Near-Data Processing: Insights from a MICRO-46 Workshop

Rajeev Balasubramonian
University of Utah

Jichuan Chang
Google

Troy Manning
Micron

Jaime H. Moreno
IBM Thomas J. Watson Research Center

Richard Murphy
Micron

Ravi Nair
IBM Thomas J. Watson Research Center

Steven Swanson
University of California, San Diego

After a decade-long dormancy, interest in near-data processing (NDP) has spiked. A workshop on NDP was organized at MICRO-46 and was well attended. Given the interest, the organizers and keynote speakers have attempted to capture the key insights from the workshop for wider dissemination. This article describes why NDP is compelling today and identifies upcoming challenges in realizing its potential.

Processing in large-scale systems is shifting from the traditional computing-centric model successfully used for many decades into one that is more data centric. This transition is driven by the evolving nature of computing, which is no longer dominated by the execution of arithmetic and logic calculations but instead by the handling of large data volume and the cost of moving data to the locations where computations are performed. The computing-centric model where data lives on disk, or even tape, and moves as needed to a central computing engine across a deep storage hierarchy is sufficient when computational aspects dominate data movement aspects. In contrast, in the data-centric model, data lives in different storage levels within the hierarchy, with processing engines surrounding the data and operating on such data without moving it across the system.

The trend toward big data is leading to changes in the computing paradigm, and in particular to the notion of moving computation to data, in what we call the *near-data processing* (NDP) approach. Data movement impacts performance, power efficiency, and reliability, three fundamental attributes of a system. NDP seeks to minimize data movement by computing at the most appropriate location in the hierarchy, considering the location of the data and the information that needs to be extracted from that data. Thus, in NDP, computation can be performed right at the data's home, either in caches, main memory, or persistent storage. This is in contrast to the movement of data toward a CPU independent of where it resides, as is done traditionally. Examples of NDP already exist in systems that execute computations close to the disk, filtering or preprocessing the data streaming from the disks so that a minimal number of items are transferred for processing at other parts of the system. Conceptually, the same principle can be applied at other levels of a system's memory and storage hierarchy by placing computing resources close to where data is located, and restructuring applications to exploit the resulting distributed computing infrastructure.

At the MICRO-46 conference, a workshop was held to bring together experts from academia and industry, who presented recent advances in the development of large systems employing NDP principles (www.cs.utah.

Published by the IEEE Computer Society

Recent Work in Near-Data Processing

This sidebar lists some recent products and papers in NDP as evidence of the resurgence of interest in this area, and provides an initial reading list for interested researchers.

Recent companies and products in the NDP space include Netezza, Venray Technology, EMU Technology, Micron (the Hybrid Memory Cube and Automata Processor), Convey Computer, DSSD, Adapteva, and Oracle (the Exadata).

The following six papers were presented at the Workshop on Near-Data Processing (http://www.cs.utah.edu/wondp):

- S. Kumar et al., "SQRL: Hardware Accelerator for Collecting Software Data Structures"
- G. Loh et al., "A Processing in Memory Taxonomy and a Case for Studying Fixed-Function PIM"
- B. Cho et al., "XSD: Accelerating MapReduce by Harnessing GPU inside SSD"
- A. Anghel et al., "Spatio-Temporal Locality Characterization"
- H. Tseng and D. Tullsen, "Data-Triggered Multithreading for Near-Data Processing"
- M. Chu et al., "High-Level Programming Model Abstractions for Processing in Memory"

Other recent papers on NDP include the following:

- A. De et al., "Minerva: Accelerating Data Analysis in Next-Generation SSDs," *Proc. IEEE 21st Ann. Int'l Symp. Field-Programmable Custom Computing Machines,* 2013, pp. 9-16.
- J. Chang et al., "A Limits Study of Benefits from Nanostore-Based Future Data-Centric System Architectures," *Proc. 9th Conf. Computing Frontiers,* 2012, pp. 33-42.
- S. Pugsley et al., "NDC: Analyzing the Impact of 3D-Stacked Memory+Logic Devices on MapReduce Workloads," *Proc. Int'l Symp. Performance Analysis of Systems and Software* (ISPASS), 2014.
- Q. Guo et al., "A Resistive TCAM Accelerator for Data-Intensive Computing," *Proc. 44th Ann. IEEE/ACM Int'l Symp. Microarchitecture,* 2011, pp. 339-350.
- Q. Guo et al., "AC-DIMM: Associative Computing with STT-MRAM," *Proc. 40th Ann. Int'l Symp. Computer Architecture,* 2013, pp. 189-200.

edu/wondp). This first workshop in this emerging area of computer architecture was very successful, with more than 60 people in attendance. The workshop program consisted of four keynotes and six presentations based on submitted papers. This article aims to convey the insights developed at this workshop to a wider audience.

Resurgence of interest in NDP

NDP is not a new concept; intelligent controllers near memory, I/O, and disk have been considered for several decades, most notably as processing-in-memory (PIM). Multiple groups of researchers in the 1990s built PIM prototypes and demonstrated the potential for significantly improved performance in many application classes.[1-4]

Most PIM projects were based on a single PIM chip having a number of DRAM or embedded DRAM (eDRAM) arrays and a few simple processing cores located near these arrays. Such PIM chips were connected to a traditional host processor with a custom interconnect. A runtime system was responsible for spawning and migrating tasks so that the computation was performed in close proximity to the data handled by that task.

Although PIM research yielded a number of promising results, widespread commercial adoption remained elusive. This could be attributed to several factors. Incorporating DRAM and logic on a single chip meant that DRAM was being designed with a costlier logic process, or that logic was being designed with a process optimized for DRAM. The memory industry is extremely cost sensitive, and the inevitable increase in cost per bit implied by PIM did not help. Moreover, exploitation of PIM required programmers to grapple with a new programming model.

After a dormant decade, NDP research is currently trending upward. In its most recent reincarnation, NDP has assumed a scope broader than that captured by the PIM research of the 1990s. NDP is now being applied at many levels of the memory hierarchy, from caches to DRAM to nonvolatile storage-class memory to hard disk drives.[5-9] This resurgence is motivated by new technologies such as 3D stacking, big-data workloads with high degrees of parallelism, programming models for distributed big-data applications, accelerator use in specific domains, and the need to overcome the diminished benefits from technology scaling through new

Everything Old is New Again: How Moore's Law Continues to Drive the Future

Processing-in-memory (PIM) and processing-near-memory have been rich areas of research for decades, but are now poised to enter the mainstream. With the end of Dennard scaling, performance improvements cannot depend on increases in the transistor's switching frequency; they need fundamental changes in computer architecture. The multicore paradigm represents one such change. The inability to scale clock frequency forced the industry to exploit the continued availability of transistors through the integration of multiple cores, each with a simpler architecture, to get double the performance at constant power with each technology generation. However, even this change is running out of steam. Moore's law would have allowed us to have 64 to 128 cores on the chip by now, but commercial chips have a more modest number of cores. Some of the reasons for this are

- the complexity of scaling cache-coherence mechanisms and shared memory architectures, in general;
- the challenge of programming in parallel and, hence, the limited consumer demand for parallel applications at the low-end;
- the attractive cost of smaller chips with a more modest number of cores; and
- the opportunity provided by the available real estate to integrate specialized architectures for high-volume functions instead of more cores.[1]

At the same time, Moore's law itself is slowing as we approach the 10-nm technology node, signaling the need for computer architectures beyond the dominant von Neumann model. The demand for increased capability on a chip will continue unabated, but at a reasonable price, especially with the new need to process massive amounts of data with high inherent parallelism. The window opens for creative computing architectures to meet this demand economically. Freed from the traditional von Neumann architecture with its high energy cost of moving data, processing data much closer to memory or storage, or even in memory or storage, can now realize its potential.

Communication between the processor and its various subsystems was clearly identified as the bottleneck in the 1990s, and PIM research emerged to address that problem, wrestling with the limits of traditional approaches such as instruction level parallelism, which have today fully played out. Indeed, McKee and Wulf identified the classic memory wall in 1995,[2] pointing out that as processor clock rates increase, the CPU will forever wait on memory, performing little or no useful work. The low utilization of modern CPUs demonstrates the correctness of this prediction, despite a flattening of clock rates in 2003. Today, the key problems with memory and storage subsystems are

- limited capabilities across relatively weak interfaces (memory or I/O), which provide insufficient bandwidth for data movement; and
- lack of concurrency of access across those interfaces, which impedes overall system throughput.

DDR-style memory interfaces are largely optimized for cost rather than for addressing these two problems. The traditional RAS/CAS architecture provides little opportunity to facilitate parallel access; memory banks typically exist in slave interfaces to improve the latency of access but not overall system throughput. Additionally, because of the bus architecture, increasing the number of DIMMs per memory channel typically results in a decrease in interface speed, forcing a tradeoff between capacity and performance. As an example, Micron's Hybrid Memory Cube (HMC) architecture addresses both of these challenges by moving to an abstracted serialized network interface and increasing the overall number of banks (or vaults in HMC parlance) available to the system. Storage systems suffer from similar problems.

Again, using DRAM as an example, there is typically a reduction in available bandwidth of six orders of magnitude between the sense amplifiers and the CPU edge. In addition, the cost of access in terms of energy increases from hundreds of femtojoules to tens of picojoules over a span of the same distance. When combined with the failure of processor architects to produce the same kinds of performance and energy improvements, the trend toward NDP will continue over the next decade.

Architectures such as STARAN or MPP by Goodyear Aerospace or that of Thinking Machines might get fresh scrutiny, not only from a parallel-processing perspective but also from a memory- and storage-centric perspective. Aspects of previously explored PIM architectures such as DiVA, the Terasys PIM Array, or IBM's Execube might be resurrected, but substantially integrated in silicon. These machines did not fade away because of lack of performance gains or energy efficiency, but simply because Moore's law allowed the basic von Neumann machines to outrun them with economic affordability.

The adoption of new architectures and the exploitation of their advantages in terms of power and performance will not be easy. The software ecosystem, regardless of any specific host CPU architecture and manufacturer, will be impacted. The burden will initially be on software engineers and computer scientists to make these new architectures programmable for application developers. Heterogeneous and distributed processing development environments must be made understandable and easy to use by the general programmer.

Architecturally, NDP is the right place at a system level to provide an overall increase in performance while simultaneously reducing power. With proper ecosystem enablement and continued economic demand for increased computational performance, and with the nature of silicon processing beginning to fundamentally change over the next decade, the time has come for nearer-to-memory processing to help extend the effect of Moore's law for several more years.

References

1. N. Hardavellas et al., "Toward Dark Silicon in Servers," *IEEE Micro*, vol. 31, no. 4, 2011, pp. 6-15.
2. W.A. Wulf and S.A. McKee, "Hitting the Memory Wall: Implications of the Obvious," *ACM SIGARCH Computer Architecture News*, vol. 23, no. 1, 1995, pp. 20-24.

Near-Data Computation: Looking Beyond Bandwidth

Moving computation off a system's main processors and into processors embedded in storage devices like SSDs can offer great improvements in bandwidth for data-centric computations, but the potential goes far beyond that:

- Avoiding data movement across a PCI Express or SATA bus saves power.
- Processors in the SSDs are vastly more efficient in terms of energy per operation than those in the host.
- Code running on the SSDs can be trusted, because it runs in an independent execution environment.
- Latency for accessing data from within the SSDs is much lower than it is from the host.

Latency improvements have obvious performance benefits, but leveraging the first three advantages can lead to significant gains in energy efficiency and reductions in latency for complex operations that require data-dependent storage accesses (for example, enforcing ordering or atomicity guarantees).

The software that implements these application-specific semantics makes modest computational demands, but moving it into the storage device can have a disproportionate impact on system performance. For instance, a recently proposed multipart, atomic, write mechanism reduces latency for transactional updates by up to 65 percent and nearly eliminates the bandwidth overheads that logging schemes incur.[1] Likewise, implementing portions of a key-value store on a processor in an SSD can improve throughput between 7.5 and 9.6 times.[2]

Leveraging trusted execution could provide further gains. A recently proposed storage interface that allows applications to bypass the operating system on storage accesses that do not modify file system metadata could provide even better performance if the file system could delegate simple metadata updates to software running in the SSD.[3]

To explore the potential of implementing application-specific semantics within the storage system, we implemented a prototype SSD that makes programmability the central abstraction for the storage interface.[4] Applications can download SSD applications to modify the device's behavior and add novel features. Our results show that defining SSD semantics can allow programmers to exploit all of the advantages of NDP that we have described. Our experience also shows that providing flexible programmability in the SSD helps programmers ensure that the new SSD functionality works seamlessly with the host-side application. Indeed, it is relatively easy to migrate portions of legacy programs to the SSD with minimal effort. For instance, we have embedded information about file system data structures in an SSD app, so that the SSD can take over common metadata updates. Because the SSD app uses the same data structures and algorithm as the original host-side code, the opportunity for errors is reduced, and the result is simpler host-side code and better performance owing to reduced I/O traffic to the SSD.

The performance gains that programmable, application-specific semantics can provide in a storage system can rival those approaches that exploit intra-SSD memory bandwidth. With careful design, near-data computing architectures can provide both programmable, extensible semantics as well as data-intensive computational off-load, ensuring the maximum benefit for the widest range of applications.

References

1. J. Coburn et al., "From ARIES to MARS: Transaction Support for Next-Generation Solid-State Drives," *Proc. 24th ACM Symp. Operating Systems Principles*, 2013, pp. 197-212.
2. A. De et al., "Minerva: Accelerating Data Analysis in Next-Generation SSDs," *Proc. IEEE 21st Int'l Symp. Field-Programmable Custom Computing Machines*, 2013, pp. 9-16.
3. A. Caulfield et al., "Providing Safe, User Space Access to Fast, Solid State Disks," *Proc. 17th Int'l Conf. Architectural Support for Programming Languages and Operating Systems*, 2012, pp. 387-400.
4. S. Seshadri et al., "Software-Defined Solid State Disks," to be published in *Proc. 5th Non-Volatile Memories Workshop*, 2014.

paradigms. The "Recent Work in Near-Data Processing" sidebar lists a few example papers and products to demonstrate this resurgence.

Research challenges in NDP

Although emerging technology presents several opportunities to implement NDP, many issues must be addressed and many problems solved before NDP can become a ubiquitous computing paradigm. At the hardware level, packaging and thermal constraints must be addressed, communication interfaces designed, synchronization mechanisms defined, and processing cores optimized for effective exploitation of NDP. NDP is not just a means to improve efficiency (cost or energy or latency), it is also an opportunity to design systems with fundamentally new capabilities—ones with important societal impact that will justify continued billion-dollar investments as those that have sustained the computing industry through the CMOS scaling era (see the "Everything Old is New Again: How Moore's

Top 10 Reasons Near-Data Processing Might Be Real This Time

Motivated by technology trends such as nonvolatile memories, die stacking, and dark silicon, and the rapid development and adoption of distributed big-data software such as MapReduce, researchers have started directing system architectures toward a new paradigm of resource-efficient, data-centric computing.[1] This approach is based on a few key design principles—namely, using NDP; specializing computation to accelerate data processing; redesigning the cache, memory, and storage hierarchy; and redesigning the software/hardware interface to enable cross-layer optimizations.

In proposing and evaluating NDP architectures, we must answer what makes these schemes different from earlier PIM designs, and why NDP might be real this time. The optimism toward NDP comes from recent studies that demonstrate its significant performance and energy-efficiency potentials. Our research in NDP has received positive feedback from colleagues and funding agencies, and has enabled discovery of potential use cases in the oil and gas industry, financial industry, databases and NoSQL systems, new storage systems, and graph and event processing. However, widespread adoption of NDP will require a system architecture roadmap backed by the larger R&D community. Users need early access to prototypes, programming, and tool-chain support, as well as a clear software migration path in anticipation of future NDP hardware. Wide adoption of the NDP paradigm hinges on solving these critical issues to address the software versus hardware cause-and-effect dilemma.

Here, we summarize the top 10 reasons for a revitalized NDP 2.0, based on insights gained from our collaborative research efforts.

1. *Necessity.* The renewed focus on efficiency and the increasing overheads of computing-centric architectures make NDP a promising alternative for the following reasons: moving computation close to data reduces data movement and cache hierarchy overhead; matching computation to data capacity, bandwidth, and locality needs enables the rebalance of computing-to-memory ratios; and specializing computation for the data transformation tasks further improves efficiency.
2. *Technology.* 3D and 2.5D die-stacking technologies have matured to enable the integration of computing and data stores (memory, storage, or unified) without the previous disadvantages of merged logic+memory fabrication. The close proximity of computation and data also enables high bandwidth at a low energy overhead. These two reasons make the strongest case for NDP.
3. *Software.* Distributed software frameworks such as MapReduce have popularized the concept of moving computation to data and smoothed the learning curve of programming NDP hardware. Such frameworks can also handle tough NDP software issues such as data layout, naming, scheduling, and fault tolerance.
4. *Interface.* NDP requires a host- and memory-decoupled interface that would have been impossible with today's DDRx standard. However, the dominance of a desktop- and server-based memory interface is likely to change because of two trends: mobile DRAM is rapidly replacing desktop and server DRAM as the new commodity memory, and there is already a proliferation of new memory interfaces, such as DDR4, LPDDRx, Wide I/O, HBM, and HMC. More importantly, new interfaces such as HMC have already included preliminary NDP support, such as smart

Law Continues to Drive the Future" sidebar). At higher levels of the system stack, existing programming models and runtimes must be adapted, or new ones developed, and big-data algorithms must be restructured with an awareness of the data's location. Security and programmability are looming as formidable issues in future large-scale computing, and NDP could hold the key to effective solutions in these areas (see the "Near-Data Computation: Looking Beyond Bandwidth" sidebar). NDP is also strongly influenced by the needs of different workload domains, accelerators at different levels of the hierarchy, and our ability to pull off hardware–software codesign (see the "Top 10 Reasons Near-Data Processing Might Be Real This Time" sidebar).

The four keynote talks at the Workshop on Near-Data Processing, summarized in three sidebars to this article, provided many compelling reasons to strongly consider NDP for future systems. "Everything Old is New Again: How Moore's Law Continues to Drive the Future" cites the difficulties of further technology scaling combined with recent advances in main memory technology that make near-memory processing worth revisiting, and argues that traditional architectures' bandwidth and energy constraints will push designs toward NDP in future systems. "Near-Data Computation: Looking Beyond Bandwidth" lists the varied advantages of processing data near storage devices such as solid-state drives. "Top 10 Reasons Near-Data Processing Might Be Real This Time" emphasizes the confluence of technology, workload, and IT ecosystem trends that could enable the wide adoption of NDP; it also provides a list of reasons to strongly consider NDP.

refresh and host-device decoupling. More sophisticated NDP protocols can be further developed by leveraging these basic services.

5. *Hierarchy.* New nonvolatile memories (NVMs) that combine memory-like performance with storage-like capacity enable a flattened memory/storage hierarchy and self-contained NDP computing elements. In essence, this flattened hierarchy eliminates the bottleneck of getting data on and off the NDP memory.
6. *Balance.* Within each NDP element, the computation-to-data bottleneck is removed by tight coupling, but communication could emerge as a new bottleneck between NDP elements. Fortunately, the balance can be regained through new system-on-a-chip (SoC) and die-stacking technologies that enable network-on-chip integration, a more efficient network software stack, and potentially new opportunities for NDP-customized interconnect designs.
7. *Heterogeneity.* NDP often involves heterogeneity for specialization and for the flexibility needed to support a wide range of workloads. This could have been a barrier, but recent advances in programming and in managing heterogeneity (as in GPU/APU, big.LITTLE, FPGA/SoC) are already clearing the road to NDP adoption.
8. *Capacity.* Advantages of using NVM in NDP are the much-larger device capacities and typically lower cost per bit—key factors in reducing system cost. A more fundamental, yet often overlooked, benefit of having larger capacity per NDP element is the reduction in system scale for a fixed dataset size. This is an advantage because smaller systems have lower failure rates and reduce software parallelization barriers. In contrast, early NDP designs were limited by small device capacities that forced too much fine-grained parallelism and interdevice data movement.
9. *Anchor workloads.* A commercially viable anchor market is critical for the adoption of a new technology. Codesigned big-data appliances appear to present an ideal market for introducing NDP. Indeed, IBM's Netezza and Oracle's Exadata are already marketing NDP products in limited forms.
10. *Ecosystem.* Prototypes, tools, and training are also critical elements for the adoption of NDP by nonexpert programmers. The confluence of several recent technologies is helping the emergence of such an ecosystem. Software programming models such as OpenMP4.0, OpenCL, and MapReduce, along with hardware prototypes by companies such as Adapteva, Micron, Vinray, and Samsung, can provide the basis for developers to build NDP apps.

We hope that reexamining such fundamental issues in today's context helps make the case for NDP and the need for more research in this direction.

Reference

1. P. Ranganathan, "From Microprocessors to Nanostores: Rethinking Data-Centric Systems," *Computer*, vol. 44, no. 1, 2011, pp. 39-48.

The six papers presented at the workshop provide new thoughts on the potential evolution of this field, and new results with evidence of related innovations and developments.

Locality of computation has always been a key aspect in designing algorithms to solve a problem on a computing system, and also in designing computing systems for solving a given problem. We have reached an inflection point where the nature of technology available to continue scaling the capabilities of computing systems has changed, while at the same time, the nature of problems that must be solved has also changed. Locality of computation in this new environment is likely to come from computing paradigms that distribute computing into the memory and storage hierarchy, closer to where the data reside. NDP is a paradigm that shows significant promise in helping to transport us into the new computing landscape. The Workshop on Near-Data Processing served to highlight the need for this transition, and broaden the interest to a larger community. MICRO

Acknowledgments

Some of this work was performed while Jichuan Chang was at HP Labs.

References

1. P. Kogge, "A Short History of PIM at Notre Dame," July 1999; www.cse.nd.edu/~pim/projects.html.
2. C.E. Kozyrakis et al., "Scalable Processors in the Billion Transistor Era: IRAM," *Computer*, vol. 30, no. 9, 1997, pp. 75-78.
3. T.L. Sterling and H.P. Zima, "Gilgamesh: A Multithreaded Processor-in-Memory Architecture

for Petaflops Computing," *Proc. ACM/IEEE Conf. Supercomputing*, 2002, pp. 1-23.

4. J. Draper et al., "The Architecture of the DIVA Processing-in-Memory Chip," *Proc. 16th Int'l Conf. Supercomputing*, 2002, pp. 14-25.
5. "Hybrid Memory Cube—A Revolution in Memory," *Micron*, www.micron.com/products/hybrid-memory-cube.
6. C. Ulmer et al., "Exploring Data Warehouse Appliances for Mesh Analysis Applications," *Proc. 43rd Hawaii Int'l Conf. System Sciences*, 2010, doi:10.1109/HICSS.2010.200.
7. D. Tewari et al., "Active Flash: Towards Energy-Efficient, In-Situ Data Analytics on Extreme-Scale Machines," *Proc. 11th USENIX Conf. File and Storage Technologies*, 2013, pp. 119-132.
8. J. Do et al., "Query Processing on Smart SSDs: Opportunities and Challenges," *Proc. ACM SIGMOD Int'l Conf. Management of Data*, 2013, pp. 1221-1230.
9. S. Cho et al., "Active Disk Meets Flash: A Case for Intelligent SSDs," *Proc. 27th Int'l Conf. Supercomputing*, 2013, pp. 91-102.

Rajeev Balasubramonian is an associate professor in the School of Computing at the University of Utah. His research focuses on memory systems and interconnects. Balasubramonian has a PhD in computer science from the University of Rochester. He is a member of IEEE and the ACM.

Jichuan Chang is a member of the Platforms Performance team at Google. His research interests include computer system architecture and memory systems, with a focus on co-designing cost- and energy-efficient platforms. Chang has a PhD in computer science from the University of Wisconsin-Madison.

Troy Manning is director of advanced memory systems in the DRAM Solutions Group at Micron. His research focuses on DRAM design and architecture and solid-state storage systems. Manning has a BS in electrical and computer engineering from the University of Iowa.

Jaime H. Moreno is senior manager of microprocessor architecture at the IBM Thomas J. Watson Research Center. His research focuses on microprocessor architecture and performance analysis topics, with an emphasis on emerging technologies. Moreno has a PhD in computer science from the University of California, Los Angeles. He is a senior member of IEEE and a member of the ACM and the IBM Academy of Technology.

Richard Murphy is a senior advanced memory systems architect for the DRAM Solutions Group at Micron. His research focuses on future memory platforms, research and development of computer architecture, advanced memory systems, and supercomputing systems for physics and data-intensive problems. Murphy has a PhD in computer science and engineering from the University of Notre Dame. He is a senior member of IEEE.

Ravi Nair is a researcher at the IBM Thomas J. Watson Research Center. His research interests include supercomputer design, processor microarchitecture, dynamic compilation, virtual machine technology, and approximate computing. Nair has a PhD in computer science from the University of Illinois at Urbana-Champaign. He is a member of the IBM Academy of Technology and a fellow of IEEE.

Steven Swanson is an associate professor in the Department of Computer Science and Engineering and the director of the Non-Volatile Systems Laboratory at the University of California, San Diego. His research interests include the systems, architecture, security, and reliability issues surrounding nonvolatile, solid-state memories. Swanson has a PhD in computer science and engineering from the University of Washington.

Direct questions and comments about this article to Rajeev Balasubramonian, 50 S. Central Campus Dr., Room 3190, Salt Lake City, UT 84112; rajeev@cs.utah.edu.