that test and verify the working of the mobile SDRAM interface.

1. Application software: This software is written in C language under the Windows® platform and runs on PC, which is interfaced to the Microsemi board through a USB port. The program uses a command prompt user interface to perform write and read operations on the mobile SDRAM and verify that the read data is correct.
2. Firmware software: This software is written in C language and must be initially downloaded to the Program Memory of Core8051. This program can be used for Write and Read operations with the Mobile SDRAM memory.

The offset addresses corresponding to various registers are hardcoded in the code. Read and write operations are performed on a Bank basis. Each bank can be individually read or written sequentially. Each Bank consists of four MBytes. Before performing any operation, status signal Ready/Busy is verified. Table 3 shows the register mapping.

***Table 3 •* Register Mapping**

| Address | Register Name | R/W | Description |
|---|---|---|---|
| 0x0000 | BASEADDRESS | | Base address of the design example |

The set of software files used for testing the IP is provided in the software folder.

# Software Files

## Firmware Files

### Main.c

This file contains source code for the SDRAM drivers. The code handles the command for UART communication and performs the read and write operations on the RAM device.

## Application Files

### Main.c

This file provides the main functionality of the program. The user interaction, data validation, and communication with the USB port are done inside this file. User input is validated and sent to the USB port sequentially.

### UsbCom.c

This source file takes care of the USB communication.

## Program Execution (SDRam.exe)

The executable program is run from a Windows environment. When the program is run, the menu options are displayed. The option for communication port is selected based on the USB port where the board is connected. The option for Read, Write, or verify is displayed following that. Make sure you program the device and load the hex file before running the executable.

When the application program SDRam.exe is executed (SDRam.exe is provided with the design example), the screen shown in Figure 3 appears on the laptop or PC monitor.

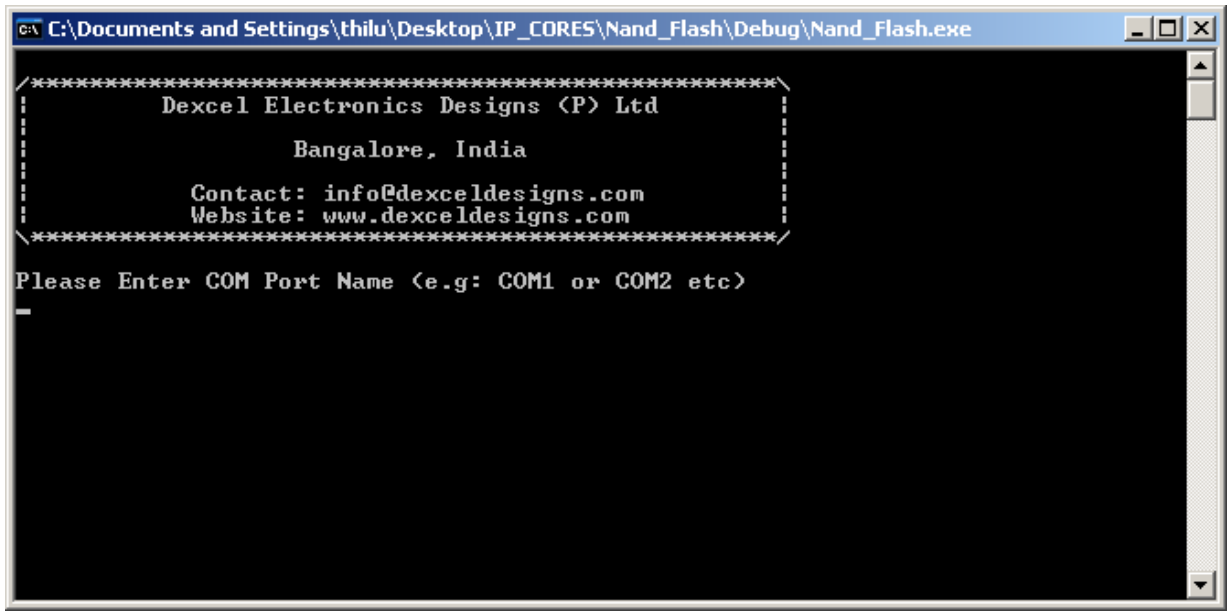


***Figure 3* • COM Port Identification**

Enter the COM port name. You can find the COM port in Device Manager (in this example it is COM3).

After you specify the COM port, the screen shown in Figure 4 appears.

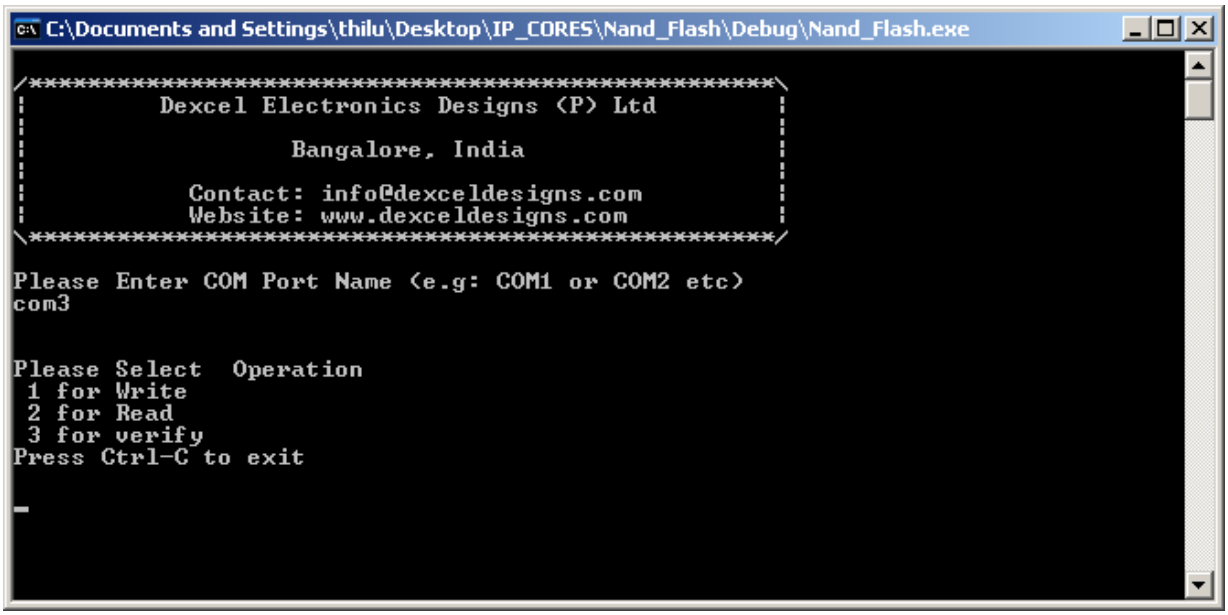


***Figure 4* • Selecting an Operation**

The options are 1 for Write, 2 for Read, and 3 for verify.

For testing each bank, follow the sequence of Write, Read, verify.

First select 1 for a Write operation. The screen shown in Figure 5 appears.

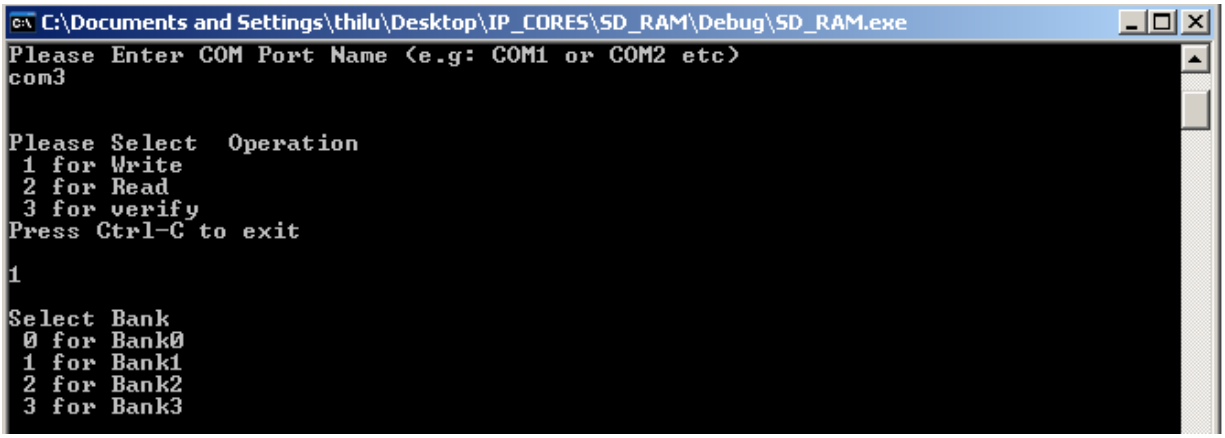


***Figure 5* • Selection of Bank**

Select each bank, 0 to 3, one at a time, to perform the test on that particular bank.

When you select a bank, the screen shown in Figure 6 appears.

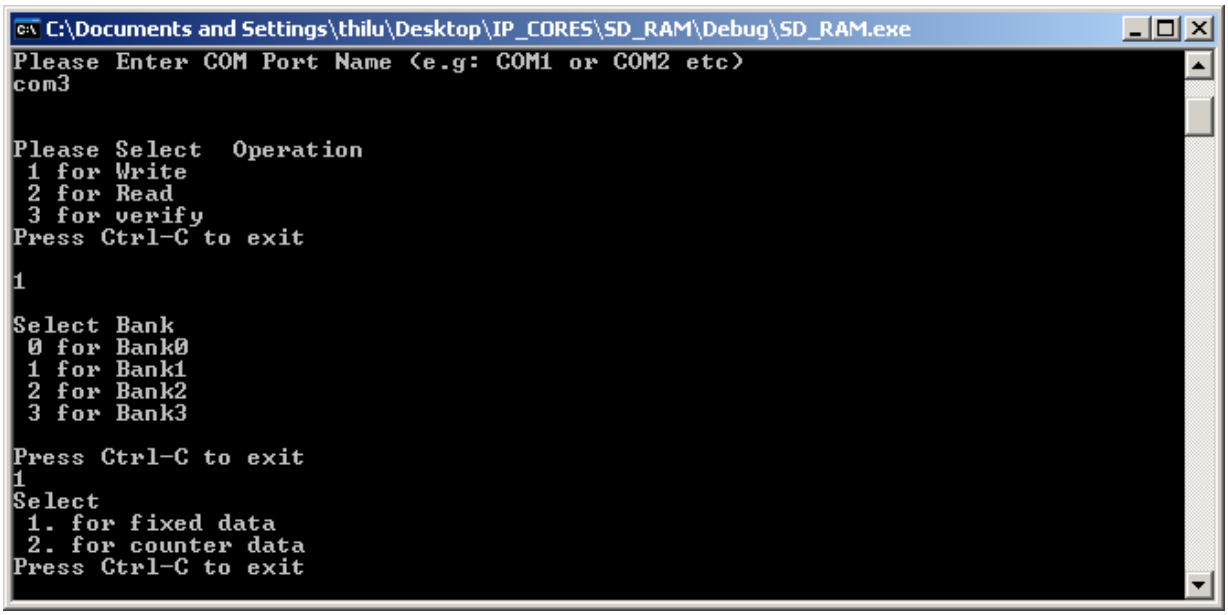


***Figure 6* • Selection of Data Type**

Select 1 for filling a SDRAM location with fixed data. Select 2 for selecting an incremental counter value from 0x0000 to 0xFFFF.

If option 1 is selected, the screen shown in Figure 7 appears.

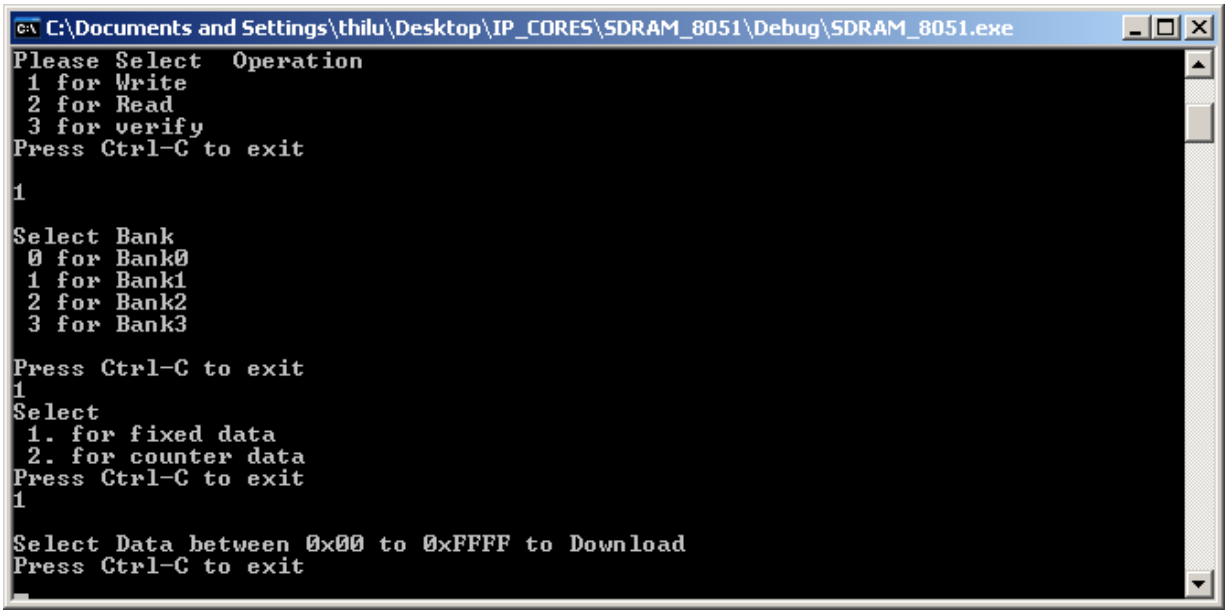


***Figure 7* • Selection of Data**

Select a data value from 0x00 to 0xFFFF.

During the write operation, a status message appears on the screen: “Writing Data please wait.” (Figure 8). The writing process takes a few minutes to complete. A temporary file, WRITE_DATA.txt, is created for the type of data selected. This file is used later during the verification process.

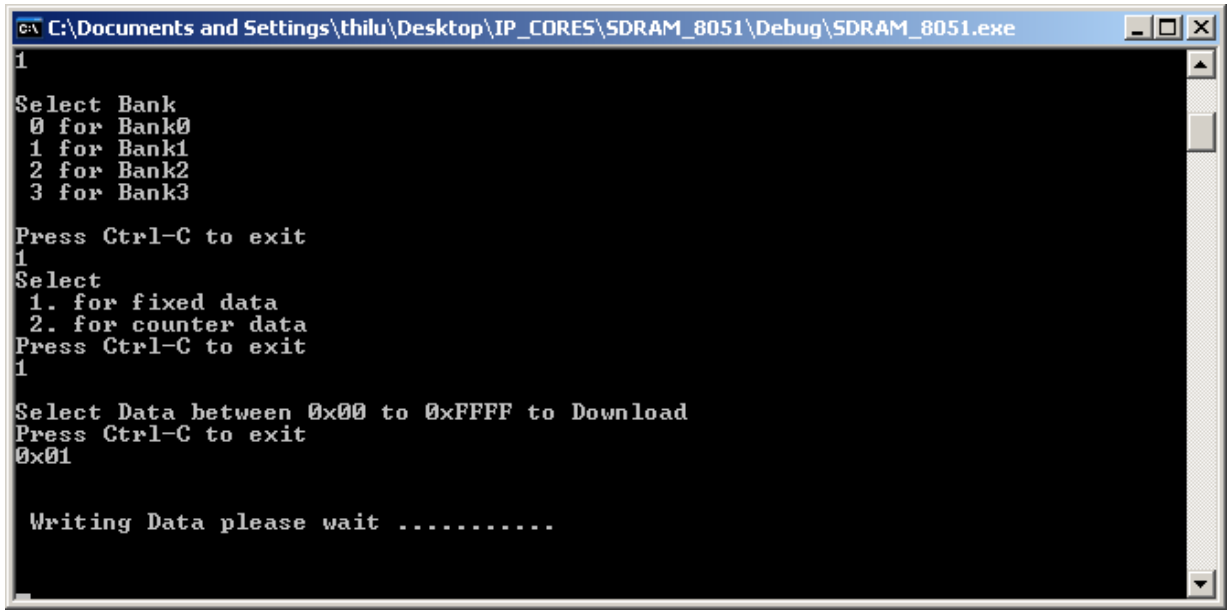


***Figure 8* • Status Message while Writing**

After completion of the write operation, a window indicating, “WRITE COMPLETED” appears, as shown in Figure 9.

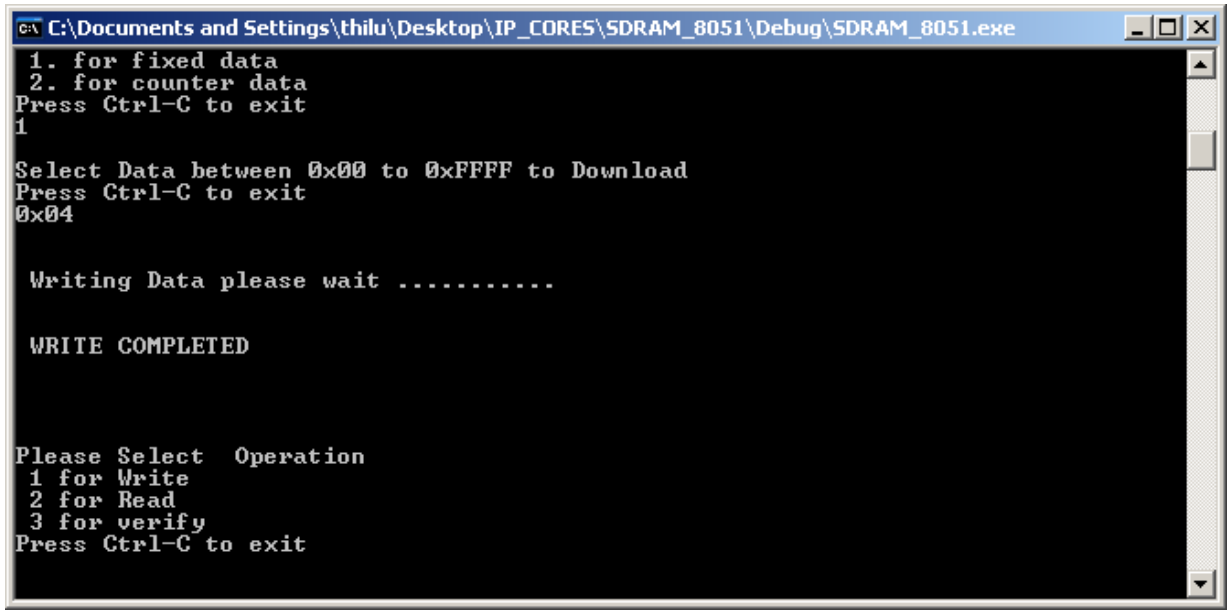


***Figure 9* • Status Message for Write Completion**

After the write operation finishes, the selection menu appears again and you can select option 2 for a read operation.

When Read is selected, the screen shown in Figure 10 appears.

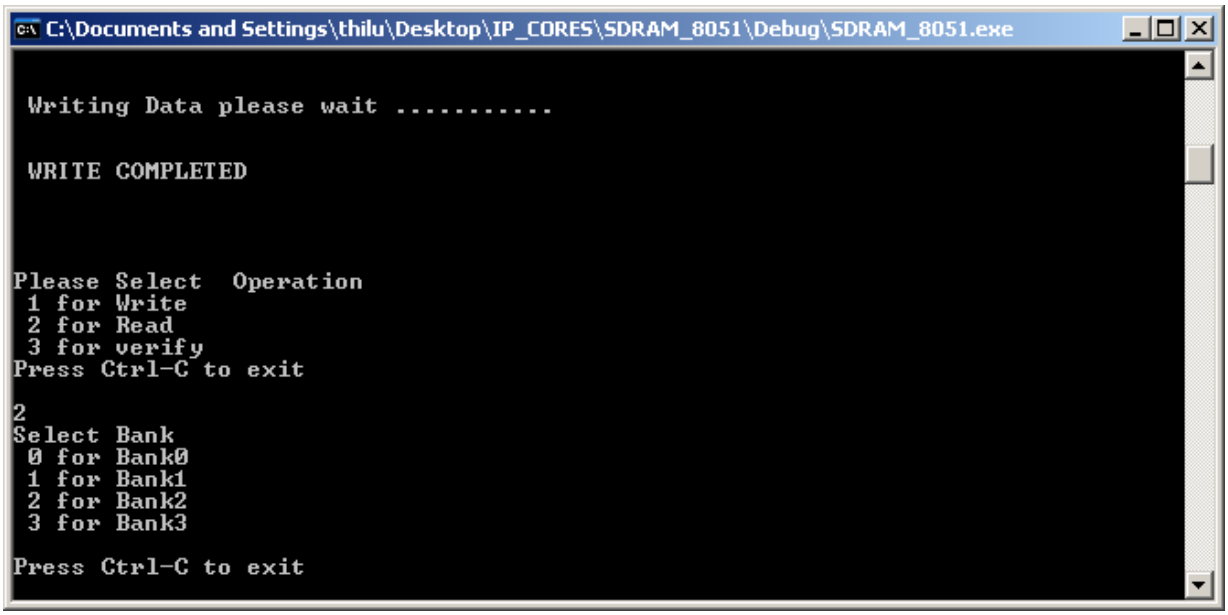


***Figure 10* • Read Operation Selection**

Select the bank number to read.

After the bank is selected, a status message, “Reading Data please wait,” appears on the screen (Figure 11). The reading process may take a few minutes more than write, since the read data must be transferred using a slower UART interface. The read data is stored in a temporary file, READ_DATA.txt.

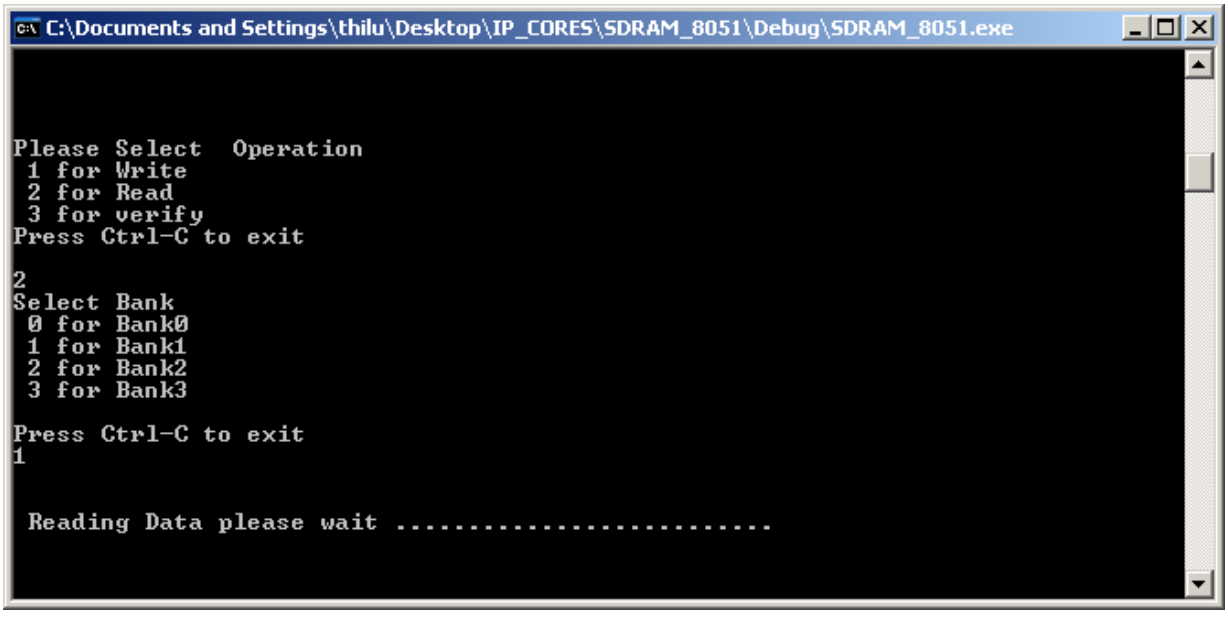


***Figure 11* • Status Message for Read Operation**

After completion of the read operation, the screen shown in Figure 12 appears.

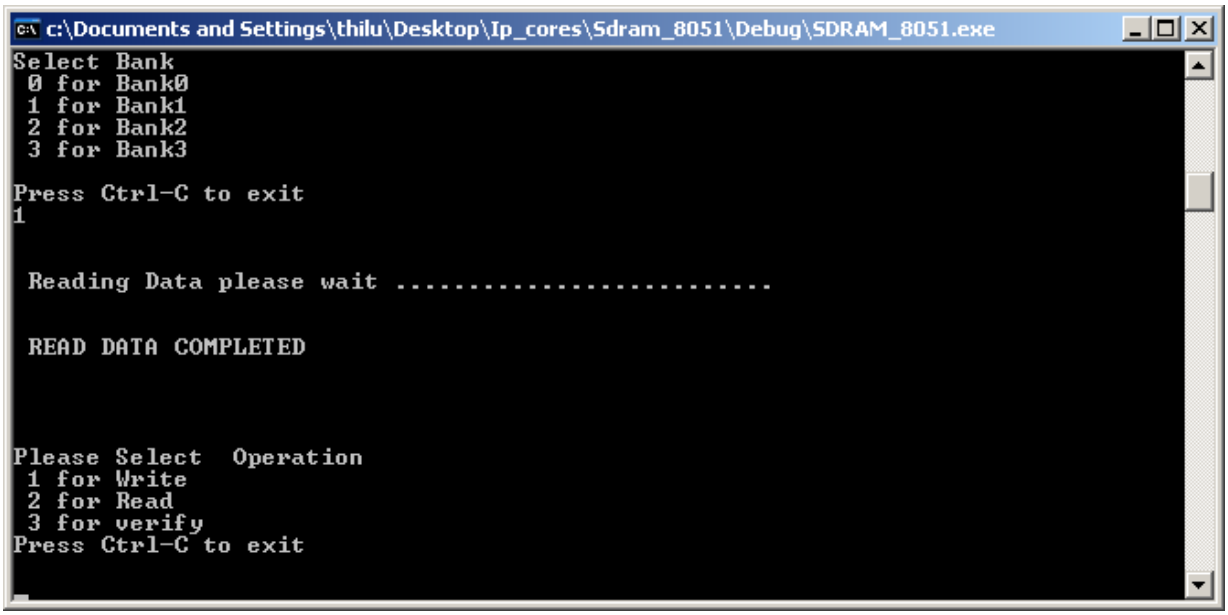


***Figure 12* • Status Message after Read Operation**

To verify whether the data written and read were correct, select 3 for verification. The Verify command compares an immediately read value with the last written data. A message appears (Figure 13), detailing the results of the verification process

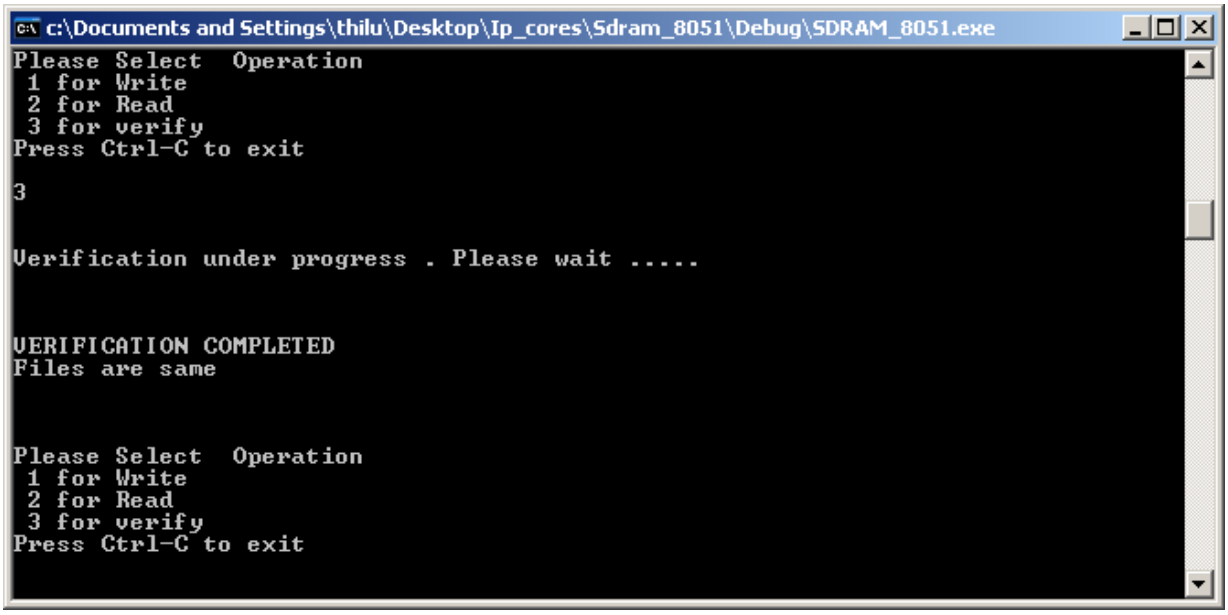


***Figure 13* • Verification Status Message**

Note: During a write operation, a text file named Write_Data.txt is created in the path from which SDRAM.exe was executed. Similarly, during a read operation, Read_Data.txt is created. These files are overwritten by a new operation. During verification, these two files are compared and the result is displayed. In this process, the immediately read data content and the last written data content are compared. Hence for valid testing, a write and

read must be performed for the same bank; otherwise the result obtained would not be correct, because the data written to one bank and read from another bank might not be the same.

# Timing Diagram

The simulation waveforms are shown in Figure 14 and Figure 15 on page 11.

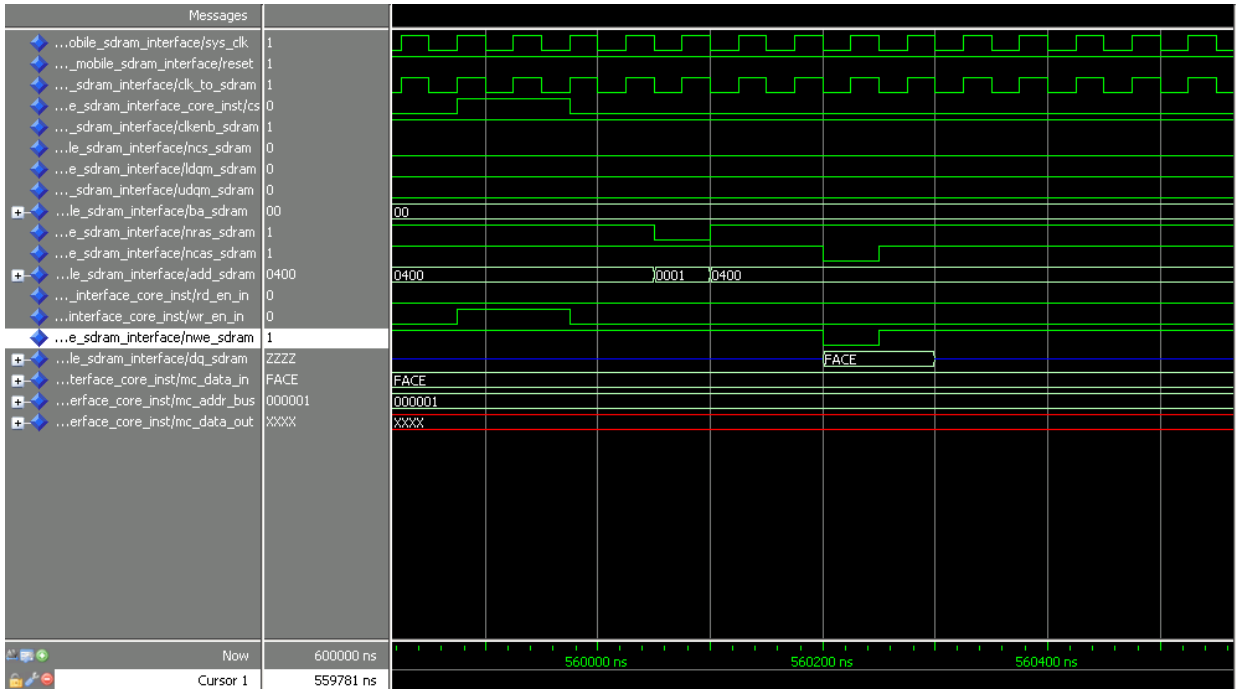


***Figure 14* • Write Cycle**

Figure 14 gives the Write Cycle on mobile SDRAM. It shows the data being written is 0xFACE on location 0x0001 of the mobile SDRAM.

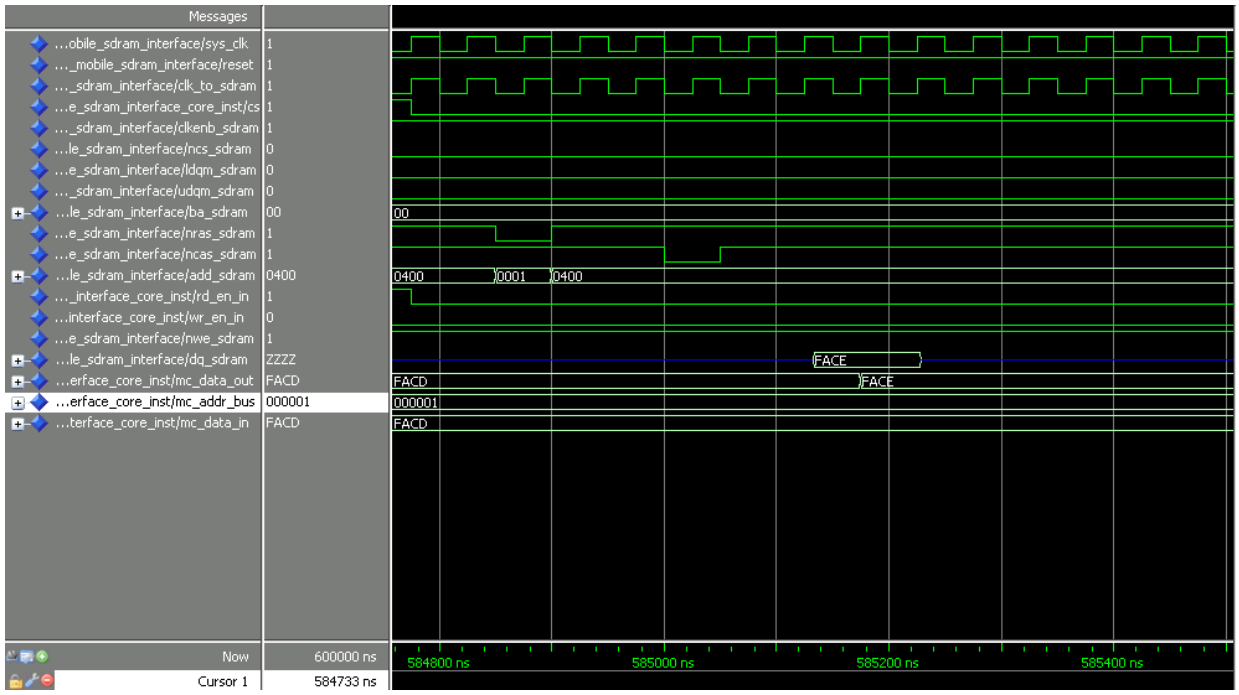


***Figure 15* • Read Cycle**

# Application Area

Mobile SDRAM is used in almost all compact handhelds, due largely to its compact size and low-power architecture. While the majority of the volume applications are found in the consumer market segment, the portable market for industrial and military applications is growing at a rapid pace.

# Conclusion

Mobile SDRAMs provide many benefits for low-power applications, such as mobile phones, handheld media players, cameras, book readers, and other portable applications. Combining Mobile SDRAM with Microsemi's low-power FPGAs provides a high-value solution for power- and cost-sensitive mobile applications.

# List of Changes

The following table lists critical changes that were made in each revision of the document.

| Revision | Changes | Page |
|---|---|---|
| Revision 1<br>(March 2015) | Non-Technical Updates. | N/A |
| Revision 0<br>(June 2009) | Initial Release. | N/A |

**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; 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 3,400 employees globally. Learn more at **www.microsemi.com**.



51900197-1/03.15