Beyond the Datasheet: Using Test Beds to Probe Non-Volatile Memories' Dark Secrets

Laura M. Grupp*, Adrian M. Caulfield*, Joel Coburn*, John D. Davis†, and Steven Swanson*

*University of California, San Diego
†Microsoft Research, Silicon Valley Lab
[lgrupp, acaulfie, jdcoburn, swanson]@cs.ucsd.edu
john.d@microsoft.com

Abstract

Non-volatile memories (such as NAND flash and phase change memories) have the potential to revolutionize computer systems. However, these technologies have complex behavior in terms of performance, reliability, and energy consumption that make fully exploiting their potential a complicated task. As device engineers push bit densities higher, this complexity will only increase. Managing and exploiting the complex and at times surprising behavior of these memories requires a deep understanding of the devices grounded in experimental results. Our research groups have developed several hardware test beds for flash and other memories that allow us to both characterize these memories and experimentally evaluate their performance on full-scale computer systems. We describe several of these test bed systems, outline some of the research findings they have enabled, and discuss some of the methodological challenges they raise.

1. Introduction

Non-volatile memory has recently emerged as a possible replacement for main memory and hard disk drives (HDDs). While these devices have some limitations, the potential benefits that the components promise more than outweigh them. In current systems, non-volatile memories usually appear as solid state disks (SSDs) that use NAND flash to build "black box" replacements for conventional HDDs.

Constructing SSDs or other components built from non-volatile memories presents two problems that researchers and designers must confront. First, they must determine how to best construct a system to exploit the strengths and mitigate the weaknesses of the technology. The latency, bandwidth, bit error rate, and energy efficiency will all combine to determine which designs are viable and which are not. Second, they must understand how the non-volatile memory array will affect the rest of the system. If a new storage technology significantly increases performance, it may expose bottlenecks in other parts of the system.

Addressing the first challenge requires a comprehensive understanding of non-volatile memory performance, failure modes, and energy consumption. This information is difficult to come by. Device manufacturers hide the details of their devices behind non-disclosure agreements, and even when datasheets are freely available, they provide scant details: average performance characteristics, vague recommendations about error correction requirements, and inexact bounds on energy consumption. The datasheets make no mention of many "warts" that these devices possess. It is not that datasheets are inaccurate; it is that they are woefully inadequate if our goal is to fully exploit the capabilities of these memories.

To address the second challenge and fully evaluate a new non-volatile storage array, it is easiest to measure and observe the array *in situ*, in a working system, running real applications. The benefits of this approach – versus, for example an analytical model or simulation – is that it reveals unexpected hardware and/or software bottlenecks and reveals the actual benefit of this application.

The only way to address either of these challenges is to construct working hardware test beds – custom-built hardware systems that incorporate non-volatile memories or can emulate them at very high fidelity. These test beds can provide "ground truth" data on the behavior of both memory devices and systems that incorporate them.

This paper describes the non-volatile test beds that we have built and provides some results from them. We describe some of the insights we have gleaned from them and discuss some of the challenges that building and using these test beds presents. Our experiences with these systems demonstrate that they can provide a wealth of useful data and raise unexpected and important questions that motivate and inform research into applying non-volatile memories.

2. Hardware Test beds

Hardware test beds come in two flavors: characterization platforms and application platforms. Characterization platforms allow researchers to observe the behavior of the device under test at a finer granularity than the information provided in the device datasheet. Application platforms allow researchers to evaluate the impact of a memory technology at the application level, in a working system, and in the presence of real-world software and hardware overheads. In some cases, the same platform can be used for both characterization and applications.

Characterization platforms

A device's datasheet provides some information about how it will behave, but it does not provide the level of detail that a characterization platform can extract. Whereas a datasheet may provide a latency range for a device operation, the characterization platform can observe the distribution of latencies for that operation across the chip or a collection of chips. In general, these platforms may be able to observe characteristics of the device that are not reported in the datasheet.

We have used a platform called Ming to characterize many NAND flash devices. Ming is composed of an off-the-shelf Xilinx FPGA-based development board as well as a custom-built flash testing board, enabling detailed measurements of almost all aspects of flash memory devices. The test board holds two "burn in" sockets that accept standard TSOP flash devices. Each socket has a separate power plane to support per-chip power measurement. Figure 1 provides a block diagram of the Ming hardware platform.

In addition to the hardware, Ming also includes several software components that make it useful as a characterization system. The Xilinx FPGA contains a microprocessor and we configure it with a custom-built flash memory controller. We run a full-fledged version of Linux on the processor, and provide a custom driver

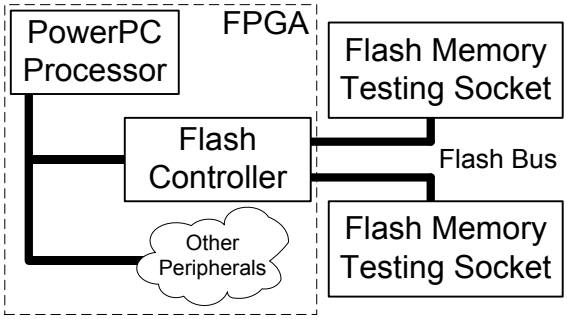

Figure 1. The Ming/Zarkov system uses a PowerPC processor and a custom flash controller built into an FPGA to provide direct access to flash devices for either characterization (in Ming) or FTL prototyping (in Zarkov).

for the flash controller. The driver is unique in that it provides user-level access to low-level flash operations (read, program, and erase, etc.). The result is that developing test code for Ming is very easy. For instance, to test a new error correction or data encoding technique, it is sufficient to implement it in C.

The combination of the Ming hardware and software provides a wealth of information about each flash memory access. The driver returns the operation latency at 10ns resolution and allows straight forward measurement of the energy consumed by each chip.

Application platforms

Application platforms offer insight into how a new technology will affect application-level behavior. They offer insight into how storage devices will interact with other hardware and software components. This can, for instance, reveal bottlenecks in the system that the new technology exposes. Below we describe the application platforms our groups have developed and used.

The first is Zarkov. Zarkov relies on the same FPGA-based controller and FPGA prototyping board as Ming but holds up to 32, non-removable flash chips. Zarkov is a key component of the BlueSSD project [5]. BlueSSD aims to provide an open platform for experimenting with SSD optimizations.

The second application platform is the Flash Research Platform (FRP). The FRP hardware is a combination of the BEE3 multi-FPGA research platform [1] and a custom printed circuit board, a Flash Dual Inline Memory Module (FDIMM). The combination of an FPGA and DIMM slot provide the ability to interface to other non-volatile memories, like PCM, for future research, leveraging the same BEE3 hardware and software. Each FDIMM exposes 8 independent flash channels and up to four FDIMMs

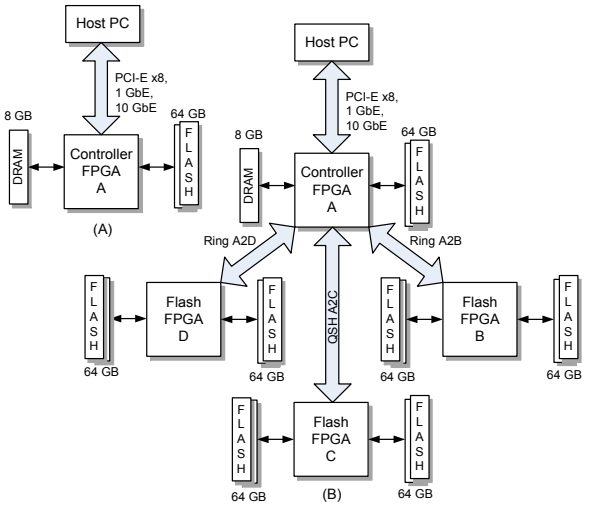

Figure 2. FRP configuration ranges from (A) a single FPGA and associated DIMM cards to (B) a multiple FPGA system with up to 16 DIMM cards. These configurations assume 4 GB DRAM DIMMs and 32 GB FDIMMs.

can be connected to an FPGA. The BEE3 has 16 DIMM slots that we can populate with DRAM, FDIMMs, or other special-purpose modules. Figure 2 illustrates the flexibility of FRP to build small or large systems using one to four FPGAs and their associated DIMM slots.

We have built a variety of FDIMMs ranging from 8GB to 128 GB per card or 128 GB to 2TB per BEE3, using current generation TSOP NAND flash devices. The majority of the FDIMMs are built using eight 4GB SLC Samsung NAND flash TSOP package (Writes: 20 MB/s, Reads: 40 MB/s), making a 32 GB module [7]. More details describing the hardware and software components of the FRP system as an application platform can be found in [2].

There is also an FRP software layer that is responsible for translating user-level commands into NAND flash specific operations. This software management layer abstracts away the details of the flash devices operations, such as block erase, and provides a well-defined hardware independent interface for read/write /flush or APIs like *trim* and *secure erase*. Likewise, flash-specific commands can be easily implemented in this layer and sent to the flash controller.

The FRP management software implements various algorithms used in the Flash Translation Layer (FTL) such as cache management, garbage collection, wear leveling and logical to physical sector remapping [3]. The FRP management software translates various tasks into the gateware layer commands and monitors the execution of these commands on the actual hardware. The FRP management software can be executed on an embedded CPU in the FPGA, or on the host CPU in the device driver or even as a user-level application. Currently, we organize the management layer as a user-level application on a PC running Windows XP, trading off performance for ease of implementation. Co-development of the user-level application and the FPGA flash controller enable implementing and instrumenting all flash operations. Like the Ming/Zarkov platform, the FRP can be a characterization or application platform. The main difference is that the FRP platform does not have separate power planes per flash package on the FDIMM, enabling only module power measurement and not individual package power measurement.

The final application platform is a prototyping system for developing PCIe-attached storage devices based on next-generation non-volatile memories. The platform is called Moneta and it targets memories such as phase change memory, the memristor, or scalable MRAM technologies.

Moneta uses the same BEE3 FPGA platform as the FRP, but instead of incorporating actual non-volatile memories, it uses DRAM to emulate a range of fast non-volatile memories that are not yet commercially available. To do this, Moneta uses a modified DRAM controller that modifies the read and write latencies of the DRAM to match those of the non-volatile memory. This provides a very high-fidelity model for memories and allows us to engineer both the storage array architecture and the driver stack to take full advantage of the technology.

3. Results

These non-volatile memory test beds have led to some interesting and unexpected findings. Below, we describe results from both types of test beds.

Characterization results

Both Ming and FRP have proven to be very capable tools for understanding flash memory behavior under a range of circumstances, from operating outside the normal datasheet specification to observing behaviors not specified by the datasheet.

One of the most interesting results came from a characterization of write-once-memory (WOM) codes applied to flash memory [4]. WOM codes are useful for memories, like flash, that only allow bit transitions

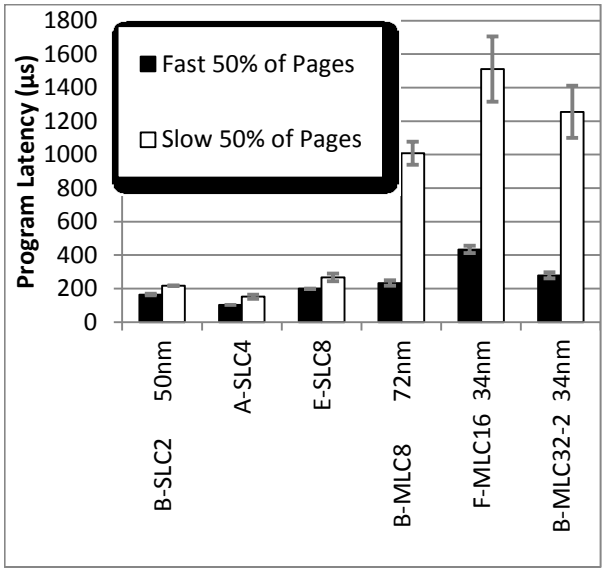

Figure 3. Measured program latencies for SLC and MLC flash pages divided into fast and slow groups.

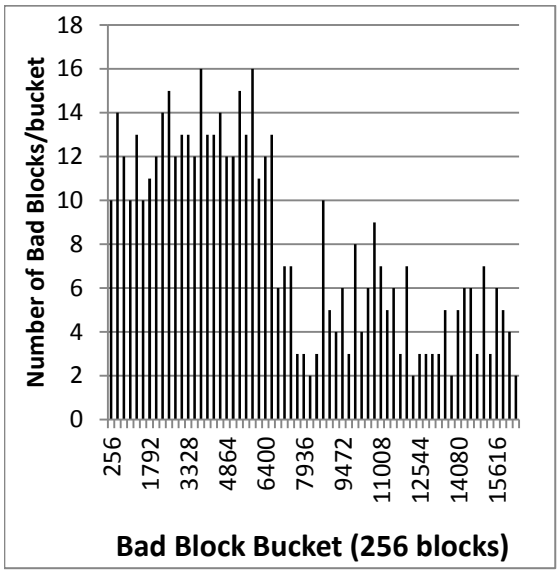

Figure 4. Bad block distribution for 16 MLC NAND flash components.

in one direction (1 to 0). They store two logical bits using three physical bits, but the system can program those physical bits twice before erasing them. Simple calculations show that WOM codes should increase flash lifetime by 33% by eliminating one third of erase operations needed to write a given amount of data to an SSD. Measurements on Ming demonstrated much greater increases (5.2x) for some flash chips, but a dramatic reduction in lifetime for others. Ming has also allowed us to empirically measure the effectiveness of different ECC schemes [8].

One of the most important applications we have found for Ming is in developing and validating models for flash memory performance, power consumption, and reliability. Analytical models such as [6] can incorporate significant errors, but by closely examining mismatches between the models predictions and measurements on Ming, we can increase the accuracy of those models. This type of work will increase in importance for error correction and data coding as flash memory's reliability falters with continued scaling. It is likely that developing more robust error correction schemes will require integrating increasingly detailed physical models into the channel model, mirroring the development of the advanced coding techniques for hard drives.

As NAND flash feature sizes scale down with each successive silicon technology generation, the device behavior resembles more of a distribution than absolute discrete values. We can leverage these distributions and learn about the device if we can observe the distributions. The key is determining if the observations are device specific or not.

Figure 3 shows how the analysis of many data points reveals unexpected patterns in flash's behavior. The figure measures program operation latency for three SLC and three MLC devices. For SLC devices, the datasheets report a "typical" latency of around 200us, and our measurements corroborate that. For MLC devices, the datasheets give between 600 and 800us, and while that is the average latency, the data in graph show that it is not the whole story. For MLC devices, exactly half the pages are almost as fast as SLC devices, but the other half are up to 5x slower. The "fast" pages are faster to read and consume less energy per program operation than the "slow" pages and for some chips, "slow" pages have higher bit error rates.

Figure 4 shows results from the FRP measuring the distribution of bad blocks across multiple flash devices. From a set of 16 MLC multi-die packages, we can see that the lower half of the blocks contains more bad blocks than the upper half. However, when producing a histogram for a set of 168 SLC multi-die packages, we saw a more uniform distribution (results not shown). Initial bad block distributions can be used in the FTL metadata and control structure design. This information may also provide insight into defect distribution and on-die process variation for a particular silicon technology generation and/or fab.

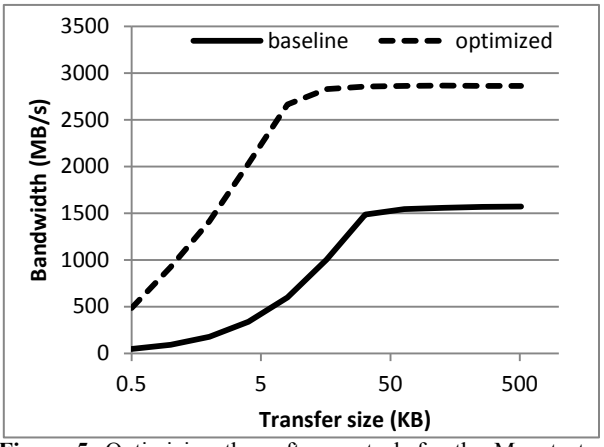

Figure 5. Optimizing the software stack for the Moneta test bed can improve performance by up to 10x and provides insights into designing software for fast non-volatile memories.

As with all characterization platforms, most of the implementation effort is concentrated on implementing the software features or making the device compatible with existing software. The FRP platform eases this development cycle by enabling FTL components either in the FPGA, user-level management software or the driver. We implemented 8-bit BCH ECC in the user-level application and this required almost full utilization of a single core in a dual core Intel Core 2 Duo 2.4 GHz processor. This demonstrates that future ECC implementations for next generation MLC or SLC devices will require complex hardware for error correction, making software implementation difficult.

By using these hardware platforms, we can observe how device characteristics change over time. These changes may be indicators of device failure, device reliability, data retention time, asymmetry in page or block performance for reads, programs or erase, etc. Leveraging these characteristics in the FTL can lead to improved performance, reliability, advanced coding techniques, or other properties that can make the system viable or differentiate the system from its competitors.

Application results

Moneta has allowed us to explore issues that span the entire system from the storage hardware through the applications. Existing software assumes that storage (i.e., disks) is slow. PCM-based SSDs will be orders of magnitude faster, and will require re-engineering both the operating system and the applications.

Figure 5 shows the results of this optimization for the Linux storage driver. The horizontal axis measures transfer size, while the vertical axis measures sustained throughput. The "baseline" data shows the performance with the stock Linux IO stack. The "optimized" data is the performance of the same Moneta array with an optimized software stack. For 4KB accesses, optimizing the software stack improves performance by 4.6 times and reduces per-IOP software overhead by 60%. The data also show that, for 512 byte requests, the remaining software overheads limit Moneta's performance far more than the hardware.

These results are useful for operating system designers, but they also provide the basis for understanding the changes needed at other layers as well. Our lab is now studying how to modify these layers to run well on Moneta, and everything we learn will inform the design of software systems that will run on next-generation arrays when fast non-volatile memories become available.

4. Discussion

Hardware test beds offer unique advantages, but they also increase the difficulty of performing some research. The main benefit of building real, working hardware prototypes and measuring real system is that it places the results and analysis on a firm footing. It also significantly expands the range of workloads that researchers can run on the prototype systems, since the hardware prototypes are much faster than simulated versions of the same system.

Systems like Ming provide a different kind of experimental foundation. Ming provides concrete, detailed measurements that can inform other work. The alternative, using device models derived from datasheets is limiting in several ways. First, datasheets provide limited, coarse-grain information about device performance. For instance, almost all of the interesting results in [4] were uncovered behaviors not documented in datasheets. Without measuring these properties directly, it would be impossible to exploit them.

However, there is no free lunch and abandoning datasheets comes at a cost. Trends that appear in experimental data may not hold across different hardware revisions, over temperature ranges, or remain valid over the lifetime of the device. Measured variation between devices from different manufacturers or in different manufacturing technologies also raises the danger of encouraging the researchers to over-

specialize systems rather than designing for flash memory in general.

The solution to both of these difficulties is thorough, ongoing testing to both identify unintended consequences of particular usage patterns and to understand the variation and trends in behavior between manufacturers and over time.

Hardware test beds can also significantly increase the time required to evaluate an idea. Developing Moneta took about 10 months, but a simulator for the same system would have taken, perhaps, 3 months to build. The results from a simulator would have been less reliable, and, more important, fully implementing the system has suggested several directions for further research.

The final danger in building hardware prototypes arises because presently available technologies may constrain the design of a system. For instance, to reduce cost, Zarkov uses an off-the-shelf FPGA prototyping board that has limited IO capacity. This restriction limits the number of flash chips the controller can communicate with at once, and this, in turn, constrains the space of SSD designs it can model.

There are two potential solutions to this problem. The first is to treat the hardware as an emulation system rather than a prototype implementation. This is the approach we took with Moneta, and it requires the researcher to maintain a clear distinction between the *modeled system* and the test bed system. In Moneta, this manifests itself in the memory controller that inserts delays to model fast non-volatile memories. More generally, the modeled system and the test bed may differ in terms of capabilities and performance. For instance, the modeled system might emulate faster memory by artificially slowing the rest of the system and scaling the measured performance of the system afterward.

The second solution is to build the test bed system from scratch rather than leveraging off-the-shelf hardware components. This approach is more expensive and time consuming than using general-purpose prototyping platforms, and it is not always possible: if the memory technology of interest is not yet commercially available, emulation (as described above) is the only alternative.

5. Conclusion

Hardware test beds for non-volatile memory-based systems can provide researchers with a wealth of concrete information about both device and full system performance. We have described a few of the systems we have built to study these systems and highlighted both the advantages that these systems offer as well as the challenges they present. On balance, we believe that taking the time to construct working prototypes and gather data firsthand about memory technology performance is well-worth the effort. Indeed, as system-level interactions become more complex, these test beds will become an increasingly important means to fully understand the implications of non-volatile memories in computer system design.

References

[1] J. D. Davis, C. P. Thacker, and C. Chang, BEE3: Revitalizing Computer Architecture Research, no. MSR-TR-2009-45, April 2009

[2] J. D. Davis and L. Zhang, FRP: a Nonvolatile Memory Research Platform Targeting NAND flash, in The First Workshop on Integrating Solid-state Memory into the Storage Hierarchy, Held in Conjunction with ASPLOS 2009, ACM, Inc., March 2009

[3] E. Gal and S. Toledo, Algorithms and data structures for flash memories, ACM Computing Surveys (CSUR), v.37 n.2, p.138-163, June 2005

[4] L. Grupp, A. M. Caulfield, J. Coburn, E. Yaakobi, S. Swanson, P. H. Siegel, Characterizing Flash Memory: Anomalies, Observations, and Applications, In the Proceedings of the 42nd International Symposium on Microarchitecture.

[5] S. Lee, K. Fleming, J. Park, K. Ha, A. M. Caulfield, S. Swanson, Arvind, J. Kim, BlueSSD: An Open Platform for Cross-layer Experiments for NAND Flash-based SSDs, In the 5th Workshop on Architectural Research Prototyping, 2010.

[6] V. Mohan, S. Gurumurthi, M. R. Stan, FlashPower: A detailed power model for NAND flash memory, DATE 2010: 502-507.

[7] Samsung Corporation, K9XXG08XXM Flash Memory Specification, http://www.samsung.com/global/system/business/semiconductor/product/2007/6/11/NANDFlash/SLC_LargeBlock/8Gbit/K9F8G08U0M/ds_k9f8g08x0m_rev10.pdf, 2007.

[8] E. Yaakobi, J. Ma, L. Grupp, P. H. Siegel, S. Swanson, J. K. Wolf, Error Characterization and Coding Schemes for Flash Memories, To appear in: Workshop on the Application of Communication Theory to Emerging Memory Technologies, 2010.