Opportunities for Nonvolatile Memory Systems in Extreme-Scale High Performance Computing

Jeffrey S. Vetter
Sparsh Mittal

Oak Ridge National Laboratory, USA

Abstract

For extreme-scale high performance computing systems, system-wide power consumption has been identified as one of the key constraints moving forward, where the DRAM main memory systems account for about 30-50% of a node's overall power consumption. Moreover, as the benefits of device scaling for DRAM memory slow, it will become increasingly difficult to keep memory capacities balanced with increasing computational rates offered by next-generation processors. However, a number of emerging memory technologies - nonvolatile memory (NVM) devices – are being investigated as an alternative for DRAM. Moving forward, these NVM devices may offer a number of solutions for HPC architectures. First, as the name, NVM, implies, these devices retain state without continuous power, which can, in turn, reduce power costs. Second, certain NVM devices can be as dense as DRAM, facilitating more memory capacity in the same physical volume. Finally, NVM, such as contemporary NAND flash memory, can be less expensive than DRAM in terms of cost per bit. Taken together, these benefits can provide opportunities for revolutionizing the design of extreme-scale HPC systems. Researchers are investigating how to integrate these emerging technologies into future extreme-scale HPC systems, and how to expose

these capabilities in the software stack and applications. Current results show a number of these strategies may offer high-bandwidth I/O, larger main memory capacities, persistent data structures, and new approaches for application resilience and output post-processing, such as transaction-based, incremental-checkpointing and in situ visualization, respectively.

Keywords: nonvolatile memory, high performance computing, supercomputing, DRAM, ReRam, PCM, STT-RAM, Flash.

1 Introduction

Currently, exascale computing systems are planned for 2020 through 2022 in Japan and the USA, respectively. Although the precise details of these systems are unknown, it is clear that these systems will need significant technological innovations on several fronts. The first and most pervasive of these challenges is that of managing power consumption [11] and energy-efficiency. To illustrate this point, consider that in 2010, the Jaguar supercomputer at the Oak Ridge National Laboratory operated at 0.25 Gigaflops/Watt. To meet the target goal of 20 Megawatts of aggregate system power, any 1 Exaflops supercomputer will require a ratio of roughly 50 Gigaflops/Watt: an improvement of two orders of magnitude in this decade! In this respect, the high performance computing community has already witnessed a number of trends over the past five years, foreshadowing these challenges. For example, the majority of the top 10 supercomputers on the TOP500 list since 2010 have used heterogeneous computing to increase their computational rates while keeping system wide power consumption and density lower than earlier estimates [1, 11].

One key contributing factor to system power consumption is a system's main memory [27]. Currently, DRAM main memory systems account for about 30-50% of a node's overall power consumption; this consumption is determined by several factors including memory capacity and configuration. In addition, experts predict three orders of magnitude increase in concurrency, and two orders of magnitude decrease in network interface bandwidth per core. Simply put, any attempt to limit the power consumption of a future system will necessarily constrain DRAM memory capacity. Based on current technology trends, some estimates forecast that an Exascale system would be limited to a main memory capacity of only 32-64 Petabytes, which is dramatically smaller than previous and contemporary systems.

From the technology perspective, most experts predict the end of device scaling for DRAM in the next decade (cf. Section 3.1). In fact, over the past five years, we have seen the trend for DRAM device scaling slow, and as it plateaus in the coming years, improvements in power, performance, and capacity will flatten.

These constraints on main memory capacity of future extreme-scale systems have several important implications, which are often subtle. Most importantly, a limited main memory capacity throws the system architecture out of balance, impacting other system parameters and efficiencies. That is, in most scientific applications, a smaller main memory per node reduces the amount of computation a node can execute without internode communication, increases the frequency of communication, and reduces the sizes of the respective messages communicated. All of these factors are known to lower application efficiencies; this relatively small memory capacity will be efficient for only the most computationally intense workloads (e.g., large matrix factorization, naïve pairwise molecular dynamics).

In this regard, recent trends in the research, development, and manufacturing of nonvolatile memory (NVM) technologies provide optimism that these technologies can be used to reverse this trend toward very small main memory capacities in these future systems. Research in new devices, such as Phase Change RAM (PCRAM) and resistive RAM (ReRAM), and innovative integration strategies could provide alternative memory designs that have broad impact on future extreme-scale architectures and applications.

2 Memory Hierarchies in Extreme-Scale HPC Systems

To lay the groundwork for our discussion, Figure 1 illustrates a notional diagram of a general node architecture in a future extreme-sale HPC system; it represents a design point, taking into account the trajectory of numerous technologies, such as heterogeneous computing [29]. In this architecture, we expect to have multiple types of processing cores including latency tolerant (LT) cores and throughput (TP) cores: similar to today's general purpose X86 cores and lightweight, highly multithreaded GPU cores, respectively [8]. Furthermore, we expect these cores to be augmented with specialized hardware to improve performance for particular functionality, such as encryption, random number generation, or other capabilities. The balance of the quantities of these cores and specialized hardware is clearly workload dependent and an open research question. All of these cores will be connected to a Network-on-a-Chip (NOC) that provides communication among the cores, memories, and external networks. The NOC will provide an interface to other nodes through the Network Interface (NI).

The memory controller (MC) connects the cores, and potentially the NI, to the local memory, whether the memory is directly integrated into the package or some off package device like Micron's Hybrid Memory Cube [23]. The integrated package may be stacked in three dimensions with memory and logic (3-D stacking) connected with through-silicon vias (TSVs), a multi-chip module (MCM), or a silicon interposer (2.5-D stacking) [10]. More importantly, we expect the node to contain several different types of memory device technologies, and, perhaps, even made available to applications through a unified address space. These memory technologies could include commodity double data rate synchronous dynamic random-access (DDR SDRAM) memory with varying levels of error detection and correction (e.g., SECDED, Chipkill), specialized memories like Graphics DDR (GDDR) and High Bandwidth Memory (HBM), or nonvolatile memory (NVM) technologies. Clearly, the proportion and specific configurations of these memories is workload and architecture dependent, and an open research question.

Unequivocally, this configuration of cores will represent an increasing amount of computational throughput, requiring a significant amount of local memory capacity and bandwidth to allow efficient use of these cores without stalling on remote accesses (over the NI or to other NUMA partitions). Given the significant and increasing differences in latency between local and remote accesses to memory, it will be imperative to have a large memory capacity on the node in order to keep these cores busy.

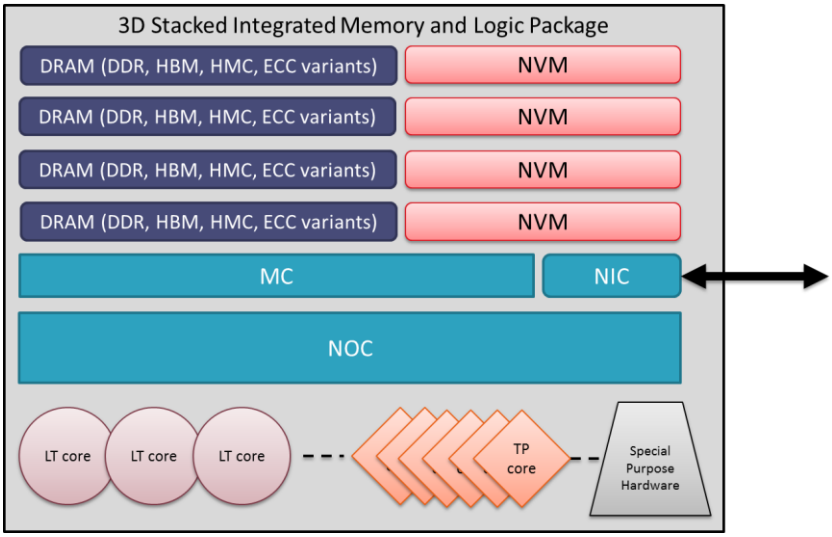

Figure 1: Notional diagram of node architecture for future extreme-scale HPC system.

Notably lacking from our diagram is local disk storage. An interesting trend over the past decade has been the removal of hard-disk drives from the node designs of large scale supercomputers. Virtually none of today's large scale systems have mechanical hard disks physically present in the node. This action was

prompted by the failure rates of HDDs and complexity of system management and maintenance [26]. Rather, contemporary systems forward all parallel I/O requests over the interconnection network to specialized nodes that connect directly to a storage area network, typically using Infiniband links and commodity storage targets to retrieve or store the data as requested by the application. In some systems, a small amount of main memory can be reserved as a RAM Disk for the operating system and application use. An important consequence of this configuration is that these systems cannot support any demand paging of virtual memory as is typical in most other computing systems. As such, HPC applications are designed so that all application data explicitly fits well within the size of main memory capacity, while allowing space for other system functionality, such as those needed for the operating system and message passing runtime systems.

3 Memory Technology Overview

The memory hierarchy, from the registers of a processor core out to the blocks on a disk or tape drive, is a fundamental component in every computing system, be it mobile, enterprise, or high performance computing. However, the memory technologies used within and across these systems vary dramatically due to the requirements and uses demanded by the workload and other system configuration parameters, such as energy efficiency and reliability.

Given the significance of memory technology, the International Technology Roadmap for Semiconductors (ITRS) (cf. http://public.itrs.org) and research community have been monitoring the roadmaps for memory technologies carefully. This roadmap includes widely-deployed, existing technologies, such as DRAM, SRAM, and NAND Flash, to newer, emerging technologies including Phase-change RAM (PCRAM), metal-oxide resistive RAM (RRAM) (c.f. memristor), and spin-torque transfer RAM (STT-RAM), molecular memory, FeFET memory, carbon-based memory [12], and others. While many of these emerging technologies are not in volume production now, researchers and vendors are aggressively pursuing them.

Since a full description of all of these technologies is beyond the scope of this paper, we limit our discussion to several prominent nonvolatile memories: Flash, PCRAM, ReRAM, and STT-RAM. ([12] and [20] provide more thorough descriptions.) Table 1 outlines these technologies along several important dimensions. For more details, we refer readers to ITRS and other comprehensive device surveys, such as [12]. Note that even these surveys may omit important information pertinent to this comparison, such as the performance and power costs of peripheral circuits, and manufacturing and operational data, like failure rates.

Table 1: Overview Comparison of Emerging Memory Technologies.

	HDD	DRAM	NAND SLC Flash	PCRAM SLC	STTRAM	ReRAM
Data retention	Y	N	Y	Y	Y	Y
Cell Size	N/A	6-10F^2	4-6F^2	4-12F^2	6-50F^2	4-10F^2
Access Granularity (B)	512	64	4192	64	64	64
Endurance	$>10^{15}$	$>10^{15}$	10^4-10^5	10^8-10^9	$>10^{15}$	10^{11}
Read Latency	5ms	50ns	25us	50ns	10ns	10ns
Write Latency	5ms	50ns	500us	500ns	50ns	50ns
Standby Power	Disk access mechanisms	Refresh	N	N	N	N

3.1 DRAM

To set a baseline for our discussion, we describe DRAM. For nearly four decades, dynamic access random memory has provided the primary means of providing main memory in computer architectures

[6]. DRAM technology underpins nearly every computer, ranging from mobile phones to servers in data centers to extreme-scale HPC systems. This pervasive use has led to huge commodity markets, with recent estimated annual, global revenues of $30B to $40B. Over this period, DRAM technology has undergone dramatic scaling improvements in capacity and performance, while remaining inexpensive, plentiful, and reliable.

Yet, many experts predict these scaling trends to plateau over the next five to ten years. In fact, these improvements started slowing several years ago. The key consideration is in the design and fabrication of the DRAM cell per se. The basic DRAM cell consists of one transistor and one capacitor. The capacitor must be large enough for reliable sensing; however, as feature sizes shrink, capacitors become increasingly difficult to fabricate. Also, due to the need for periodic refreshing of the cells, DRAM cells and peripheral circuits consume power even when they are not storing useful application data or being accessed by the processing cores.

Additionally, unforeseen problems with DRAM (as well as other memories) are causing even greater concern. These problems include challenges like cell interference (or disturbances [34]), and developing high-volume manufacturing processes that can produce memory cells within reliable tolerances.

Viewed in this light, two possible solutions seem evident. First, the DRAM architecture could be redesigned to mitigate these device-level constraints [5, 17, 28]. For DRAM, there are possible improvements from redesigning and optimizing DRAM protocols, moving DRAM closer to processors, and improved manufacturing processes. In fact, this integration of memory onto the package in future systems may provide for performance and power benefits of about one order of magnitude [5].

Second, emerging memory technologies with different characteristics could replace or complement DRAM [13, 15, 19, 24]. In fact, interest in memory and storage technologies [12] has grown over the past decade, in anticipation of these limits on DRAM, and for the use in enterprise and mobile systems markets.

3.2 Nonvolatile memory

Nonvolatile, solid-state memory devices (NVMs) exist across a range of maturities. NVMs, such as NAND flash memory, are already in widespread use and deployment. Other NVMs, namely phase-change memory (PCRAM), spin-torque transfer memory (STTRAM), and resistive RAM (RRAM), are being aggressively investigated for potential uses in future computing systems. These new memory devices provide the non-volatility of conventional disks (HDDs) while providing performance approaching that of DRAM. These technologies are no panacea, however. Over the next decade, as researchers and industry gain experience with these various devices, the community will identify shortcomings (and potential solutions) for these new devices in terms of power, cost, reliability, and manufacturability that will influence their adoption. Note that nonvolatile memory devices are certainly not new. In fact, early supercomputers, such as the Cray EL92, offered nonvolatile memory technology, but it was typically built for a specific workload, had low density, and was not designed for energy efficiency.

3.2.1 NAND Flash memory

NAND flash memory is the most pervasive form of nonvolatile memory today; it is used in most mobile devices including laptop computers, cameras, smartphones, flash sticks, and elsewhere. Moreover, flash memory has been injected transparently to most computing systems in the form of solid state drives (SSDs): a replacement for HDDs using the same block-oriented device interfaces. These new SSDs show much better performance than HDDs on many workloads that have frequent, irregular I/O operations. Currently, a range of technologies, such as the FusionIO's PCIe card, are capable of providing huge amounts of memory to a node, directly off of the PCIe interface, and bypassing traditional filesystems interfaces, such as POSIX. These benefits propelled sales of flash memory to exceed DRAM sales in 2012. Currently, the cost per bit of flash memory is lower than DRAM and this trend is predicted to continue.

NAND flash memory is typically formed by creating a memory cell using a floating-gate MOSFET transistor, which allows easy manufacturing and high density. Since the transistor is 'floating,' it remains set to its written state for a long period of time. However, the act of changing the state degrades the transistor, and over time, many writes can cause the flash memory cells to fail permanently. Techniques, such as advanced error correction codes and wear leveling, have been designed to mitigate these endurance challenges. In fact, wear-leveling has become a very complex and intricate research area, where researchers seek to balance reliability, power, and performance simultaneously.

Manufacturers continue to develop and introduce increasingly dense flash memory chips based on new processes and designs [2, 7], such as multi-level cells (MLCs). Experts expect the effective scaling and density of flash memory to continue for several more years, but they predict a plateau at a feature size of 15nm, and before the end of the decade.

3.2.2 Emerging Nonvolatile Memories

Aside from flash memory, several other forms of nonvolatile memories are emerging: phase-change memory (PCRAM) [15, 24], spin-torque transfer memory (STTRAM) [13], resistive RAM (RRAM) [33], and carbon-based nanotube memory [6, 12]. (For the purposes of this article, we label these Emerging Nonvolatile Memories or ENVMs). In addition to persistence, these other nonvolatile memory technologies have some possible advantages over contemporary flash memory as shown in Table 1. One of the key disadvantages of flash memory is its poor endurance. If any of these new technologies could improve on this issue, then they could be integrated into higher levels of the memory hierarchy more readily, and also avoid the high overheads of wear-leveling techniques and block-oriented interfaces. Not surprisingly, these emerging technologies are in various forms of maturity and they could face many hurdles to being deployed in a production system.

Although a comprehensive description of these technologies can be found elsewhere [6, 12], we highlight some of the common traits of these technologies. First, all of ENVMs have better read performance than write performance in terms of both power and access latencies. Second, write operations distress the cells of ENVMs more than read operations in terms of endurance; however, initial evidence shows that the endurance of ENVMs is better than the endurance of flash memory. Finally, most ENVMs use power only when they are being accessed, and they use much less, if any, power when idle.

3.2.3 Nonvolatile Memory Summary

Simply put, the benefit of NVM is its lack of a need for energy to keep memory cells refreshed, whether or not any particular memory cell is actively used by an application during execution. Better still, the persistence of NVM may allow applications to store checkpoints and other datasets locally without any costs for static power for extended periods of time. However, given the device-level costs in terms of the often asymmetric latencies and energies for read and writes, the application usage scenarios for this memory will determine if such configurations is indeed performance and energy efficient for future extreme-scale systems.

Although it is now difficult to predict which of these ENVMs will ultimately dominate the market, the community continues gaining experience with the manufacturability, cost, performance, and reliability of ENVMs, and continues to propose practical solutions with these technologies.

4 Opportunities for NVM in HPC Systems

As mentioned in Section 1, the amount of main memory available on a node in future extreme scale systems is expected to be very limited given the predictions for traditional DRAM; this limitation has forced the community to explore alternative technologies. As the community gains experience with these alternatives, the applications, programming systems, and system architectures must adapt to gain the full benefits of these NVM.

As such, two questions become important for extreme scale HPC systems. First, how should architects integrate NVM devices into the node architecture? Second, how should applications, programming systems, and system software manage this new capability? Clearly, the two questions are interrelated. For example, if NVM is a simple replacement for a HDD as a SSD, how will the operating system need to change to offer higher performance beyond the typical POSIX I/O interface? In another scenario, if NVM is attached to the main memory hierarchy, how should applications make use of this persistent memory, and how can the applications or intervening software prevent hotspots that may cause reliability problems? Of course, more interesting problems emerge with data stored in NVM; these include topics such as ensuring security and access controls for the persistent data, and resurrecting corrupted data structures.

4.1 Architectural Integration of NVM

Not surprisingly, it is possible to integrate NVM into one or more levels of the memory hierarchy of a HPC compute node. However, the integration should account for the advantages and disadvantages of NVM. For example, the write endurance of most NVM devices is short when compared to DRAM, so the design should avoid those scenarios that may have excessive writing to a hotspot location. Nevertheless, the best integration strategy remains an open, active research question, and the technical feasibility of these strategies must be balanced against the requirements of the workload, and the costs to manufacturing and operating the systems. The research community has already started exploring these potential opportunities for NVM in HPC systems (as well as other architectures, such as mobile devices). While the scope of these possibilities is large and continues to increase, we refer the interested reader to the comprehensive survey [20] for complete details. Here, we illustrate a sampling of these possibilities.

Returning to our node description in Section 2, we start with the most common and straightforward integration of NVM into a node as a HDD replacement, and then move up the storage hierarchy to the processor caches.

4.1.1 Solid State Disks (SSDs)

The first and most predictable method to introduce NVM into HPC nodes is as solid state disks with standard I/O and device interfaces: POSIX and SATA, respectively. Because applications and operating systems already support block-I/O interfaces, the NVM device drivers can simply emulate the standard protocols, and hide any complexity of NVM. Currently, SSDs are widely available and inexpensive; a 256GB SSD based on Flash memory sells for approximately $100 or 2.5 GB/$. Even as a straightforward replacement, SSDs have demonstrated a significant I/O performance improvements for most systems, especially for workloads dominated by many, small IO operations (IOPs). These benefits have led to SSDs being deployed widely in laptop, enterprise, and HPC systems [29], even though magnetic, mechanical hard-disk drives continue to be denser and cheaper than their SSD counterparts. In HPC, SSDs have found their way into HPC systems through the metadata servers for HPC filesystems.

4.1.2 PCIe Integration

Moving up the storage hierarchy, the next possible integration point is the PCIe expansion slots. PCIe is a standardized interface that supports thousands of possible devices including network interfaces, disk drives, GPUs, FPGAs, etc. PCIe has many benefits including field upgradability, interoperability, and standardized device driver development.

The benefit of integrating NVM into the system via PCIe is that the device interface to NVM can be specialized for specific operations (as opposed to a standard filesystem interface), and it can exploit the full bandwidth of PCIe. In addition, the design of these expansion cards can be specialized to perform very high transaction rates and optimized for mixed workloads. Several options exist today; these include appliances from FusionIO, Violin Memory, and Texas Memory. For example, high-end, contemporary appliances offer over 5TB of usable flash memory storage in a standard height, half-length card, while

consuming less than 25W. Many of these devices provide extremely high IOPS (e.g., 500,000 or higher) relative to the performance of traditional HDDs.

Despite the flexibility offered by PCIe expansion cards, the latency and bandwidth offered by these devices remain an order of magnitude or more less than that provided by traditional main memory systems.

4.1.3 Hybrid Memory Hierarchies

A more aggressive approach for using NVM is to directly integrate it into the node's memory hierarchy. This integration could be as another level of memory, or a combination of memory with DRAM and NVM. This method could provide more memory capacity to applications in the same power envelope, and much lower latency to and fine-grained control of NVM than by using I/O interfaces (as mentioned above). However, this memory would need to be managed very carefully due to its endurance challenges; the memory system would need to guard against processor cores creating hotspots in these NVM hierarchies. Indeed, the memory traffic generated by today's cores could quickly exhaust the endurance of any given NVM cell.

The two primary components of today's node memory hierarchies are caches and main memory (e.g., DRAM). NVM could be integrated into these components either horizontally or vertically. Adding a layer of NVM vertically would replace a level of the existing memory hierarchy or add a new level, say to back a large DRAM cache [19]. Adding NVM horizontally, on the other hand, would offer both types of memory through possibly a uniform interface, hiding the distinct details of accessing each type of memory device. In both cases, the processor, memory controller, and software stack, including the application, may have to be aware of the differences in the memory system to manage performance, hotspots, and possibly endurance; however, this type of awareness is common in today's systems, for, say, managing NUMA memory hierarchies or programming heterogeneous systems.

Adding NVM to supplement DRAM memory of a node's main memory seems like a straightforward and highly likely possibility [15]. The main benefits of this approach is that processors and node design may not have to change dramatically if the NVM interface follows existing memory interface protocols. In fact, companies like MICRON and Hynix have created such NVM DIMMs. This approach has the benefit that memory capacities and ratios are field configurable. However, as mention earlier, the software and applications need to avoid creating performance or endurance problems with their resulting access patterns. In some cases, the memory system may be able to manage the migration of pages with the suitable characteristics to NVM [25, 31], while in other cases, applications may be able to place their data structures directly in the most appropriate memory regions [3, 16]. For example, researchers studying extreme-scale scientific applications found a number of common use cases for scientific data structures in applications that effectively match the characteristics of NVM [16]. In fact, using binary instrumentation, the researchers identified large numbers of data structures of these applications appeared to be dominated by reads (30-40% of all data structures in several applications), which are a good match for a hybrid, byte-addressable NVM. These scientific data structures have common uses: lookup, index, and permutation tables, inverted and 'element-lagged' mass matrices, geometry arrays for grids, boundary condition data, and constants for transforms and interpolation, while others have application specific data structures like thermal conductivity for soils, and strain and conductivity rates.

Lastly, a NVM technology could be combined with SRAM or eDRAM to create a hybrid cache. This configuration could help reduce the large power and area requirements of SRAM; however, the memory system would need to use clever new algorithms to prevent the processor from creating either performance or endurance hotspots in the NVM technology, which, if left unaddressed, could lead to a very short cell lifetime. Such algorithms are currently being investigated [21, 32], but numerous manufacturing and deployment hurdles remain.

4.1.4 Global NVM Storage Devices

Finally, another interesting option for NVM to find its way into large scale HPC and enterprise systems is the option of building NVM memory servers that can be included, scaled, and accessed globally from all the nodes in the system over the interconnection network. These memory servers could allow remote access to the NVM via several types of interfaces including a key value store, a file, a global memory space using traditional GAS semantics, or a mapped remote memory region. Although these devices are just emerging, they can be very effective for specific workloads, such as those in the enterprise market (e.g., MemcacheD, MongoDB), where datasets are unstructured and difficult to partition, and access patterns are unpredictable and require consistency.

4.2 Functional Integration of NVM

Given this range of architectural options, researchers are investigating and proposing application and software solutions to explore the pros and cons of each integration strategy while optimizing for NVM's strengths and avoiding NVM's limitations [9, 20, 24, 25]. Additional research focuses on redefining system interfaces or operating primitives [3, 4, 20, 22] to accommodate NVM. For example, appliactions may have to use new I/O interfaces to access NVM, requiring a departure from POSIX I/O. Much of the work currently underway is exploring NVM's potential impact on applications, programming systems, and system software. It is critical to demonstrate these benefits on large-scale mission critical scientific applications. In some scenarios, like substituting a SSD for a HDD, the application will see a performance improvement without changes. However, in other cases, if NVM is integrated into the architecture as a peer or replacement for main memory, applications and programming systems will need to be redesigned to benefit fully from this integration. If they are not redesigned, they may still reap a benefit, but it may be suboptimal. Here, we illustrate some more creative functional strategies for exploiting NVM in extreme-scale HPC systems.

4.2.1 Parallel I/O Burst Buffers

Conceivably, the first use of NVM memory in extreme scale HPC systems will be as burst buffers for parallel I/O. The idea of burst buffers is not new as it is commonly used in areas such as video processing, but rather it is a timely idea for extreme-scale HPC architectures [18]. As was mentioned in Section 2, current extreme-scale HPC architectures do not include local HDD storage. Instead, these systems offload all I/O activity over their interconnection network to I/O nodes. These I/O nodes then access data on SANs. The weakness of this configuration is that the I/O of the entire system is funneled into a small number of I/O nodes and disks, leading to congestion and often poor and unpredictable performance.

With NVM in HPC nodes, applications could quickly dump a checkpoint or other output data (e.g., for visualizations) onto the NVM in each node (i.e., burst), and proceed with the application computation using the freed DRAM. Meanwhile, the data stored in each node's NVM (i.e., buffer) is concurrently drained onto the traditional HPC storage infrastructure (mentioned above). The benefit of this approach is the fact that the application can continue to compute (and not stall) as data is drained to the HPC SAN, and that this I/O model does not require major modifications to applications or their respective I/O frameworks. In a node configuration including NVM, NVM could serve as a burst buffer using any number of integration strategies including as a SSD in the node, added as an expansion card (e.g., PCI-Express), or, perhaps, integrated directly with the memory hierarchy. The I/O software stack could be minimally adapted so that many of these changes could be hidden from the application.

4.2.2 Persistent Data Structures

An important benefit of NVM is that its data persists even after the application terminates (or crashes). This trait can be valuable in several ways. First, as said earlier, if data is stored in NVM, then the system can save power. Second, since NVM can be more considerably more plentiful in a node than DRAM, the data stored in these persistent data structures can live between invocations of the application,

or post-processing of the data from an application, thereby reducing pressure on the systems' interconnection network and parallel I/O subsystem. If proper access controls are provided, application data, such as materials tables or grids, can be stored on the node indefinitely for use by requisite applications, that open them as necessary. However, this persistence can also require careful mechanisms for preserving and maintaining this state. Said another way, if NVM contains data that was corrupted by an earlier application crash, it may prove impossible to use by later applications, similar to a filesystem.

In this regard, a number of researchers have developed systems to provide software interfaces to this NVM memory, so that its state is managed properly and consistently in the case of any failure. In one example, NV-heaps [3] and Mnemosyne [30] provides a library and runtime system to manage changes to NVM as transactions. This interface requires that users specifically declare data and pointers to be stored in NVM, and then use transaction semantics to invoke multiple changes to these data structures. The proper strategy for balancing the programming complexity against the performance and endurance of NVM remains an open research question.

4.2.3 Improving application reliability

As discussed earlier, one of the major changes in extreme-scale system architectures over the past decade has been the fact that nodes to not have local disk storage, but rather all I/O must traverse the interconnection network and the attached SAN. This is a critical limitation for the traditional checkpoint/restart model of maintaining application integrity and reliability. In this model of checkpointing application state and restarting when necessary, this architectural trend means that much of the system's memory must be flushed to disk hourly.

Rather than use the burst buffer approach mentioned earlier, applications could save and restore versions of their state incrementally locally on NVM as they progress, restoring to earlier versions as deemed necessary. This strategy has the benefit that the local NVM has considerably higher bandwidth, thus consuming less of the application's valuable execution time. This incremental checkpointing strategy has been also adapted in mobile and transiently-powered systems [20]. The only limitation of this strategy is a catastrophic node failure where node hardware cannot restart, and prevents access to the application's local checkpoint versions from other nodes.

4.2.4 In-situ visualization and post-processing

In the same way that system-wide defensive checkpoints must be funneled to I/O nodes in contemporary systems, so does productive I/O for visualization, movie creation, or feature detection [14]. Rather than flush the data to an external filesystem, applications could write time-series scientific data to NVM, and then pause periodically to use the node's processors for in-situ post-processing. That is, the data would remain on the node, and the processors would post-process the application data that it just created with its simulation, coordinating with other nodes as needed. Although this usage scenario for NVM is new, it is expected to gain in importance as systems grow more constrained by limited parallel I/O subsystems.

5 Summary

For extreme-scale high performance computing systems, system-wide power consumption has been identified as one of the key constraints moving forward, with a significant contribution to power consumption identified as main memory. In this article, we review a number of emerging memory technologies - nonvolatile memory (NVM) devices – that are being investigated as an alternative for DRAM. Moving forward, these NVM memory systems may offer a number of solutions for HPC architectures. The benefits of NVM when properly balanced with its limitations can provide opportunities for revolutionizing the design extreme-scale HPC systems. Researchers are investigating how to integrate these new technologies into future extreme-scale HPC systems, and how to expose these capabilities to their applications. Current results show a number of these strategies may provide more main memory

capacity at the same or reduced power costs, offer higher bandwidth I/O, and new opportunities for application resilience and output post-processing, such as in situ visualization and incremental checkpointing.

Acknowledgements

This manuscript has been authored by UT-Battelle, LLC under Contract No. DE-AC05-00OR22725 with the U.S. Department of Energy. The United States Government retains and the publisher, by accepting the article for publication, acknowledges that the United States Government retains a non-exclusive, paid-up, irrevocable, world-wide license to publish or reproduce the published form of this manuscript, or allow others to do so, for United States Government purposes. The Department of Energy will provide public access to these results of federally sponsored research in accordance with the DOE Public Access Plan (http://energy.gov/downloads/doe-public-access-plan). This research is sponsored by the Office of Advanced Scientific Computing Research in the U.S. Department of Energy.

References

[1] J. Ang, K. Bergman, S. Borkar, W. Carlson, L. Carrington, G. Chiu, R. Colwell, W. Dally, J. Dongarra, A. Geist, G. Grider, R. Haring, J. Hittinger, A. Hoisie, D. Klein, P. Kogge, R. Lethin, R. Lucas, V. Sarkar, R. Schreiber, J. Shalf, T. Sterling, and R. Stevens, "Top Ten Exascale Research Challenges," DOE Office of Science, Advanced Scientific Computing Advisory Committee, Subcommittee for the Top Ten Exascale Research Challenges 2014, http://science.energy.gov/~/media/ascr/ascac/pdf/meetings/20140210/Top10reportFEB14.pdf,

[2] S. Choi, D. Kim, S. Choi, B. Kim, S. Jung, K. Chun, N. Kim, W. Lee, T. Shin, and H. Jin, "19.2 A 93.4 mm 2 64Gb MLC NAND-flash memory with 16nm CMOS technology," Proc. Solid-State Circuits Conference Digest of Technical Papers (ISSCC), 2014 IEEE International, 2014, pp. 328-9,

[3] J. Coburn, A.M. Caulfield, A. Akel, L.M. Grupp, R.K. Gupta, R. Jhala, and S. Swanson, "NV-Heaps: making persistent objects fast and safe with next-generation, non-volatile memories," in *Proceedings of the sixteenth international conference on Architectural support for programming languages and operating systems*. Newport Beach, California, USA: ACM, 2011, pp. 105-18, 10.1145/1950365.1950380.

[4] J. Condit, E.B. Nightingale, C. Frost, E. Ipek, B. Lee, D. Burger, and D. Coetzee, "Better I/O through byte-addressable, persistent memory," in *ACM SIGOPS 22nd symposium on Operating systems principles (SOSP)*. Big Sky, Montana, USA: ACM, 2009, pp. 133-46, 10.1145/1629575.1629589.

[5] B. Giridhar, M. Cieslak, D. Duggal, R.G. Dreslinski, R. Patti, B. Hold, C. Chakrabarti, T. Mudge, and D. Blaauw, "Exploring DRAM Organizations for Energy-Efficient and Resilient Exascale Memories," in *SC13*. Denver: ACM/IEEE, 2013

[6] B. Jacob, S. Ng, and D. Wang, *Memory systems: cache, DRAM, disk*: Morgan Kaufmann, 2010.

[7] K. Kanda, N. Shibata, T. Hisada, K. Isobe, M. Sato, Y. Shimizu, T. Shimizu, T. Sugimoto, T. Kobayashi, and N. Kanagawa, "A 19 nm 112.8 mm 64 Gb Multi-Level Flash Memory With 400 Mbit/sec/pin 1.8 V Toggle Mode Interface," *Solid-State Circuits, IEEE Journal of*, 48(1):159-67, 2013,

[8] S.W. Keckler, W.J. Dally, B. Khailany, M. Garland, and D. Glasco, "GPUs and the future of parallel computing," *IEEE Micro*, 31(5):7-17, 2011,

[9] D. Kim, S. Lee, J. Chung, D.H. Kim, D.H. Woo, S. Yoo, and S. Lee, "Hybrid DRAM/PRAM-based main memory for single-chip CPU/GPU," in *Proceedings of the 49th Annual Design Automation Conference*. San Francisco, California: ACM, 2012, pp. 888-96, 10.1145/2228360.2228519.

[10] J. Knickerbocker, P. Andry, E. Colgan, B. Dang, T. Dickson, X. Gu, C. Haymes, C. Jahnes, Y. Liu, and J. Maria, "2.5 D and 3D technology challenges and test vehicle demonstrations," Proc. Electronic Components and Technology Conference (ECTC), 2012 IEEE 62nd, 2012, pp. 1068-76,

[11] P. Kogge, K. Bergman, S. Borkar, D. Campbell, W. Carlson, W. Dally, M. Denneau, P. Franzon, W. Harrod, K. Hill, J. Hiller, S. Karp, S. Keckler, D. Klein, R. Lucas, M. Richards, A. Scarpelli, S. Scott, A. Snavely, T. Sterling, R.S. Williams, and K. Yelick, "ExaScale Computing Study: Technology Challenges in Achieving Exascale Systems," DARPA Information Processing Techniques Office 2008,

[12] M.H. Kryder and K. Chang Soo, "After Hard Drives: What Comes Next?," *IEEE Transactions on Magnetics*, 45(10):3406-13, 2009, http://dx.doi.org/10.1109/TMAG.2009.2024163.

[13] E. Kultursay, M. Kandemir, A. Sivasubramaniam, and O. Mutlu, "Evaluating STT-RAM as an energy-efficient main memory alternative," Proc. Performance Analysis of Systems and Software (ISPASS), 2013 IEEE International Symposium on, 2013, pp. 256-67,

[14] M. Kwan-Liu, "In Situ Visualization at Extreme Scale: Challenges and Opportunities," *Computer Graphics and Applications, IEEE*, 29(6):14-9, 2009, 10.1109/MCG.2009.120.

[15] B.C. Lee, E. Ipek, O. Mutlu, and D. Burger, "Architecting phase change memory as a scalable dram alternative," in *36th annual international symposium on Computer architecture*. Austin, TX, USA: ACM, 2009, pp. 2-13, 10.1145/1555754.1555758.

[16] D. Li, J.S. Vetter, G. Marin, C. McCurdy, C. Cira, Z. Liu, and W. Yu, "Identifying Opportunities for Byte-Addressable Non-Volatile Memory in Extreme-Scale Scientific Applications," in *IEEE International Parallel and Distributed Processing Symposium (IPDPS)*. Shanghai: IEEEE, 2012

[17] J. Liu, B. Jaiyen, Y. Kim, C. Wilkerson, and O. Mutlu, "An experimental study of data retention behavior in modern DRAM devices: implications for retention time profiling mechanisms," in *Proceedings of the 40th Annual International Symposium on Computer Architecture*. Tel-Aviv, Israel: ACM, 2013, pp. 60-71, 10.1145/2485922.2485928.

[18] N. Liu, J. Cope, P. Carns, C. Carothers, R. Ross, G. Grider, A. Crume, and C. Maltzahn, "On the role of burst buffers in leadership-class storage systems," Proc. IEEE 28th Symposium on Mass Storage Systems and Technologies (MSST), 2012, pp. 1-11,

[19] J. Meza, J. Chang, H. Yoon, O. Mutlu, and P. Ranganathan, "Enabling efficient and scalable hybrid memories using fine-granularity DRAM cache management," *Computer Architecture Letters*, 11(2):61-4, 2012,

[20] S. Mittal and J.S. Vetter, "A Survey of Software Techniques for Using Non-Volatile Memories for Storage and Main Memory Systems," Oak Ridge National Laboratory, Technical Report ORNL/TM-2014/633, 2014,

[21] S. Mittal, J.S. Vetter, and D. Li, "LastingNVCache: A Technique for Improving the Lifetime of Non-volatile Caches," Proc. IEEE Computer Society Annual Symposium on VLSI (ISVLSI), 2014,

[22] X. Ouyang, D. Nellans, R. Wipfel, D. Flynn, and D.K. Panda, "Beyond block I/O: Rethinking traditional storage primitives," Proc. High Performance Computer Architecture (HPCA), 2011 IEEE 17th International Symposium on, 2011, pp. 301-11,

[23] J.T. Pawlowski, "Hybrid memory cube (HMC)," Proc. Hotchips, 2011, pp. 1-24,

[24] M.K. Qureshi, V. Srinivasan, and J.A. Rivers, "Scalable high performance main memory system using phase-change memory technology," in *Proceedings of the 36th annual international symposium on Computer architecture*. Austin, TX, USA: ACM, 2009, pp. 24-33, 10.1145/1555754.1555760.

[25] L.E. Ramos, E. Gorbatov, and R. Bianchini, "Page placement in hybrid memory systems," in *International Conference on Supercomputing*. Tucson, Arizona, USA: ACM, 2011, pp. 85-95, 10.1145/1995896.1995911.

[26] B. Schroeder and G.A. Gibson, "A large-scale study of failures in high-performance computing systems," in *Proceedings of the International Conference on Dependable Systems and Networks*: IEEE Computer Society, 2006, pp. 249-58, 10.1109/dsn.2006.5.

[27] M.E. Tolentino, J. Turner, and K.W. Cameron, "Memory MISER: Improving Main Memory Energy Efficiency in Servers," *Computers, IEEE Transactions on*, 58(3):336-50, 2009, http://ieeexplore.ieee.org/xpl/articleDetails.jsp?arnumber=4633346, 10.1109/TC.2008.177.

[28] A.N. Udipi, N. Muralimanohar, N. Chatterjee, R. Balasubramonian, A. Davis, and N.P. Jouppi, "Rethinking DRAM design and organization for energy-constrained multi-cores," *ACM SIGARCH Computer Architecture News*, 38(3):175-86, 2010,

[29] J.S. Vetter, Ed., *Contemporary High Performance Computing: From Petascale Toward Exascale*, 1 ed. Boca Raton: Taylor and Francis, 2013, pp. 750.

[30] H. Volos, A.J. Tack, and M.M. Swift, "Mnemosyne: Lightweight persistent memory," *ACM SIGPLAN Notices*, 46(3):91-104, 2011,

[31] B. Wang, B. Wu, D. Li, X. Shen, W. Yu, Y. Jiao, and J.S. Vetter, "Exploring Hybrid Memory for GPU Energy Efficiency through Software-Hardware Co-Design," in *International Conference Parallel Architectures and Compilation Techniques (PACT)*. Edinburgh, Scotland: IEEE/ACM, 2013, 10.1109/pact.2013.6618807.

[32] X. Wu, J. Li, L. Zhang, E. Speight, R. Rajamony, and Y. Xie, "Hybrid cache architecture with disparate memory technologies," Proc. ACM SIGARCH Computer Architecture News, 2009, pp. 34-45,

[33] C. Xu, D. Niu, N. Muralimanohar, N.P. Jouppi, and Y. Xie, "Understanding the trade-offs in multi-level cell ReRAM memory design," Proc. Design Automation Conference (DAC), 2013 50th ACM/EDAC/IEEE, 2013, pp. 1-6,

[34] K. Yoongu, R. Daly, J. Kim, C. Fallin, L. Ji Hye, L. Donghyuk, C. Wilkerson, K. Lai, and O. Mutlu, "Flipping bits in memory without accessing them: An experimental study of DRAM disturbance errors," Proc. Computer Architecture (ISCA), 2014 ACM/IEEE 41st International Symposium on, 2014, pp. 361-72, 10.1109/ISCA.2014.6853210.

Author Bios

Jeffrey S. Vetter, Ph.D., holds a joint appointment between ORNL and Georgia Institute of Technology (GT). At ORNL, he is a Distinguished R&D Staff Member, and the founding group leader of the Future Technologies Group. At GT, he is a Joint Professor in the Computational Science and Engineering School, where he leads the NSF Keeneland Project that provides GPU computing resources to NSF computational scientists, and the NVIDIA CUDA Center of Excellence. His research interests include massively multithreaded processors, non-volatile memory, and heterogeneous multicore processors. Vetter is a Senior Member of the IEEE, and a Distinguished Scientist Member of the ACM. (Email: vetter@computer.org).

Sparsh Mittal received the B.Tech. degree in electronics and communications engineering from IIT, Roorkee, India and the Ph.D. degree in computer engineering from Iowa State University, USA. He is currently working as a Post-Doctoral Research Associate at ORNL. His research interests include non-volatile memory, memory system power efficiency, cache, and GPU architectures. (Email: mittals@ornl.gov).