IPSJ Transactions on System LSI Design Methodology Vol.8 2–11 (Feb. 2015)

[DOI: 10.2197/ipsjtsldm.8.2]

Invited Paper

Memory and Storage System Design with Nonvolatile Memory Technologies

Jishen Zhao[1,a)] Cong Xu[2,b)] Ping Chi[3,c)] Yuan Xie[3,d)]

Received: September 16, 2014, Released: February 12, 2015

Abstract: The memory and storage system, including processor caches, main memory, and storage, is an important component of various computer systems. The memory hierarchy is becoming a fundamental performance and energy bottleneck, due to the widening gap between the increasing bandwidth and energy demands of modern applications and the limited performance and energy efficiency provided by traditional memory technologies. As a result, computer architects are facing significant challenges in developing high-performance, energy-efficient, and reliable memory hierarchies. New byte-addressable nonvolatile memories (NVMs) are emerging with unique properties that are likely to open doors to novel memory hierarchy designs to tackle the challenges. However, substantial advancements in redesigning the existing memory and storage organizations are needed to realize their full potential. This article reviews recent innovations in rearchitecting the memory and storage system with NVMs, producing high-performance, energy-efficient, and scalable computer designs.

Keywords: computer architecture, memory, storage, nonvolatile memory technologies

1. Introduction

The memory and storage system is a critical component of various computer systems. It is a hierarchy of memory and storage devices with various capacities, costs, and access times. In particular, processor caches act as staging areas for a subset of the data and instructions stored in the main memory; the main memory stages data stored in large, slow storage devices, such as disks and flash. Modern applications rely on both real-time and offline processing of data from infinite sea of books, maps, photos, audios, videos, references, facts, and conversations. Their data sets can be gigabytes, terabytes, or even larger in size. Storing and manipulating such a large amount of data raises significant challenges in designing high-performance, energy-efficient memory hierarchy.

Unfortunately, performance and energy scaling of mainstream memory technologies is in jeopardy. Commodity memory technologies, such as SRAM and DRAM, are facing scalability challenges due to the constraints of their device cell size and power dissipation. In particular, the increasing leakage power for SRAM and DRAM and the increasing refresh dynamic power of DRAM pose challenges to circuit and architecture designers of future memory hierarchy designs. Energy consumption has become key design limiters as the memory hierarchy continues to contribute a significant fraction of overall system energy and power [1]. The lack of memory technology scaling can make it difficult for the memory hierarchy to achieve high capacity and efficiency at low cost. As a result, the memory subsystem is becoming a fundamental performance and energy bottleneck in various computer systems.

Recently, emerging byte-addressable nonvolatile memory (NVM) technologies, such as spin-transfer torque memory (STT-MRAM), phase-change memory (PCM), and resistive memory (ReRAM), have been studied as the replacement of traditional memory technologies used in the memory hierarchy [2], [3], [4], [5]. NVMs have several promising characteristics: they have much higher density than SRAM, need no refresh (which is required by DRAMs), have much lower leakage power than SRAM and DRAM. Yet they also have various shortcomings (e.g., some have cell endurance problems, some have very high write latency/power) that need to be overcome. Therefore, NVMs and traditional memory technologies trade off density, speed, power, and reliability. Furthermore, NVMs promise the game-changing feature – comprising both traditional memory (fast and byte-addressable) and storage (nonvolatile) properties. Consequently, NVMs yield abundant design challenges and opportunities that will significantly change the landscape of memory subsystem. It is our position that computer architects need to fundamentally rethink the way we design the memory subsystem today to enable the effective use of emerging NVM technologies to tackle the scaling challenges with traditional memory technologies.

This article reviews recent research efforts on computer architecture design with NVMs and presents their implications on memory subsystem design. In particular, we investigate answers to the following questions:

- How to devise efficient and scalable memory and storage system design by leveraging NVMs to replace traditional memory technologies.

[1] University of California, Santa Cruz, Santa Cruz, CA 95064
[2] Pennsylvania State University, University Park, PA 16802, USA
[3] University of California, Santa Barbara, Santa Barbara, CA 93106, USA
[a)] jzhao31@ucsc.edu
[b)] czx102@cse.psu.edu
[c)] pingchi@ece.ucsb.edu
[d)] yuanxie@ece.ucsb.edu

- How to redesign the memory subsystem to fully exploit the full potential of of NVMs.

2. NVM Technologies

This article focuses on examining three NVM technologies, STT-MRAM, PCM, and ReRAM, which are being actively investigated by academia and industry. Most of the discussion can also apply to other NVM technologies that share the common properties – being byte-addressable and nonvolatile, such as ferroelectric RAM.

2.1 STT-MRAM

STT-MRAM is the latest generation of magnetic RAM (MRAM) [6], [7]. The basic difference between the STT-MRAM and the conventional RAM technologies (such as SRAM/DRAM) is that the information carrier of STT-MRAM is a Magnetic Tunnel Junction (MTJ) instead of electric charges. Each MTJ contains two ferromagnetic layers and one tunnel barrier layer. **Figure 1** shows a conceptual illustration of an MTJ. One of the ferromagnetic layer (reference layer) has fixed magnetic direction while the other one (free layer) can change its magnetic direction by an external electromagnetic field or a spin-transfer torque. If the two ferromagnetic layers have different directions, the MTJ resistance is high, indicating a "1" state (the anti-parallel case in Fig. 1 (a)); if the two layers have the same direction, the MTJ resistance is low, indicating a "0" state (the parallel case in Fig. 1 (a)). STT-MRAM changes the magnetic direction of the free layer by directly passing a spin-polarized current through the MTJ structure. Comparing to the previous generation of MRAMs that uses external magnetic fields to reverse the MTJ status, STT-MRAMs has the advantage of scalability, which means the threshold current to make the status reversal will decrease as the size of the MTJ becomes smaller.

In the STT-MRAM memory cell design, the most popular structure is composed of one NMOS transistor as the access controller and one MTJ as the storage element ("1T1J" structure). As illustrated in **Fig. 2**, the storage element, MTJ, is connected in series with the NMOS transistor. The NMOS transistor is controlled by the word-line (WL) signal. The detailed read and write operations for each MRAM cell is described as follows:

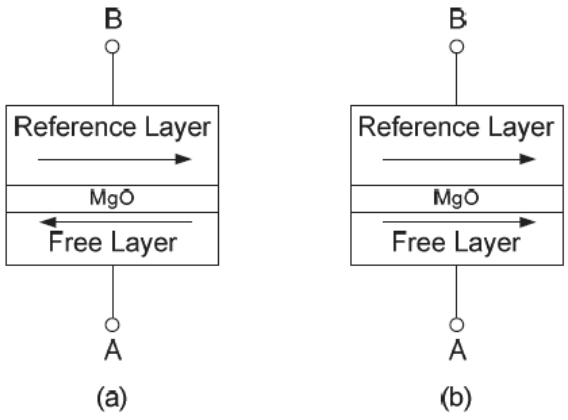


Fig. 1 A conceptual view of MTJ structure. (a) Anti-parallel (high resistance), which indicates "1" state; (b) Parallel (low resistance), which indicates "0" state.

- Read Operation: When a read operation happens, the NMOS is turned on and a voltage ($V_{BL} - V_{SL}$) is applied between the bit-line (BL) and the source-line (SL). This voltage is negative and is usually very small. This voltage difference will cause a current passing through the MTJ, but it is not small enough to invoke a disturbed write operation. The value of the current is determined by the equivalent resistance of MTJs. A sense amplifier compares this current with a reference current and then decides whether a "0" or a "1" is stored in the selected MRAM cell.
- Write Operation: When a write operation happens, a positive voltage difference is established between SLs and BLs for writing for a "0" or a negative voltage difference is established for writing a "1." The current amplitude required to ensure a successful status reversal is called threshold current. The current is related to the material of the tunnel barrier layer, the writing pulse duration, and the MTJ geometry.

2.2 PCM

PCM [8], [9] uses chalcogenide-based material to storage information. The crystalline and amorphous states of chalcogenide materials have a wide range of resistivity, about three orders of magnitude, and this forms the basis of data storage. The amorphous, high resistance state is used to represent a bit "0," and the crystalline, low resistance state represents a bit "1." When germanium-antimony-tellurium (GeSbTe or GST) is heated to a high temperature (normally over 600 °C), it will get melted and its chalcogenide crystallinity is lost. Once cooled, it is frozen into an amorphous and its electrical resistance becomes high. This process is called RESET. One way to achieve the crystalline state is by applying a lower constant-amplitude current pulse for a time longer than the so-called SET pulse. This is called SET process. The time of phase transition is temperature-dependent.

Figure 3 gives an illustration of a PCM cell. The read and

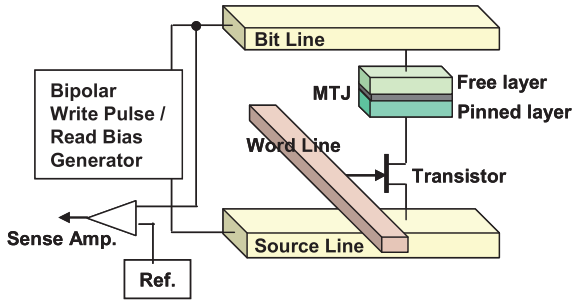


Fig. 2 An illustration of an STT-MRAM cell with read/write circuitry.

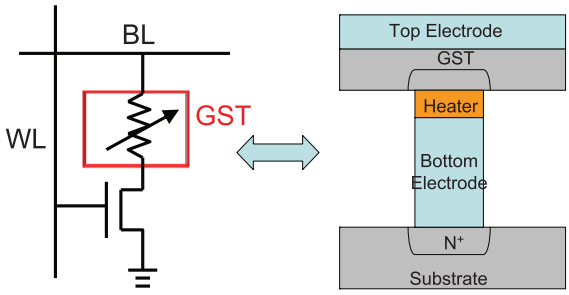


Fig. 3 Demonstration of a PCM cell.

write operations for a PRAM cell is described as follows:

- Read Operation: To read the data stored in PCM cells, a small voltage is applied across the GST. Since the SET status and RESET status have a large variance on their equivalent resistance, the data is sensed by measuring the pass-through current. The read voltage is set to be sufficiently strong to invoke detectable current but remains low enough to avoid write disturbance. Like other RAM technologies, each PRAM cell needs an access device for control purpose. As shown in Fig. 3, every basic PRAM cell contains one GST and one NMOS access transistor. This structure has a name of "1T1R" where "T" stands for the NMOS transistor and "R" stands for GST. The GST in each PCM cell is connected to the drain-region of the NMOS in series so that the data stored in PRAM cells can be accessed by wordline controlling.
- Write Operation: There are two kinds of PRAM write operations, the SET operation that switches the GST into crystalline phase and the RESET operation that switches the GST into amorphous phase. The SET operation crystallizes GST by heating it above its crystallization temperature, and the RESET operation melt-quenches GST to make the material amorphous. These two operations are controlled by electrical current: high-power pulses for the RESET operation heat the memory cell above the GST melting temperature; moderate power but longer duration pulses for the SET operation heat the cell above the GST crystallization temperature but below the melting temperature. The temperature is controlled by passing through a certain amount of electrical current and generating the required Joule heat.

2.3 ReRAM

The general definition of ReRAM includes any memory technology that represents digital information using their variable resistance. In this work, we restrict the definition of resistive memory to the metal oxide ReRAM which typically uses an intermediate layer of metal oxide for resistive switching. In ReRAM [10], [11], [12], a normally insulating dielectric is conducted through a filament or conduction path generated by applying a sufficiently high voltage. The conduction path can be generated by different mechanisms, including defects, metal migration, etc. The filament may be reset (broken, resulting in high resistance) or set (re-formed, resulting in lower resistance) by applying an appropriate voltage.

The basic structure of a ReRAM cell is one metal oxide layer sandwiched by two metal electrodes - the top electrode (TE) and the bottom electrode (BE), shown in **Fig. 4**. A low resistance state (LRS) represents digital "1" while a high resistance state (HRS) represents digital "0." The switching from HRS to LRS is defined as a SET operation while the opposite switching is defined as a RESET operation. Here we focus on bipolar switching, which means that a SET and a RESET occur at opposite voltage polarities. When a positive voltage is applied, a SET process leads to the formation of conductive filaments (CFs) made of oxygen vacancies. Once the CFs are formed, the ReRAM cell is in LRS. In contrast, when a negative voltage is applied across the cell, a RESET process leads to the rupture of the CFs, causing the cell into HRS.

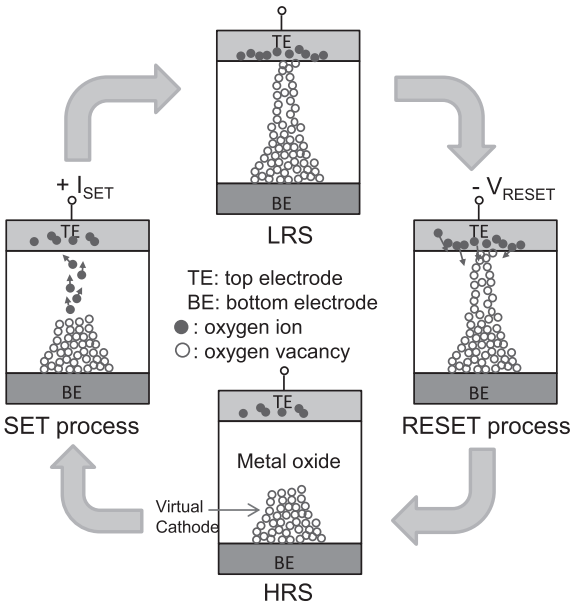


Fig. 4 Demonstration of a ReRAM cell and its SET/RESET operations.

2.4 Comparisons between Different NVMs

Table 1 compares NVMs [13], [14], [15], [16], [17], [18] with traditional memory and storage technologies, in terms of performance, energy, density, and endurance.

Benefits: NVMs promise superior density, performance, and energy characteristics compared with traditional memory technologies. For example, PCM is predicted to scale to much smaller feature sizes beyond the scaling limit of traditional memory technologies [8]. STT-MRAM, PCM and ReRAM can store multiple bits per cell [19], [20], [21], promising much higher density than traditional memory technologies. Due to the non-volatility, NVMs do not require refresh operations, which is a key scaling challenge of DRAM due to the performance degradation and high energy consumption imposed by periodical refresh [22]. Moreover, NVMs also have low idle power dissipation. Beyond the density, performance, and energy benefits, NVMs can accommodate both byte-addressable, fast data accesses and nonvolatile data storage. Therefore, they incorporate both memory (fast) and storage (retaining data without power supply) properties. This feature will disrupt current two-level memory-storage stack with the capability of accommodating fast access to permanent data storage in a unified nonvolatile memory.

Drawbacks: Nevertheless, existing NVM technologies impose performance, energy, and endurance issues – they are not strictly superior than SRAM and DRAM. Therefore, existing NVM technologies are unlikely to completely replace traditional memory technologies in the memory hierarchy. All the three types of NVMs have higher write latency and write energy than SRAM and DRAM. Two of the NVM technologies, PCM and ReRAM, have limited endurance [23], [24] that is much lower than SRAM and DRAM (STT-MRAM has near-SRAM endurance at 10^{15} [25], hence its endurance is less of an issue). The endurance of existing PCM implementations is in the range of

Table 1 Comparison of traditional memory and NVM technologies [13], [14], [15], [16], [17], [18].

Feature	SRAM	DRAM	Disk	Flash	STT-MRAM	PCM	ReRAM
Cell size	$> 100F^2$	$6 - 8F^2$	$2/3F^2$	$4 - 5F^2$	$37F^2$	$8 - 16F^2$	$> 5F^2$
Read latency	< 10 ns	10-60 ns	8.5 ms	$25\mu s$	< 10 ns	48 ns	< 10 ns
Write latency	< 10 ns	10-60 ns	9.5 ms	$200\mu s$	12.5 ns	40-150 ns	10 ns
Energy per bit access	> 1 pJ	2 pJ	100-1,000 mJ	10 nJ	2 pJ	100 pJ	0.02 pJ
Leakage power	High	Medium	High	Low	Low	Low	Low
Endurance	$> 10^{15}$	$> 10^{15}$	$> 10^{15}$	10^4	$> 10^{15}$	10^5-10^9	10^5-10^{11}
Nonvolatility	No	No	Yes	Yes	Yes	Yes	Yes
Scalability	Yes	Yes	Yes	Yes	Yes	Yes	Yes

10^5 to 10^9 writes [26]. That of existing ReRAM implementations varies between 10^5 and 10^{11} writes [24], [27], [28]. Future NVM technologies may improve endurance by orders of magnitude. A projected plan by ITRS [29] highlighted that the endurance of PCM and ReRAM will be at the order of 10^{15} or higher by 2024.

Products: NVM technologies are at early product development. They have achieved substantial innovations in capacity and performance in recent products. For example, SanDisk and Toshiba recently presented their 32 Gb ReRAM device [30], stepping forward for the technology to far exceed in capacity of previous NVM devices. Everspin recently launched the DDR3 compatible STT-MRAM components [31], which transfers data at a speed comparable to current DDR3-1600 DRAM.

3. Replacing Traditional Memory Technologies with NVMs

In principle, we can implement both on-chip and off-chip memory components with NVMs. For example, we can implement the off-chip NVM main memory as dual in-line memory modules (DIMMs), which is compatible to the commodity off-chip DRAM implementations [32]. Most NVM technologies are not compatible with CMOS technology, which is the traditional technology used to implement the processor cores and caches. Consequently, with most types of NVMs, we need to implement on-chip memory components by leveraging silicon interposer [33] or 3D stacking [34], [35], [36] technologies. With the silicon interposer technology, we can package CMOS and NVM components side-by-side in a single chip. This technology has been widely explored by academia and industry to develop high-performance system-in-package designs [33], [37]. With 3D stacking technology, we can deploy CMOS and NVM components on separate dies and vertically stack the dies on top of each other [36], [38], [39], [40], [41], [42], [43], [44], [45], [46], [47]. For instance, Samsung recently announced a 3D-stacked wide-I/O DRAM targeting mobile systems [43]. The presented two-layer DRAM with four 128-bit wide buses has 12.8 GB/s peak bandwidth, 2 Gb of capacity, and only 330.6 mW of power consumption. Woo et al. [42] rearchitected the memory hierarchy, including the L2 cache and DRAM interface, and take full advantage of the massive bandwidth provided by stacking the DRAMs on top of processor cores. Tezzaron corporation has implemented true 3D DRAMs, where the individual bitcell arrays are stacked in a 3D fashion [45]. The peripheral control logic and circuitry are placed on a separate, dedicated layer, incorporated with different process technologies. Micron developed a Hybrid Memory Cube (HMC) [44] that combines high-speed logic process technology with a stack of through-silicon via (TSV) bonded memory die. HMC increases density per bit and reduces the overall package form factor. Recently, AMD and SK Hynix announced joint development of high bandwidth memory (HBM) stacks [48], which leverages TSV and wide I/O technology and conforms JEDEC HBM standardization [49]. Recently, Lin et al. demonstrated a CMOS-compatible STT-MRAM implementation [50], which allows STT-MRAM based memory components to be directly integrated with processor cores on the same die.

3.1 Processor Cache Design with NVMs

In addition to low power dissipation and high density, processor caches have a couple of distinctive requirements: they need to be very fast; they need to have high endurance. Essentially, SRAM is currently preferred for caches because it is very fast. However, the fact that it is expensive, less dense, and its high leakage power has hindered its continuous scaling as the memory technology of processor caches. NVMs offer much higher density and lower leakage power than SRAM. Therefore, they have been actively studied as processor cache replacement.

With near-SRAM endurance and the best performance among various NVMs, STT-MRAM is a practical solution for cache design [51], [52], [53]. Dong et al. [14] proposed a 3D cache architecture design with STT-MRAM. The study demonstrated that an STT-MRAM-based level-2 (L2) cache can dramatically reduce system power by 70% and moderately improve system performance, due to the low leakage power and nonvolatility property of STT-MRAM. The same study also observed that STT-MRAM also improves system performance when being used as L3 cache. Sun et al. [54] demonstrated that leveraging an STT-MRAM-based shared L2 cache in a multicore processor can substantially reduce system power by 73.5%, compared with using an SRAM L2 cache in a similar silicon area. With higher cell density than SRAM, STT-MRAM-based cache offers much larger capacity.

To address NVMs' high write latency issue, Wu et al. [2], [55] investigated various hybrid cache architecture designs combining SRAM, eDRAM, STT-MRAM, and PCM in an IBM Power7 processor. The key idea is to create a cache hierarchy with two different regions: a write region implemented by SRAM or eDRAM which has low write latency; a read region implemented by STT-MRAM or PCM which has the benefit of low leakage power. The study also explored the potential of hardware support for intra-cache data movement and power consumption management within hybrid caches. Under the same area constraint across a collection of 30 workloads, the study found that such an aggressive hybrid cache design yields a 10% to 16% performance improvement over the baseline design with a level-3, SRAM-only

cache design, and achieves up to a 72% power reduction. Wang et al. [56] proposed an adaptive block placement and migration policy used in an SRAM/STT-MRAM hybrid last-level cache. The proposed mechanism improves both performance and power by placing a block into either SRAM or STT-MRAM by adapting to the access pattern of writes that are categorized as prefetching, demand, and core access.

One critical bottleneck for CPU performance scaling is the widening gap between the increasing bandwidth demand created by processor cores and the limited bandwidth provided by off-chip memories [57], [58], [59]. Due to such limitation, memory-demanding applications with a large working set spends additional cycles on off-chip memory accesses, and thus decreases the parallelism. In addition, even moderately memory-demanding applications will reach the bandwidth limitation as the number of cores scales up [60]. Consequently, memory bandwidth becomes one of the most important factors that influence high performance system design. To address the bandwidth issue, Zhao et al. [61] proposed a bandwidth-aware reconfigurable cache hierarchy to enhance system performance of multicore processors with hybrid memory technologies, by leveraging NVM's bandwidth benefits at large capacities. The hybrid cache hierarchy maximizes the provided bandwidth of processor caches and minimizes the bandwidth demand to the off-chip main memory. The design also dynamically reconfigures the hybrid cache hierarchy to accommodate the actively changing bandwidth demands of various applications. As a result, the proposed design can improve system throughput by up to 58%.

The low endurance of PCM and ReRAM makes them less feasible to be used as processor caches. To address this issue, Wang et al. [62] propsed a hard failure-tolerant architecture designed for nonvolatile caches. It imposes gradual performance overhead and small storage overhead, however, improves the lifetime of ReRAM-based caches by 4.6× compared with conventional cache architecture. Prior studies also developed various schemes to address the endurance issue by reducing the writes to NVM-based caches. Examples include efficient cache replacement policies [63] and hybrid PCM/STT-MRAM cache architecture for extend the lifetime of PCM caches [64].

3.2 Main Memory Design with NVMs

Main memory needs to be sufficiently large to hold most of the working set of executing programs; it can be slower than processor caches, because the most frequently accessed data are already cached. Commodity computer systems almost exclusively use DRAM as main memory. Yet the low leakage power and nonvolatility (and therefore requires no refresh and can be shut down when they are idle) benefits of NVMs make them attractive replacements of DRAM.

Lee et al. [5] pointed out that DRAM has inherent scalability limits, and proposed a PCM array architecture that enables the use of PCM as main memory. They showed that a pure PCM-based main memory can be 1.6× slower and consumes 2.2× higher energy than DRAM-based main memory, due to the high write latency and energy consumption. Their design incorporates two strategies to address these issues: reorganizing a single, wide buffer into multiple, narrow buffers; only updating the modified data rather than an entire row with partial writes.

Qureschi et al. [18] conducted another study to address PCM's high write latency and energy issues. The propsed main memory design consists of a PCM storage coupled with a small DRAM buffer. The DRAM buffer serves memory accesses that requires low latency; the PCM storage offers high capacity to store the working set of executing applications. Their design results in up to 3× speedup. Another study by Zhou et al. [65] tackled the same problem with a different architecture design. They proposed a 3D die stacked chip multiprocessor design which deploys processors and the main memory in the same chip. This is difficult to achieve with DRAM because such design requires tight power and thermal constraints which do not apply to DRAM with high leakage power. They demonstrated that PCM-based main memory consumes only 65% of the energy of a DRAM-based main memory with the same capacity.

Various techniques have been proposed to address the endurance issue of PCM and ReRAM based main memory. Examples include ECP [66], dynamically replicated memory [67], SAFER [68], start-gap [4], security refresh [69], and FREE-p [70].

4. Storage Design with NVMs

Because NVMs are nonvolatile and have much lower latency than disk and flash, they have been explored as fast storage components.

Freitas and Wilcke [71] studied potential applications that can leverage PCM as a disk replacement. They noted that the density of PCM can scale up to a point where it can compete with disks. They also argued that NVM-based storage will have simpler and more predictable behavior than disks, simplifying performance tuning. Kryder and Kim [15] evaluated STT-MRAM, PCM, and ReRAM as disk replacement, and showed that STT-MRAM and PCM are more feasible to replace disks than ReRAM in 2020 time frame.

Sun et al. [72] proposed hybrid storage architecture, which employs NVMs as the log region of NAND-flash-based SSDs. This design combines the advantages of NAND-flash memory and NVMs (specifically, STT-MRAM and PCM). The NAND-flash offers large capacity with low cost, while the NVM-based log region can significantly improve the performance of storage access by supporting in-place, fine-granularity log updates and reducing the read traffic from SSDs to main memory. In addition, the design can improve the lifetime of NAND-flash by reducing the number of erase operations.

5. Specialized Memory Design with NVMs

Beyond the conventional layers of the memory hierarchy, which includes processor caches, main memory, and storage, a few studies investigated the use of NVMs for specialized memories.

NVMs have been studied in checkpointing mechanism design for better performance and higher power efficiency. Dong et al. [73] leveraged PCM as the local storage of each node to implement checkpointing mechanisms in Massively Parallel Process-

ing (MPP) systems. To accommodate various component failures, modern MPP systems employ centralized storage to periodically take checkpoints of processing states. The study showed that petascale MPP systems can incur more than 25% performance overhead by committing the checkpoints in disks. The proposed PCM-based checkpointing mechanism adopts both local and global checkpointing. With local checkpointing, a node can reboot and recover from failures. With global checkpointing, the system can restore the complete system state when a node is removed. They demonstrated that their MPP system design can scale up to 500 petaflops with only 4% checkpointing overheads.

Kannan et al. [74] proposed to optimize checkpoints by using NVM as virtual memory. NVM such as PCM is utilized as extended memory to efficiently store checkpoints on both local and remote nodes. However, it is not directly exposed to applications. Specialized NVM interface is provided for applications to write checkpoints. To reduce the NVM and interconnect bandwidth, a pre-copy scheme is proposed, incrementally moving checkpoint data from DRAM to NVM before a local checkpoint is started. Their experimental results demonstrate that the local checkpoint time is reduced by 15% compared to using NVM as RAM disk, and the peak interconnect usage is decreased by up to 46% with the pre-copy method.

Chi et al. [75] proposed to leverage MLC STT-MRAM as the main memory to accelerate local checkpointing. MLC STT-RAM can be implemented with two types of designs, known as serial MLC and parallel MLC, respectively. A serial MLC has one small MTJ and one big MTJ, which are two vertically stacked free layers. The combination of the magnetization directions of the two free layers can generate four resistance levels. Alternatively, a parallel MLC possesses a single MTJ whose free layer has two domains, a hard domain and a soft domain. Similar to serial MLCs, parallel MLCs can achieve four resistance levels by combining each state of the two domains. Serial MLCs are easier to fabricate, whereas parallel MLCs are more energy-efficient because they need smaller switching current density to change the state. In parallel MLC, the hard domain in the free layer needs a larger current than the soft domain to switch its state. By encoding the low resistance state as "0" and the high resistance state as "1" in a 2-bit cell, four different states are available: "00," "01," "10," and "11," where the most significant bit (MSB) is harder to flip (a.k.a. hard-bit), and the least significant bit (LSB) is easier to switch (a.k.a. soft-bit). In their proposed design, the soft-bit of each cell is used to store the working data and the hard-bit is used to save a checkpoint of the soft-bit. According to the unique features of the write operations in MLC STT-MRAM, only one-step write is needed during the entire execution time. Local checkpoints can be easily created by moving data within memory cells, which substantially eliminates the data transfer overhead between the main memory and the backup storage. Their evaluation results demonstrate that using MLC STT-MRAM for local checkpointing is a fast and energy-efficient solution. For a multi-programmed four-core process node, the average local checkpointing overhead is less than 1% with a 1-second local checkpointing interval.

NVMs also bring in opportunities in designing energy-efficient memory hierarchy in General-purpose Graphic Processing Units (GPGPUs), which have been an attractive solution for applications that demand high computational performance. GPGPUs exploit extreme multithreading to target high throughput. To accommodate such high- throughput demands, the energy consumption of GPGPU systems continues to increase. As existing and future computer systems are becoming power limited, reducing system energy consumption while maintaining high energy efficiency is a critical challenge for GPGPU system design. To satisfy the demands of high-throughput computing, the GPGPUs require substantial amounts of on-chip and off-chip memories that can support a very large number of read and write accesses. Consequently, the memory subsystem consumes a significant portion of energy in a GPGPU system.

Today, GPGPUs adopt SRAM and DRAM to implement on-chip and off-chip memories, which impose high energy consumption and face scalability issues. Satyamoorthy and Parthasarathy [76] propose a GPGPU design that employs STT-MRAM as on-chip global memory, increasing the density by 4× with 3.5x less power dissipation compared with using SRAM. Zhao et al. [77] proposed an NVM-based graphics memory design, which effectively reduces graphics memory energy consumption without hurting GPU system performance by leveraging the unique data access patterns of the workloads running on GPGPUs.

6. Leveraging NVMs as Persistent Memory

An ideal memory system would be fast, persistent, and big (highly dense). Until recently, all known memory and storage technologies address only some of these characteristics. SRAM is very fast, but has low density and is not persistent. DRAM is fast and has higher densities than SRAM, but is not persistent either. Disks and flash are highly dense and persistent, however, are much slower than SRAM and DRAM. As a result, traditional computer systems adopt two-level storage model to achieve the ideal requirement in speed, capacity, and persistence: a fast, non-persistent memory hierarchy temporarily storing applications' working sets, which will be lost when system halts or reboots; a slow, persistent storage system storing persistent data that can survive across system boots.

NVMs promise near-memory latency and nonvolatility, and therefore incorporate the properties from both commodity memory and storage technologies. They can accommodate byte-addressable, fast memory accesses like memories, and offer permanent data storage without power supply like disks and flash. These unique properties enriched the traditional storage model with the new **persistent memory** technology, which supports persistence property in memory. It allows applications to perform loads and stores, as if they are accessing the main memory; yet it is the permanent home of data.

Persistence is a property which guarantees that the data (e.g., database records, files, and the corresponding metadata) stored in nonvolatile devices retain a consistent state in case of power loss or program crashes, even when all the data in higher-level volatile components are lost. To achieve persistence in memory is nontrivial, due to the presence of volatile processor caches and

Table 2 Comparison of various persistent memory designs [78]. (⋆ means In-place updates are only performed for memory stores to a single variable or at the granularity of the bus width.)

Designs	Mechanisms					Persistence Support	
	In-place	Logging	COW	clflush/ msync/ fsync	mfence/ barrier	Atomicity	Consistency
System without persistence support	Yes	No	No	No	No	×	×
BPFS [79]	⋆	**No**	Yes	**No**	Yes	√	√
Mnemosyne [80]	⋆	Yes	Yes	Yes	Yes	√	√
NV-heaps [81]	No	Yes	**No**	**No**	Yes	√	√
CDDS [82]	No	**No**	Yes	Yes	Yes	√	√
Kiln [78]	Yes	No	No	No	No	√	√

memory request reordering performed by the write-back caches and memory controllers. For instance, a power outage may occur while an application is inserting a node to a linked list stored in NVM. To optimize system performance, processor caches and memory controllers may reorder the write requests, writing the pointer into the NVM before writing the values of the new node. The linked list can lose consistency with dangling pointers, if values of the new node remaining in processor caches are lost due to power outage, and lead to unrecoverable data corruption.

In commodity computer sytsems, persistence is enforced in storage systems, which incorporate atomicity, consistency, and durability properties. Persistent memory also needs to guarantee these properties in memory system. First of all, a persistent memory system contains nonvolatile devices so each data update is retained during power loss, crashes, or errors. This is referred to as the **durability** property. Second, because the granularity of programmer-defined data updates can be larger than the interface width of the persistent memory, a single update is typically serviced as multiple requests. Therefore, sudden power losses or crashes can leave an update partially completed, corrupting the persistent data structures. To address this issue, each single update must be "all or nothing," i.e., either successfully completes or fails completely with the data in persistent memory intact. This property is **atomicity**. Third, **consistency** requires each update to convert persistent data from one consistent state to another. Taking an example where an application inserts a node to a linked list stored in persistent memory, a system (including software programs and hardware) needs to ensure that the initial values of the node are written into the persistent memory before updating the pointers in the list. Otherwise, the persistent data structure can lose consistency with dangling pointers in a sudden crash, leading to a permanent corruption not recoverable by restarting the application or the system. Typically, programmers are responsible for defining consistent data updates, because only the programmers know what it means for application data to be in harmony with itself. Of course, programmers can leverage runtime API to do this. While executing the software programs, hardware and system software need to preserve the demanded consistency.

One common method to ensure atomicity is multiversioning, where multiple copies of data exist in the memory system. When performing updates to one copy of data, another copy is left intact. Therefore, if one copy of data is corrupted by a partial update, another copy is still valid and available for recovery. Most persistent memory studies focus on developing software interface and programming models to enable the use of persistent memory; the memory architecture remains intact. The software interfaces of these persistent memory designs are developed by modifying traditional file sytems and databases, which employ one of two techniques to maintain multiversioning: write-ahead logging (or journaling) [80], [81] and copy-on-write (COW) [79], [82]. For example, NV-heaps [81] and Mnemosyne [80] adopt durable software transactional memory (STM) to support persistence for in-memory data objects. Both designs enforce atomic transactional updates by maintaining a redo log. Venkataraman et al. [82] developed a B-Tree implementation that maintains multiple versions of data by COW to allow atomic updates. Both logging and COW mechanisms impose significant performance overhead by explicitly executing logging or data copying instructions. While the software overhead is tolerable with traditional disk-based persistent memories where the I/O delay dominates the performance overhead, the fraction of software overhead increases dramatically when the persistent memory can be accessed at a much faster speed [83]. To address this issue, Kiln [78] leverages processor cache and main memory hierarchy to naturally maintain atomicity without performing logging or COW.

A primary mechanism to preserve consistency in persistent memory is ordering control, i.e., to enforce that the order of writes arriving at persitent memory must match the order in which they are issued by the processor. A mismatch can happen when processor caches and memory controllers reorder memory requests to optimize performance. Most persistent memory designs ensure the ordering by write-through caching [80] or bypassing the processor caches entirely, flush, memory fence [80], [82], [84], and msync operations, each imposing high performance costs. With write-through caching, each memory store needs to wait until reaching the main memory. Flush and memory fence mechanisms can cause a burst of memory traffic and block subsequent memory stores. Furthermore, flushing an entire cache can also evict the working sets of other applications from the cache. To address this issue, Condit et al. [79] developed a new file system, called BPFS, which adopted an epoch barrier mechanism to minimize the flush traffic, however at the cost of reduced durability strength that leads to potential data loss. Pelley et al. [85] recently introduced a relaxed persistence model to minimize the ordering control to buffer and coalesce writes to the same data. This study introduced a strand persistency model, which allows flexible reordering of independent data updates. **Table 2** qualitatively compares various persistent memory designs and a memory system

that support no persistence, in terms of memory update mechanisms and support of atomicity and consistency.

Leveraging NVMs' high density and nonvolatility benefits, a couple of studies [86], [87] envision that NVMs can be used as persistent memory and main memory simultaneously. In this case, significant resource contention can exist between applications/threads that leverage persistent memory (persistent applications/threads) and those using NVMs as working memory (general applications/threads). For example, Kannan et al. [86] observed that concurrently running persistent and general applications tend to compete for the shared processor caches due to the doubled writes issued by persistent applications. To address this issue, the study proposed a page contiguity algorithm to reduce the interference-related cache misses by up to 12%. Liu et al. [87] recently observed that contention exists write requests issued by persistent and general applications at the shared NVM interface. They proposed a use-case-aware memory scheduling policy to manipulate the scheduling ordering between the two types of writes. The policy allows writes from general applications to be serviced without being blocked by those from persistent applications.

7. Conclusion

For decades, computer systems adopt SRAM as caches, DRAM as main memory, and disks/flash as storage. However, limitations of these technologies threaten the sustainable growth of performance and energy efficiency of computer systems. With superior density, power, and nonvolatility characteristics, emerging NVM technologies provide opportunities to break this traditional system organization. However, NVMs also have drawbacks in performance and endurance compared with traditonal memory technologies. This article reviews recent research efforts in rearchitecting processor caches, main memory, and storage system by taking the full advantages of three types of NVM technologies, including STT-MRAM, PCM, and ReRAM. With an introduction of recent innovations in NVM-based memory and storage design, we hope this article will drive both the technology advancement and extensive adoption of NVM technologies in computer systems.

Acknowledgments The work in this article was supported in part by National Science Foundation grants 1218867, 1213052, 1409798, and Department of Energy under Award Number DESC0005026.

References

[1] Lefurgy, C., Rajamani, K., Rawson, F., Felter, W., Kistler, M. and Keller, T.: Energy management for commercial servers, *Computer*, Vol.36, No.12, pp.39–48 (2003).

[2] Wu, X., Li, J., Zhang, L., Speight, E., Rajamony, R. and Xie, Y.: Hybrid cache architecture with disparate memory technologies, *Proc. International Symposium on Computer Architecture*, pp.34–45 (2009).

[3] Sun, G., Dong, X., Xie, Y., Li, J. and Chen, Y.: A novel architecture of the 3D stacked MRAM L2 cache for CMPs, *Proc. International Symposium on High Performance Computer Architecture*, pp.239–249 (2009).

[4] Qureshi, M.K., Karidis, J., Franceschini, M., Srinivasan, V., Lastras, L. and Abali, B.: Enhancing lifetime and security of pcm-based main memory with start-gap wear leveling, *Proc. 42nd Annual IEEE/ACM International Symposium on Microarchitecture*, pp.14–23 (2009).

[5] Lee, B.C., Ipek, E., Mutlu, O. and Burger, D.: Architecting phase change memory as a scalable DRAM alternative, *International Symposium on Computer Architecture*, pp.2–13 (2009).

[6] Hosomi, M., Yamagishi, H., Yamamoto, T., Bessho, K., Higo, Y., Yamane, K., Yamada, H., Shoji, M., Hachino, H., Fukumoto, C., Nagao, H. and Kano, H.: A novel nonvolatile memory with spin torque transfer magnetization switching: spin-RAM, *Electron Devices Meeting, IEDM Technical Digest*, pp.459–462 (Dec. 2005).

[7] Zhao, W., Belhaire, E., Mistral, Q., Chappert, C., Javerliac, V., Dieny, B. and Nicolle, E.: Macro-model of spin-transfer torque based magnetic tunnel junction device for hybrid magnetic-CMOS design, *Behavioral Modeling and Simulation Workshop, Proc. 2006 IEEE International*, pp.40–43 (Sep. 2006).

[8] Raoux, S., Burr, G.W., Breitwisch, M.J., Rettner, C.T., Chen, Y.-C., Shelby, R.M., Salinga, M., Krebs, D., Chen, S.-H., Lung, H.-L. and Lam, C.H.: Phase-change random access memory: A scalable technology, *IBM J. Res. Dev.*, Vol.52, No.4, pp.465–479 (2008), available from ⟨http://dx.doi.org/10.1147/rd.524.0465⟩.

[9] Sousa, V.: Phase change materials engineering for RESET current reduction, *Proc. Memory Workshop* (2012).

[10] Degraeve, R., Fantini, A., Clima, S., Govoreanu, B., Goux, L., Chen, Y.Y., Wouters, D., Roussel, P., Kar, G., Pourtois, G., Cosemans, S., Kittl, J., Groeseneken, G., Jurczak, M. and Altimime, L.: Dynamic hourglass model for SET and RESET in HfO2 RRAM, *Proc. Symposium on VLSI Technology*, pp.75–76 (2012).

[11] Goux, L., Fantini, A., Kar, G., Chen, Y., Jossart, N., Degraeve, R., Clima, S., Govoreanu, B., Lorenzo, G., Pourtois, G., Wouters, D., Kittl, J., Altimime, L. and Jurczak, M.: Ultralow sub-500na operating current high-performance TiN\Al2O3\HfO2\Hf\TiN bipolar RRAM achieved through understanding-based stack-engineering, *Proc. Symposium on VLSI Technology*, pp.159–160 (2012).

[12] Cagli, C.: Characterization and modelling of electrode impact in HfO2-based RRAM, *Proc. Memory Workshop* (2012).

[13] Burr, G.W., Kurdi, B.N., Scott, J.C., Lam, C.H., Gopalakrishnan, K. and Shenoy, R.S.: Overview of candidate device technologies for storage-class memory, *IBM J. Res. Dev.*, Vol.52, No.4, pp.449–464 (2008), available from ⟨http://dx.doi.org/10.1147/rd.524.0449⟩.

[14] Dong, X., Wu, X., Sun, G., Xie, Y., Li, H. and Chen, Y.: Circuit and microarchitecture evaluation of 3d stacking magnetic ram (mram) as a universal memory replacement, *Proc. 45th Annual Design Automation Conference*, ser. DAC '08, pp.554–559, New York, NY, USA: ACM (2008), available from ⟨http://doi.acm.org/10.1145/1391469.1391610⟩.

[15] Kryder, M. and Kim, C.S.: After hard drives: What comes next? *IEEE Trans. Magnetics*, Vol.45, No.10, pp.3406–3413 (2009).

[16] Lewis, D. and Lee, H.-H.: Architectural evaluation of 3D stacked RRAM caches, *IEEE International Conference on 3D System Integration, 2009, 3DIC 2009*, pp.1–4 (2009).

[17] Mogul, J.C., Argollo, E., Shah, M. and Faraboschi, P.: Operating system support for nvm+dram hybrid main memory, *Proc. 12th Conference on Hot Topics in Operating Systems*, ser. HotOS'09, pp.14–14, Berkeley, CA, USA: USENIX Association (2009), available from ⟨http://dl.acm.org/citation.cfm?id=1855568.1855582⟩.

[18] Qureshi, M.K., Srinivasan, V. and Rivers, J.A.: Scalable high performance main memory system using phase-change memory technology, *Proc. 36th Annual International Symposium on Computer Architecture*, ser. ISCA '09, pp.24–33, New York, NY, USA: ACM (2009).

[19] Aoki, M., Noshiro, H., Tsunoda, K., Iba, Y., Hatada, A., Nakabayashi, M., Takahashi, A., Yoshida, C., Yamazaki, Y., Takenaga, T. and Sugii, T.: Novel highly scalable multi-level cell for STT-MRAM with stacked perpendicular MTJs, *Symposium on VLSI Technology*, pp.T134–T135 (2013).

[20] Bedeschi, F., Fackenthal, R., Resta, C., Donze, E., Jagasivamani, M., Buda, E., Pellizzer, F., Chow, D., Cabrini, A., Calvi, G., Faravelli, R., Fantini, A., Torelli, G., Mills, D., Gastaldi, R. and Casagrande, G.: A bipolar-selected phase change memory featuring multi-level cell storage, *IEEE Journal of Solid-State Circuits*, Vol.44, No.1, pp.217–227 (2009).

[21] Xu, C., Niu, D., Muralimanohar, N., Jouppi, N.P. and Xie, Y.: Understanding the trade-offs in multi-level cell ReRAM memory design, *Proc. 50th Annual Design Automation Conference*, ser. DAC '13, pp.108:1–108:6, New York, NY, USA: ACM (2013), available from ⟨http://doi.acm.org/10.1145/2463209.2488867⟩.

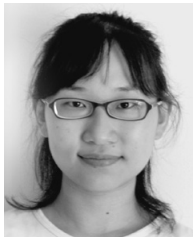
[22] Janzen, J.: The Micron system-power calculator, available from ⟨http://www.micron.com/products/dram/syscalc.html⟩.

[23] Wu, J., Breitwisch, M., Kim, S., Hsu, T.H., Cheek, R., Du, P.Y., Li, J., Lai, E., Zhu, Y., Wang, T., Cheng, H., Schrott, A., Joseph, E., Dasaka, R., Raoux, S., Lee, M.-H., Lung, H.L. and Lam, C.: A low power phase change memory using thermally confined TaN/TiN bottom electrode, *Electron Devices Meeting (IEDM), 2011 IEEE International*, pp.3.2.1–3.2.4 (2011).

[24] Kim, K.-H., Hyun Jo, S., Gaba, S. and Lu, W.: Nanoscale resistive

memory with intrinsic diode characteristics and long endurance, *Applied Physics Letters*, Vol.96, No.5, pp.1–3 (2010).
[25] Chen, E., Apalkov, D., Diao, Z., Driskill-Smith, A., Druist, D., Lottis, D., Nikitin, V., Tang, X., Watts, S., Wang, S., Wolf, S., Ghosh, A.W., Lu, J., Poon, S.J., Stan, M., Butler, W., Gupta, S., Mewes, C.K.A., Mewes, T. and Visscher, P.: Advances and future prospects of spin-transfer torque random access memory, *IEEE Trans. Magnetics*, Vol.46, No.6, pp.1873–1878 (2010).
[26] Ahn, S., Song, Y., Jeong, C., Shin, J., Fai, Y., Hwang, Y., Lee, S., Ryoo, K., Lee, S., Park, J.-H., Horii, H., Ha, Y., Yi, J., Kuh, B., Koh, G., Jeong, G., Jeong, H., Kim, K. and Ryu, B.-I.: Highly manufacturable high density phase change memory of 64Mb and beyond, *Proc. International Electron Devices Meeting*, pp.907–910 (2004).
[27] Lin, W.S., Chen, F.T., Chen, C.H.L. and Tsai, M.-J.: Evidence and solution of over-RESET problem for HfOx based resistive memory with sub-ns switching speed and high endurance, *Proc. International Electron Devices Meeting*, pp.19.7.1–19.7.4 (2010).
[28] Kim, Y.-B., Lee, S., Lee, D., Lee, C., Chang, M., Hur, J.H., Lee, M.-J., Park, G.-S., Kim, C.J., Chung, U., Yoo, I.-K. and Kim, K.: Bi-layered RRAM with unlimited endurance and extremely uniform switching, *Proc. Symposium on VLSI Technology*, pp.52–53 (2011).
[29] International technology roadmap for semiconductors, process integration, devices, and structures 2010 update, available from ⟨http://www.itrs.net⟩.
[30] yi Liu, T., Yan, T.H., Scheuerlein, R., Chen, Y., Lee, J., Balakrishnan, G., Yee, G., Zhang, H., Yap, A., Ouyang, J., Sasaki, T., Addepalli, S., Al-Shamma, A., Chen, C.-Y., Gupta, M., Hilton, G., Joshi, S., Kathuria, A., Lai, V., Masiwal, D., Matsumoto, M., Nigam, A., Pai, A., Pakhale, J., Siau, C.H., Wu, X., Yin, R., Peng, L., Kang, J.Y., Huynh, S., Wang, H., Nagel, N., Tanaka, Y., Higashitani, M., Minvielle, T., Gorla, C., Tsukamoto, T., Yamaguchi, T., Okajima, M., Okamura, T., Takase, S., Hara, T., Inoue, H., Fasoli, L., Mofidi, M., Shrivastava, R. and Quader, K.: A 130.7mm2 2-layer 32Gb ReRAM memory device in 24nm technology, *Solid-State Circuits Conference Digest of Technical Papers* (*ISSCC*), *2013 IEEE International*, pp.210–211 (2013).
[31] Janesky, J., Rizzo, N., Houssameddine, D., Whig, R., Mancoff, F., DeHerrera, M., Sun, J., Schneider, M., Chia, H., Aggarwal, S., Nagel, K., Deshpande, S., Andre, T., Alam, S., Tan, C., Slaughter, J., Hellmold, S. and LoPresti, P.: Device performance in a fully functional 800MHz DDR3 spin torque magnetic random access memory, *Memory Workshop* (*IMW*), *2013 5th IEEE International*, pp.17–20 (2013).
[32] Janesky, J.: Device performance in a fully functional 800MHz DDR3 Spin Torque Magnetic Random Access Memory, *IMW* (2013).
[33] Dorsey, P.: Xilinx stacked silicon interconnect technology delivers breakthrough FPGA capacity, bandwidth, and power efficiency, *Xilinx White Papers* (2010).
[34] Zhao, J., Dong, X. and Xie, Y.: Cost-aware three-dimensional (3D) many-core multiprocessor design, *Proc. 47th Design Automation Conference*, pp.126–131 (2010).
[35] Xie, Y., Loh, G.H., Black, B. and Bernstein, K.: Design space exploration for 3D architectures, *J. Emerg. Technol. Comput. Syst.*, Vol.2, No.2, pp.65–103 (2006).
[36] Loh, G.H.: 3d-stacked memory architectures for multi-core processors, *Proc. International Symposium on Computer Architecture*, pp.453–464 (2008).
[37] Dong, X., Xie, Y., Muralimanohar, N. and Jouppi, N.P.: Simple but effective heterogeneous main memory with on-chip memory controller support, *Proc. 2010 ACM/IEEE International Conference for High Performance Computing, Networking, Storage and Analysis*, ser. SC '10, pp.1–11, Washington, DC, USA, IEEE Computer Society (2010).
[38] Kgil, T., D'Souza, S. and Saidi, A., et al.: PicoServer: Using 3D stacking technology to enable a compact energy efficient chip multiprocessor, *Proc. International Conference on Architectural Support for Programming Languages and Operating Systems*, pp.117–128 (2006).
[39] Liu, C.C., Ganusov, I., Burtscher, M. and Tiwari, S.: Bridging the processor-memory performance gap with 3D IC technology, *IEEE Design Test*, Vol.22, pp.556–564 (2005).
[40] Loi, G.L., Agrawal, B., Srivastava, N., Lin, S.-C., Sherwood, T. and Banerjee, K.: A thermally-aware performance analysis of vertically integrated (3-D) processor-memory hierarchy, *Proc. Design Automation Conference*, pp.991–996 (2006).
[41] Gu, S., Marchal, P., Facchini, M., Wang, F., Suh, M., Lisk, D. and Nowak, M.: Stackable memory of 3D chip integration for mobile applications, *Proc. Intl. Electron Devices Meeting*, pp.1–4 (2008).
[42] Woo, D.H., Seong, N.H., Lewis, D.L. and Lee, H.-H.: An optimized 3D-stacked memory architecture by exploiting excessive, high-density TSV bandwidth, *Proc. International Conference for High Performance Computing*, pp.1–12 (2010).
[43] Kim, J.-S., Oh, C.S. and Lee, H., et al.: A 1.2V 12.8GB/s 2Gb mobile wide-I/O DRAM with 4 × 128 I/Os using TSV-based stacking, *Proc. Intl. Solid-State Circuits Conf.*, pp.496–498 (2011).
[44] Micron: Hybrid memory cube specification 1.0 (2013).
[45] Tezzaron Semiconductors: FaStack 3D stackable DRAM (2010), available from ⟨http://www.tezzaron.com/memory/⟩.
[46] Loi, I. and Benini, L.: An efficient distributed memory interface for many-core platform with 3D stacked DRAM, *Proc. Conference on Design, Automation and Test in Europe*, pp.99–104 (2010).
[47] Jevdjic, D., Volos, S. and Falsafi, B.: Die-stacked DRAM caches for servers: Hit ratio, latency, or bandwidth? have it all with footprint cache, *Proc. International Symposium on Computer Architecture*, pp.404–415 (2013).
[48] AMD and Hynix announce joint development of HBM memory stacks (2014), available from ⟨http://electroiq.com/blog/2013/12/amd-and-hynix-announce-joint-development-of-hbm-memory-stacks/⟩.
[49] JEDEC: High bandwidth memory (HBM) DRAM (2013), available from ⟨http://www.jedec.org/standards-documents/docs/jesd235⟩.
[50] Lin, C.J., Kang, S.H., Wang, Y.J., Lee, K., Zhu, X., Chen, W.C., Li, X., Hsu, W.N., Kao, Y.C., Liu, M.T., Chen, W.C., Lin, Y.C., Nowak, M., Yu, N., and Tran, L.: 45nm low power CMOS logic compatible embedded STT MRAM utilizaing a reverse-connection 1T/1MTJ cell, *Proc. International Electron Devices Meeting*, pp.11.6.1–11.6.4 (2009).
[51] Kitagawa, E., Fujita, S., Nomura, K., Noguchi, H., Abe, K., Ikegami, K., Daibou, T., Kato, Y., Kamata, C., Kashiwada, S., Shimomura, N., Ito, J. and Yoda, H.: Impact of ultra low power and fast write operation of advanced perpendicular MTJ on power reduction for high-performance mobile CPU, *Proc. International Electron Devices Meeting*, pp.29.4.1–29.4.4 (2012).
[52] Yoda, H., Fujita, S., Shimomura, N., Kitagawa, E., Abe, K., Nomura, K., Noguchi, H. and Ito, J.: Progress of STT-MRAM technology and the effect on normally-off computing systems, *Proc. International Electron Devices Meeting*, pp.11.3.1–11.3.4 (2012).
[53] Abe, K., Noguchi, H., Kitagawa, E., Shimomura, N., Ito, J. and Fujita, S.: Novel hybrid DRAM/MRAM design for reducing power of high performance mobile CPU, *Proc. International Electron Devices Meeting*, pp.10.5.1–10.5.4 (2012).
[54] Sun, G., Dong, X., Xie, Y., Li, J. and Chen, Y.: A novel architecture of the 3D stacked MRAM L2 cache for CMPs, *Proc. International Conference on High-Performance Computer Architecture*, pp.239–249 (2009).
[55] Wu, X., Li, J., Zhang, L., Speight, E. and Xie, Y.: Power and performance of read-write aware hybrid caches with non-volatile memories, *Proc. Conference on Design, Automation and Test in Europe*, ser. DATE '09, pp.737–742, 3001 Leuven, Belgium, Belgium, European Design and Automation Association (2009), available from ⟨http://dl.acm.org/citation.cfm?id=1874620.1874803⟩.
[56] Wang, Z., Jimenez, D., Xu, C., Sun, G. and Xie, Y.: Adaptive placement and migration policy for an STT-RAM-based hybrid cache, *2014 IEEE 20th International Symposium on High Performance Computer Architecture* (*HPCA*), pp.13–24 (2014).
[57] McKee, S.A.: Reflections on the memory wall, *Proc. Conference on Computing Frontiers*, p.162 (2004).
[58] Burger, D., Goodman, J.R. and Kägi, A.: Memory bandwidth limitations of future microprocessors, *Proc. International Symposium on Computer Architecture*, pp.78–89 (1996).
[59] Rogers, B.M., et al.: Scaling the bandwidth wall: Challenges in and avenues for cmp scaling, *Proc. International Symposium on Computer Architecture*, pp.371–382 (2009).
[60] Huh, J., Burger, D. and Keckler, S.W.: Exploring the design space of future CMPs, *Proc. International Conference on Parallel Architectures and Compilation Techniques*, pp.199–210 (2001).
[61] Zhao, J., Xu, C. and Xie, Y.: Bandwidth-aware reconfigurable cache design with hybrid memory technologies, *Proc. International Conference on Computer-Aided Design*, pp.48–55 (2011).
[62] Wang, J., Dong, X. and Xie, Y.: Point and discard: A hard-error-tolerant architecture for non-volatile last level caches, *Design Automation Conference* (*DAC*), *2012 49th ACM/EDAC/IEEE*, pp.253–258 (2012).
[63] Rodríguez-Rodríguez, R., Castro, F., Chaver, D., Pinuel, L. and Tirado, F.: Reducing writes in phase-change memory environments by using efficient cache replacement policies, *Proc. Conference on Design, Automation and Test in Europe*, ser. DATE '13, pp.93–96, San Jose, CA, USA, EDA Consortium (2013), available from ⟨http://dl.acm.org/citation.cfm?id=2485288.2485312⟩.
[64] Joo, Y. and Park, S.: A hybrid PRAM and STT-RAM cache architecture for extending the lifetime of PRAM caches, *Computer Architecture Letters*, Vol.12, No.2, pp.55–58 (2013).
[65] Zhou, P., Zhao, B., Yang, J. and Zhang, Y.: A durable and energy efficient main memory using phase change memory technology, *Proc. 36th Annual International Symposium on Computer Architecture*, ser. ISCA '09, pp.14–23, New York, NY, USA, ACM (2009), available

from ⟨http://doi.acm.org/10.1145/1555754.1555759⟩.
[66] Schechter, S., Loh, G.H., Straus, K. and Burger, D.: Use ECP, not ECC, for hard failures in resistive memories, *Proc. International Symposium on Computer Architecture*, pp.141–152 (2010).
[67] Ipek, E., Condit, J., Nightingale, E.B., Burger, D. and Moscibroda, T.: Dynamically replicated memory: Building reliable systems from nanoscale resistive memories, *Proc. Fifteenth Edition of ASPLOS on Architectural Support for Programming Languages and Operating Systems*, ser. ASPLOS XV, pp.3–14, New York, NY, USA, ACM (2010),
[68] Seong, N.H., Woo, D.H., Srinivasan, V., Rivers, J.A. and Lee, H.-H.S.: SAFER: Stuck-at-fault error recovery for memories, *Proc. International Symposium on Microarchitecture*, pp.115–124 (2010).
[69] Seong, N.H., Woo, D.H. and Lee, H.-H.S.: Security refresh: Prevent malicious wear-out and increase durability for phase-change memory with dynamically randomized address mapping, *Proc. International Symposium on Computer Architecture*, pp.383–394 (2010).
[70] Yoon, D.H., Muralimanohar, N., Chang, J., Ranganathan, P., Jouppi, N. and Erez, M.: FREE-p: Protecting non-volatile memory against both hard and soft errors, *Proc. International Symposium on High Performance Computer Architecture*, pp.466–477 (2011).
[71] Freitas, R.F. and Wilcke, W.W.: Storage-class memory: The next storage system technology, *IBM J. Res. Dev.*, Vol.52, No.4, pp.439–447 (2008), available from ⟨http://dx.doi.org/10.1147/rd.524.0439⟩.
[72] Sun, G., Joo, Y., Chen, Y., Niu, D., Xie, Y., Chen, Y. and Li, H.: A hybrid solid-state storage architecture for the performance, energy consumption, and lifetime improvement, *2010 IEEE 16th International Symposium on High Performance Computer Architecture* (*HPCA*), pp.1–12 (2010).
[73] Dong, X., Muralimanohar, N., Jouppi, N., Kaufmann, R. and Xie, Y.: Leveraging 3D PCRAM technologies to reduce checkpoint overhead for future exascale systems, *Proc. Conference on High Performance Computing Networking, Storage and Analysis*, ser. SC '09, pp. 57:1–57:12, New York, NY, USA: ACM (2009), available from ⟨http://doi.acm.org/10.1145/1654059.1654117⟩.
[74] Kannan, S., Gavrilovska, A., Schwan, K. and Milojicic, D.: Optimizing checkpoints using NVM as virtual memory, *IPDPS*, pp.29–40 (2013).
[75] Chi, P., Xu, C., Zhang, T., Dong, X. and Xie, Y.: Using multi-level cell stt-ram for fast and energy-efficient local checkpointing, *ICCAD* (2014).
[76] Satyamoorthy, P. and Parthasarathy, S.: MRAM for shared memory in GPGPUs (2010), available from ⟨http://www.cs.virginia.edu/s̃p5ej/MRAM.pdf⟩.
[77] Zhao, J. and Xie, Y.: Optimizing bandwidth and power of graphics memory with hybrid memory technologies and adaptive data migration, *Proc. International Conference on Computer-Aided Design*, pp.81–87 (2012).
[78] Zhao, J., Li, S., Yoon, D.H., Xie, Y. and Jouppi, N.P.: Kiln: Closing the performance gap between systems with and without persistence support, *Proc. 2013 46th Annual IEEE/ACM International Symposium on Microarchitecture*, Washington, DC, USA, IEEE Computer Society (2013).
[79] Condit, J., Nightingale, E.B., Frost, C., Ipek, E., Lee, B., Burger, D. and Coetzee, D.: Better i/o through byte-addressable, persistent memory, *Proc. ACM SIGOPS 22Nd Symposium on Operating Systems Principles*, ser. SOSP '09, pp.133–146, New York, NY, USA, ACM (2009),
[80] Volos, H., Tack, A.J. and Swift, M.M.: Mnemosyne: Lightweight persistent memory, *Proc. Sixteenth International Conference on Architectural Support for Programming Languages and Operating Systems*, ser. ASPLOS XVI, pp.91–104, New York, NY, USA, ACM (2011).
[81] Coburn, J., Caulfield, A.M., Akel, A., Grupp, L.M., Gupta, R.K., Jhala, R. and Swanson, S.: NV-heaps: Making persistent objects fast and safe with next-generation, non-volatile memories, *International Conference on Architectural Support for Programming Languages and Operating Systems*, pp.105–118 (2011).
[82] Venkataraman, S., Tolia, N., Ranganathan, P. and Campbell, R.H.: Consistent and durable data structures for non-volatile byte-addressable memory, *Proc. 9th USENIX Conference on File and Storage Technologies*, pp.1–15 (2011).
[83] Caulfield, A.M., Mollov, T.I., Eisner, L.A., De, A., Coburn, J. and Swanson, S.: Providing safe, user space access to fast, solid state disks, *Proc. International Conference on Architectural Support for Programming Languages and Operating Systems*, pp.387–400 (2012).
[84] Narayanan, D. and Hodson, O.: Whole-system persistence, *Proc. 17th International Conference on Architectural Support for Programming Languages and Operating Systems*, ser. ASPLOS XVII, pp.401–410, New York, NY, USA, ACM (2012).
[85] Pelley, S., Chen, P.M. and Wenisch, T.F.: Memory persistency, *Proc. International Symposium on Computer Architecture*, pp.1–12 (2014).
[86] Kannan, S., Gavrilovska, A. and Schwan, K.: Reducing the cost of persistence for nonvolatile heaps in end user devices, *Proc. International Symposium on High Performance Computer Architecture*, pp.1–12 (2014).
[87] Liu, R.-S., Shen, D.-Y., Yang, C.-L., Yu, S.-C. and Wang, C.-Y.M.: NVM Duet: Unified working memory and persistent store architecture, *Proc. International Conference on Architectural Support for Programming Languages and Operating Systems*, pp.455–470 (2014).



Jishen Zhao received her Ph.D. degree from Pennsylvania State University in 2014. She was a research scientist at HP Labs before joining UCSC as an assistant professor in 2015. Her research is concerned with a broad range of computer architecture, electronic design automation, and VLSI design, with a particular emphasis on memory systems, high-performance computing, and energy efficiency. She is a member of IEEE and ACM.

Cong Xu received his B.S. degree from Peking University, China, in 2009. He received his Ph.D. in the Computer Science and Engineering department at the Pennsylvania State University in 2014. His research interests include emerging memory technologies, neuromorphic computing paradigms, and computer architectures. He has authored/co-authored about 30 publications in these fields. He is a member of IEEE and ACM.

Ping Chi received both her B.S. and M.S. degrees from Tsinghua University, Beijing, China in 2008 and 2011. She joined the Pennsylvania State University as a Ph.D student in August, 2011. She transferred to University of California, Santa Barbara in August, 2014. Her research interest includes emerging non-volatile memory technologies, electronic design automation, and low power system design. She is a member of IEEE and ACM.

Yuan Xie received his B.S. degree and Ph.D. degree from Tsinghua University and Princeton University, respectively. He has worked for IBM and AMD, and was with Pennsylvania State University for 11 years, before joining UCSB in Fall 2014. He is a Fellow of IEEE.

(Invited by Editor-in-Chief: *Hiroyuki Tomiyama*)