Proposal for an Open Source SLC NAND Flash Failure Analysis Platform

By:

Garrett Thomas, Greg Bray, Jeffrey Gorton, Jonathan Morgan, and Kyle Stewart

Date: November 12th, 2007

Clinic Sponsor:
Micron Technology, Inc.

Advisor:
Ken Stevens
University of Utah

Table of Contents

Abstract

The use of NAND Flash has increased worldwide each year as the cost of manufacturing decreases and memory density increases. Unfortunately, as manufacturing processes improve, the reliability and failure rates of NAND Flash memory have remained unchanged. Current SLC NAND Flash memory is guaranteed to operate for up to 100,000 Program-and-Erase cycles, but failure rates beyond this point are usually unavailable since they will vary significantly between manufactures and product lines. The lack of information beyond 100,000 cycles has been the greatest barrier to adoption in a wider market of devices. A testing platform that can run a specified use-pattern, detect and record memory failures, and then return a graphical report of actual failure rates would be of great benefit to product engineers interested in using NAND Flash memory in their devices. By developing an open platform for NAND Flash failure analysis, we aim to improve the use of NAND Flash technology in future devices and provide reliable data on failure rates beyond the SLC cycle limit specifications.

Introduction

NAND flash memory has become the preferred choice for high-density, nonvolatile memory storage in applications such as digital photography, USB flash drives, embedded systems, mp3 players, cellular phones, and solid state drives. Because of its small size, low cost, and low power requirements NAND flash is frequently used to replace or complement magnetic disk, static ROM, or optical drive storage. New applications for NAND flash memory continue to emerge each year, and the total market for NAND flash memory is predicted to reach $15 billion by 2010 [1].

Due to the specific nature of flash technology, permanent and temporary failures may occur during regular use and will increase in frequency over the lifespan of the memory eventually leading to complete memory failure. Failure can occur on a single bit or over an entire block and can be exacerbated by specific use patterns (partial page programming, high block read cycle count, high block erase/program count) [2]. Manufactures currently guarantee error-free functionality of their SLC NAND flash devices for up to 100,000 program and erase cycles when using basic Error Code Correction (ECC) algorithms [3]; however very little data is available regarding failure rates and usability past the manufacture specified limits as each unique NAND design, vendor, and fabrication process will demonstrate different failure mechanisms and behaviors.

The goal of this project is to create a simple and low cost platform for empirically determining the failure behavior of a SLC NAND flash chip beyond the manufacture specified limits. The system will allow users to define a custom use-pattern for wear testing over a specific number of cycles and will report the bit and block failure behavior measured under these conditions. This will allow product designers to make better use of NAND flash technology by understanding the cycle limitations and planning for graceful device degradation using bad block retirement and ECC algorithms to obtain the desired product lifespan.

Also the entire testing platform is to be released as an open source project to help improve the current state of NAND flash testing for the entire industry. A Google Code website has been setup to store all of the project files and act as a permanent public presence for an ongoing community of users [4]. This will allow future users to expand the project to work with additional NAND components not covered by the initial release.

Project Tasks

We have divided the project tasks according to the various components and assigned team members to each task:

- Greg – Front-end GUI and Database
- Kyle – Firmware and NAND Flash Controller
- Jon – System Architecture (Verilog) and Database
- Garrett – Random Number Generator and Debugging/Quality Assurance
- Jeff – Documentation, NAND Flash Controller and PCB Board/Daughter Card

Each team member will be responsible for developing and documenting the interfaces used by the components they are assigned to. The System Architecture involves the overall view of how the various Verilog components connect and interface to each other on the FPGA. It also includes the pin mappings that connect the Verilog flash controller through the daughter card and to the flash device itself.

Specific Task Interfaces

This section describes how the various components will interface with one another. The nine major components with specific interface requirements are listed in the table below and range from the user interaction with a front-end graphical user interface (GUI) to the hardware controlling the flash module itself. The interfacing model follows the flow of information from the high-level user input to the low-level implementation hardware. This section describes the interfaces in a top-down manner beginning with the high-level GUI. Commands flow through the interfaces until they are finally executed on the NAND flash hardware. The results of these executions then flow back up the chain of interfaces and are stored in a database. The user can then view reports using the application GUI.

Component	Physical location	Technology/Platform
SQL Database	Host Windows PC	Microsoft SQL Server
Front-end GUI	Host Windows PC	Microsoft .NET C#
USB 2.0 Driver	Host Windows PC	libusb-win32 and LibUsbDotNet
USB 2.0 Driver	Altera DE2 FPGA	Phillips ISP1362 USB controller
C Firmware	Altera DE2 FPGA	Altera Nios II IDE
NAND Flash Controller	Altera DE2 FPGA	Verilog, Altera Quartus II
Daughter Card	Daughter Card (DC)	PCB provided by Micron
Nand Flash Module	Daughter Card (DC)	SLC provided by Micron

Table 1: List of project interfaces

1. User <=> Front-end Graphical User Interface (GUI) on PC

The GUI allows the user to customize various parameters that affect how the program tests the NAND flash. The user must be able to choose various testing techniques and verify the results. The GUI must allow the user to execute the following set of basic commands.

- Erase a block
- Check block for erase failure flag
- Report the number of erasure failures (i.e. the number of 1's in an erased block)
- Program a page
- Verify programming was successful
- Read Page
- Read Verify - compare/verify a block against originally programmed pattern

The GUI must also support the ability to allow any sequence of the above commands to be automated. This may be done through the GUI itself or through a scripting language. In addition to the basic command set, the GUI must also allow the user to specify the following parameters.

- Range of blocks to be tested
- Number of cycles to repeat the command (minimum range of [1, 1048575])
- Auto-generated random number generation or a specific pattern

2. Front-end Graphical User Interface (GUI) on PC <=> USB 2.0 driver on PC

In order to interface the host to the USB 2.0 host driver, we plan to use the libusb-win32 library version 0.1.12.1. The libusb-win32 library is a Windows port of the libusb generic usb driver for Linux/UNIX systems. This library serves as a wrapper to a generic USB driver for Windows XP. Using this library, we will be able to talk to the bus using code developed and executed in user space and avoid the potential hazards of developing a custom, kernel space driver. There is an actively developed C# wrapper for the libusb-win32 library called LibUsbDotNet [5]. This wrapper allows us to develop our GUI code on the C# .NET platform and interact with the device through USB without using with unmanaged code or native device drivers.

3. Frontend GUI <=> SQL Database

The GUI will collect the results of each test by using the USB interface described above. These results will then be stored in a SQL database for further processing. The GUI will communicate with the database using the standard ActiveX Data Objects classes of the Microsoft .NET framework (ADO.NET). The GUI will then use Transactional SQL (T-SQL) to query the information stored in this database and present to the user the testing results. Initially the SQL table will contain a single table with six columns holding the following information:

- ID - A unique identification number assigned every time a new instruction is performed
- Cycle - The cycle count for this instruction
- Memory Address - The place in memory where this instruction begins
- Function Name - The name of the instruction performed (read block, erase block, program, etc.)
- Status - Pass/Fail

Eventually, the table will be expanded to include more fine grained detail that provides more insight into the failure characteristics of flash device.

4. C Firmware on the FPGA <=> USB 2.0 driver on the FPGA

The firmware running on the FPGA will be written in the C language and compiled to run on the Nios II soft processor available for the DE2 FPGA development board [6]. The firmware will interface with the Phillips ISP1362 USB controller chip built on the DE2 board. This board is compatible with the USB 2.0 specification, but will only communicate up to full-speed (12 Mbits/sec). The firmware must be able to process packets according to the USB 2.0 specification as well as interface with the libusb-win32 and C# code written for the GUI. The firmware must receive a 16 byte command from the GUI and send the results back to the host (up to 2112 bytes for read page) using as much of the 12 Mbits/sec bandwidth available to it.

5. C Firmware on FPGA <=> NAND Flash Controller on FPGA

The firmware module on the FPGA will interpret the commands sent to it through the USB 2.0 interface and issue them to the NAND Flash Controller. This involves listening for communication from the host and properly parsing the 16 byte commands into a sequence of instructions for the NAND Flash Controller. After the command has been parsed, the firmware will then communicate those instructions to the NAND flash controller through the on-chip RAM buffer. The processor will then wait for the command to execute on the flash and gather any results it needs. This information is then sent back over the bus through the USB interface to the host.

6. NAND Flash Controller on FPGA <=> NAND Flash Module device under test

The flash controller has already been developed by last year's team. It has been developed in Verilog and interfaces to the flash device through the daughter card provided by Micron. The interface is well defined in the Micron datasheets. Stretch goals include developing new controllers that allow flash devices from different manufacturers to be tested alongside Micron chips.

Testing and Integration Strategy

Because of the considerable time required for testing and integration, we will attempt to minimize problems in the beginning stages of development. For this reason the project has

been segmented to allow for individual component testing, debugging, and interfacing before integrating with other components. Each development task will be completed by two or more team members to ensure a knowledge overlap on critical sections. Also one team member will be responsible for guiding the overall development and testing of each component.

After the individual components are completed, system integration and debugging will move on to make sure that the system can endure extended testing sessions that may last upwards of 3 months time. We will initially start with short test sessions over small quantities of memory and make sure that all the data acquisition and analysis is correct. Once the collected data has been found to exhibit the expected results, testing will be expanded to larger memory blocks and the process will be repeated.

Group Management and Communication Plan

Communication is essential to completing and delivering the NAND Flash test platform, but we must also consider communication with developers that will become the eventual end users of our system. We have setup a Google Code website to store all the documents and meeting notes during our project development and be the central repository for all of our project source code [4]. The website will also provide a permanent public presence on the Internet that can be used for ongoing issue tracking and collaboration between users.

During the project development we will hold a weekly meeting in which all members report on the progress of their assignments. A meeting log containing a detailed report of what was accomplished and what the next assignment are will be posted on the project website. In this manner everyone will be accountable for completing the project on time. To further provide communication, each member will be assigned a major role in the group and will be responsible for that aspect of the project.

Schedule and milestones

Task	Start Date	End Date
Become familiar with project code and FPGA	October 21, 2007	October 31, 2007
System integration	November 1, 2007	November 15, 2007
System testing	November 16, 2007	November 30, 2007
Final testing and integration	December 1, 2007	December 10, 2007
Present working system to Micron, Begin collecting data	Week of December 10, 2007	
Collect and analyze data, Work on stretch goals and adding new features, Prepare final project documentation, Prepare final project deliverables	January and February, 2008	
Present project at ECE Technical Open House	March, 2008	
Present project findings at Flash Summit	June, 2008	

Table 2: Schedule and milestones

Risk Assessment

One of the first challenges that needs to be addressed will be debugging preexisting code and completing a thorough testing of the existing application. Following this, the code will then need to be modified and adapted to allow for a random bit generator, reports for the GUI and accurate data collection. This will require that the SQL database be structured to allow for potentially millions of records. Reports generated from this data will be important in understanding the characteristics of the tested NAND flash chip. Patterns that will be of most interest will be the number of read/write cycles it will take to reach 10, 20, or 50 percent failure rates and whether there is a grouping pattern associated to the bit failures.

The USB driver that was developed last year might not be adequate for this year. It will be required to evaluate the driver and assess if it will be useful or if a new driver will need to be written.

Another challenge will be to complete the design and debugging portion of this project before the end of the year so that adequate data can be collected, analyzed and reported on. This will require that the program communicate flawlessly with each of the separate components, which really means that development should be completed as quickly as possible so that there will be adequate time for testing to assure that there will be minimal data corruption.

Bill of Materials

NAND FLASH VENDOR INFORMATION

I **Micron Technology** contact: Dennis Z.
8000 S Federal Way, Boise, ID 83706 F: 651-994-8186 P: 208-994-8200
Part # : 29F2G08AAC 2 Gbit w/ 8 bit I/O

Time:	3-5 Days	Price/Unit Cost:	$0.00
Quantity:	6	Total Cost:	$0.00

II **Luscombe Engineering Company Inc.**
6465 S 3000 E Suite 101, SLC, UT 84121 P: 801-944-1280
Part # : JS29FO4G08AANB1 2 Gbit w/ 8 bit I/O

Time:	3-5 Days	Price/Unit Cost:	$
Quantity:	1	Total Cost:	$

FPGA BOARD VENDOR INFORMATION

I **Altera** contact: university@altera.com
357 S McCaslin Blvd, Louisville, CO 80027 F: 303-926-4945 P: 303-926-4955
Part # : DE2 DE2 Development & Education Board

Time:	3-5 Days	Price/Unit Cost:	$269.00
Quantity:	1	Total Cost:	$269.00

II **Digilent Inc.** contact: www.digilentinc.com
P.O. Box 428, Pullman, WA 99163 F: 509-334-6306 P: 509-334-6306
Part # : XUPV2P Virtex-II Pro Development System

			Price/Unit	
Time:	3-5 Days		Cost:	$299.00
Quantity:		1	Total Cost:	$299.00

USB 2.0 VENDOR INFORMATION

I **L-com, Inc.** contact: www.l-com.com/home.aspx
45 Beechwood Drive, North Andover, MA 01845 P: 978-682-6936
Part # : CSMUAA-1M Premium USB Type A-A Cable, 1.0m

			Price/Unit	
Time:	3-5 Days		Cost:	$8.45
Quantity:		1	Total Cost:	$8.45

II **Cables To Go** contact: www.cablestogo.com
1501 Webster Street, Dayton, OH 45404 F: 800-331-2841 P: 937-224-8646
Part # : 39979 JETLAN USB 2.0 Net/Data Transfer Cable

			Price/Unit	
Time:	3-5 Days		Cost:	$22.99
Quantity:		1	Total Cost:	$22.99

Vendor List

The following hardware and software components will be used to complete this project:

NAND Flash Daughter board – Provided by Micron

NAND Flash SLC chips – Provided by Micron [3]

Altera DE2 FPGA development board – Provided through the University of Utah [6]

TortoiseSVN 1.4.5 subversion client – Free Download [7]

Subclipse plugin for Eclipse/NIOS IDE – Free Download [8]

Altera Quartus II Web Edition Verilog development environment – Free Download [9]

Altera Nios II Embedded Design Suite – Free Download [9]

LibUSBDotNet – C# LibUSB-Win32 wrapper for generic USB drivers – Free Download [5]

Visual Studio 2005 - IDE for C# GUI development – VS 2005 Express download [10]

Microsoft SQL 2005 Server - SQL database engine to store results -- MS SQL Express [11]

Conclusion

The integration of NAND Flash memory into new devices can lead to many improvements and technological innovations. Because of the physical limitations on read and write cycles and the lack of empirical data on failure rates, NAND Flash memory is often not utilized to its full capacity. This test platform will allow users to ensure that NAND Flash is a reliable memory system for their specific use-scenario and will provide failure data beyond the manufacture's stated 100,000 cycle limit. The test platform will also allow users to compare different NAND Flash chips or vendors to find the product offering that best fits their application's needs.

References

[1] Micron NAND Flash 101 Technical Notes - Figure 1, http://download.micron.com/pdf/technotes/nand/tn2919.pdf

[2] NAND Flash Design and Use Considerations, http://download.micron.com/pdf/technotes/nand/tn2917.pdf

[3] http://2007-uofu-micron-clinic.googlecode.com/files/2_4_8gb_nand_m49a.pdf

[4] Google Code Website http://code.google.com/p/2007-uofu-micron-clinic/

[5] LibUsbDotNet, http://www.picmicrochip.com/index.php?page=LibUsbDotNet_Project

[6] Altera DE2 Board, http://www.altera.com/education/univ/materials/boards/unv-de2-board.html

[7] TortoiseSVN Client, http://tortoisesvn.net/downloads

[8] Eclipse Subversion plugin, http://www-128.ibm.com/developerworks/opensource/library/os-ecl-subversion/

[9] Altera Software https://www.altera.com/support/software/download/altera_design/quartus_we/dnl-quartus_we.jsp

[10] Visual Studio Express, http://msdn.microsoft.com/vstudio/express/

[11] MS SQL Express, http://msdn2.microsoft.com/en-us/express/aa718378.aspx