2017 30th International Conference on VLSI Design and 2017 16th International Conference on Embedded Systems

DyPhase: A Dynamic Phase Change Memory Architecture with Symmetric Write Latency

Ishan G Thakkar, Sudeep Pasricha
Department of Electrical and Computer Engineering
Colorado State University, Fort Collins, CO, U.S.A.
{ishan.thakkar, sudeep}@colostate.edu

***Abstract*— A major challenge for the widespread adoption of phase change memory (PCM) as main memory is its asymmetric write latency. Generally, for a PCM, the latency of a SET operation (i.e., an operation that writes '1') is 2-5 times longer than the latency of a RESET operation (i.e., an operation that writes '0'). For this reason, the average write latency of a PCM system is limited by the high-latency SET operations. This paper presents a novel PCM architecture called *DyPhase*, which uses partial-SET operations instead of the conventional SET operations to introduce a symmetry in write latency, thereby increasing write performance and throughput. However, use of partial-SET decreases data retention time. As a remedy to this problem, *DyPhase* employs novel distributed refresh operations in PCM that leverage the available power budget to periodically rewrite the stored data with minimal performance overhead. Experiments with PARSEC benchmarks indicate that our *DyPhase* architecture achieves 16.2% less average latency and 27.6% less EDP on average over other PCM architectures from prior works.**

***Index Terms*—phase change memory, asymmetric write latency, distributed refresh, endurance, lifetime**

I. Introduction

For decades since its emergence, dynamic random access memory (DRAM) has supported the demands on main memory capacity and performance [1]-[5]. In the "big data" era, the next generation memory systems must be capable of offering the memory capacity required to maintain large data structures [1]. However, scaling DRAM below 22nm to increase capacity is currently unknown [2], and at 22nm, DRAM dissipates a large amount of leakage power [3]. These limitations make DRAM less suitable for next generation main memory in the "big data" era.

Recent advances have enabled Phase Change Memory (PCM) as a leading technology that can alleviate the leakage and scalability shortcomings of traditional DRAM [6]. Compared to DRAM, PCM has non-volatility, superior scalability, lower standby leakage power, and comparable read latency. Single-Level-Cell (SLC) PCM differentiates between two resistance levels of phase change material to store a logic bit (logic '0' or '1'). An SLC PCM cell requires different durations and strengths of programming current to write '0' and '1'. The RESET operation, which writes '0' to a PCM cell, uses a short but high-amplitude current pulse to program the phase change material to the amorphous state that has high resistance. In contrast, the SET operation, which writes '1' to a PCM cell, utilizes 2-5× longer and 2-4× lower-amplitude current pulse to program the phase change material to the polycrystalline state that has lower resistance. When a page/line of data is written to PCM, the longer SET operation limits the write latency. The resultant longer average write latency in PCM may incur as much as 60% performance degradation [16]. Therefore, it is imperative to reduce the inherently long write latency of PCM to enable PCM as a viable replacement for DRAM in the next generation memory systems.

In recent years, many techniques have been proposed to minimize the effect of longer write latency on PCM performance [8]-[17]. These methods either hide longer write latency by scheduling writes among idle bank cycles [8] or provide architecture-level solutions for reducing write latency in multi-level cell (MLC) PCMs [9]-[13]. The write scheduling methods are not suitable for high-performance systems where there are little-to-no idle cycles between memory accesses, whereas the architecture-level methods in [9]-[13] are specific to MLC PCM technology, which is less preferred over SLC PCMs due to reliability and variation susceptibility issues. Thus, neither of these methods is general enough to address the long write latency in SLC PCMs. Some other techniques (e.g., [14]-[16]) have been proposed that utilize latency-aware coding schemes to encode data words, which relax the need to write '1' bits in some write operations resulting in a reduced average write latency [14]-[16]. However, these methods cannot eliminate the need to write '1' bits in every write operation, thus, limiting the improvement in average write latency.

Different from these methods, B. Li et al. in [17] propose a PCM architecture (referred to as *pSET_PCM* architecture henceforth) that uses partial-SET operations instead of the conventional SET operations to reduce the average write latency of PCM. But partial-SET operations result in lower data retention periods. To overcome data retention related reliability issues, the *pSET_PCM* architecture reactively schedules full-SET (conventional SET) operations on the partial-SET cells. Reactive scheduling of full-SET operations stall regular read/write operations, which in turn limits the improvements in average write latency of the *pSET_PCM* architecture. In contrast, this paper presents a novel PCM architecture called *DyPhase*, which overcomes the shortcomings of the *pSET_PCM* architecture and enables more reliable and energy-efficient use of partial-SET operations, thereby achieving greater write performance. The novel contributions of this paper are:

- A dynamic phase change memory (PCM) architecture called *DyPhase* that introduces a symmetry in PCM write latency by using partial-SET operations to write '1's, and achieves improved write performance;
- A distributed refresh method called *Reset-pSet Refresh* as part of the *DyPhase* architecture, which utilizes available power budget to periodically restore data bits in the PCM cells, thereby ensuring reliable data retention in partial-SET cells.

II. Background and Motivation

A. Phase Change Memory

As shown in Fig. 1(a), a PCM storage cell is comprised of two electrodes separated by a resistive heater element and phase change material, which is typically a chalcogenide material $Ge_2Sb_2Te_5$ (GST) [18]. The bottom electrode is connected to the heater element at one end, whereas it is connected to an access

2380-6923/16 $31.00 © 2016 IEEE
DOI 10.1109/VLSID.2017.6

device at the other end. The access device is typically one of the following three devices: a standard NMOS transistor, a bipolar junction transistor (BJT), or a PN-junction diode. As shown in Fig. 1(b), the GST can be switched between two states (i.e., polycrystalline and amorphous) with dramatically different electrical resistance. The amorphous high-resistance (usually in the MΩ range) state is used to represent a binary '0', while the crystalline low-resistance (usually in the KΩ range) state represents a '1'.

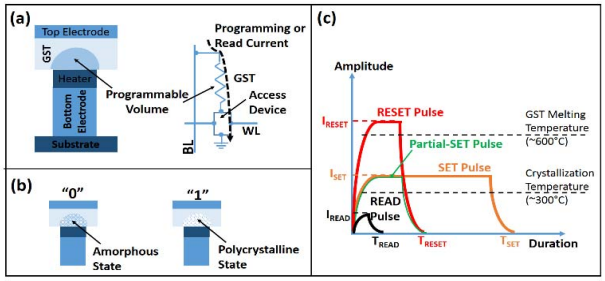

Fig. 1: (a) The basic structure of a PCM cell, (b) different states of a PCM cell, (c) programming (write) and read pulses for PCM cell.

A PCM memory system has three basic operations: read, SET, and RESET. A PCM cell can be read by simply sensing the current flow through it. Due to the large gap between the two resistance levels ('0' and '1') of the GST material, the sensed currents of these two states differ by a large magnitude. The latency of the read operation is typically tens of nanoseconds.

As shown in Fig. 1(c), in the write operations, different heat-time profiles are applied to switch cells from one state to the other. As shown in the figure, to RESET a PCM cell, a strong programming current pulse (I_{RESET}) of short duration (T_{RESET}) is required. This programming pulse raises the temperature of the GST material to its melting point, after which the pulse is quickly terminated. Consequently, the small region of melted material cools quickly, leaving the GST material programmed in the amorphous state. As the region of the GST material that melts with I_{RESET} is small, the required T_{RESET} is short. In contrast, to SET a PCM cell, a programming current pulse of a longer duration T_{SET} and weaker strength I_{SET} is applied to program the cell from the amorphous state to the polycrystalline state. As the crystallization rate is a function of temperature, and given the variability of PCM cells within an array, reliable crystallization requires a longer programming pulse than amorphization [6].Therefore, the SET latency of a PCM cell is longer than both the RESET and the read latency.

From a system perspective, it is highly likely that both SET and RESET operations occur in a typical PCM write operation, where hundreds of bits (typically 512 bits of a cacheline) are programmed in PCM cells. For this reason, the average write latency of a PCM system is limited by the slower SET operations. Therefore, the write latency in PCM memory is longer than the read latency. As mentioned earlier, our *DyPhase* architecture uses partial-SET operations to reduce its write latency to a value that is comparable to its read latency.

B. Concept of Partial-SET Operations

The concept of partial-SET operations used in this paper is based on the studies reported in [17] and [19]. As mentioned earlier, in a typical PCM, SET bits have about 1000× less resistance than RESET bits. However, the resistance contrast of only 10×, which corresponds to 10× contrast in read sense current, is sufficient to detect logic '1' bit as separate from logic '0' bit. As described in [17] and [19], the resistance of a RESET bit (logic '0') can be decreased by 8-10× by applying a SET current pulse of I_{SET} amplitude but only of T_{RESET} duration. In other words, a SET pulse of T_{RESET} duration (Fig. 1(c)) can decrease the resistance of a logic '0' bit by 8-10× to program it as a logic '1' bit. We refer to the SET pulse of I_{SET} magnitude and T_{RESET} duration as a partial-SET pulse and the corresponding cell-programming event as a partial-SET operation. The conventional SET operation with current pulse of I_{SET} amplitude and T_{SET} duration is referred to as a full-SET operation henceforth. Thus, the latency of SET operations can be reduced to be equal to the latency of RESET operations by using partial-SET pulses instead of full-SET pulses, which can achieve the same performance from a system perspective as can be achieved by the ideal write operations with symmetric latency.

Unfortunately, as described in [17], there exists a reliability challenge with partial-SET operations. Due to the resistance drift caused by the thermally activated atomic rearrangement of the amorphous structure [17][20], the resistance of a partial-SET PCM cell increases with time. The phenomenon of resistance drift exhibits a power-law model, $R_t = R_0 \times t^{\nu}$, where R_0 is the initial resistance of cells after write, t is the elapsed time (in seconds) and ν is the drift exponent. When R_t crosses the reference value between the RESET and SET states, the data in PCM cells become invalid. As described in [17], partial-SET cells give readout errors due to resistance drift for elapsed time of more than 4 seconds. Hence, the retention window of partial-SET cells is 4 seconds.

To deal with the low retention period of partial-SET cells, the *pSET_PCM* architecture described in [17], which uses partial-SET operations, maintains a separate partial-SET queue, each entry of which stores the address and elapsed time for a partial-SET line. Then, the *pSET_PCM* controller reactively schedules full-SET write operations on partial-SET PCM lines within their retention window [17]. *Due to reliability concerns, the full-SET operations are prioritized over regular read/write requests. As a result, the regular read/write requests in [17] are stalled while the full-SET requests are being served, which increases the queuing latency of the stalled requests, resulting in an increased average latency for the system.* Thus, the use of partial-SET queue and having to rewrite the partial-SET lines with full-SET operations for reliable data retention limits the latency improvements achieved by *pSET_PCM* from [17]. Our proposed *DyPhase* architecture overcomes these shortcomings and achieves better improvements in PCM latency, throughput, and energy-efficiency. The next section describes our *DyPhase* architecture in detail.

III. DyPhase PCM Architecture: Overview

Before diving into the specifics of our *DyPhase* architecture, we present the general structure and data-organization of the baseline PCM system used in this work. The *DyPhase* and all other PCM architectures used in this paper (for comparison purposes) are implemented on this baseline PCM system with their specific microarchitecture-level attributes. Fig. 2 depicts the structure and data-organization of this baseline PCM system.

As shown in Fig. 2(a), a 4GB PCM DIMM has 8 PCM chips of 512MB size each. The total PCM capacity is divided in 8 logical banks, each of which has 32768 logical rows. Each of these rows

is of 16KB in size and is striped across 8 PCM chips, which makes the single-chip part of each row to be of 2KB in size. As shown in Fig. 2(b), in each single-chip part of a logical bank, the PCM cells are hierarchically organized into blocks and sub-blocks. Peripheral circuitry, such as global decoders, buffers, sense amplifiers (S/As), and write drivers (W/Ds) are shared among blocks and are multiplexed across sub-blocks. For read and write operations, both global (GBL_DEC) and local (LBL_DEC) bit-line decoders, select some bit-lines and connect them to the S/As for reading or the W/Ds for writing. In the case of a PCM write operation, the write current flows from the W/D to the cell ground line through the bitline decoder, bit-line, GST material, and access device. For a read operation, the read current follows the same path except that it originates from the S/A.

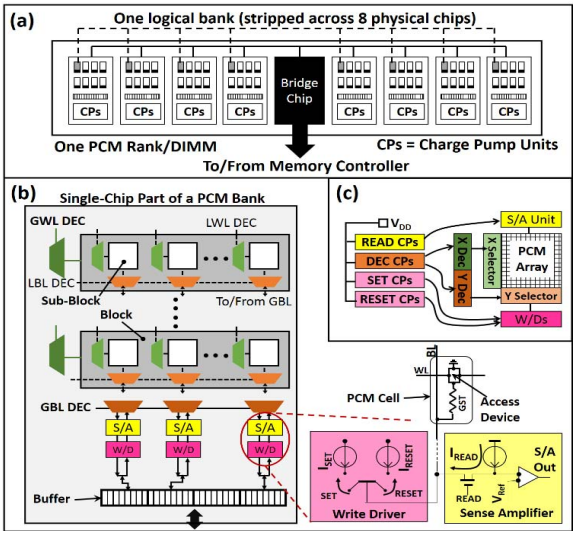

Fig. 2: (a) Architecture of the baseline SLC PCM rank, (b) hierarchical organization of a single-chip part of the baseline PCM bank, (c) a charge pump (CP) system for the baseline PCM array.

As shown in Fig. 2(a), each PCM chip has dedicated charge pump (CP) circuits, which are responsible for providing large amplitude voltage and currents for READ, RESET, and SET operations. For the baseline PCM system, we assume the use of multi-unit modular CPs (as described in [13]). As the amplitudes of currents required for READ, RESET, and SET operations are different, these operations require dedicated multi-unit CPs. Fig. 2(c) shows a multi-unit CP system for a PCM array with a dedicated multi-unit CP per PCM operation.

Table 1 gives the parameter values (voltage, current, energy, delay etc.) for various operations of the basic diode-switched 11nm 4GB SLC PCM system with one PCM DIMM or rank. These values are consistent across all the PCM architectures used in this paper. We first used NVsim [23], a CACTI-based non-volatile memory-modeling tool, to calculate the 22nm diode-switch PCM chip parameters. Then we scaled the calculated parameter values to 11nm PCM technology values following the PCM scaling guidelines given in [6] and [24]. According to the scaling guidelines described in [24], the SET/RESET resistance increases linearly (by *k*) and programming current reduces linearly (by *1/k*) as the PCM feature size scales down by *k*. Note that the scaled values of programming currents (SET/partial-SET/RESET) given in Table 1 are greater than ITRS projected values for 11nm PCM process [2]. The use of higher than ITRS-projected programming currents in our simulations provides significant tolerance against possible discrepancies between the ITRS projections and actual manufacturable solutions. As both SET/partial-SET and RESET resistances increase with process scaling, the dynamic range of resistance (difference in resistance between SET/partial-SET and RESET states) is preserved with process scaling. Moreover, process scaling does not affect read and write latencies, as the read/write latencies are primarily determined by the phase change material (GST in this paper) [6].

As implied in [13], the total number of modular CP units present in the assumed baseline PCM chip decide the peak power provisioning to the PCM chip, which in turn decides how many 1-bit RESET operations can be performed concurrently per chip. Hay et al. calculated in [25] that the peak cell-write current of 168mA provided by a typical DDR3-1066×16 DRAM memory can allow up to 560 simultaneous single-bit RESETs in a typical PCM with I_{RESET} of 300μA. But our assumed baseline PCM system has I_{RESET} of 40μA (Table 1), and modern DRAMs (e.g., [26]) can easily provide peak cell-write current of 200mA (or more) per chip. This implies that our assumed baseline PCM system can support up to 5000 (200mA/40μA) 1-bit RESETs per chip. To be conservative, we assume that our baseline PCM can support up to 4096 (i.e., 512B, equal to eight 64B cache lines) RESET operations per chip. Note that the required number of CP units to support this peak power provisioning and their area and power overheads are given in Table 1.

Table 1: Various parameters for 11nm diode-switched SLC PCM system of 4GB capacity with eight chips per rank

	RESET	SET	READ	Partial-SET
Current pulse amplitude (I - in μA)	40	20	4.2	20
Voltage	2.5	1.5	1.5	1.5
Current pulse duration (T – in ns)	50	150	40	50
Energy per bit (pJ)	5	4.5	0.25	1.5
#CPs per chip	4096	8192	16384	8192
Total CP area per chip (mm^2)	3.7	1.2	0.5	1.2
Total power wastage in CPs (mW)	1250	460	190	460

As mentioned earlier, we implement the key attributes of our proposed *DyPhase* architecture on the above-described baseline PCM. Similar to the *pSET_PCM* architecture described in [17], the *DyPhase* architecture also uses partial-SET operations instead of full-SET operations to write '1's to PCM cells. In the next subsection, we describe *Reset-pSet Refresh*, which is a key microarchitecture-level attribute of our *DyPhase* PCM architecture.

A. Reset-pSet Refresh

As the *DyPhase* PCM architecture uses partial-SET operations instead of full-SET operations, it has to deal with the retention related reliability issues induced by the resistance drift in the partial-SET PCM cells. To deal with these issues, *DyPhase* employs *Reset-pSet Refresh*, which is a distributed refresh technique that periodically restores the data bits in *DyPhase* PCM cells every 4 seconds (the retention window of partial-SET cells as reported in [17]) and ensures reliable data retention. *Reset-pSet Refresh* is different from the distributed/interleaved refresh technique used in

state-of-the-art DRAM systems [4][5][26]-[28] in the following ways: *(i)* the distributed refresh in DRAM restores the charge in DRAM storage cells, whereas *Reset-pSet Refresh* restores the difference in resistance between the partial-SET and RESET states of *DyPhase* PCM cells. *(ii) Reset-pSet Refresh* has longer refresh interval due to the longer data retention time for PCMs than DRAMs. *(iii) Reset-pSet Refresh* refreshes smaller number of cells in parallel per refresh command.

Typically, in the popular interleaved refresh scheme, each refresh command refreshes a certain number of rows depending on the size of the row and memory capacity. One refresh command is scheduled after every refresh interval (*tREFI*). The refresh interval (*tREFI*) is defined as *Retention Time/Total Number of Rows*. The number of rows that need to be refreshed during every refresh command is calculated as *Memory Capacity / (Row Size × Total Number of Rows)* [28]. The time taken by every refresh command to refresh the required number of rows is defined as refresh cycle time (*tRFC*), which depends on the granularity at which the required rows are refreshed.

As shown in Fig. 3(a), in our *DyPhase* PCM, due to the drift in resistance of partial-SET cells, the difference in resistance between the RESET and partial-SET cells becomes very small after 4s (retention period). As a result, logic '1' bits become very marginally distinguishable from logic '0's, potentially harming the validity of stored data. As shown in Fig. 3(b), *Reset-pSet Refresh* restores the difference in resistance between partial-SET and RESET cells to be 8-10×, which ensures that logic '1's are clearly distinguishable from '0's. To accomplish that, as shown in Fig. 3(b), *Reset-pSet Refresh* first reads the cells that are being refreshed and stores their data in a buffer, and then rewrites the buffered data in the cells in multiple write cycles (four write cycles in this case) depending on the granularity of refresh operation. To rewrite the buffered data, *Reset-pSet Refresh* uses RESET (for '0's) and partial-SET (for '1's) operations, which in turn ensures restoration of the difference in resistance between partial-SET and RESET cells. In a PCM system, the granularity of refresh operation is decided from the peak power provisioning to the constituent PCM chips.

As mentioned earlier, each row of a *DyPhase* bank is 16KB in size (striped across 8 *DyPhase* chips) and there are a total of 32768×8=262144 rows in a 4GB *DyPhase* rank, which implies that exactly one row should be refreshed during every refresh command. Moreover, as the retention time for partial-SET cells in *DyPhase* PCM is 4 seconds, the *tREFI* for the *DyPhase* PCM becomes 15.26µs (4s/262144). Our assumed baseline PCM system, and hence the *DyPhase* PCM which is derived from the baseline PCM, can support only up to 4096 1-bit writes per chip, as discussed earlier. But all eight constituent chips of a *DyPhase* rank can function in parallel, which allows up to 4096×8=4KB cells to be written in parallel per *DyPhase* rank. As a result, refreshing/rewriting a row of 16KB size requires four write cycles where one write cycle rewrites only 4KB cells in parallel across all eight constituent chips. The required number of write cycles per refresh command defines the granularity of the refresh operation, which in turn defines the refresh cycle time, *tRFC*.

To evaluate *tRFC* for *Reset-pSet Refresh*, consider Fig. 3(b) again. As shown in the figure, a *Reset-pSet Refresh* cycle is comprised of one read cycle followed by four write cycles. During the read cycle, first, the global and local wordline decoders (GWL DEC and LWL DEC) decode the target row address in 1.5ns (12ns at 90nm technology taken from [6], and scaled down to 11nm by *1/k*). Then, the S/As read data from the target row in 40ns (Table 1) and store it in the buffer in 2ns (scaled down value taken from [6]). As this buffer is used for refresh operations, we refer to it as refresh-buffer henceforth. After the read cycle, the first write cycle (out of total 4 write cycles) starts after providing 2ns time for data stability in buffers. Every write cycle starts with bit-line decode phase, which takes about 2ns. Then, the W/Ds write the buffered data back in the row cells in 50ns (defined by partial-SET operations; Table 1). Every write cycle of a *Reset-pSet Refresh* command refreshes 4KB cells in parallel (512B per chip) and precedes an idle period of 10ns to limit the average power within the budget. From this analysis, the *tRFC* time for *Reset-pSet Refresh* is reckoned to be 285.5ns (45.5ns (40ns+1.5ns+2ns+2ns) for the read cycle and 240ns (4× (50ns+10ns)) for total 4 write cycles).

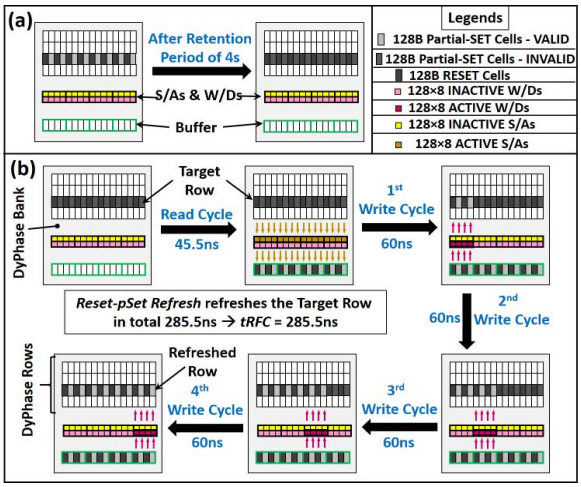

Fig. 3: (a) Schematic of drift in resistance of partial-SET cells of a single-chip portion of a DyPhase bank over the retention period of 4s, (b) schematic of a *Reset-pSet Refresh* cycle.

In summary, the *Reset-pSet Refresh* method schedules one refresh command to a DyPhase rank every 15.26µs, which refreshes one logical row of 16KB size in 285.5ns. As the *Reset-pSet Refresh* fully utilizes the available power budget, the entire *DyPhase* rank is stalled for the *tRFC* time of 285.5ns during which it does not serve any other memory requests. Thus, *Reset-pSet Refresh* enables *DyPhase* PCM to use partial-SET operations without any retention related reliability issues, and stalls the memory operation only for 1.87% of the total run time of memory.

Note that, as only one row is refreshed at a time in the entire PCM system, the refresh-buffer can be shared among all the banks. Therefore, only one refresh-buffer is required per *DyPhase* rank and its size is exactly equal to the size of a logical row (i.e., 16KB). According to our area and power analysis, using NVSim [23] based standard cell library scaled for 11nm technology node, the refresh-buffer consumes 0.02mm^2 area and dissipates 10.3pJ dynamic energy and 2mW leakage power.

IV. Evaluation

A. Evaluation Setup

We performed trace-driven simulation analysis to compare the *Reset-pSet Refresh* enabled *DyPhase* architecture (referred to as

Reset-pSet DyPhase henceforth) with the baseline PCM (Section III) and other PCM architectures from prior works such as Partial-SET PCM (*pSET_PCM*) [17] and Write-Once-Memory-Code PCM (*WOMC_PCM*) [16]. We implement a cycle-accurate model of the baseline PCM system described in Section III in DRAMSim2 [29] and use it as our simulation platform. We model the *Reset-pSet DyPhase*, *pSET_PCM*, and *WOMC_PCM* architectures by implementing their unique characteristics on the cycle-accurate model of the baseline PCM system. We use the resulting cycle-accurate models of PCM architectures as PCM-only main memory systems and not as PCM-DRAM hybrid memory systems, because a PCM-only memory system draws forth the true unmasked behavior of system performance.

Memory access traces for the PARSEC benchmark suite [22] were extracted from detailed cycle-accurate simulations using gem5 [30]. We use the PARSEC benchmark suite because it covers a wide spectrum of working sets, locality, data sharing, synchronization, and off-chip traffic [22]. We ran each PARSEC benchmark for a "warm-up" period of 1 billion instructions and captured memory access traces from the subsequent 1 billion instructions extracted. These memory traces were then provided as inputs to the DRAMSim2-based simulator platform. Table 1 and Table 2 give the configuration of the PCM-only main memory system. Table 3 gives the configuration of Gem5 that was used for this study. We chose a clock frequency of 5GHz, as the processors of the future are projected to operate at such frequencies, e.g., IBM's zEC12 and z13 processors [7]. An FCFS scheduling scheme and rank:row:col:bank address mapping scheme were used for all simulations. Average latency, average write throughput, and energy-delay product values for the memory subsystem were obtained from DRAMSim2.

Table 2: Configuration of the PCM-only main memory system

Main memory	11nm process; 4GB capacity; 1 channel; 64-bit channel width; 1 rank per channel; 8 chips per rank; 8 banks per rank; 8 banks interleaved on 8 chips; LPDDR2-NVM interface
PCM chip	$4F^2$ diode-switched cell; 8-bit width; 512MB capacity; 1V V_{DD}; 133MHz; 200mA peak current provisioning
Memory controller	First-Come-First-Serve scheduling; rank:row:col:bank address mapping; 32-entry R/W queues; in-order R/W issue

Table 3: Gem5 simulation configuration

Number of Cores	4	L2 Coherence	MOESI
L1 I Cache	16KB	Frequency	5 GHz
L1 D Cache	16KB	Issue Policy of cores	In-order
L2 Cache	128KB	# Memory Controllers	1
ISA/OS	ARM/64-bit	Cache Associativity	Direct Mapped

B. *Evaluation Results*

This section presents the average latency, write throughput, net power, and energy-delay product results for various PCM architectures obtained for PARSEC benchmark workloads.

Fig. 4 shows average memory access latency for various PCM architectures across the PARSEC benchmarks. The values in Fig. 4 are normalized to the average latency values of the baseline PCM architecture. As evident from the figure, our *Reset-pSet DyPhase* architecture yields 16.5% less average latency on average compared to the other PCM architectures. More specifically, *Reset-pSet DyPhase* yields 7.8%, 8.2%, and 29.6% less average latency compared to *WOMC_PCM*, *pSET_PCM*, and baseline PCM respectively. In spite of using partial-SET operations instead of full-SET operations to write '1's, the use of a partial-SET queue and having to rewrite the partial-SET cells with full-SET operations reduces the latency improvements achieved by the *pSET_PCM* architecture compared to *WOMC_PCM* and *Reset-pSet DyPhase*. Along the same lines, as *WOMC_PCM* needs to use full-SET operations for every alternate write to the same memory location, it cannot completely eliminate the need to use full-SET operations. As a result, it incurs greater average latency compared to *Reset-pSet DyPhase* architecture.

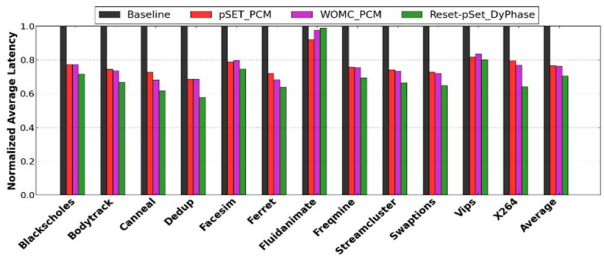

Fig. 4: Average memory access latency values for various PCM architectures across PARSEC benchmarks. The values are normalized w.r.t. average latency values of the baseline PCM.

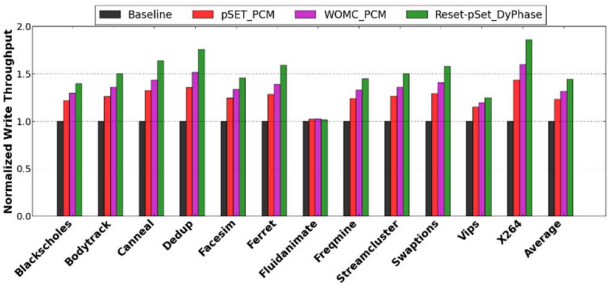

Fig. 5: Write throughput values for various PCM architectures across PARSEC benchmarks. The values are normalized w.r.t. the write throughput values of the baseline PCM.

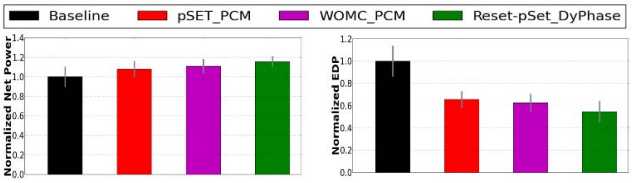

Fig. 6: (a) Net power, (b) energy delay product (EDP) values for various PCM architectures averaged across PARSEC benchmarks. The values are normalized w.r.t. the baseline PCM. The error bars represent standard deviation of values across the PARSEC benchmarks.

Fig. 5 shows write throughput values for various PCM architectures across the PARSEC benchmarks. As evident from the figure, *Reset-pSet DyPhase* yields 9.6%, 17.3%, and 44.3% more write throughput over *WOMC_PCM*, *pSET_PCM*, and baseline PCM respectively. The smaller latency for partial-SET operations and smaller value of *tRFC* overhead result in the greatest write throughput for *Reset-pSet DyPhase* compared to the other PCMs. As evident from Fig. 4 and Fig. 5, *pSET_PCM* yields a comparable value of average latency with *WOMC_PCM*, but yields much greater write throughput. This is because the smaller value of per-access write latency results in a greater value of write throughput for *pSET_PCM*. But as explained earlier, the use of a partial-SET queue in *pSET_PCM* stalls regular read/write requests, which results in comparable value of average latency with *WOMC_PCM*.

As can be observed from Fig. 4 and Fig. 5, the average latency and write throughput values of *Reset-pSet DyPhase* for the *Fluidanimate* application are worse than *WOMC_PCM* and *pSET_PCM*. As discussed in [22], the *Fluidanimate* application exhibits very high parallelization and inter-thread communication overheads, both of which increase with increase in main memory stall time, deteriorating overall system performance. As *Reset-pSet DyPhase* has greater memory stall time due to periodic refresh operations, it has worse average latency and write throughput for the *Fluidanimate* application than *WOMC_PCM* and *pSET_PCM*.

Fig. 6(a) shows net power values for various PCM architectures averaged over 12 applications of the PARSEC benchmark suite. We obtain the net power by adding the burst power (read-write power), background power (leakage power), and refresh power (if any) values. The error bars in the figure represent standard deviation of net power values across the PARSEC benchmarks. It can be observed that *Reset-pSet DyPhase* consumes about 15.4%, 4.2%, and 7% more net power on average over the baseline PCM, *WOMC_PCM*, and *pSET_PCM* respectively. The addition of periodic refresh operations results in greater power consumption for *Reset-pSet DyPhase*. The necessity of rewriting the partial-SET cells with full-SET operations results in greater power consumption for *pSET_PCM* than *WOMC_PCM*.

Fig. 6(b) shows the energy-delay product (EDP) values for various PCM architectures averaged over 12 applications of the PARSEC benchmark suite. The EDP values are obtained by multiplying the energy per access values with the average latency values. The energy per access values are obtained by dividing the net power values with the corresponding throughput values (in terms of number of accesses/sec). The error bars in the figure represent standard deviation of EDP values across the PARSEC benchmarks. As evident from the figure, *Reset-pSet DyPhase* yields about 45.5%, 12.6%, and 16.8% less EDP over the baseline PCM, *WOMC_PCM*, and *pSET_PCM* respectively. It can be concluded from these results that in spite of consuming greater power, our *Reset-pSet DyPhase* yields better energy-efficiency in terms of EDP compared to the other PCM architectures. The significant decrease in average memory access latency for our proposed *Reset-pSet DyPhase* architecture helps drive down its energy costs.

V. CONCLUSIONS

This paper presented a novel PCM architecture called *DyPhase*, which uses partial-SET operations to reduce the average write latency. To remedy retention-related reliability issues associated with partial-SET operations, *DyPhase* periodically refreshes the stored data by using the *Reset-pSet Refresh* technique. *Reset-pSet Refresh* enables *DyPhase* PCM to use partial-SET operations without any retention related reliability issues, and stalls the memory operation only for 1.87% of the total run time of memory. Our evaluation with PARSEC benchmarks indicate that the *Reset-pSet DyPhase* architecture achieves about 16.5% less average latency, 22% more write throughput, and 28.3% less EDP on average over the state-of-the-art *WOMC_PCM*, *pSET_PCM* and baseline PCM architectures. Thus, it can be concluded from the evaluation results that our proposed *Reset-pSet DyPhase* PCM architecture provides a low-latency and energy-efficient solution for future PCM implementations.

ACKNOWLEDGMENTS

This research is supported by grants from SRC and NSF (CCF-1252500, CCF-1302693, ECCS-1646576).

REFERENCES

[1] P. Kogge et al., “ExaScale Computing Study: Technology Challenges in Achieving Exascale Systems”, *tech. rep.*, *DARPA*, 2008.
[2] “Process Integration, Devices and Structures,” *International Technology Roadmap for Semiconductors (ITRS)*, 2013.
[3] M. T. Chang et al., “Technology comparison for large last-level caches (L3Cs): Low-leakage SRAM, low write-energy STT-RAM, and refresh-optimized eDRAM,” in *HPCA*, 2013.
[4] I. Thakkar and S. Pasricha, " 3D-WiRED: A Novel Wide I/O DRAM with Energy-Efficient 3D Bank Organization", *IEEE D&T*, 2015.
[5] I. Thakkar, S. Pasricha, “A Novel 3D Graphics DRAM Architecture for High-Performance and Low-Energy Memory Accesses”, in *ICCD*, 2015.
[6] B. C. Lee et al., “Architecting Phase Change Memory As a Scalable Dram Alternative,” in ISCA, 2009.
[7] "IBM zEnterprise System". URL: https://en.wikipedia.org/wiki/IBM_zEnterprise_System. Accessed: Oct 2016.
[8] Y. Kim et al., “Write performance improvement by hiding R drift latency in phase-change RAM,” in *DAC*, 2012.
[9] L. Jiang et al., “Improving write operations in MLC phase change memory,” in *HPCA*, 2012.
[10] J. Yue and Y. Zhu, “Making Write Less Blocking for Read Accesses in Phase Change Memory,” in *MASCOTS*, *2012.*
[11] C. Pan et al., “3M-PCM: Exploiting multiple write modes MLC phase change main memory in embedded systems,” in *CODES+ISSS*, 2014.
[12] L. Jiang et al., “FPB: Fine-grained Power Budgeting to Improve Write Throughput of Multi-level Cell Phase Change Memory,” in MICRO, 2012.
[13] L. Jiang et al., “A low power and reliable charge pump design for Phase Change Memories,” in *ISCA*, 2014.
[14] S. Cho and H. Lee, “Flip-N-Write: A simple deterministic technique to improve PRAM write performance, energy and endurance,” in MICRO, 2009.
[15] J. Yue and Y. Zhu, “Accelerating write by exploiting PCM asymmetries,” in HPCA, 2013.
[16] J. Li and K. Mohanram, “Write-once-memory-code phase change memory,” in DATE, 2014.
[17] B. Li et al., “Partial-SET: Write speedup of PCM main memory,” in *DATE*, 2014.
[18] H. Horii et al., “A novel cell technology using N-doped GeSbTe films for phase change RAM,” in *Symposium on VLSI Technology,* 2003.
[19] G. W. Burr et al., “The inner workings of phase change memory: Lessons from prototype PCM devices,” in *IEEE GLOBECOM Workshops,* 2010.
[20] W. Zhang and T. Li, “Helmet: A resistance drift resilient architecture for multi-level cell phase change memory system,” in *DSN,* 2011.
[21] M. K. Qureshi et al., “Scalable High Performance Main Memory System Using Phase-change Memory Technology,” in *ISCA*, 2009.
[22] C. Bienia et al., “The PARSEC Benchmark Suite: Characterization and Architectural Implications,” in *PACT*, 2008.
[23] X. Dong et al., “NVSim: A Circuit-Level Performance, Energy, and Area Model for Emerging Nonvolatile Memory,” *TCAD,* vol. 31, no. 7, 2012.
[24] A. Pirovano et al., “Scaling analysis of phase-change memory technology,” in *IEDM*, 2003.
[25] A. Hay et al., “Preventing PCM Banks from Seizing Too Much Power,” in *MICRO*, 2011.
[26] “DDR4 SDRAM MT40A1G4.” Datasheet by Micron, 2014.
[27] I. Thakkar and S. Pasricha, “3D-ProWiz: An Energy-Efficient and Optically-Interfaced 3D DRAM Architecture with Reduced Data Access Overhead", *IEEE TMSCS*, vol. 1, no. 3, 2015.
[28] I. Thakkar and S. Pasricha, “Massed Refresh: An Energy-Efficient Technique to Reduce Refresh Overhead in Hybrid Memory Cube Architectures,” in VLSID, 2016.
[29] P. Rosenfeld et al., “DRAMSim2: A Cycle Accurate Memory System Simulator,” in *CAL*, vol. 10, no. 1, pp. 16–19, Jan. 2011.
[30] N. Binkert et al., “The Gem5 Simulator,” *Comput Arch. News*, vol. 39, no. 2, 2011.