
NAND Flash Interface Design Example

Table of Contents

Overview	1
Design Description	1
Interface Description	3
Utilization Details	4
Software Interface and Design Details	4
Testing Scheme	12
Application Area	16
Conclusion	16
List of Changes	17

Overview

NAND flash memories are known for their relatively simple structure, low cost, and high capacity. NAND flash memories are used extensively in solid state drives (SSD) and portable consumer products, such as smart phones, GPS, digital cameras, and MP3 players. NAND flash is economical and dense compared to the alternative NOR flash. This document details the design example of a NAND flash memory interface.

Associated files for this design example can be downloaded from the Microsemi website:

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

Design Description

This design can be easily implemented in Microsemi IGLOO® or ProASIC®3 low-power FPGAs that contain 5600 tiles, such as AGL250 or A3P250 devices. However, this design was verified with the Microsemi M1AGL600V2-484FBGA IGLOO device and interfaces with an Microsemi Core8051 and a Micron® MT29F2G08AADWP NAND flash device. The NAND flash interface is universal and supports similar devices.

NAND flash devices have a multiplexed bus for data, address, and instructions and support page access rather than the random access used by NOR flash. The MT29F2G08AADWP NAND flash device contains 2,048 blocks; each block is subdivided into 64 programmable pages; each page consists of 2,112 bytes; and each page is divided into a 2,048-byte data storage region with a separate 64-byte area, typically used for error management functions.

Figure 1 shows the top-level block diagram of the NAND flash interface.

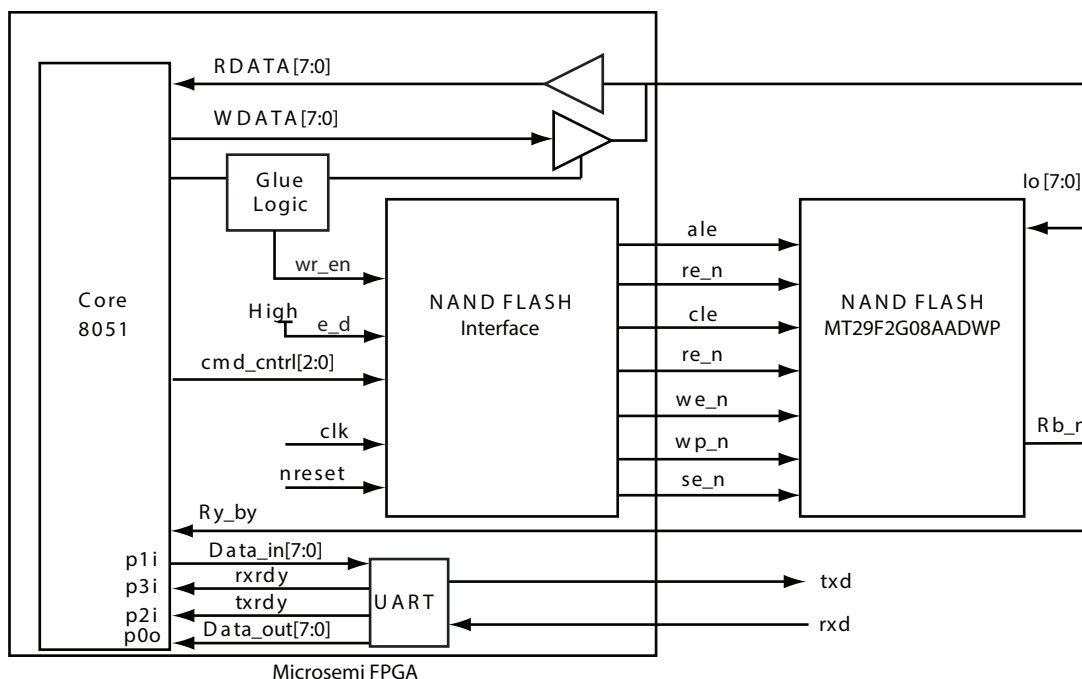


Figure 1 • Top-Level Block Diagram

The data bus of the NAND flash is directly connected to the microcontroller data bus. This 8-bit bus acts as a multiplexed bus to transfer data, address, and instructions.

The 3 least significant bits (LSB) of the address bus and the other control signals of the controller are used to generate the control signals used to decode the needed commands. The NAND flash interface block decodes the commands and generates the control signals for the bus cycles required for the flash read, write, and erase operations. The bus cycle decoding is performed as described in [Table 1](#).

Table 1 • Command Decoding

CMD_CNTRL [2:0]	E_D	Operation Performed
000	1	Idle condition
001	1	Read data/status/device, depending on the command sent on the data lines
010	1	Write data/command/address, depending on the command sent on the data lines
011	1 0	ALE is asserted (High). ALE is disabled (Low).
100	1 0	SE is asserted (Low). SE is disabled (High).
101	1 0	WP is asserted (Low). WP is disabled (High).
110	1 0	CE is asserted (Low). CE is disabled (High).
111	X	Command latch enabled (CLE = 1)

This design considers the lowest 3 bits of the address bus as CMD_CNTRL, so the valid addresses are from 0x00h to 0x07h. These lines and the enable/disable line generate the required five control signals for the flash device: CLE, ALE, CE_n, RE_n, and WE_n.

Two additional pins control hardware write protection (WP_N) and monitor device status (RB_N). Write protect is controlled using the software, whereas RB_n is directly connected to one of the general purpose input/output (GPIO) pins of Core8051.

Programming and reading of the flash device occurs on a page basis, whereas the erase takes place on a block basis. Prior to programming the flash device with fresh data, erase the flash device and check the ready/busy (RB_N) status of the device before giving any new command to the flash device. The new command on the I/O bus will be latched into the command register and executed only when the device is in ready state.

Associated files for this design example include firmware that can be used to perform erase, write, and read operations with the flash memory. If you are using a different flash device, RTL code remains the same, but the firmware must be altered (refer to appropriate device datasheet for more information about the parameter changes).

Interface Description

Table 2 provides interface details for the IP.

Table 2 • Interface Description

Port	Direction	Description
NRESET	Input	Active Low reset signal aborts current operation and resets all the control signals.
WR_EN	Input	Write enable signal generated by the host for the interface.
E_D	Input	Enable/disable signal generated by the host.
CMD_CNTRL [3:0]	Input	Control signal generated by the host to determine the functions that the interface will perform (for decoding information refer to Table 1 on page 2).
CLK	Input	48 MHz input clock to the NAND flash interface
RY_BY	Input	Active Low output from the NAND flash device, indicating the status of the device: RB_BY = 0: Device is still busy performing an operation. RB_BY = 1: Device is ready to accept the next command.
IO_BUS [7:0]	Inout	Bidirectional 8-bit multiplexed bus used to send data/command/address to their respective registers in the NAND flash device. The data read from the NAND flash device is also available on these lines.
ALE	Output	Address latch enable: Controls the writing to the address register
CE_N	Output	Active Low chip enable: Controls the active and standby modes of the device.
CLE	Output	Command latch enable: Controls the writing to the command register.
RE_N	Output	Active Low read enable: Controls the data and status output on the I/O lines.
WE_N	Output	Active Low write enable: Controls the data and command on the I/O lines during a write sequence.
WP_N	Output	Active Low write protect: Provides protection when programming or erasing the device
SE_N	Output	Active Low spare area enable: Required only for AMD® devices. se = 0: 16 bytes of spare area on each page are enabled. se = 1: Spare area is disabled.

Utilization Details

This design was verified using an Microsemi AGL600V2-484 FBGA IGLOO device, but can easily be instantiated in other IGLOO and ProASIC3 devices that contain the minimum required resources. The utilization details for the AGL600V2-484 FBGA device described in [Table 3](#) include the NAND flash interface, Core8051, and other glue logic.

Table 3 • Resource Utilization

Resource	Utilization	Total	Percentage
Core	5,607	13,824	40.56%
I/O (with clocks)	20	235	8.51%
Differential I/O	0	60	0.00%
GLOBAL (chip + quadrant)	3	18	16.67%
PLL	0	1	0.00%
RAM/FIFO	17	24	70.83%
Low Static I _{CC}	0	1	0.00%
FlashROM	0	1	0.00%
User JTAG	1	1	100%

Software Interface and Design Details

The software modules consist of two applications to test and verify the working of the NAND flash interface:

- Application software: This software is written in C language for a Windows® platform and must run from a PC connected to the Microsemi board via a USB port. The program is used to send data to the NAND flash, read back the data from the NAND flash, and verify the data is correct.
- Firmware software: The software is written in C language and can be used for erase, write, and read operation with the flash memory. The program must be initially downloaded to the program memory of Core8051.

The offset address corresponding to the different registers is hardcoded. Prior to performing a write operation to a block, that particular block must be erased. Erasing can be performed on a block-to-block basis. If any blocks are corrupted during erase operation, the corresponding blocks cannot perform read or write operations. This provision is based on the error management feature of the device, as mentioned in the appropriate device datasheet.

Read or write operations can be performed on a page-to-page basis. Each page can be individually read or written sequentially for all 2,112 bytes. Before performing any operation, the status signal Ready/Busy is verified. For testing from the user interface, read or write operations are performed based on the block numbers. When using a different device from the MT29F2G08, obtain data from the datasheet and modify the software program if required.

Firmware Files

Nandflash.c

This file contains source code for the NAND I/O drivers, which represent the standard command set for Micron NAND flash devices. The NAND I/O drivers handle the commands such as page read, page program, and block erase for a Micron NAND device. The source code also contains UART communication operations.

Nandflash.h

This is the NAND low-level driver header file.

SDK MACRO USED

Erase Block - `int NAND_EraseBlock(unsigned long a_uiBlockNum);`

Parameters:

`a_uiBlockNum`: Block number to be erased.

Description:

This function erases (returns all bytes in the block to 0xFF) a block of data in the NAND device

Return code:

`NAND_IO_RC_PASS = 0`: This function completes its operation successfully

`NAND_IO_RC_FAIL = 1`: This function does not complete its operation successfully

`NAND_IO_RC_TIMEOUT=2`: The function times out before operation completes.

Program page - `int NAND_ProgramPage(unsigned long a_uiPageNum,unsigned short a_usColNum, unsigned short a_usReadSizeByte,unsigned char *a_pucReadBuf);`

Parameters:

`a_uiPageNum`: Programming will occur at this page address.

`a_usColNum`: Programming will begin at this column address.

`a_usReadSizeByte`: Number of bytes to program.

`a_pucReadBuf`: Data buffer with data which will be programmed to flash.

Description:

The `NAND_ProgramPage` function is used program a page of data into the NAND device.

Return code:

`NAND_IO_RC_PASS = 0`: The function completes operation successfully.

`NAND_IO_RC_FAIL = 1`: The function does not complete operation successfully.

`NAND_IO_RC_TIMEOUT =2`: The function times out before operation completes.

Read Page - `int NAND_ReadPage(unsigned long a_uiPageNum,unsigned short a_usColNum,unsigned short a_usReadSizeByte,unsigned char *a_pucReadBuf);`

Parameters:

`a_uiPageNum`: Page number for reading

`a_usColNum`: Column number for reading

`a_usReadSizeByte`: Number of byte to read

`a_pucReadBuf`: Read data buffer

Description:

This function reads data from the input page `a_uiPageNum` and column `a_usColNum` number into the buffer `a_pucReadBuf` for `a_usReadSizeByte` number of bytes.

Return code:

`NAND_IO_RC_PASS =0`: The function completes operation successfully.

`NAND_IO_RC_FAIL =1`: The function does not complete operation successfully.

`NAND_IO_RC_TIMEOUT =2`: The function times out before operation completes.

NAND Read Status - `int NAND_ReadStatus(void);`

Parameters:

None.

Description:

This function Reads the status register of the NAND device by issuing a 0x70 command.

Return code:

NAND_IO_RC_PASS = 0: This function completes its operation successfully
 NAND_IO_RC_FAIL = 1: This function does not complete its operation successfully
 NAND_IO_RC_TIMEOUT = 2: The function times out before operation completes.

Table 4 describes the register mapping operations.

Table 4 • Register Mapping

Address	Register Name	R/W	Description
0x0001	RD_OFFSET	R	Data to be read
0x0002	WR_OFFSET	W	Data to be written
0x0003	ALE_OFFSET	W	Address to be written
0x0006	CE_OFFSET	W	Chip enable register
0x0007	CLE_OFFSET	W	Commands to be transmitted
0x0100	GPIO_STATUS	R	Read the Ready/Busy Signal

Application Files

Nandflash.c

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

UsbCom.c

This file controls the USB communication. The set of software files used for testing the IP are provided in the software folder.

Program Execution (Nand_Flash.exe)

This executable runs from a Windows environment. The following menu options are available:

- Communication port: Select the USB port connected to the board.
- Operation: Select the operation (read, write, or verify).

Note: You must program the device and load the hex file before running the executable.

Once Nand_Flash.exe is executed, the screen shown in Figure 2 appears.

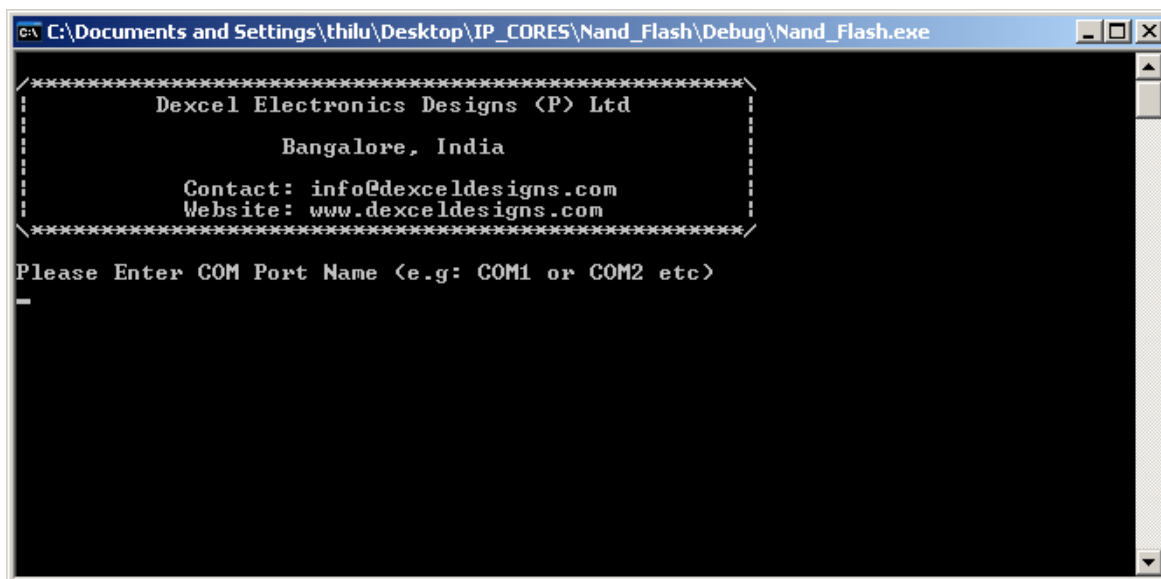


Figure 2 • Identifying a COM Port

Enter the COM port name (available from the Device Manager), the screen shown in Figure 3 appears.

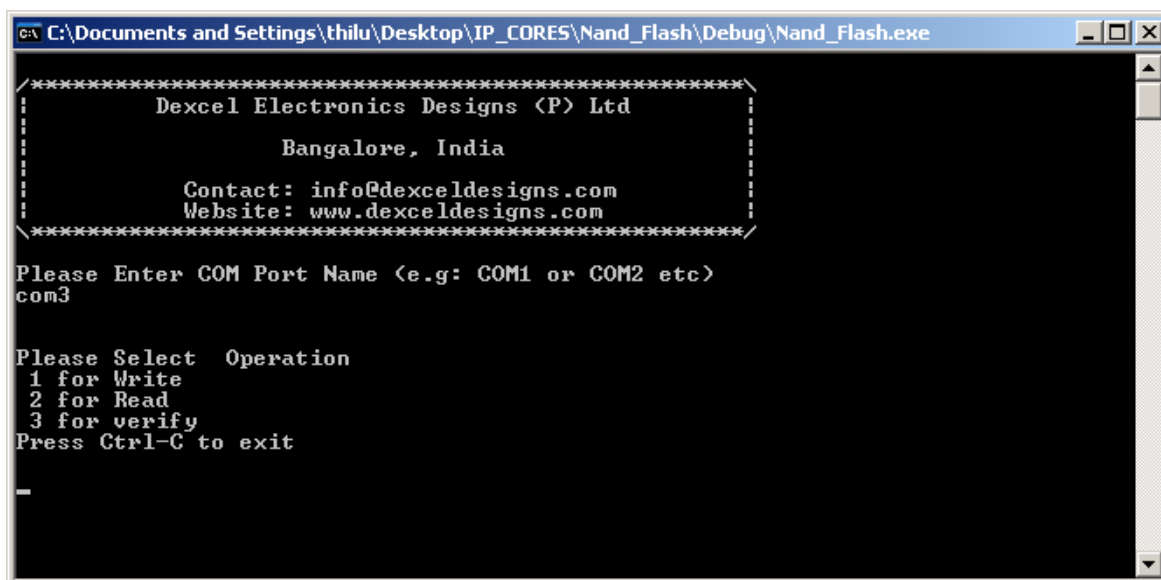


Figure 3 • Selecting an Operation

Select 1 for write operation (2 for read or 3 for verify). To test each bank, follow this sequence: write, read, and verify. The following screen shown in Figure 4 appears.

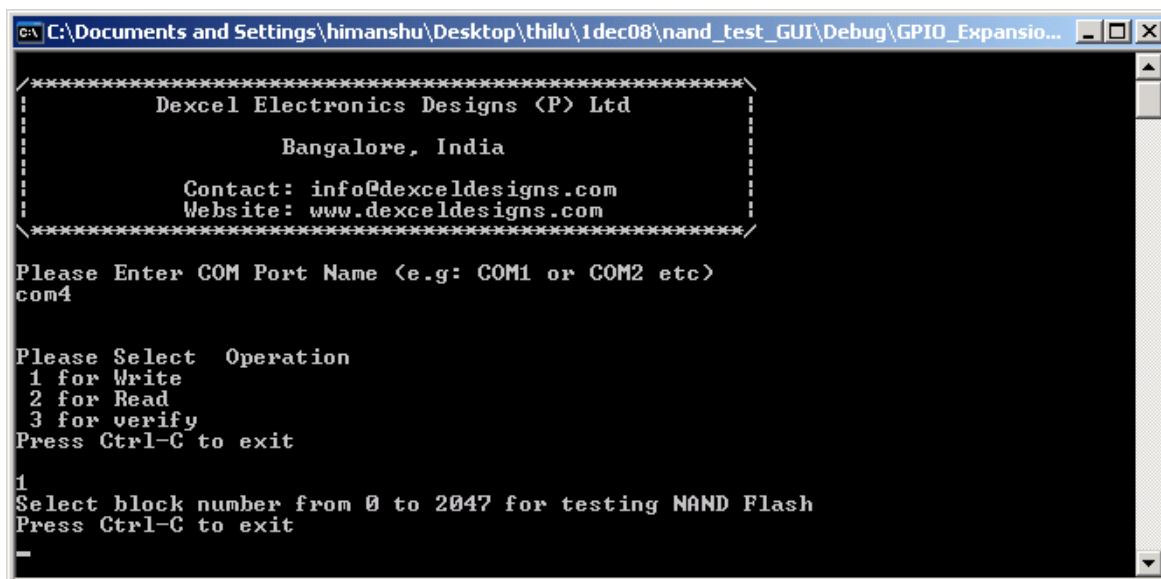


Figure 4 • Selecting a Block Number

Select a block number between 0 to 2,047. The corresponding block will be erased (Figure 5).

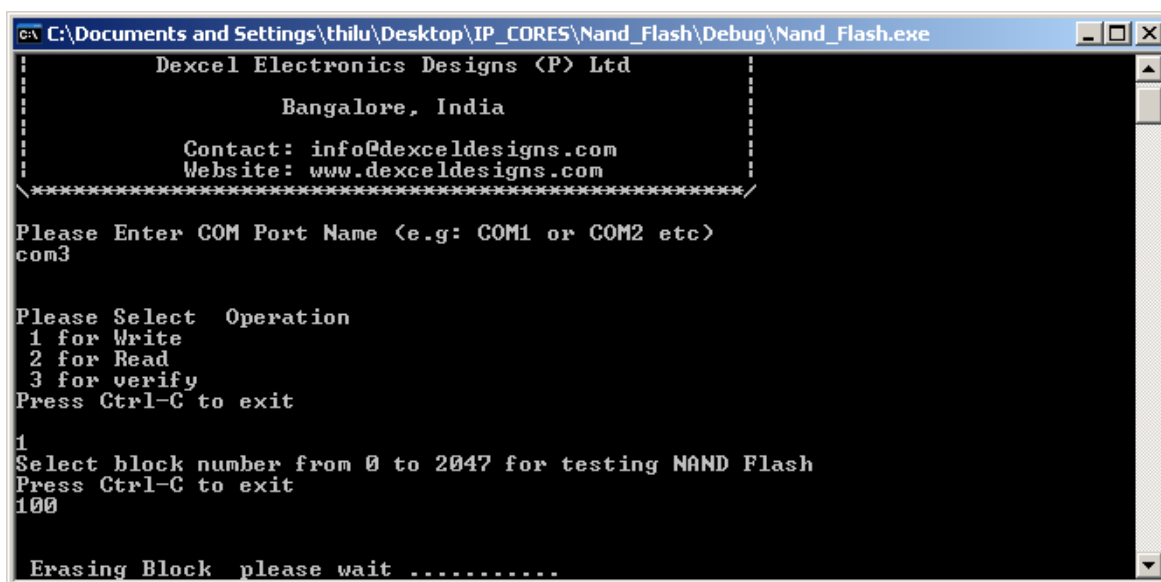
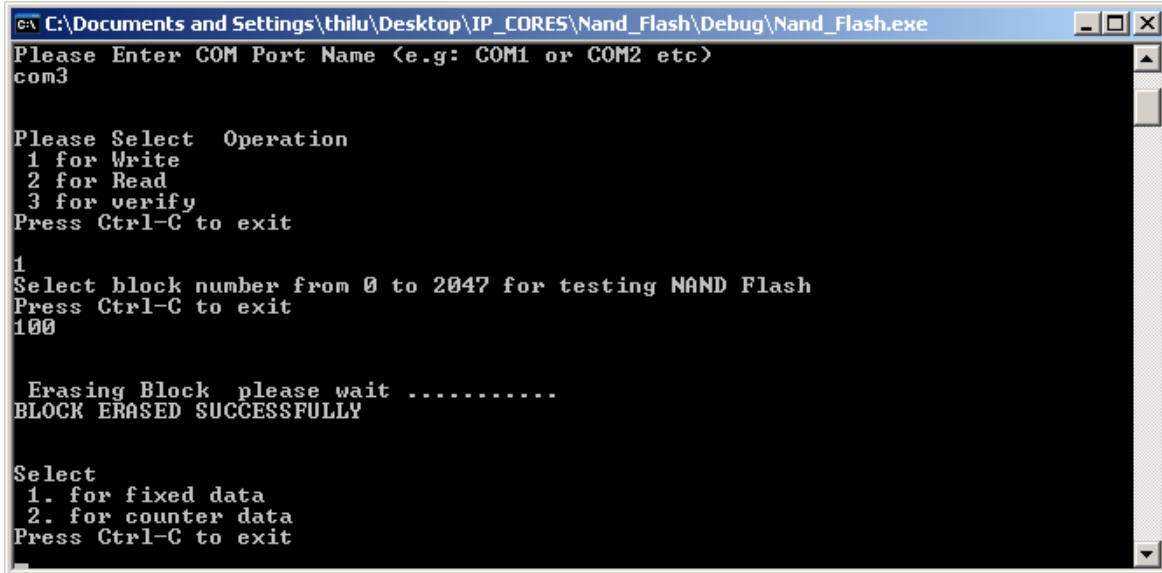


Figure 5 • Block Erasing Message

The result is displayed in [Figure 6](#).



```
C:\Documents and Settings\thilu\Desktop\IP_CORES\Nand_Flash\Debug\Nand_Flash.exe
Please Enter COM Port Name (e.g: COM1 or COM2 etc)
com3

Please Select Operation
1 for Write
2 for Read
3 for verify
Press Ctrl-C to exit

1
Select block number from 0 to 2047 for testing NAND Flash
Press Ctrl-C to exit
100

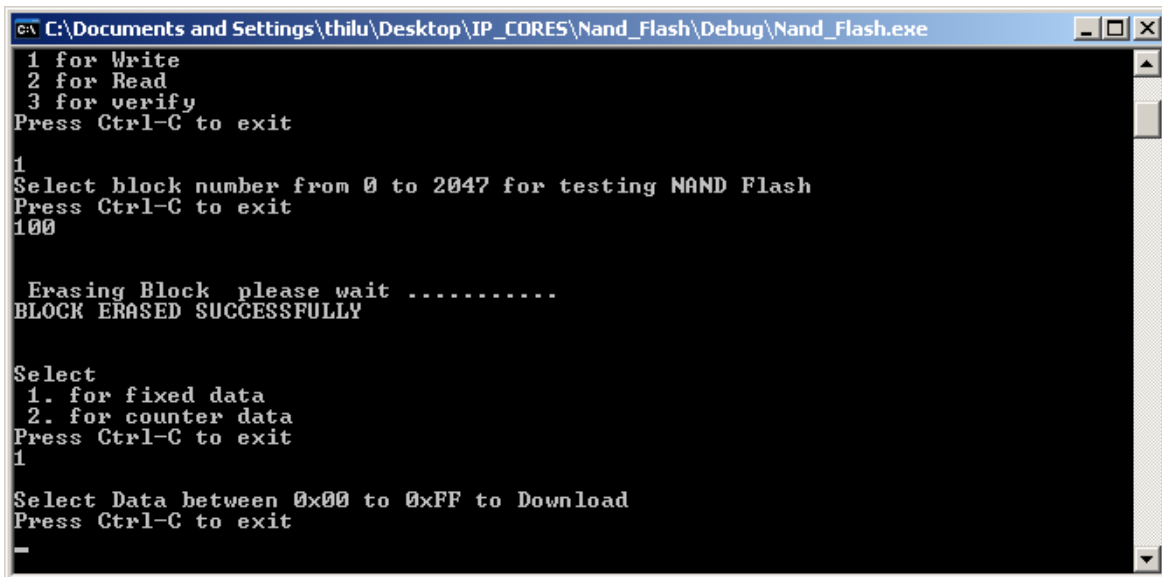
Erasing Block please wait .....
BLOCK ERASED SUCCESSFULLY

Select
1. for fixed data
2. for counter data
Press Ctrl-C to exit
```

Figure 6 • Selecting a Data Type

Note: A write operation can only be performed on the block if the block is erased successfully. Otherwise, a failure message is displayed and a write operation cannot be performed on the block.

Select **1** for fixed data, or **2** for counter data (counter value from 00 to 0xff, incremented sequentially for each page). If option 1 is selected, the screen shown in [Figure 7](#) appears.



```
C:\Documents and Settings\thilu\Desktop\IP_CORES\Nand_Flash\Debug\Nand_Flash.exe
1 for Write
2 for Read
3 for verify
Press Ctrl-C to exit

1
Select block number from 0 to 2047 for testing NAND Flash
Press Ctrl-C to exit
100

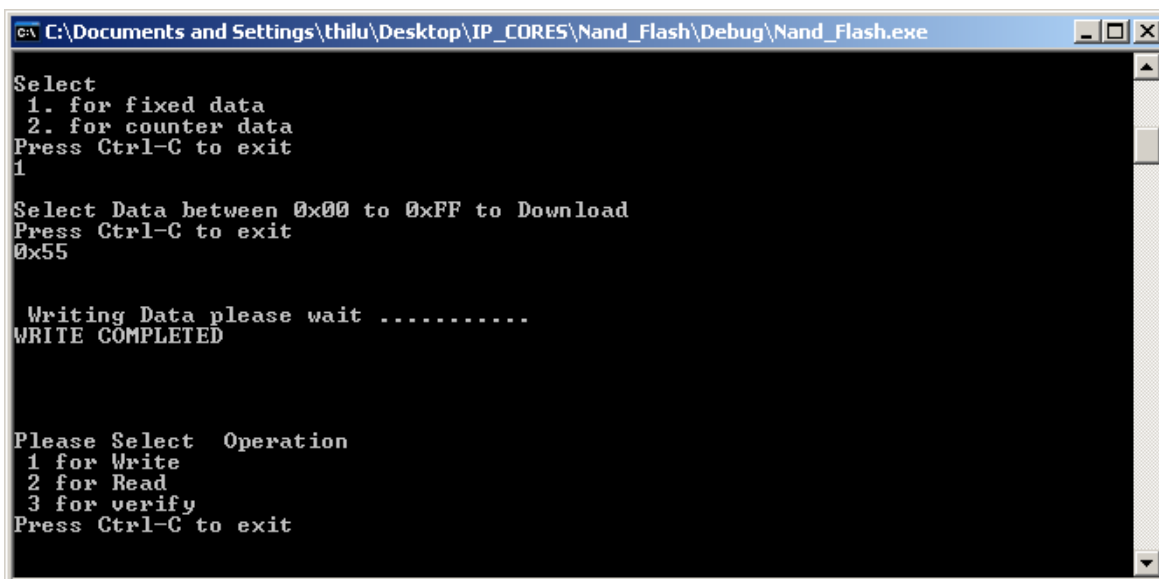
Erasing Block please wait .....
BLOCK ERASED SUCCESSFULLY

Select
1. for fixed data
2. for counter data
Press Ctrl-C to exit
1

Select Data between 0x00 to 0xFF to Download
Press Ctrl-C to exit
-
```

Figure 7 • Selecting Data

Select data between 0x00 to 0xFF. During write operation, a writing data status message appears on the screen (Figure 8).



```
C:\Documents and Settings\thilu\Desktop\IP_CORES\Nand_Flash\Debug\Nand_Flash.exe

Select
  1. for fixed data
  2. for counter data
Press Ctrl-C to exit
1

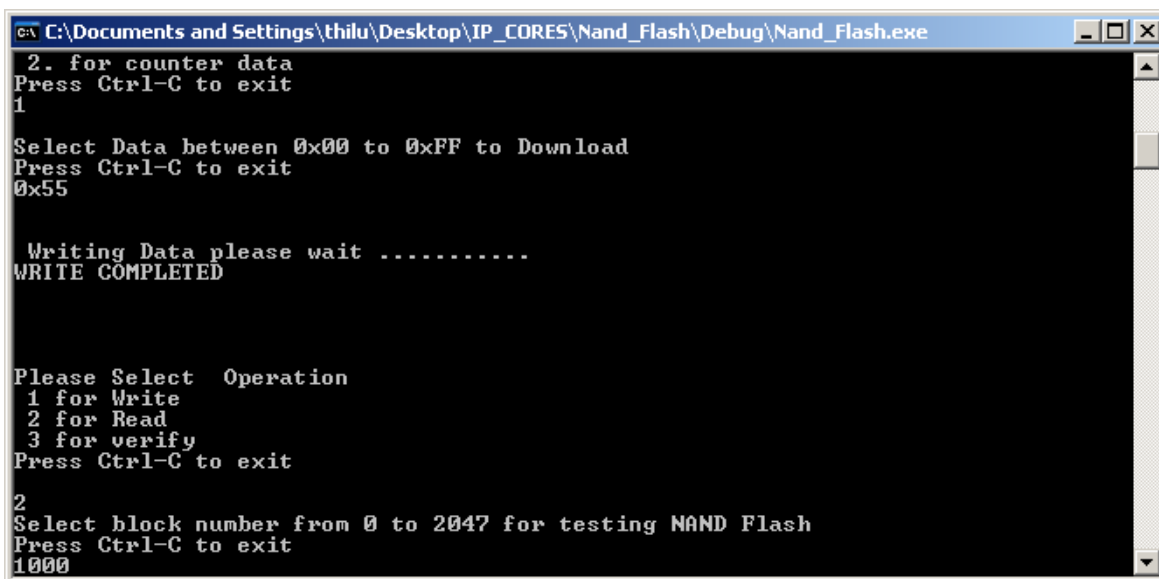
Select Data between 0x00 to 0xFF to Download
Press Ctrl-C to exit
0x55

Writing Data please wait .....
WRITE COMPLETED

Please Select Operation
  1 for Write
  2 for Read
  3 for verify
Press Ctrl-C to exit
```

Figure 8 • Status Message for Write Complete

After completing the write operation, a window appears indicating that writing is complete. Select 2 for read. The screen shown in Figure 9 appears.



```
C:\Documents and Settings\thilu\Desktop\IP_CORES\Nand_Flash\Debug\Nand_Flash.exe

  2. for counter data
Press Ctrl-C to exit
1

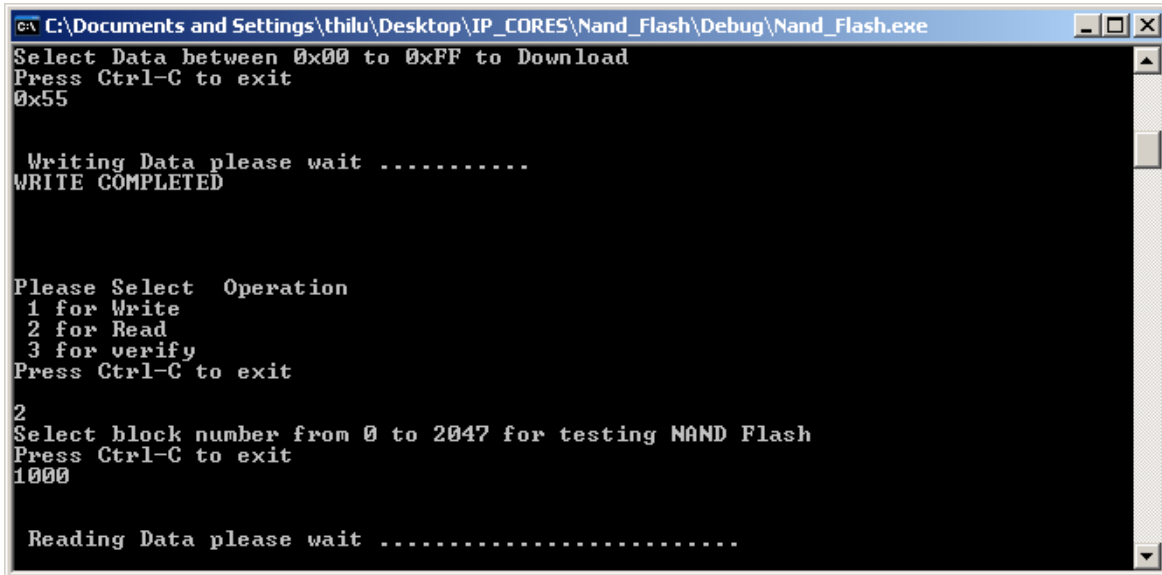
Select Data between 0x00 to 0xFF to Download
Press Ctrl-C to exit
0x55

Writing Data please wait .....
WRITE COMPLETED

Please Select Operation
  1 for Write
  2 for Read
  3 for verify
Press Ctrl-C to exit
2
Select block number from 0 to 2047 for testing NAND Flash
Press Ctrl-C to exit
1000
```

Figure 9 • Selecting a Read Operation Block

Select the required block number to be read. A read data status message appears on the screen (Figure 10).



```
C:\Documents and Settings\thilu\Desktop\IP_CORES\Nand_Flash\Debug\Nand_Flash.exe
Select Data between 0x00 to 0xFF to Download
Press Ctrl-C to exit
0x55

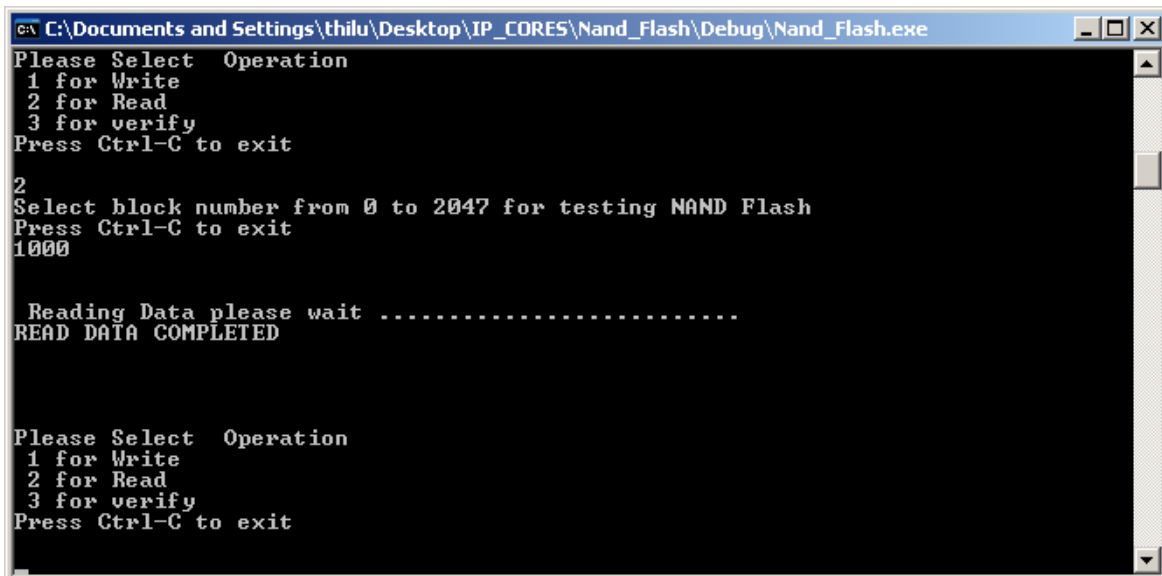
Writing Data please wait .....
WRITE COMPLETED

Please Select Operation
1 for Write
2 for Read
3 for verify
Press Ctrl-C to exit
2
Select block number from 0 to 2047 for testing NAND Flash
Press Ctrl-C to exit
1000

Reading Data please wait .....
```

Figure 10 • Status Message for Read Operation

When the reading operation is complete, a screen appears (Figure 11).



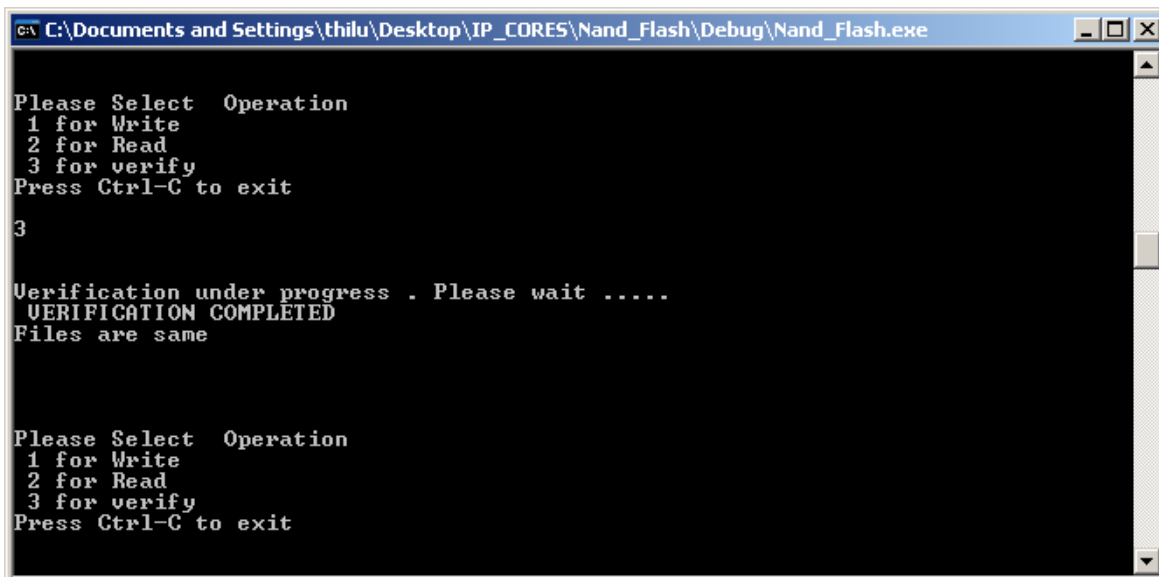
```
C:\Documents and Settings\thilu\Desktop\IP_CORES\Nand_Flash\Debug\Nand_Flash.exe
Please Select Operation
1 for Write
2 for Read
3 for verify
Press Ctrl-C to exit
2
Select block number from 0 to 2047 for testing NAND Flash
Press Ctrl-C to exit
1000

Reading Data please wait .....
READ DATA COMPLETED

Please Select Operation
1 for Write
2 for Read
3 for verify
Press Ctrl-C to exit
```

Figure 11 • Read Data Completed

To verify whether the data written and read are correct, select **3** for verification. A message appears detailing the verification status (Figure 12).



```
C:\Documents and Settings\thilu\Desktop\IP_CORES\Nand_Flash\Debug\Nand_Flash.exe

Please Select Operation
1 for Write
2 for Read
3 for verify
Press Ctrl-C to exit

3

Verification under progress . Please wait .....
VERIFICATION COMPLETED
Files are same

Please Select Operation
1 for Write
2 for Read
3 for verify
Press Ctrl-C to exit
```

Figure 12 • Verification Message

Note: During the write operation a text file (Write_Data.txt) is created in the path from which Nand_Flash.exe was executed. Similarly, during read operation, Read_Data.txt is created. During verification, these two files are compared and the result is displayed. To obtain valid testing results, Microsemi recommends performing a write and read operation for the same block; data written to one block and read from another block will not provide accurate results.

Testing Scheme

Hardware verification of this design is done on Microsemi's IGLOO Development Kit and a customized daughter board with NAND flash memory (MT29F2G08AADWP). NAND flash memory erasing, writing, and verification are performed using an FS2® debugger and software executable.

Simulation of the design example is done using ModelSim®, using a timing model for the NAND flash device. Simulation can be performed for best case and worst case timing delays.

Figure 13 through Figure 16 on page 16 show the simulation waveforms for the various cycles of operation. Figure 13 shows the erase command cycle.

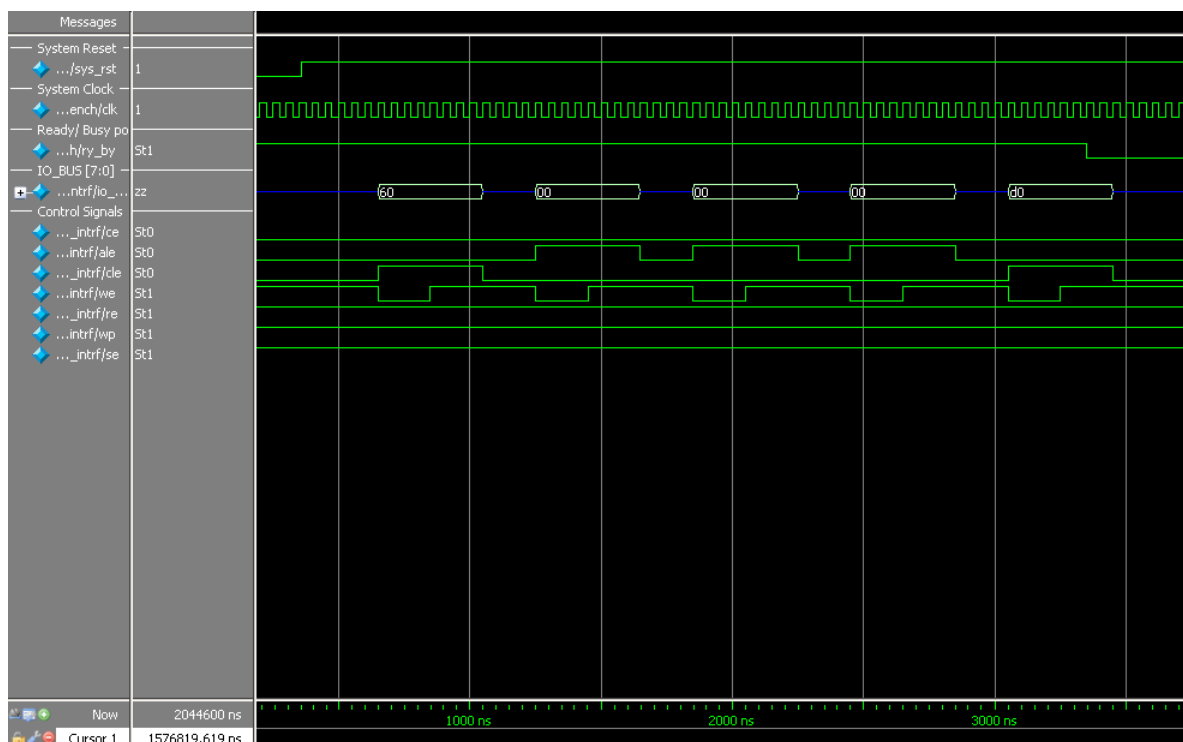


Figure 13 • Erase Command Cycle

Figure 14 shows a write cycle for data 0x55.

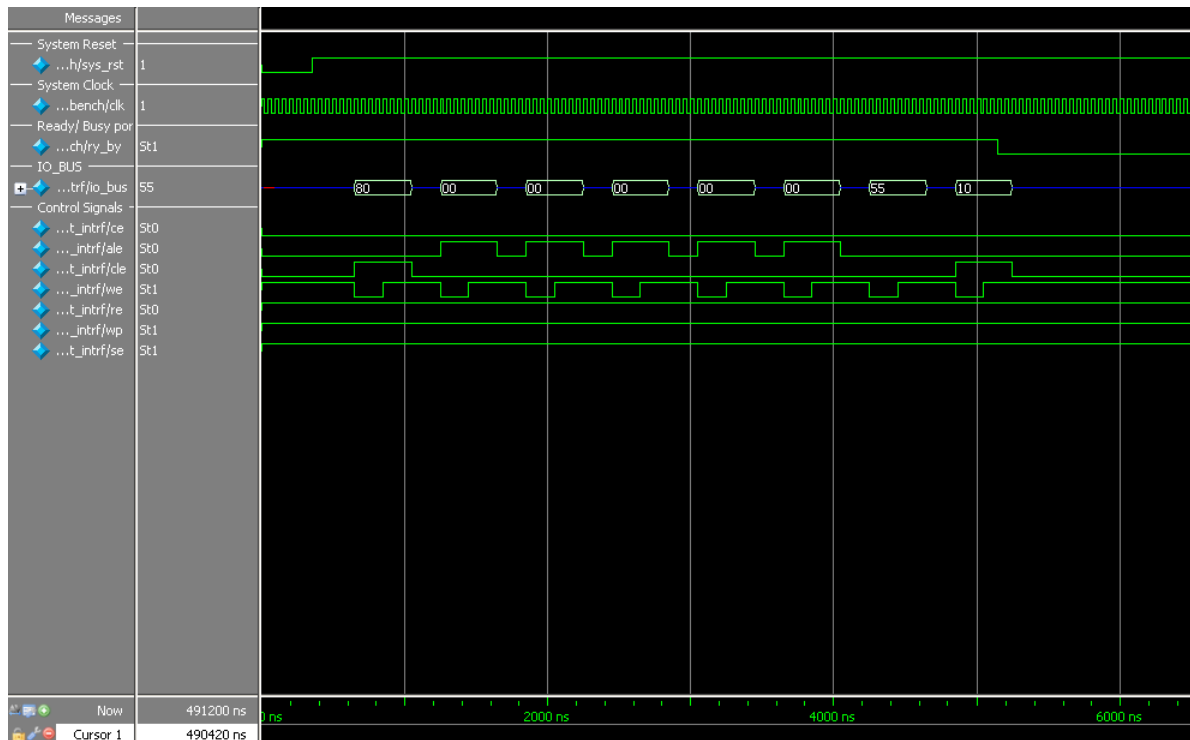


Figure 14 • Write Cycle

Figure 15 shows the read command cycle.

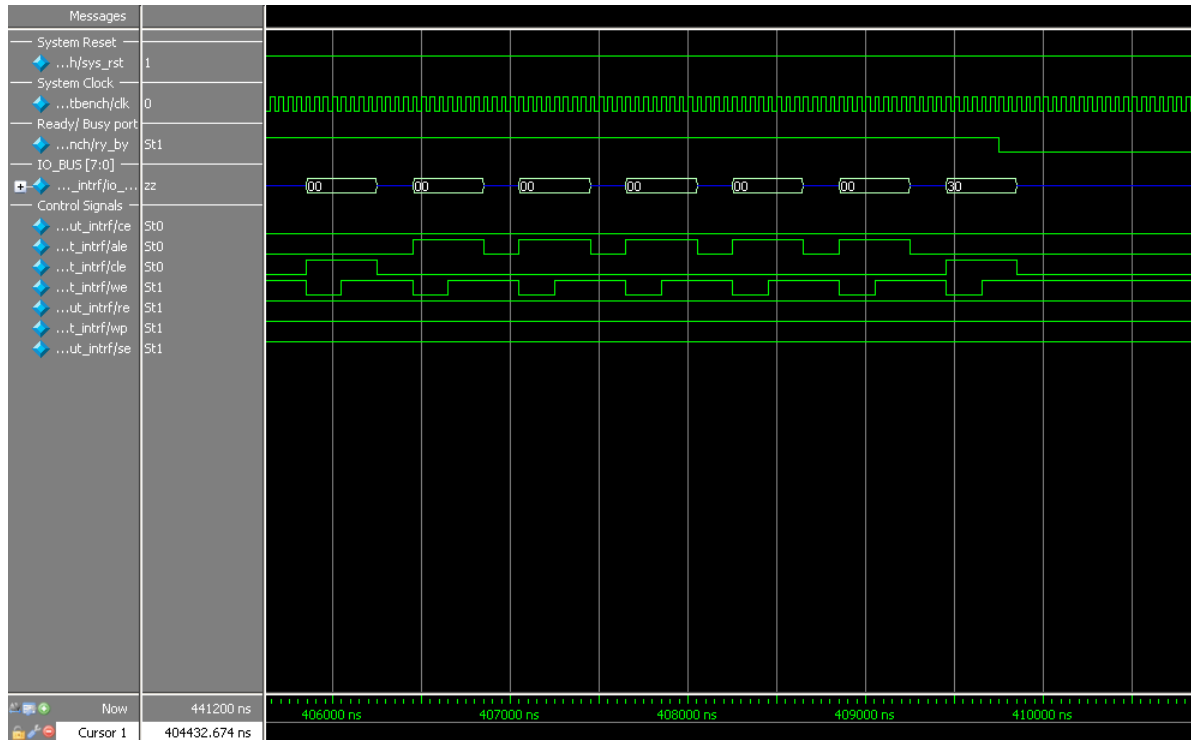


Figure 15 • Read Command Cycle

Figure 16 shows the complete read cycle.

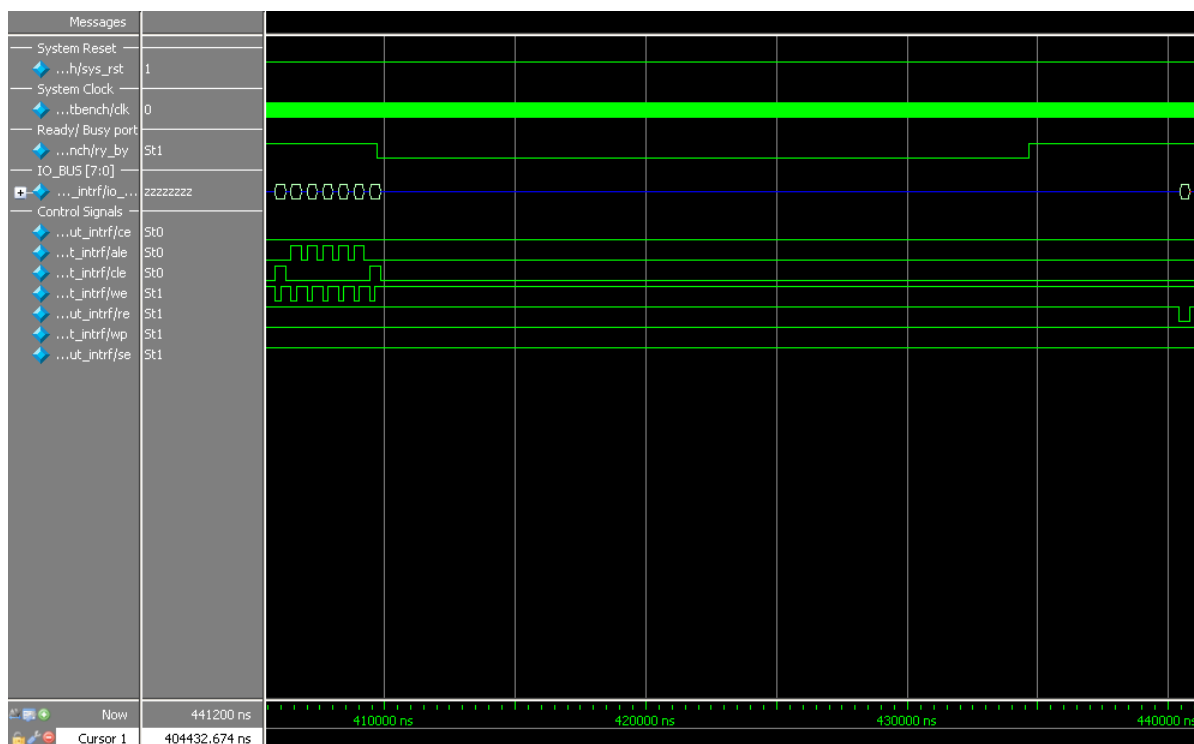


Figure 16 • Read Cycle

Application Area

NAND flash devices are commonly used in most solid state drive units, such as memory cards and compact media players. Due to the rapid growth of the consumer market, FPGAs too are becoming part of this market. This core, along with Microsemi's FPGAs, provide an easy-to-use, ready-made solution for similar requirements.

Conclusion

With their flexibility and system integration value, FPGAs are an intelligent choice for rapidly changing consumer markets. Most audio and video applications require large amounts of inexpensive and readily available memory such as NAND flash. This design example, when targeted to Microsemi's low-power IGLOO or ProASIC3 devices, provides a power efficient, programmable, secure, single-chip, compact solution that can be easily upgraded.

List of Changes

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

Revision	Changes	Page
Revision 1 (March 2015)	Non-Technical Updates.	N/A
Revision 0 (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

© 2015 Microsemi Corporation. All rights reserved. Microsemi and the Microsemi logo are trademarks of Microsemi Corporation. All other trademarks and service marks are the property of their respective owners.

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.

Microsemi makes no warranty, representation, or guarantee regarding the information contained herein or the suitability of its products and services for any particular purpose, nor does Microsemi assume any liability whatsoever arising out of the application or use of any product or circuit. The products sold hereunder and any other products sold by Microsemi have been subject to limited testing and should not be used in conjunction with mission-critical equipment or applications. Any performance specifications are believed to be reliable but are not verified, and Buyer must conduct and complete all performance and other testing of the products, alone and together with, or installed in, any end-products. Buyer shall not rely on any data and performance specifications or parameters provided by Microsemi. It is the Buyer's responsibility to independently determine suitability of any products and to test and verify the same. The information provided by Microsemi hereunder is provided "as is, where is" and with all faults, and the entire risk associated with such information is entirely with the Buyer. Microsemi does not grant, explicitly or implicitly, to any party any patent rights, licenses, or any other IP rights, whether with regard to such information itself or anything described by such information. Information provided in this document is proprietary to Microsemi, and Microsemi reserves the right to make any changes to the information in this document or to any products and services at any time without notice.