A Survey of Power Management Techniques for Phase Change Memory

Sparsh Mittal

Future Technologies Group
Oak Ridge National Laboratory
Tennessee, USA 37830
Email: mittals@ornl.gov

Abstract: The demands of larger memory capacity in high-performance computing systems have motivated the researchers to explore alternatives of DRAM (dynamic random access memory). Since PCM (phase change memory) provides high-density, good scalability and non-volatile data storage, it has received significant amount of attention in recent years. A crucial bottleneck in wide-spread adoption of PCM, however, is that its write latency and energy are very high. Recently, several architecture and system-level techniques have been proposed to address this issue. In this paper, we survey several techniques for managing power consumption of PCM. We also classify these techniques based on their characteristics to highlight their similarities and differences. The aim of this paper is to provide insights to researchers into working of PCM power-management techniques and also motivate them to propose even better techniques for designing future 'green' PCM-based main memory systems.

Keywords: Phase change memory (PCM), phase change RAM (PCRAM), non-volatile memory (NVM), energy saving techniques, architectural techniques, green computing

Reference to this paper should be made as follows: Mittal, S. 'A Survey of Power Management Techniques for Phase Change Memory', Int. J. of Computer Aided Engineering and Technology, 2014.

Biographical notes: 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 Oak Ridge National Laboratory (ORNL). He has been awarded scholarship and fellowship from IIT Roorkee and Iowa State University. His research interests include non-volatile memory, cache and main memory systems, memory system power efficiency, and GPU architectures.

.

1 Introduction

Recent years have witnessed increasing power costs of memory subsystems, which have also led to increased costs of operating large scale supercomputers and data centers. Further, several modern computing systems provide several 24×7 services and execute data-centric computing applications, which demand larger capacity memory systems. Since DRAM has low-density, DRAM-based memory system may contribute as much as 40% in the total processor power [1–3]. Table 1 shows the power consumption breakdown for an IBM p670 processor [1]. Here the power under the row "processor" includes power of processor cores and L1 and L2 caches, cache controllers, and directories [1] etc. Also, the power under the row "memory" includes the power consumption of off-chip L3 caches, DRAM memory, memory controllers, and their interface chips. The power is measured when the system is idle and these values are expected to be same when the system is active, due to the performance-optimized design and use of system-level power-management features [1]. It is clear that memory system may consume very large amount of power in commercial processors. Further, the scaling of DRAM to smaller feature size has become difficult [4] and hence, computer architects and designers have explored alternative memory technologies that provide high density and can balance the costs against capacities.

Phase change memory (PCM, also called phase change RAM or PCRAM) is an emerging technology, which has recently received a lot of attention. PCM is a non-volatile device which has high retention property (over 10 years) and very good operation characteristics and scalability [5]. PCM has high storage density and PCM prototypes with feature size as small as 3nm have been fabricated [5]. It also has higher write endurance than the flash memory [2, 6], although its write endurance is several orders of magnitude worse than that of DRAM. The read power and delay of PCM

Table 1 Power consumption breakdown for an IBM p670 processor (large configuration) [1].

	Power (watt)	Percentage
Processor	840	28.3%
Memory	1223	41.1%
I/O etc.	90	3.0%
Processor and memory fans	676	22.7%
I/O component fans	144	4.8%
Total	2973	100%

are in the same range as that of DRAM, however, its write power is significantly higher than that of DRAM [7, 8]. Writing to a PCM cell requires high current density over a large period of time. Hence, to ensure correct operation, hard limits on the number of simultaneous writes must be enforced which limits write throughput and overall performance. Thus, failure to save write energy may nullify the energy saving advantage gained due to low leakage power of PCM.

Further, large power consumption of PCM can have deleterious effect on its operation. It may lead to violating power limits, which may in turn lead to voltage drops in the power supply or excessive currents flowing through the processor. It may increase the temperature which may further increase the leakage energy consumption of other components of the system. It may also create logical errors, incomplete PCM phase transitions, PCM read errors, etc. which may lead to chip failures or chip-aging. Thus, power management of PCM is extremely important to ensure its wide-spread adoption and also avoid the problems due to heating of the system.

In this paper, we review several architectural and system-level techniques for managing the power consumption of PCM. We classify these techniques based on their properties to highlight their similarities and differences. We only discuss the essential idea of the techniques and not the quantitative results, since different techniques have been evaluated using different platforms. Also, we do not include device/fabrication level approaches for improving PCM performance/power. The aim of this paper is to provide a synthetic overview of the research field of PCM power management. We believe that this overview will encourage researchers to propose even better techniques for managing PCM based main memory systems.

The rest of the paper is structured as follows. Section 2 provides a background on phase change memory. Section 3 provides an overview of power management techniques and related research works on PCM. Section 4 discusses some of these power management techniques in detail. Finally, Section 5 discusses the future work and provides the conclusion.

Table 2 Overview and classification of research works on PCM

Classification	References
Architectural Techniques	
Hybrid PCM-DRAM architecture	[10, 12–35]
Data-migration in DRAM-PCM hybrid memory	[2, 12, 15, 17, 18, 20]
Read-before write techniques	[6, 7, 36–41]
Reducing PCM write-traffic	[42–45]
Task-scheduling techniques	[19, 34]
PCM-aware cache management	[9, 46–50]
Data-compression based techniques	[21, 51]
Memory access scheduling techniques	[15, 28, 35, 52]
Techniques utilizing asymmetry in write times	[38, 53–56]
Data-encoding based techniques	[36, 38, 41, 42]
Novel PCM memory architecture	[57, 58]
Managing multi-level cell (MLC) PCM	[36, 39, 59–61]
Application Domain	
GPU	[14, 24]
Embedded systems	[44, 62–65]
Real-time systems	[66]
Video applications	[40, 67]
Designing caches	[68–75]
Others	
Simulator and modeling tools	[76–79]

2 A Brief Background on Phase Change Memory

We briefly review the design of PCM and refer the reader to previous works for more details [9, 10]. A PCM cell comprises an NMOS access transistor and a storage resistor which is made of a chalcogenide alloy [11, 12]. To store a binary value on PCM, heat is applied to it which transitions the physical state of the alloy with particular resistances. When the alloy is heated to a very high temperature (greater than 600 degree Celsius) and quickly cooled down, it transitions into an amorphous substance with high electrical resistance which represents binary “0”. On the other hand, when the alloy is heated to a temperature between the crystallization (300 degree Celsius) and melting (600 degree Celsius) points and cools down slowly, it crystallizes to a physical state with lower resistance, which represents binary “1”. Since the state of chalcogenide is retained in the absence of electrical power, PCM is non-volatile.

It is clear that while DRAM stores data as a small amount of electric charge in a capacitor, the PCM’s approach to storing data is fundamentally different. This difference gives PCM the potential for better technology scaling, leading to higher density, lower cost per bit, and larger capacity memory than DRAM. The difference in resistance values between the two states of PCM is typically 3 orders of magnitude. PCM memories achieve high density by exploiting this high resistance range to store multiple bits in a single cell, this structure is known as multi-level cell or MLC. PCM is byte-addressable and is immune to radiation-induced soft errors.

3 An Overview of Power Management Techniques for PCM Main Memory

To meet the demands of power-budget, power management techniques have been proposed for all range of computing systems, ranging from embedded systems to data-centers and supercomputers [80] and for all components of computing systems, such as cache and main memory [81, 82]. Specifically, in the context of PCM main memory systems, computer architects have proposed several power management techniques. Some researchers have proposed hybrid PCM-DRAM design [10, 12–34]. These techniques aim to achieve the best of both DRAM and PCM, viz. the short latency and high write endurance of DRAM and low leakage power and high density of PCM.

Some techniques convert PCM write operation to read-before-write (or data comparison write) operation to reduce write energy [6, 7, 36–41]. These techniques, referred to as "differential write" based techniques, read out the old value in the PCM array before writing the new one and compare them to write only those bits that need to change. Some researchers propose task-scheduling based techniques to address the challenges in hybrid DRAM-PCM based main memory [19, 34].

Several other techniques are based on reducing write-traffic to PCM memory, e.g. [42–45]. Some researchers propose last level cache management techniques for improving energy efficiency of PCM main memory [46, 47]. Other researchers have proposed compression based techniques to reduce write-traffic to PCM [21, 51]. Several other techniques aim to address the write latency issue, and its harmful impact on read latency arising due to bank conflicts and try to utilize write locality to coalesce all possible changes to the data by using buffers before they are finally written to PCM [15, 52]. PCM also offers the ability to store multiple bits per cell and several researchers propose techniques to achieve this in an energy efficient manner [36, 39, 59–61].

Since PCM has much smaller leakage energy than DRAM, recently, there has been a significant increase in application-areas of PCM. PCM has been evaluated in context of GPUs (graphics processing unit) [14, 24], embedded systems [44, 62–65], real-time systems [66], video applications [40, 67]) and so on. In fact, use of PCM has also been explored for designing caches and the insights gained from these techniques can be applied for managing PCM main memory also [68–75]. Finally, some researchers have proposed architecture or device-level simulators for studying non-volatile memories [76–79], which facilitate study of PCM. Table 2 provides an overview of research works on PCM.

4 Power Management Techniques

In this section, we discuss some power management techniques. Many of the techniques are orthogonal to each other and hence, can be synergistically integrated with each other.

4.1 Hybrid PCM-DRAM Architecture

Several authors propose PCM-DRAM hybrid main memory system architectures, which aim to utilize the best features of different technologies, such as performance, cost, energy, reliability, endurance. We now discuss a few of these architectures.

Qureshi et al. [10] propose a hybrid memory design where PCM memory is augmented with a small DRAM that acts as a "page cache" for the PCM memory. The page cache buffers frequently accessed pages and thus helps performance and improves PCM endurance by reducing the number of writes to PCM with write combining and coalescing. Further, at cache line level, only the lines modified in a page are written to the main memory, which reduces the effective number of writes to the main memory. Finally, at block-level, swapping is used for achieving wear-leveling. Their technique also reduces the page faults which improves the performance of the system and leads to saving of energy. However, when the applications have poor locality, the advantage of using page cache reduces.

Zhang et al. [15] study PCM in the context of 3D die-stacking. Using analytical and circuit-level modeling for PCM characterization, they show that the programming power of PCM cells can be reduced as the chip temperature is elevated. This high-temperature friendly operation of PCM can be advantageously used to design 3D die-stacking memory systems. They propose a hybrid memory design where a large portion of PCM is used as a primary memory space and a small portion of DRAM is used as a write-buffer to reduce the number of writes to PCM. They also propose an OS-level paging scheme that takes into account the memory reference characteristics of applications and migrates the hot-modified pages from PCM to DRAM so that the lifetime degradation of PCM is alleviated. Since access to DRAM takes much less energy than that to PCM, their technique also improves the energy efficiency of the memory system.

Lee et al. [18] propose a memory management technique for hybrid PCM-DRAM memory to hide the slow write performance of PCM. Their technique uses methods such as dirty bit clearing and frequency accumulation to accurately estimate future write references. They observe that using write history alone performs better than using both read and write history in estimating future write references. Also, by using temporal locality and frequency characteristics, more accurate estimates of write references can be obtained. Based on these observations, they propose a page replacement algorithm called CLOCK-DWF (CLOCK with dirty bits and write frequency) that reduces the number of PCM write operations by using DRAM to absorb most of the write references. Reduction in number of writes to PCM translates into improved energy efficiency.

Seok et al. [20] propose a migration based page caching technique for PCM-DRAM hybrid main memory system. Their technique aims to overcome the problem of the long latency and low endurance of PCM. For this, read-bound access pages are kept in PCM and write-bound access pages are kept in DRAM. Their technique uses separate read and write queues for both PCM and DRAM and uses page monitoring to make migration decisions. Write-bound pages are migrated from PCM to DRAM and read-bound pages are migrated from DRAM to PCM. The decision to migrate is taken as follows: when a write access is hit and the accessed page is in PCM write queue, it is migrated. Similarly, if a read access is hit and the accessed page is in the DRAM read queue, it is migrated.

Ramos et al. [12] propose a PCM-DRAM hybrid design for improving energy efficiency of main memory. Their technique uses a hardware-driven page placement policy. Their policy leverages the memory controller to monitor program access patterns and uses this information to migrate pages between DRAM and PCM, and translate the memory addresses coming from the cores. Further, the operating system periodically updates its page mappings based on the translation information used by the memory controller. Since most frequently accessed pages reside in DRAM, the high write latency of accessing PCM is avoided, which also improves performance and saves energy.

Dhiman et al. [2] propose a hybrid main memory system composed of DRAM and PCM. Their memory system exposes DRAM and PCM addressability to the software. In their technique, data placement is performed based on the write frequency to data. If the number of writes to a PCM page exceeds a threshold, the contents of the page are copied to another page (either in DRAM or PCM) for achieving PCM wear-leveling. Movement of hot pages to DRAM leads to saving of energy due to faster and more energy efficient DRAM accesses.

Liu et al. [16] study the variable partitioning problem on a hybrid main memory designed with PCM and DRAM for DSP systems. They propose ILP (integer linear programming) formulations and heuristic algorithms, such that the energy efficiency of PCM can be leveraged while also minimizing the performance and lifetime degradation caused by PCM writes. The limitation of this approach is that the ILP formulation may not capture all the factors. Also, several approximations may be required for managing its complexity and simplifying the model.

In context of PCM-DRAM hybrid main memory, Meza et al. [27] propose a technique for efficiently managing the metadata (such as tag, replacement-policy information, valid, and dirty bits) for data in a DRAM cache at a fine granularity. Their technique uses the observation that storing metadata off-chip in the same row as their data can exploit DRAM row buffer locality. Further, it also reduces the access latency from two row buffer conflicts (one for the metadata and another for the datum itself). Based on this, their technique only caches the metadata for recently accessed rows on-chip using a small buffer. Since metadata needed for data with temporal or spatial locality is cached on-chip, it can be accessed with the same latency as an SRAM tag store. This provides better energy efficiency than using a large SRAM tag store.

Yoon et al. [28] propose row-buffer locality aware caching policies for hybrid PCM-DRAM main memories. Their technique works on the observation that both DRAM and PCM have row buffers, with (nearly) same latency and bandwidth. However, the cost of row buffer misses in terms of latency, bandwidth, and energy is much higher in PCM than in DRAM. Based on this, their technique avoids allocating in PCM data that frequently causes row buffer misses. Such data are allocated (cached) in DRAM, whereas the data that frequently hits in the row-buffer are stored in PCM. Further, since PCM has much higher write latency/power than read latency/power, their technique uses a caching policy such that the pages that are written frequently are more likely to stay in DRAM.

Tian et al. [19] present a task-scheduling based technique for addressing the challenges of hybrid DRAM-PCM main memory. They study the problem of task-scheduling, assuming that a task should be entirely placed in either PCM bank or DRAM bank. Their approach works for different optimization objectives such as 1. minimizing the energy consumption of hybrid memory for a given PCM and DRAM size and given PCM endurance 2. minimizing the number of writes to PCM for a given PCM and DRAM size and given threshold on energy consumption and 3. minimizing PCM size for a given DRAM size, given threshold on energy consumption and PCM endurance.

Zhou et al. [35] propose a writeback-aware bandwidth partitioning scheme for hybrid DRAM-PCM main memory system. Due to the slow writes to PCM, the bandwidth to PCM acts as a bottleneck. Their technique uses a writeback-aware analytical model to derive the allocation strategy for bandwidth partitioning of PCM. Using this, PCM service cycles are partitioned among applications to improve performance and energy efficiency. They also use a scheme which dynamically selects the partitioning weights to different applications.

4.2 Read-before Write Techniques

Several authors utilize the fact that reads from PCM are significantly faster than writes to PCM. Thus, by reading the data before writing and taking intelligent action, write operations can be partially or fully avoided, which leads to saving of energy. The limitation of these techniques is that the amount of improvement which can be achieved using them depends on the data-pattern and bit-level variations. Also, for these techniques to be effective, the latency and energy cost of the read operation must be significantly lower than that of the write operation. We now summarize a few of these techniques.

Zhou et al. [8] propose a technique which works by avoiding redundant bit writes to PCM. Their technique performs a read before write, and writes only those bits to PCM which have changed. They also propose wear-leveling techniques which uniformly spread the writes to PCM. Taken together, these techniques significantly reduce the write to PCM main memory which improves its energy efficiency and lifetime.

Cho et al. [6] propose a technique named, Flip-N-Write to improve PCM write bandwidth, write energy, and write endurance under an instantaneous write power constraint. Their technique works on the observation that many bit-writes to PCM are redundant. Their technique replaces a write operation with read-modify write operation to skip writing a bit if the bit being written is same as the originally stored bit. Further, to restrict the maximum number of bits which are written, they use a "flip" bit. If storing the flipped value of data requires less number of bit-write operations, their technique stores the data in flipped form and changes the flip bit to ON. Using their technique, the write bandwidth can be potentially doubled, which also improves the write endurance and reduces the write energy. Compared to the redundant-bit write-avoidance technique of [8], Flip-N-Write introduces the additional overhead of the flip bit and retrieving the original data when the flip bit is ON.

Xu et al. [42] propose data manipulation techniques to reduce the write energy of PCM main memory. Their technique works on the observation that PCM read incurs much less energy than PCM writes. Also, write of different value to a PCM cell incurs significantly different energy. Their technique uses selective-XOR operations to bias the data value distribution. For a given word to be written and originally stored word, their technique finds an optimal bit-pattern such that writing a XOR-masked value of word to be written with bit pattern leads to minimum write energy.

Jacobvitz et al. [41] propose a technique named FlipMin, which uses coset coding to reduce the number of writes to main memory. Coset coding performs a one-to-many mapping from each dataword to a coset of vectors. Using this, for each write, their technique selects a vector which minimizes the number of bits that must flip. This reduces the write-traffic to PCM main memory, which also translates into saving of energy. They have shown that Flip-N-Write is a degenerate instance of FlipMin. Thus, FlipMin provides a generalization of Flip-N-Write, which is one of its advantage. However, the extra flexibility comes at the cost of additional overhead of finding suitable coset of vectors.

Hay et al. [4] propose a technique to reduce write power consumption of PCM banks. Their technique is based on the observation that typically only a small portion of the bits (for example, less than 25% on average) are written to which consume power. Their technique monitors the number of bits that will change on a write and hence, need to be written. This gives an estimate of the number of bits and hence, amount of power consumed in a write. Then, to not exceed the power budget, the memory controller issues writes only when there is enough power to support them.

Fang et al. [40] propose a technique called "SoftPCM" which utilizes the error tolerance characteristic of video applications to relax the accuracy of write operations. It is well-known that several multimedia applications have the inherent ability of error tolerance [83]. Thus, a slight error in multimedia data may not be perceived by the human end-users. Their technique leverages this fact and provisions that if the stored old data in PCM are very close to the new data to be written, the write operation is cancelled and the old data are taken as the new data. This leads to significant reduction in write-traffic which also reduces the energy consumption of PCM. The difference between SoftPCM and the other techniques is that SoftPCM may introduce data-errors which may not be tolerable in several application domains. Other techniques do not introduce such errors. Thus, SoftPCM is only useful for those workloads or application domains, which are tolerant to minor errors in the data.

4.3 Techniques for Reducing PCM Write Traffic

As discussed before, the write latency and energy of PCM are significantly higher than that of DRAM. Towards this, researchers have proposed techniques which reduce the number of writes to PCM memory. The reduction in write-traffic improves the performance and also reduces the dynamic energy of PCM main memory. Note that redundant writes may be avoided by either actual comparison with originally stored data [6, 7, 36–41] or using flags which track whether the portions of the data being written have actually changed and then writing only the changed data-words [10, 43]. We now discuss a few techniques for reducing PCM write-traffic.

Hu et al. [44] propose a technique for reducing the number of writes to PCM main memory. Their technique is based on data migration and re-computation. In an embedded CMP (chip multiprocessor) having scratch-pad memory, their technique migrates data to the scratch-pad memory of a different core to avoid write-backs of shared data. Thus, by temporarily storing the data on scratch-pad, their technique reduces the number of write-backs. Their technique uses program analysis to determine when and where the data should be migrated. They also propose data re-computation to reduce the number of write activities by discarding the data which should have been written back to the main memory and recomputing these data when they are needed. They model the problem of data migration as a shortest path problem. Also, they propose an approach to find the optimal data migration path with minimal cost for both dirty data and clean data. The limitation of this technique is that in performance-critical systems, recomputing the data may introduce unacceptable latency overheads. Also, program analysis may not be feasible in all scenarios.

Huang et al. [45] propose a register-allocation based technique with re-computation to reduce the number of store instructions to non-volatile memory. Register allocation refers to multiplexing a large number of target program variables onto a small number of physical registers. The less the number of physical registers a processor contains, the more number of spills will be generated. Each spill is mapped to one store instruction and one (or few) load instructions during the compilation process. Traditional register allocation process does not distinguish read and write activities and does not try to minimize writes. Huang et al. use graph-coloring approach to extend traditional register allocation technique with re-computation to reduce non-volatile memory write activities by reducing store instructions. Their technique discards a set of carefully-selected actual spills and re-computes them when they are needed.

Xia et al. [43] propose a technique for reducing writes to PCM main memory. Their technique works on the observation that a large fraction of words in a cache line being written back to memory are not actually modified. Their technique utilizes the burst writes of unmodified data and consolidates multiple writes targeting the same row into one write. Their technique records whether a data-word is modified and propagates this information to the memory controller. Their technique consolidates those writes that access the same row of the same bank, have no read command between them that accesses the same column and also fulfill the condition that the sum of modified data blocks of these write commands is equal to or less than the burst length. The saving of time due to write-consolidation leads to improvement in energy efficiency of the memory system. The difference between the technique of [43] and other redundant-write avoidance techniques, such as Flip-N-Write [6] is that the former reduces the writes at the level of 8-byte words, while the latter ones reduce it at the level of individual bits.

4.4 Techniques Utilizing Asymmetry in Write Times

Due to the physical properties of PCM material, the latency and energy of writing a "one" to PCM array is different from that of writing a "zero". This introduces inefficiency in PCM write, since the total write time is decided by the slower of the two write operations. Several techniques work on this observation and aim to remove this inefficiency. This leads to improvement in energy efficiency and performance of PCM memory systems. We now discuss some of these techniques.

Mirhoseini et al. [38] propose a coding-level technique for saving PCM energy, which is based on the observation that PCM set and reset energy costs are not equal. Their technique aims at minimizing the energy cost of rewriting to PCM by designing low overhead data encoding methods. Their encoding scheme utilizes PCM bitwise manipulation ability during the word overwrites such that only the bits which are changing for the new word compared to the original word would require overwriting. They propose an ILP (integer linear programming) method and employ dynamic programming to produce codes for uniformly distributed data. The limitation of data-encoding based techniques, such as [36, 38, 41, 42] is that encoding and decoding of data may introduce non-trivial overheads in terms of hardware and latency.

Yue et al. [55] a two-stage-write scheme for reducing PCM write latency and the energy consumption. Their scheme works on the observation that in PCM, writing a one takes longer time but less electrical current than writing a zero. Based on this, their scheme distinguishes writing the bit zero and writing the bit one by dividing a write into two stages. In the writing zero stage, all zeros are written at a fast speed. In writing one stage, all ones are written with increased parallelism such that the power constraint is not violated. This becomes possible since writing a one takes much less electric current than writing a zero. Thus, two-stage-write can effectively reduce write latency to PCM array. This improves the performance and also reduces the energy consumption. The difference between the work of [55] and [4] is that the former accounts for the difference between response times and power consumptions for writing a zero and a one, while the latter minimizes the write power requirement by tracking the number of modified bits in a block.

Qureshi et al. [53] propose a technique named PreSet to alleviate the problem of slow writes. Their technique works on the observation that PCM writes are slow only in one direction (SET operation) and are almost as fast as reads in the other direction (RESET operation). Thus, a write operation to a line in which all memory cells have been SET before the write will consume much less time. Based on this, their technique pro-actively SETs all the bits in a given memory line much before the anticipated write to that memory line. As soon as a line becomes dirty in the cache, their technique initiates a SET request for that line, which allows a large window of time for the SET operation to complete. The reduction in effective write latency of PCM also leads to saving of energy. The difference between the work of [53] and [55] is that the former exploits only 'time' asymmetry while the latter exploits both 'time' and 'energy' asymmetry of write operations. Also, PreSet may increase write-traffic to main memory, while the technique of Yue et al. does not generate extra write-traffic to main memory.

4.5 Memory Access Scheduling Techniques

Several researchers propose architectural techniques to manage accesses to main memory with a view to optimize performance and save energy. While the read-before-write techniques discussed above aim to reduce writes to PCM at the level of bits and depend on 'difference' between new and originally stored data, memory access scheduling techniques reduce writes at

the level of memory system or memory devices and do not depend on the difference between new and originally stored data. The discussion of a few techniques follows.

Lee et al. [52] propose using multiple row-buffers inside a PCM chip, which reduces the read latency and also the write energy through write coalescing. Multiple writes to the same location are absorbed in the buffers, thus resulting in much smaller number of write-backs to the PCM array. They also propose a technique which uses multiple dirty bits in the cache blocks to enable partial writes. Using this, the number of bit updates are reduced by not writing untouched, clean data portion in a dirty cache block to the main memory when the cache block is replaced. Their techniques save PCM energy and also increase the lifetime of PCM.

Qureshi et al. [84] propose a write cancellation and write pausing technique to indirectly improve the PCM read performance. Although a higher value of write latency can be tolerated using buffers and large write bandwidth, once a write request is scheduled for service to a PCM bank, a subsequent read access to the same bank needs to wait until the write access has completed. Thus, the slow write can increase the effective latency of read accesses and since read accesses are latency-critical, this may severely affect the program performance. Write cancellation policy aborts an on-going write if a read request arrives to the same bank and the write operation is not close to completion. It avoids aborting an ongoing write that has completed more than a threshold percentage of its service time and this threshold can be adapted during runtime.

A limitation of write-cancellation technique is that it requires some writes to be re-executed which incurs power and bandwidth overhead. To avoid this overhead, Qureshi et al. [84] propose write pausing technique. This technique utilizes fundamental characteristic of PCM that most multi-bit PCM devices use iterative write algorithms. In each iteration data are written and the current state of the device is compared with the desired state. Write pausing allows iterative write algorithms to potentially pause a write request at the end of each write iteration, complete a pending read request, and then resume the paused write request.

4.6 Minimizing PCM Main Memory Power by Managing Caches

In the memory hierarchy of modern processors, main memory and caches exist close to each other and thus, intelligent management of caches can translate into optimization of performance and energy efficiency at main memory. The design goal of these techniques is to minimize cache miss-rate and/or writebacks to reduce the accesses to memory, which leads to saving in energy of memory. Several researchers propose techniques for this, we now discuss a few of them.

Ferreira et al. [47] propose a cache replacement policy for saving PCM main memory energy. Their approach aims to reduce the write-back traffic to main memory. The policy is called N-Chance where N can be varied. This policy evicts the least recently accessed clean page from cache, unless all of the N least recently accessed pages are dirty, if so, it evicts the least recently accessed page. For the case when $N = 1$, this policy becomes the conventional LRU (least-recently used) policy. They have shown that for a proper choice of N, their policy can be significantly better than the LRU policy.

Fedorov et al. [48] propose a technique for simultaneously reducing both the miss rate and writeback rate in the LLC to save PCM main memory power. Their technique works on the observation that on a miss, replacing a clean block is advantageous than replacing a dirty LRU block, since it reduces the number of writeback operations. This, however, also has the disadvantage since the replaced clean cache block may generate more misses. To address this, they note that in general, with LRU replacement policy, most of the cache hits are distributed to only a few MRU (most recently used) blocks. Thus, to avoid generating writeback traffic, while also keeping low miss-rate, those clean blocks in the LRU stack can be replaced which have less hits relative to the dirty LRU ones. The techniques of both [47] and [48] work by changing the cache replacement policy to minimize writebacks, but the difference between them is that the former does not record the hit-information and works only based on LRU-ordering, while the technique of [48] aims to control miss-rate also by recording the number of hits to different ways.

Bock et al. [85] propose a technique to save PCM energy and increase its endurance by avoiding useless write-backs. They define a write-back to a lower level cache to be useless when the data that are written back are not used again by the program. As an example, a useless write-back results when a dirty cache line (block) that belongs to a dead memory region is evicted from the cache. Their technique assumes that suitable schemes can be employed to detect dead memory regions in different parts of memory, such as heap, stack and global memory. Assuming that such information is available, their technique estimates the maximum energy savings that could be achieved by avoiding useless write-backs. Further, since writes are not on critical path of execution, avoiding useless write-backs does not have a significant influence on performance.

Wang et al. [9] propose a technique to reduce the number of writebacks from LLC to PCM main memory for reducing the power consumption of PCM memory. The technique predicts blocks that are frequently written back in the LLC and then, aims to keep highly reused dirty cache blocks in the LLC. It classifies the LLC blocks into frequent writeback blocks (which are written back to main memory with high frequency within a certain access interval) and infrequent writeback blocks (which are not written into main memory frequently and can be clean or dirty). It dynamically partitions the LLC sets into a frequent writeback list and a infrequent writeback list and keeps a best size of each list in the LLC. Since the frequent-writeback data are stored in the LLC, it reduces

write-induced interference as well as energy consumption of PCM. The difference between the technique of [85] and that of [9] is that the former requires control flow analysis performed by compiler or annotations/hints by the programmer to detect dead memory regions, while the latter (the one by [9]) does not require this and works simply by using the information available during application execution.

Yoon et al. [86] use a design space exploration approach for finding the optimal configuration of cache hierarchy in a system with non-volatile memory. Use of non-volatile devices enables designing caches with higher capacity and hence, the cache hierarchy in modern systems is becoming deeper with L4 and L5 caches (e.g. off-chip DRAM caches). They consider both performance and power while performing the experiments. They consider cache hierarchy of different depth, where the cache at any level can be designed with either SRAM, DRAM or PCM. Also the main memory could be designed with DRAM or PCM. They observe that a large last level cache (LLC) designed with PCM can improve energy efficiency by reducing the costly off-chip accesses. Also, deep cache hierarchies are less energy efficient than flat hierarchies (2 or 3 levels).

4.7 Techniques for Managing MLC PCM Memory

Although PCM MLC devices offer more density than SLC (single level cell) devices, they also present significant challenges. Several techniques have been proposed to address these challenges and we now discuss a few of these techniques.

For MLC devices to work properly, precise reading of resistance values is required. As the number of levels increase, the resistance region assigned to each data value decreases significantly. Thus, the read latency of MLC devices may increase linearly or exponentially with the number of bits. Qureshi et al. [87] present a memory architecture which aims to achieve the latency, lifetime and energy of SLC devices in the common case, while still achieving the high memory capacity of MLC device. Their technique divides the main memory in two regions, one with high-density, high-latency which uses MLC mode, and another with low-latency, low-density that uses half the number of bits per cell than high-density region. By tracking the memory requirements of the workloads, their technique adapts the fraction of both regions. When the workload requires high memory capacity, the system uses capacity benefits of MLC device. When the workload requirements can be satisfied with SLC (or fewer bits per level cell), the system increases the size of SLC region to avoid increased energy and latency. This is achieved by restricting the number of levels used in a MLC device to emulate a fewer bits per cell device. To avoid high latency for frequently accessed pages, the system transfers a page from high-density region to low-density (faster) region when the page is accessed. Thus, by achieving high capacity and low latency, their technique improves the energy efficiency of PCM memory systems.

Wang et al. [36] propose a technique for mitigating the energy overhead of MLC PCM devices. Their technique works on the observation that there are significant value-dependent energy variations in programming MLC PCM. Thus, by using data encoding, the write energy can be reduced. In a 2-bit PCM, there are four states, viz. 00, 01, 10 and 11. They show that programming states 00 and 11 require significantly less energy than programming the other states. Thus, data encoding is used to increase the 00 and 11 states in writing the data. They also use data comparison write (DCW) approach to enhance the effectiveness of the data encoding scheme.

Joshi et al. [39] present a circuit and microarchitecture-level technique to address the high write latency of MLC-PCM. The write latency and energy of PCM vary significantly with target resistance level and the initial state of PCM cell. Their technique adapts the programming scheme of MLC PCM by taking into consideration the initial state of the cell, the target resistance to be programmed and the effect of process variation on the programming current profile of the MLC. For states mapped at lower resistance values, they use single reset pulse programming and for states mapped at higher resistance values, they use staircase programming. Also, data comparison writes (DCW) is used to enhance the effect of their technique. Also, when the cell is already present in the stable completely set state, their technique skips initialization sequence for programming, which further improves the write latency and energy efficiency.

5 Conclusion

Driven by the quest for exascale performance, modern processors, data-centers and supercomputers use large sized memory. To alleviate the limitations posed by conventional technologies, designers have recently explored new technologies for designing cache and memory hierarchy. The use of phase change memory (PCM) as a universal choice for main memory presents both challenges and opportunities. It is clear that while PCM provides a promising alternative to conventional DRAM, it also has several limitations such as large power consumption. We believe that in the near future, the challenges of PCM power consumption would need to be simultaneously addressed at different levels, as follows.

Device level: At device level, innovations in physical properties of PCM should improve its write endurance and reduce its write latency and energy. This may even enable integration of PCM at the level of cache, DRAM and hard-disk. Also, scaling of PCM to smaller feature sizes will help in greatly increasing the integration density.

Architecture level: At architectural level, techniques such as 3D-stacking, use of photonics

for designing energy proportional interfaces for PCM memory etc. are expected to dramatically reduce data-transfer latency and improve capacity and bandwidth of PCM memory systems.

System level: At system level, insights from algorithm and application-level analysis will aid in designing more effective management policies. For example, by utilizing the specific properties of application domains or data-structures, write coalescing techniques can be applied more effectively.

Taken together, in very near-future, these advances may enable use of PCM as a stand-alone memory technology which can be used as a complete replacement for conventional memory technologies.

In this paper, we have presented a review of techniques proposed for power management of phase change memory. We believe that our work will be useful for both beginners and experts in the field of PCM. Also, it will help them in gaining insights into the working of architectural techniques of PCM power management and encourage them to improve these techniques even further.

References

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

[2] G. Dhiman, R. Ayoub, and T. Rosing, "PDRAM: a hybrid PRAM and DRAM main memory system," in *ACM/EDAC/IEEE Design Automation Conference (DAC)*, pp. 664–669, 2009.

[3] S. Mittal, "A Survey of Architectural Techniques For DRAM Power Management," *International Journal of High Performance Systems Architecture*, vol. 4, no. 2, pp. 110–119, 2012.

[4] A. Hay, K. Strauss, T. Sherwood, G. H. Loh, and D. Burger, "Preventing PCM banks from seizing too much power," in *IEEE/ACM International Symposium on Microarchitecture (MICRO)*, pp. 186–195, 2011.

[5] S. Raoux *et al.*, "Phase-change random access memory: A scalable technology," *IBM Journal of Research and Development*, vol. 52, no. 4.5, pp. 465–479, 2008.

[6] S. Cho and H. Lee, "Flip-n-write: a simple deterministic technique to improve pram write performance, energy and endurance," in *IEEE/ACM International Symposium on Microarchitecture (MICRO)*, pp. 347–357, 2009.

[7] B.-D. Yang, J.-E. Lee, J.-S. Kim, J. Cho, S.-Y. Lee, and B.-G. Yu, "A low power phase-change random access memory using a data-comparison write scheme," in *IEEE International Symposium on Circuits and Systems (ISCAS)*, pp. 3014–3017, 2007.

[8] P. Zhou, B. Zhao, J. Yang, and Y. Zhang, "A durable and energy efficient main memory using phase change memory technology," in *ACM SIGARCH Computer Architecture News*, vol. 37, pp. 14–23, 2009.

[9] Z. Wang, S. Shan, T. Cao, J. Gu, Y. Xu, S. Mu, Y. Xie, and D. A. Jiménez, "WADE: Writeback-aware Dynamic Cache Management for NVM-based Main Memory System," *ACM Transactions on Architecture and Code Optimization (TACO)*, vol. 10, no. 4, pp. 51:1–51:21, 2013.

[10] M. K. Qureshi, V. Srinivasan, and J. A. Rivers, "Scalable high performance main memory system using phase-change memory technology," *ACM SIGARCH Computer Architecture News*, vol. 37, no. 3, pp. 24–33, 2009.

[11] J. Tominaga, T. Kikukawa, M. Takahashi, and R. Phillips, "Structure of the optical phase change memory alloy, Ag–V–In–Sb–Te, determined by optical spectroscopy and electron diffraction," *Journal of applied physics*, vol. 82, no. 7, pp. 3214–3218, 1997.

[12] L. E. Ramos, E. Gorbatov, and R. Bianchini, "Page placement in hybrid memory systems," in *International Conference on Supercomputing (ICS)*, pp. 85–95, 2011.

[13] H. Park, S. Yoo, and S. Lee, "Power management of hybrid DRAM/PRAM-based main memory," in *48th Design Automation Conference*, pp. 59–64, ACM, 2011.

[14] 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 *IEEE International Conference on Parallel architectures and compilation techniques (PACT)*, pp. 93–102, 2013.

[15] W. Zhang and T. Li, "Exploring phase change memory and 3D die-stacking for power/thermal friendly, fast and durable memory architectures," in *International Conference on Parallel Architectures and Compilation Techniques (PACT)*, pp. 101–112, 2009.

[16] T. Liu, Y. Zhao, C. J. Xue, and M. Li, "Power-aware variable partitioning for DSPs with hybrid PRAM and DRAM main memory," in *ACM/EDAC/IEEE Design Automation Conference (DAC)*, pp. 405–410, 2011.

[17] D.-J. Shin, S. K. Park, S. M. Kim, and K. H. Park, "Adaptive page grouping for energy efficiency

in hybrid PRAM-DRAM main memory," in *ACM Research in Applied Computation Symposium (RACS)*, pp. 395–402, 2012.

[18] S. Lee, H. Bahn, and S. Noh, "Characterizing Memory Write References for Efficient Management of Hybrid PCM and DRAM Memory," in *International Symposium on Modeling, Analysis and Simulation of Computer and Telecommunication Systems (MASCOTS)*, pp. 168–175, 2011.

[19] W. Tian, J. Li, Y. Zhao, C. J. Xue, M. Li, and E. Chen, "Optimal task allocation on non-volatile memory based hybrid main memory," in *ACM Symposium on Research in Applied Computation*, pp. 1–6, 2011.

[20] H. Seok, Y. Park, and K. H. Park, "Migration based page caching algorithm for a hybrid main memory of DRAM and PRAM," in *ACM Symposium on Applied Computing (SAC)* , pp. 595–599, 2011.

[21] S. Baek, H. G. Lee, C. Nicopoulos, and J. Kim, "A dual-phase compression mechanism for hybrid DRAM/PCM main memory architectures," in *ACM Great Lakes symposium on VLSI (GLSVLSI)*, pp. 345–350, 2012.

[22] S. Kwon, D. Kim, Y. Kim, S. Yoo, and S. Lee, "A case study on the application of real phase-change RAM to main memory subsystem," in *Design, Automation & Test in Europe (DATE)*, pp. 264–267, 2012.

[23] T. J. Ham, B. K. Chelepalli, N. Xue, and B. C. Lee, "Disintegrated control for energy-efficient and heterogeneous memory systems," in *IEEE International Symposium on High Performance Computer Architecture (HPCA)*, pp. 424–435, 2013.

[24] 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 *ACM/EDAC/IEEE Design Automation Conference (DAC)*, pp. 888–896, 2012.

[25] H. B. Sohail, B. Vamanan, and T. Vijaykumar, "MigrantStore: Leveraging Virtual Memory in DRAM-PCM Memory Architecture," tech. rep., Purdue University, 2012.

[26] X. Zhang, Q. Hu, D. Wang, C. Li, and H. Wang, "A read-write aware replacement policy for phase change memory," in *Advanced Parallel Processing Technologies* (O. Temam, P.-C. Yew, and B. Zang, eds.), vol. 6965 of *Lecture Notes in Computer Science*, pp. 31–45, 2011.

[27] J. Meza, J. Chang, H. Yoon, O. Mutlu, and P. Ranganathan, "Enabling efficient and scalable hybrid memories using fine-granularity dram cache management," *IEEE Computer Architecture Letters*, vol. 11, no. 2, pp. 61–64, 2012.

[28] H. Yoon, J. Meza, R. Ausavarungnirun, R. A. Harding, and O. Mutlu, "Row buffer locality aware caching policies for hybrid memories," in *IEEE International Conference on Computer Design (ICCD)*, pp. 337–344, 2012.

[29] H. G. Lee, S. Baek, C. Nicopoulos, and J. Kim, "An energy-and performance-aware DRAM cache architecture for hybrid DRAM/PCM main memory systems," in *IEEE International Conference on Computer Design (ICCD)*, pp. 381–387, 2011.

[30] G. Wu, J. Gao, H. Zhang, and Y. Dong, "Improving pcm endurance with randomized address remapping in hybrid memory system," in *IEEE International Conference on Cluster Computing*, pp. 503–507, 2011.

[31] J.-W. Hsieh and Y.-H. Kuan, "Double Circular Caching Scheme for DRAM/PRAM Hybrid Cache," in *IEEE International Conference on Embedded and Real-Time Computing Systems and Applications (RTCSA)*, pp. 469–472, 2012.

[32] M. V. Vamsikrishna, Z. Su, and K.-L. Tan, "A Write Efficient PCM-Aware Sort," in *Database and Expert Systems Applications*, pp. 86–100, Springer, 2012.

[33] L. Ramos and R. Bianchini, "Exploiting phase-change memory in cooperative caches," in *IEEE International Symposium on Computer Architecture and High Performance Computing (SBAC-PAD)*, pp. 227–234, 2012.

[34] W. Hwang and K. H. Park, "HMMSched: Hybrid Main Memory-Aware Task Scheduling on Multicore Systems," in *Future Computing*, pp. 39–48, 2013.

[35] M. Zhou, Y. Du, B. R. Childers, R. Melhem, and D. Mosse, "Writeback-aware Bandwidth Partitioning for Multi-core Systems with PCM," in *International Conference on Parallel Architectures and Compilation Techniques (PACT)*, 2013.

[36] J. Wang, X. Dong, G. Sun, D. Niu, and Y. Xie, "Energy-efficient multi-level cell phase-change memory system with data encoding," in *IEEE International Conference on Computer Design (ICCD)*, pp. 175–182, 2011.

[37] J. Chen, R. C. Chiang, H. H. Huang, and G. Venkataramani, "Energy-aware writes to non-volatile main memory," *ACM SIGOPS Operating Systems Review*, vol. 45, no. 3, pp. 48–52, 2012.

[38] A. Mirhoseini, M. Potkonjak, and F. Koushanfar, "Coding-based energy minimization for phase change memory," in *ACM/EDAC/IEEE Design Automation Conference (DAC)*, pp. 68–76, 2012.

[39] M. Joshi, W. Zhang, and T. Li, "Mercury: A fast and energy-efficient multi-level cell based phase change memory system," in *IEEE International Symposium on High Performance Computer Architecture (HPCA)*, pp. 345–356, 2011.

[40] Y. Fang, H. Li, and X. Li, "SoftPCM: Enhancing Energy Efficiency and Lifetime of Phase Change Memory in Video Applications via Approximate Write," in *IEEE 21st Asian Test Symposium (ATS)*, pp. 131–136, 2012.

[41] A. N. Jacobvitz, R. Calderbank, and D. J. Sorin, "Coset coding to extend the lifetime of memory," in *IEEE 19th International Symposium on High Performance Computer Architecture (HPCA)*, pp. 222–233, 2013.

[42] W. Xu, J. Liu, and T. Zhang, "Data manipulation techniques to reduce phase change memory write energy," in *International Symposium on Low Power Electronics and Design (ISLPED)*, pp. 237–242, 2009.

[43] F. Xia, D. Jiang, J. Xiong, M. Chen, L. Zhang, and N. Sun, "DWC: dynamic write consolidation for phase change memory systems," in *28th ACM international conference on Supercomputing (ACM)*, pp. 211–220, 2014.

[44] J. Hu, C. J. Xue, Q. Zhuge, W.-C. Tseng, and E. H.-M. Sha, "Write activity reduction on non-volatile main memories for embedded chip multiprocessors," *ACM Transactions on Embedded Computing Systems (TECS)*, vol. 12, no. 3, pp. 77:1–77:27, 2013.

[45] Y. Huang, T. Liu, and C. J. Xue, "Register allocation for write activity minimization on non-volatile main memory," in *IEEE Asia and South Pacific Design Automation Conference (ASP-DAC)*, pp. 129–134, 2011.

[46] N. Barcelo, M. Zhou, D. Cole, M. Nugent, and K. Pruhs, "Energy efficient caching for phase-change memory," in *Design and Analysis of Algorithms*, pp. 67–81, Springer, 2012.

[47] A. P. Ferreira, M. Zhou, S. Bock, B. Childers, R. Melhem, and D. Mossé, "Increasing PCM main memory lifetime," in *Design, Automation & Test in Europe (DATE)*, pp. 914–919, 2010.

[48] V. V. Fedorov, S. Qiu, A. L. N. Reddy, and P. V. Gratz, "ARI: Adaptive LLC-memory Traffic Management," *ACM Transactions on Architecture and Code Optimization (TACO)*, vol. 10, no. 4, pp. 46:1–46:19, 2013.

[49] S. Yoo, E. Lee, and H. Bahn, "LDF (less dirty first): dirtiness-aware cache replacement policy for PCM main memory," *Electronics Letters*, vol. 49, no. 25, pp. 1607–1609, 2013.

[50] R. Rodríguez-Rodríguez, F. Castro, D. Chaver, L. Pinuel, and F. Tirado, "Reducing writes in phase-change memory environments by using efficient cache replacement policies," in *Design, Automation and Test in Europe*, pp. 93–96, 2013.

[51] Y. Du, M. Zhou, B. Childers, R. Melhem, and D. Mossé, "Delta-compressed caching for overcoming the write bandwidth limitation of hybrid main memory," *ACM Transactions on Architecture and Code Optimization (TACO)*, vol. 9, no. 4, pp. 55:1–55:20, 2013.

[52] B. C. Lee, E. Ipek, O. Mutlu, and D. Burger, "Architecting phase change memory as a scalable DRAM alternative," *ACM SIGARCH Computer Architecture News*, vol. 37, no. 3, pp. 2–13, 2009.

[53] M. K. Qureshi, M. M. Franceschini, A. Jagmohan, and L. A. Lastras, "PreSET: improving performance of phase change memories by exploiting asymmetry in write times," in *International Symposium on Computer Architecture (ISCA)*, pp. 380–391, 2012.

[54] J. Yue and Y. Zhu, "Exploiting subarrays inside a bank to improve phase change memory performance," in *Design, Automation & Test in Europe (DATE)*, pp. 386–391, 2013.

[55] J. Yue and Y. Zhu, "Accelerating write by exploiting PCM asymmetries," in *IEEE International Symposium on High Performance Computer Architecture (HPCA)*, pp. 282–293, 2013.

[56] B. Li, S. Shan, Y. Hu, and X. Li, "Partial-SET: write speedup of PCM main memory," in *Design, Automation and Test in Europe Conference and Exhibition (DATE)*, pp. 1–4, IEEE, 2014.

[57] Z. Li, R. Zhou, and T. Li, "Exploring high-performance and energy proportional interface for phase change memory systems," in *IEEE 19th International Symposium on High Performance Computer Architecture (HPCA)*, pp. 210–221, 2013.

[58] R. Bheda, J. Poovey, J. Beu, and T. Conte, "Energy efficient phase change memory based main memory for future high performance systems," in *International Conference on Green Computing (IGCC)*, pp. 1–8, 2011.

[59] Q. Li, L. Jiang, Y. Zhang, Y. He, and C. J. Xue, "Compiler directed write-mode selection for high performance low power volatile PCM," in *ACM SIGPLAN/SIGBED LCTES*, pp. 101–110, 2013.

[60] L. Jiang, Y. Zhang, B. R. Childers, and J. Yang, "FPB: Fine-grained power budgeting to improve write throughput of multi-level cell phase change memory," in *IEEE/ACM International Symposium on Microarchitecture (MICRO)*, pp. 1–12, 2012.

[61] X. Dong and Y. Xie, "AdaMS: Adaptive MLC/SLC phase-change memory design for file storage," in *IEEE Asia and South Pacific Design Automation Conference (ASP-DAC)*, pp. 31–36, 2011.

[62] Z. Shao, Y. Liu, Y. Chen, and T. Li, "Utilizing PCM for Energy Optimization in Embedded Systems," in *IEEE Computer Society Annual Symposium on VLSI (ISVLSI)*, pp. 398–403, 2012.

[63] J. Hu, Q. Zhuge, C. J. Xue, W.-C. Tseng, and E. H.-M. Sha, "Software enabled wear-leveling for hybrid PCM main memory on embedded systems," in *Design, Automation & Test in Europe (DATE)*, pp. 599–602, 2013.

[64] T. Wang, D. Liu, Z. Shao, and C. Yang, "Write-activity-aware page table management for PCM-based embedded systems," in *Asia and South Pacific Design Automation Conference (ASP-DAC)*, pp. 317–322, 2012.

[65] J. Hu, Q. Zhuge, C. J. Xue, W.-C. Tseng, and E. H. Sha, "Optimizing Data Allocation and Memory Configuration for Non-Volatile Memory Based Hybrid SPM on Embedded CMPs," in *IEEE International Parallel and Distributed Processing Symposium Workshops (IPDPSW)*, pp. 982–989, 2012.

[66] M. Zhou, S. Bock, A. P. Ferreira, B. Childers, R. Melhem, and D. Mossé, "Real-time scheduling for phase change main memory systems," in *IEEE International Conference on Trust, Security and Privacy in Computing and Communications (TrustCom)*, pp. 991–998, 2011.

[67] S. Kwon, S. Yoo, S. Lee, and J. Park, "Optimizing video application design for phase-change RAM-based main memory," *IEEE Transactions on VLSI Systems (TVLSI)*, vol. 20, no. 11, 2012.

[68] S. Mittal, J. S. Vetter, and D. Li, "WriteSmoothing: Improving Lifetime of Non-volatile Caches Using Intra-set Wear-leveling," in *ACM Great Lakes Symposium on VLSI (GLSVLSI)*, pp. 139–144, 2014.

[69] P. Mangalagiri, K. Sarpatwari, A. Yanamandra, V. Narayanan, Y. Xie, M. J. Irwin, and O. A. Karim, "A low-power phase change memory based hybrid cache architecture," in *ACM Great Lakes symposium on VLSI (GLSVLSI)*, pp. 395–398, 2008.

[70] Y. Joo, D. Niu, X. Dong, G. Sun, N. Chang, and Y. Xie, "Energy-and endurance-aware design of phase change memory caches," in *Conference on Design, Automation and Test in Europe*, pp. 136–141, European Design and Automation Association, 2010.

[71] J. Li, L. Shi, C. J. Xue, C. Yang, and Y. Xu, "Exploiting set-level write non-uniformity for energy-efficient NVM-based hybrid cache," in *IEEE Symposium on Embedded Systems for Real-time Multimedia (ESTIMedia)*, pp. 19–28, 2011.

[72] J. Wang, X. Dong, Y. Xie, and N. P. Jouppi, "i2WAP: Improving non-volatile cache lifetime by reducing inter-and intra-set write variations," in *IEEE International Symposium on High Performance Computer Architecture (HPCA)*, pp. 234–245, 2013.

[73] S. Guo, Z. Liu, D. Wang, H. Wang, and G. Li, "Wear-resistant hybrid cache architecture with phase change memory," in *IEEE NAS*, pp. 268–272, 2012.

[74] G. Duan and S. Wang, "Exploiting narrow-width values for improving non-volatile cache lifetime," in *Design, Automation & Test in Europe (DATE)*, DATE '14, pp. 52:1–52:4, 2014.

[75] S. Mittal, "Using cache-coloring to mitigate inter-set write variation in non-volatile caches," tech. rep., Iowa State University, 2013.

[76] M. Poremba and Y. Xie, "NVMain: An Architectural-Level Main Memory Simulator for Emerging Non-volatile Memories," in *IEEE Computer Society Annual Symposium on VLSI (ISVLSI)*, pp. 392–397, 2012.

[77] X. Dong, C. Xu, Y. Xie, and N. P. Jouppi, "Nvsim: A circuit-level performance, energy, and area model for emerging nonvolatile memory," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 31, no. 7, pp. 994–1007, 2012.

[78] G. Zhu, K. Lu, and X. Li, "SCM-BSIM: A Non-Volatile Memory Simulator Based on BOCHS," in *Emerging Technologies for Information Systems, Computing, and Management*, pp. 977–984, Springer, 2013.

[79] X. Li, K. Lu, and X. Zhou, "SIM-PCM: A PCM Simulator Based on Simics," in *IEEE International Conference on Computational and Information Sciences (ICCIS)*, pp. 1236–1239, 2012.

[80] S. Mittal, "A survey of techniques for improving energy efficiency in embedded computing systems," *International Journal of Computer Aided Engineering and Technology (IJCAET)*, 2014.

[81] S. K. Khaitan and J. D. McCalley, "Optimizing cache energy efficiency in multicore power system simulations," *Energy Systems*, pp. 1–15, 2013.

[82] S. Mittal, "A survey of architectural techniques for improving cache power efficiency," *Sustainable Computing: Informatics and Systems*, vol. 4, no. 1, pp. 33–43, 2014.

[83] S. Mittal, Z. Zhang, and Y. Cao, "CASHIER: A Cache Energy Saving Technique for QoS Systems," *26th IEEE International Conference on VLSI Design (VLSID)*, pp. 43–48, 2013.

[84] M. K. Qureshi, M. M. Franceschini, and L. A. Lastras-Montaño, "Improving read performance of phase change memories via write cancellation and write pausing," in *IEEE International Symposium on High Performance Computer Architecture (HPCA)*, pp. 1–11, 2010.

[85] S. Bock, B. Childers, R. Melhem, D. Mossé, and Y. Zhang, "Analyzing the impact of useless write-backs on the endurance and energy consumption of PCM main memory," in *IEEE International Symposium on Performance Analysis of Systems and Software (ISPASS)*, pp. 56–65, 2011.

[86] D. H. Yoon, T. Gonzalez, P. Ranganathan, and R. S. Schreiber, "Exploring latency-power tradeoffs in deep nonvolatile memory hierarchies," in *Conference on Computing Frontiers*, pp. 95–102, ACM, 2012.

[87] M. K. Qureshi, M. M. Franceschini, L. A. Lastras-Montaño, and J. P. Karidis, "Morphable memory system: a robust architecture for exploiting multi-level phase change memories," in *ACM SIGARCH Computer Architecture News*, vol. 38, pp. 153–162, 2010.