Achieving DRAM-like PCM By Trading Off Capacity For Latency

Irina Alam and Puneet Gupta

***Abstract*—Phase Change Memory (PCM) is considered one of the most promising scalable non-volatile main memory alternatives to DRAM. It provides ∼4x-5x cost per bit advantage over DRAM, thus enabling cost-effective dense main memory solution. However, PCM accesses are slower than DRAM, which leads to significantly poorer overall system performance (upto 80% higher execution time for memory intensive applications based on our analysis). To use PCM as a viable DRAM replacement, the performance gap between the two memory technologies has to be bridged, primarily by improving PCM read latency.**

In this work we propose an optimized PCM architecture, PCM-Duplicate, that trades off capacity to improve PCM read latency. In PCM-Duplicate, every row in the PCM subarray has a duplicate row. During memory read, both the rows are activated simultaneously. As a result, the bitline discharges through two PCM cells. This reduces the discharge time significantly, bringing down the overall sensing latency by >3x compared to baseline PCM. While the overall PCM density benefit over DRAM halves, it still provides 2x more capacity than DRAM while having almost comparable read latency. PCM-Duplicate can either be used as low-cost DRAM main memory alternative or it can be used to replace the DRAM-based last level cache used in today's hybrid main memory systems for the slower PCM memories. Both these system options not only improve main memory capacity but also allow main memory based persistence by replacing DRAM and making the entire main memory non-volatile.

***Index Terms*—Phase Change Memory, Non Volatile Main Memory, Emerging Technology, Storage Class Memory, Error Correcting Code**

I. Introduction

In today's computing systems, main memories serve a pivotal role, sitting in between the processor cores and the slow storage devices. As a result, there is an ever-increasing demand for main memory capacity to fully extract and exploit the processing power of today's high-performance multicore and manycore systems. Though DRAM is still the main memory workhorse, DRAM scaling is, unfortunately, slowing down. Besides, several application contexts need different properties from the main memory such as higher density, lower cost-per-bit, non-volatility, etc. Hence, it is becoming increasingly important to consider alternative technologies that can potentially avoid the problems faced by DRAM and enable new opportunities.

Several emerging non-volatile memory (NVM) technologies are now being considered as potential replacements for or enhancements to DRAM. Most of these new non-volatile technologies (Phase Change Memory[PCM], STT-RAM, Resistive RAM[ReRAM], etc.) promise better scaling, higher density, and reduced cost-per-bit. 3D-XPoint [1], [2] is one such commercially available PCM-based non-volatile main memory that has gained a lot of attention. Unfortunately, NVMs have their own set of challenges and cannot simply replace DRAM in their current form. Compared to DRAM, NVMs have higher read and write latencies and often consume more energy. Besides, most NVM technologies have limited write endurance and also suffer from high stochastic bit error rates [3]. Hence, most current NVM-based main memory systems are hybrid in nature comprising of both DRAM and NVM [2], [4]. A hybrid memory system helps to increase capacity while reducing the impact on performance.

P. Gupta is with the Department of Electrical and Computer Engineering, University of California, Los Angeles.
E-mail: puneetg@ucla.edu
I. Alam was with University of California, Los Angeles.

A hybrid memory system can be configured in two ways [2], [4]. In one configuration (Memory Mode), the smaller but faster DRAM is used as a hardware-managed cache for the denser but slower PCM. However, the DRAM cache is transparent to the operating system (OS) and hence, the total main memory capacity is equal to the total PCM capacity. In the other configuration, the DRAM and PCM are configured as a flat address space, where the OS is aware of both memories for page allocation. Hence, the entire memory capacity can be fully utilized. However, the placement of data between the two memories has to be efficiently managed [4]. The first configuration has the advantage that it can be easily deployed, with DRAM acting as an additional level of caching between the CPU caches and main memory. However, this mode does not ensure non-volatility as data stored in the DRAM will be lost when power is lost. Besides, the DRAM is transparent to the OS. This has a non-negligible impact on overall memory capacity. DRAM has 4-5x higher cost-per-bit as compared to PCM [5], [6] and hence, using DRAM as a transparent cache to mitigate the loss in performance because of the slower PCM increases the overall system cost.

Overall, based on our analysis and past works [7], it seems that improving PCM read latency can provide significant performance gains. If PCM read latency becomes closer to that of DRAM, the use of large DRAM-based caches for performance improvement can be avoided. That would help to increase main memory capacity, reduce memory cost and make the main memory non-volatile. In this work, we propose an optimized PCM architecture (PCM-Duplicate) that helps to lower the sensing time in a PCM array by activating two wordlines in a PCM array simultaneously. In this architecture, data is duplicated across two rows in a PCM array. During a read operation, both rows are activated together. This helps to reduce the overall sensing latency since the bitline voltage now discharges through two cells instead of one, thus, increasing the rate of discharge. The overall read latency becomes almost comparable to that of DRAM while the overall PCM capacity becomes half. Thus, there is a capacity-latency tradeoff. We provide two possible ways of using this reduced capacity faster PCM in today's systems. The first option is to use it as a low-cost alternative to today's DRAM-based main memory subsystem. It provides 2x higher capacity/lower cost at ∼6% higher average execution time compared to a system using DRAM memory. The other option is to use it in a hybrid main memory setup where the PCM-Duplicate acts as the faster cache (lower cost/higher capacity compared to DRAM cache) for the slower PCM main memory. In this setup, the increased cache capacity provides up to 38% (average 5.7%) higher performance than today's baseline hybrid system with DRAM and PCM. Most importantly, in both system options, the entire main memory system is non-volatile and hence, allows easy main memory-based persistence and checkpointing. This reduces a significant overhead that is incurred in today's systems where the application state is stored in much slower non-volatile storage devices and each checkpoint restoration can take as long as 30 minutes [8].

II. Background

This section provides a brief background on two important concepts: details and working of a PCM cell, and the pros and cons of

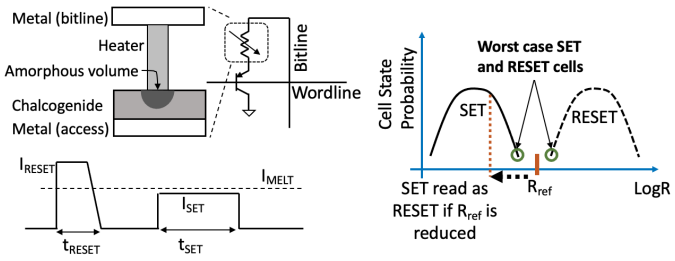

Fig. 1: Structure of PCM cell, overview of SET and RESET current pulses and variation in cell resistance for SET and RESET states.

using PCM-based memories as DRAM replacements.

A. PCM Basics

Phase change memory (PCM) is a type of non-volatile memory. PCM exploits the unique ability of chalcogenide glass of switching between high resistance amorphous and low resistance crystalline states. Figure 1 shows the structure of a PCM cell, typically comprising of Germanium-Antimony-Tellurium or GST. The state of the cell can be changed by heating and the two different states represent the stored data; high resistance RESET represents '0' and low resistance SET represents '1'. Switching from one state to the other requires two different heat-time profiles. As shown in Figure 1, to program a '0', a high power short pulse quickly raises the temperature of the PCM element above the melting point. The pulse is abruptly terminated, and the small region of melted material rapidly cools through thermal conduction, locking the material in the amorphous state. To program a '1', the amorphous element needs to be converted into a polycrystalline state. To do that, a long electric pulse is used to raise the temperature of the PCM element above the crystallization but below the melting point. The temperature needs to be sustained for a lengthy period so that most of the material crystallizes and the target cell resistance is achieved. Thus, programming a '0' (RESET) requires much lower latency than programming a '1' (SET). To read the data stored in the cell, the cell resistance is sensed without changing the state of the cell.

A PCM memory module is similar to today's DRAM array and is split up into small independent banks. As shown in Figure 2, each bank consists of multiple subarrays, where each subarray is made of horizontal wordlines and vertical bitlines. Each bitline is attached to a sense-amplifier circuitry that senses the bitline voltage and outputs a digital value of '0' or '1' based on that. The bitline sense amplifiers drive then drive the global amplifiers shared across all sub-arrays in a bank. The global sense amplifiers boost the voltage and drive the data out of the PCM chips to the processor over a DDRx-based bus.

During a read operation, the read address selects the target bank and activates the corresponding wordline. In voltage-based sensing, which has been commonly used in many industry chips [9], [10], the bitline is precharged to V_{rd}. Once the target wordline is turned on, the bitline starts discharging. The rate of discharge depends on the cell resistance. When in SET state, the discharge is faster and takes lesser time for the bitline voltage to drop below the reference voltage (V_{ref}) as compared to that when the cell is in RESET state. After a pre-determined amount of time (T_{sense}), the sense amplifiers compare the bitline voltage with V_{ref}. If it is higher than V_{ref}, then the cell is considered to be in RESET state, if not, then in SET state. The combination of V_{ref} and T_{sense} is decided based on the cell characteristics or the resistance distribution of the cell at each state and the V_{rd} voltage used. The sensing time is determined conservatively to account for device variations and drift. For target

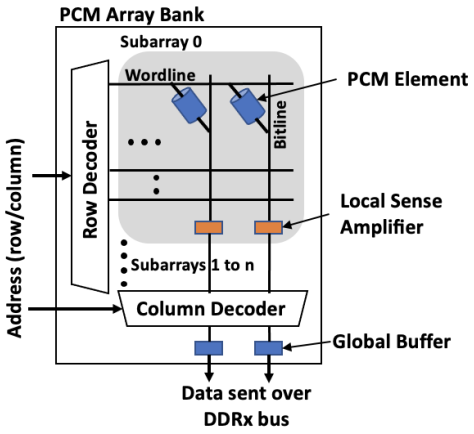

Fig. 2: Organization of a PCM bank

SET and RESET resistance values, the wait time before sensing bitline voltage should be such that the voltage separation is at least 300mV between the worst-case cells of the two states (shown in Figure 1). The sensing latency (T_{sense}) constitutes the largest fraction of the read time in a PCM array [7]. Hence, in this work, we try to reduce the sensing latency using two different optimization schemes.

B. DRAM vs. PCM

DRAM scaling is, unfortunately, slowing down [11] while the demand for main memory capacity is increasing at a very fast rate. Besides, increasing capacity and aggressive technology scaling in modern DRAM chips is significantly impacting manufacturing yield and reliability. With increasing scaling induced error rates, DRAM manufacturers are resorting to increased row/column sparing and within-dram error-correcting codes to maintain acceptable DRAM yields. These techniques add significant area, latency, and energy overheads. As a result, DRAMs have overall become a large contributor to operational cost. The other major disadvantage with DRAM is that it is volatile in nature. Several application contexts today, such as persistent database management, not only demand higher main memory density, but also the ability to store persistent data in main memories instead of using heavyweight filesystems in today's slower persistent storage devices. Phase Change Memory (PCM) has become one of the most promising scalable byte-addressable main memory alternatives to DRAM. PCM provides a significant 4x-5x [5], [6] cost-per-bit advantage over DRAM while being non-volatile and amenable to technology scaling. As a result, it has been studied extensively and manufactured commercially as DRAM replacement towards building a low cost, higher capacity main memory system [5].

However, PCM has its own set of disadvantages when compared to DRAM, the most important being higher read and write latencies, write power, and limited endurance [11], [3]. As a result, in today's systems, PCM is typically used in a hybrid main memory setup comprising of both DRAM and PCM memory modules. DRAM modules have limited capacity but lower read and write latencies as compared to the denser but slower PCM. Thus, while PCM helps to achieve higher capacity main memory at a lower cost, DRAM helps to reduce the performance impact by servicing the more frequent/recent accesses at lower latency and energy overheads. This hybrid memory can be used in two modes: (1) Memory mode, and (2) App Direct mode. In the memory mode, DRAM acts as a hardware-managed cache for the slower and denser PCM. This DRAM-based cache, sitting in between the last level cache (LLC) and the PCM main memory, is transparent to the OS [4], [2] and, therefore, the overall

main memory capacity is equal to only that of the total PCM. Besides, the main memory is not persistent since data sitting in the volatile DRAM will be lost if the system loses power. In the app direct mode, the DRAM and PCM modules are configured as a flat address space and the OS uses both memories for page allocation. This mode has the advantage of providing higher memory capacity, but faces the challenge of efficient data placement and swapping of data between the two memories. The frequently accessed hot pages would need to be identified in the PCM and swapped with the cold data in the DRAM. Since this data management requires additional hardware/software support, this mode is less frequently used as compared to the memory mode in today's systems.

III. Motivation and Past Work

There has been significant research in improving latency and limited endurance of PCM [12], [13]. However, achieving read latency similar to that of DRAM has not been talked about much. One of the past works from Nair et. al. [7] focused on improving read latency by either reducing the reference sensing resistance and using ECC to tolerate the uni-directional sensing errors or by increasing the read voltage and using ECC to tolerate the read disturb errors. We tried to combine the two techniques with stronger ECC to achieve the best possible read latency. We describe it in detail under PCM-ECC Scheme.

A. PCM-ECC Overview: Combination of Previously Proposed Improvements

In PCM-ECC, we combine the previously proposed techniques [7] to reduce the read sensing time by reducing the reference sensing resistance (R_{sense}) and increasing the read voltage (V_{rd}). PCM cells have a variation in cell resistance in both SET and RESET states as shown in Figure 1. The final resistance of the cell depends on the amount of amorphous volume in the cell and how easily the cell crystallizes in either of the two states. The sensing reference resistance is determined by the worst-case SET cell and the read voltage is determined by the worst-case RESET cell. Reducing the sensing resistance leads to lower sensing latency, but also increases sensing circuitry errors. However, this sensing error is unidirectional, where SET gets classified as RESET. On the other hand, increasing the read voltage that the bitlines are pre-charged to before it gets discharged through the cell to the sense amplifier also helps to improve read latency. However, doing so increases error rates due to read disturb. The read current flowing through the cell can accidentally flip the state of the cell. In PCM, read disturb errors are also typically unidirectional and result in RESET switching to SET. Based on prior studies [14], every 30mV increase in sensing voltage increases the read disturb error rate by 3 orders of magnitude.

From past works and PCM device characterization results [15], using a R_{sense} of 10kΩ and V_{rd} of 0.70V results in a sensing time of 69ns and a bit error rate (BER) of 10^{-16}. As proposed in [7], in PCM-ECC, we reduce R_{sense} from 10kΩ to 7kΩ and increase V_{rd} to 0.82V. The combined effect of reduced reference resistance and increased read voltage leads to a significant reduction in sensing time from 69ns to 34ns. But the BER increases exponentially to 10^{-5}, primarily coming from the reduction in reference sensing resistance. This increase in error rate can, however, be mitigated using error correcting codes (ECC). We use a default BER target of 10^{-16}, similar to that in [7]. We show that with a 4-bit rank-level error correcting code (ECC-4) for every 64-bits of data, the probability that a 512-bit memory line would have 5 errors is 3.9×10^{-18}, well below the desired target. The parity storage overhead of ECC-4 is 25-bits per 64-bits of data [16], [17] and incurs a decoding latency of 4 cycles. We assume that the PCM-based memory system will use today's standard DDRx protocol [18]. Thus, the total size of a memory line that is accessed during each memory READ/WRITE operation is 576-bits (512-bits of data and 64-bits of parity bits for rank-level within controller ECC). For ECC-4, with 25-bits of parity per 64-bits of data, the size of the memory line increases to 712-bits. Thus, accessing all the data and parity bits would require two consecutive READ/WRITE operations when using today's DDRx protocol. This is expected to significantly degrade performance. To reduce the impact on performance, we divide the ECC-4 code into two parts: (1) A Single-bit Error Correcting, Multi-bit Error Detecting code (2) A 4-bit Error Correcting Code. The first part of the code requires 8-bits per 64-bits of data and a single cycle of decoding. Thus, to achieve single-error correcting, multi-error detecting property, only one memory access is required. If the decoder detects an uncorrectable error, then the remaining parity bits are fetched using an additional READ and the decoding with error correction takes 4 cycles. For a 512-bit memory line and a BER of 10^{-5}, once every 60,000 reads would require additional access to fetch the extra parity bits.

B. Performance Analysis: PCM-ECC vs. DRAM

With the modifications in reference sensing resistance and read voltage, the PCM sensing time decreases by half. While this improvement is significant, a DRAM-based memory system still achieves higher performance. To understand the difference in performance we simulated a single-core system using cycle accurate Gem5 simulator [19]. We ran several workloads from the CPU SPEC 2017 [20] and GAP [21] suites, where we fast-forwarded 1 billion instructions and ran the simulation for a total of 3 billion instructions. We used a 2GHz single-core processor with a private 32KB I-cache, 64KB D-cache, 512KB L2 cache, and 2MB L3 cache. On average we see that the DRAM-based main memory system outperforms the system with the PCM-ECC by an average of 19%. While the sensing latency improvements in the PCM array provide significant benefit (6.7% average improvement in execution time over baseline PCM-based main memory system), it still lags behind DRAM-based main memory system. This is primarily because the PCM-ECC read latency is still more than twice that of DRAM. Thus, PCM-ECC provides $\sim$3x higher capacity/lower cost compared to DRAM, but has up to 60% higher execution time for memory intensive workloads.

C. Motivation to achieve near-DRAM latency

From the performance results we see that PCM-ECC improves system performance as compared to baseline PCM but it still significantly lags behind DRAM. One way to improve the performance is to have a hybrid main memory system with a DRAM-based cache for slower PCM. But, as mentioned before, the two biggest challenges with that are the limited capacity of the DRAM cache and the lack of non-volatility. Even though the DRAM cache is typically much larger than the LLC, capacity limited applications do not benefit much because of the frequent misses in the DRAM cache. As seen in the evaluations in [7], with a DRAM cache that is 16x larger than the total L3 capacity, the memory-intensive workloads have an average of 20.1 misses per thousand instructions (MPKI). This significantly impacts overall performance. Besides, users cannot utilize the non-volatility advantage of PCM since the DRAM cache is volatile and transparent to the OS. So, even though the OS thinks that it is writing to the non-volatile PCM, the data actually gets written to the DRAM first. If the system loses power before the data in the DRAM cache could be written back to the PCM, the data is lost. Hence, persistence is not guaranteed in such a hybrid memory system. Thus, it is necessary to have a purely PCM-based memory with higher capacity than DRAM while achieving near-DRAM read latency.

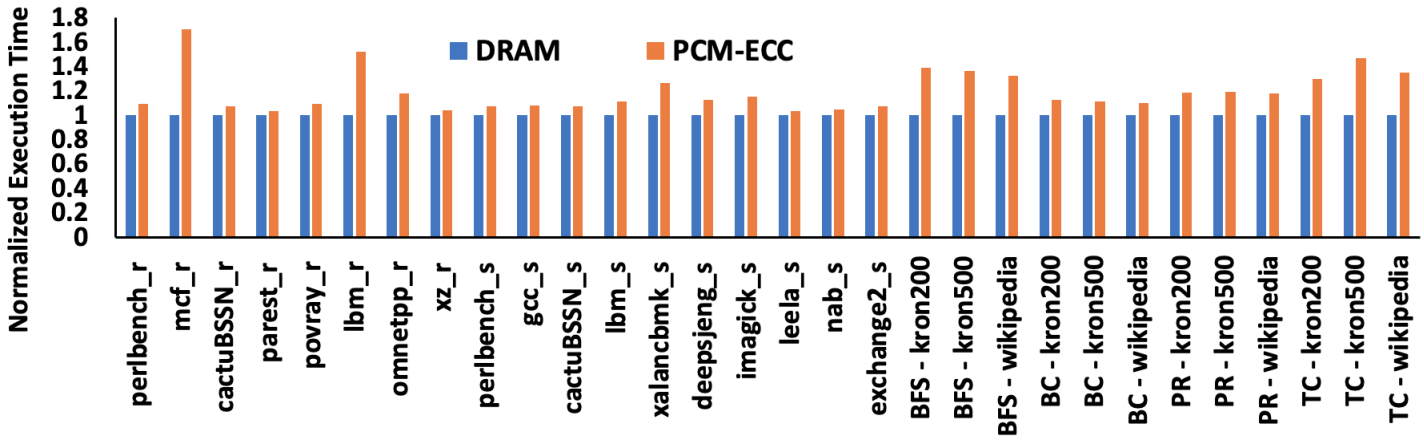

Fig. 3: Normalized Execution Time of SPEC-2017 and GAP workloads comparing DRAM and PCM-ECC based main memory systems. The execution times are normalized against the system using DRAM.

IV. Bridging the Performance Gap Between PCM and DRAM

While PCM-ECC provides larger main memory capacity than DRAM, workloads that do not require the larger capacity and fit within the DRAM suffer because of the larger read/write latencies of the PCM. As a result, we aim to bridge this performance gap by further trading off PCM capacity to achieve near-DRAM read latency. Our proposed reduced capacity PCM-Duplicate architecture has similar read performance as that of DRAM while having ∼2x larger capacity at the same cost compared to DRAM.

A. PCM-Duplicate Overview: PCM with DRAM-like read latency

We propose a PCM-Duplicate scheme that helps to significantly improve PCM read latency by trading off capacity. In this scheme, every row in a PCM subarray will have a duplicate row. The original row and the duplicate row are activated simultaneously when reading from a particular address. As a result, the overall resistance of the PCM cells being sensed is halved as they are connected in parallel. This allows us to reduce the reference resistance (R_{sense}) from 10kΩ to 5kΩ without impacting the bit error rate (BER). We can further reduce R_{sense} and increase read voltage (V_{rd}) to get additional improvements in sensing latency. With $R_{sense} = 4k\Omega$ and $V_{rd} = 0.8V$, we can achieve a sensing latency of 21ns and BER of 10^{-8}. With 2-bit error correction (ECC-2) per 512-bits of data, the probability of having 3 errors is 2.2×10^{-17}, which is well below the desired target of 10^{-16}. The exact ECC protection used is described later in Section IV-C. PCM-Duplicate provides an overall sensing latency reduction of more than 3x compared to the baseline PCM memory and brings the overall read latency closer to that of DRAM (1.3x of DRAM). However, it halves the memory capacity. But PCM has more than 4x cost-per-bit benefit over DRAM [5], [6]. Even after halving the capacity, the cost-per-bit benefit of PCM-Duplicate stands at a significant 2x over DRAM. Keeping the trade-offs of PCM-Duplicate in mind, this reduced capacity PCM with near-DRAM latency is also a good fit as the last level cache for slower PCM main memories. Using PCM-Duplicate as the last level cache for the slower PCM main memory instead of DRAM would provide 2x more cache capacity at the same cost, as well as overall main memory persistence since both last-level cache and main memory are PCM-based and hence, non-volatile.

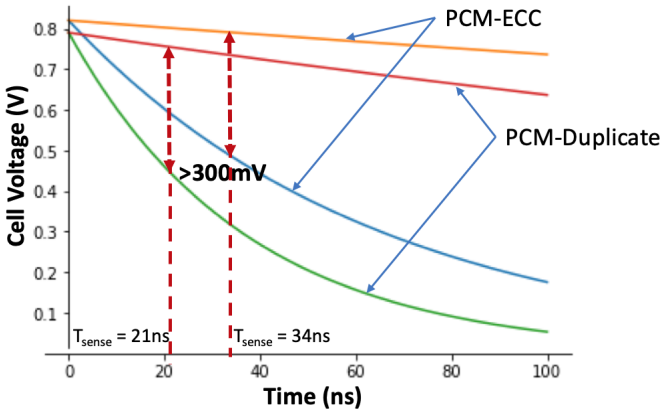

Fig. 4: Sensing latencies of PCM-ECC vs PCM-Duplicate

B. PCM-Duplicate Implementation

To activate two rows simultaneously, the row decoder is modified to activate two wordlines using a single address. So, if row address A is sent, rows A and A' are activated. Since two PCM cells in the same state are connected in parallel (as shown in Figure 5), the total cell resistance becomes half. As a result, the 300mV separation between the worst-case SET and RESET states during discharge can be achieved much faster than in the baseline PCM cell. The PCM can operate in both normal and duplicate modes. The operation mode can be set by the memory controller during boot time by setting a PCM mode register. In normal mode, the row decoder decodes using all row address bits and activates a single row at a time. In duplicate mode, each PCM subarray is halved in capacity and each row has a corresponding duplicate neighboring row. The row decoder masks the least significant bit of the row address and activates two neighboring rows that have the same A[MSB:LSB-1] bits. The original and duplicate rows must be in the same subarray since each subarray has its own set of local sense amplifiers.

C. Reducing Write Time and Energy using ECC and Infrequent Refresh

In PCM-Duplicate, the overall write current increases as two PCM cells are being programmed simultaneously. The increase in total write current translates to an overall increase in write energy. One way to reduce the write energy consumption is by reducing the write latency. As provided in [11], the SET latency is ∼4x that of RESET

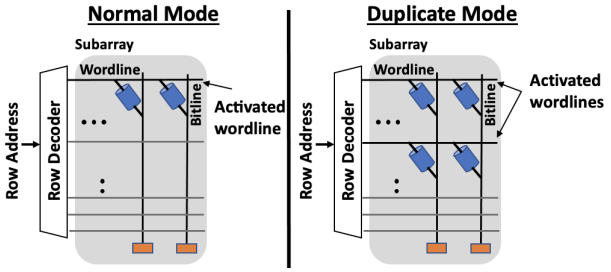

Fig. 5: The two operation modes in PCM-Duplicate

latency. Reducing the SET pulse width results in a smaller volume of crystalline chalcogenide, which increases the resistance of the cell. Besides, the resistance of these partially SET cells increases further over time [22], eventually resulting in retention error when the SET cells are sensed as RESET. Based on the characterization results presented in [13], the resistance drift follows a power-law model [13] and the SET cells begin to lose data after 4 seconds. If the PCM cells are refreshed every 4 seconds, the retention error rate has been estimated to be less than 10^{-8}. The overall BER, considering sensing error, read disturb error, and retention error is $< 2 \times 10^{-8}$. With (144,128) Double Error Correction, Triple Error Detection (DECTED) code per 128-bits of data, the probability of a triple-bit error is 2.7×10^{-18}, which is well below the desired target of 10^{-16}. Hence, to facilitate lower write time for better performance and energy, we use DECTED code per 128-bits of data and refresh memory lines every 4 seconds. This allows us to bring down the SET latency of PCM-Duplicate from $1\mu s$ to 250ns.

D. Sneak Current in Crossbar Architecture

Typically, in most non-volatile resistive main memory modules, the wordlines and bitlines are organized in a crossbar array to achieve maximum density. In such a setup, each PCM element does not have an individual access transistor. This is because having an access transistor per cell significantly increases the cell size, thereby, negating most of the density benefits of crossbar architecture. Instead, each PCM element is placed at every wordline and bitline intersection. The bitlines are first precharged to V_{rd}.

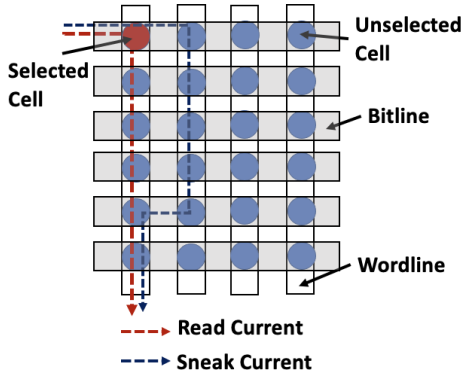

Fig. 6: Crossbar array structure showing read current and sneak current

In an ideal scenario, based on the row address, during activation, the wordline voltages should be set such that voltage V_{rd} is applied across the target cells and zero voltage is applied across the unselected cells. However, in a crossbar array, while applying the desired voltage across the cells to be read, adjacent cells get partially selected. This creates parallel 'sneak' discharge paths through the adjacent junctions, as shown in Figure 6. The parallel paths usually have higher resistance since the sneak currents have to pass through multiple resistive cells in series (three cells in Figure 6). But, these parallel circuits can alter the measured output, resulting in a read error. The impact of sneak current depends on the ratio of the current flowing through the target cells to the sneak current. Higher the ratio, the smaller the impact. With two wordlines activated at once in PCM-Duplicate, the number of sneak paths increases. However, the total current flowing through the target cells being read also increase in PCM-Duplicate due to lower total cell resistance (duplicate data cells connected in parallel). Thus, the ratio of actual to sneak current does not increase significantly to worsen the read error. Besides, in crossbar arrays, to tackle the problem of sneak currents, each PCM cell is placed in series with either a diode or a switching mechanism such as Ovonic Threshold Switch (OTS) [23] that is much smaller than an access transistor. The OTS conducts only when there is enough voltage applied across the cell. The sneak paths consist of multiple PCM/OTS cells in series. Hence, most of the cells in these paths do not see enough voltage across their terminals and hence, remain in the off state.

V. Evaluation Methodology

We modified cycle-accurate Gem5 simulator to evaluate the performance of PCM-Duplicate. The details of the simulated systems are provided in Table I. System-1 is a DRAM-based main memory system. System-2 replaces the DRAM main memory with the baseline PCM memory that has 4x higher capacity than DRAM, but has 4.6x larger read time. System-3 uses our proposed PCM-Duplicate memory that provides 2x capacity at the same cost as compared to DRAM while having comparable read latency. We also simulated hybrid memory systems where the faster memory (DRAM/PCM-Duplicate) acts as a hardware-managed cache for the slower PCM main memory. This last level cache is transparent to the OS and, therefore, does not add to the main memory capacity. In System-4, DRAM acts as the last level cache for the baseline PCM main memory. In System-5, we replaced the DRAM cache with our proposed 2x higher capacity PCM-Duplicate cache and the baseline PCM main memory with the PCM-ECC main memory. We simulated several workloads from SPEC CPU 2017 [20], Parsec [24] and GAP [21] benchmark suites. We fast-forwarded 1 billion instructions and ran the simulations for a total of 3 billion instructions. Since most of the larger Parsec and GAP benchmarks either suffer from a timeout or a kernel panic issue when simulating a multi-core system on Gem5 (similar observations reported in [25]), we had to be limited to a single core system and scale down the size of the caches. We used a 2GHz out-of-order single-core processor with a private 32KB I-cache, 64KB D-cache, 256KB L2 cache, and 512KB L3 cache in all six systems. The DRAM cache in System 4 is 8MB and the PCM-Duplicate cache is 16MB. The three graphs used when running the GAP workloads are 200MB and 500MB sized synthetic kron [21] and 600MB sized wikipedia graphs. Using graphs larger than this results in unreasonably long runtimes of more than a day. We also had to scale down the L4 sizes because of the limited-sized SPEC workloads that we used for this analysis. For an L4 size larger than 16MB, the L4 miss rate would be exceptionally low. This means that the number of memory accesses would be too less for gathering any meaningful main memory performance results. Many past works have faced similar issues [7], [12] and hence, could only evaluate a limited set of workloads.

VI. Results

We evaluate the performance improvements achieved by using our proposed optimized PCM substrates in two different main memory

TABLE I: Details of the different memory systems evaluated

	System-1	System-2	System-3	System-4	System-5
Processor					
Details	OoO, single-core, 2GHz				
L1-I/L1-D/L2/L3	32KB/64KB(2-way)/256KB/512KB				
LLC transparent to OS					
Technology	-	-	-	DRAM	PCM-Duplicate
Size	-	-	-	8MB	16MB
Latency (Read/Write)	-	-	-	15ns	21ns/250ns
Main Memory					
Technology	DRAM	Baseline PCM	PCM-Duplicate	Baseline PCM	PCM-ECC
Capacity	8GB	32GB	16GB	32GB	24GB
Read Latency	15ns	69ns	21ns	69ns	34ns
Write Latency	15ns	1us	250ns	1us	1us
Bus per Channel	DDR4				
Ranks per Channel	1				
Number of channels	2				

system configurations: (1) Replacing DRAM-based main memory system with PCM-Duplicate instead of baseline PCM, (2) Replacing the DRAM cache in hybrid main memory system with PCM-Duplicate cache for the slower PCM main memory.

A. Using PCM-Duplicate as Main Memory

PCM-Duplicate has read latency comparable to that of DRAM while providing 2x higher capacity at the same cost. In Figure 7, we compared PCM-Duplicate based main memory system (System-3) with DRAM and baseline PCM-based main memory systems (System-1 and System-2, respectively). Overall, for SPEC workloads, we see an average of 12.27% improvement of System-3 with PCM-Duplicate main memory over System-2 using baseline PCM-based main memory. For memory intensive workloads such as mcf and lbm, we see more than 30% speedup with PCM-Duplicate. The Parsec and GAP workloads are more memory intensive and also require larger memory capacity. For these workloads, the PCM-Duplicate provides a significant 23.54% average speedup. The improvement in performance is due to more than 69.5% reduction in read latency. Since the working set sizes of the workloads fit within the main memory, the reduction in main memory capacity in System-3 compared to System-2 did not impact the overall performance of System-3. PCM-Duplicate's read latency is comparable to that of DRAM. When compared against DRAM main memory (System-1), System-3 has, on an average, 4.6% higher execution time for SPEC workloads. This is primarily due to the longer write time of PCM. For larger GAP and Parsec workloads, this increases to 5.9% higher execution time. However, with only $\sim$5% slowdown, on an average, PCM-Duplicate provides 2x more capacity at the same cost or 2x lower cost for the same main memory capacity. Besides, PCM is non-volatile and, therefore, PCM-Duplicate can be easily used for main memory based checkpointing instead of having to use much slower storage.

B. Using PCM-Duplicate as Last Level Cache instead of DRAM

As mentioned before, today's systems using PCM typically implement a hybrid main memory system where each memory channel consists of both DRAM and PCM DIMMs. The DRAM becomes a hardware managed cache that is transparent to the OS. As a result, the total memory capacity reduces by a non-negligible amount as compared to a full PCM system. Also, the main memory no longer provides the benefit of persistence that comes with NVM-based main memories. In order to deal with both these problems, we propose to using optimized PCM-Duplicate as a smaller but faster cache for the slower PCM main memory. PCM-Duplicate would provide 2x more cache capacity as compared to DRAM and the entire cache-main memory system would be non-volatile. We evaluated this proposed System-5 and compared its performance with today's standard hybrid main memory system using DRAM based cache and baseline PCM-based main memory (System-4). The results are shown in Figure 8.

For less memory intensive SPEC 2017 workloads, we found that our proposed system provides an average of 4.67% speedup (upto 18% speedup for memory intensive workloads like mcf_r) as compared to the baseline hybrid system. For larger, more memory intensive Parsec and GAP workloads, the improvement in performance is upto 38.7% (average 9.43%). The performance improvement is due to the increase in the size of the last level cache. Even though PCM-Duplicate has higher write time than DRAM, the read latency is almost comparable. Since writes do not mostly fall in the critical path, the decrease in misses per thousand instruction (average 15.8%) translates to the overall speedup. For example, canneal has a 14.2% speedup while X264 has a 5.07% speedup. Both applications have similar read to write ratio (2.17:1 vs 2.89:1). But the difference in performance stems from the fact that canneal's misses per thousand instruction reduces by 102% while in x264, the reduction is by 29%. Overall, we see that using PCM-Duplicate as last level cache instead of DRAM helps in improving the overall system performance. This speedup comes from the larger sized last level cache that has similar read performance as that of the DRAM cache.

C. Enabling Lightweight Main Memory Based Persistence

In both main memory system configurations using PCM-Duplicate (System-3 and System-5), the main memory is non-volatile. Using DRAM as either the main memory (System-1) or the last level cache for the slower PCM (System-4) in hybrid memory systems makes the overall main memory system volatile. As a result, to ensure data persistence and enable checkpointing, much slower storage devices such as solid state drives(SSDs) or hard disk drives (HDDs) are used to flush and store the application state. During checkpoint recovery, the data is read back from the storage devices into the main memory before the program resumes. This overhead can be upto 30 minutes [8]. Using PCM-Duplicate as main memory or last level cache for slower PCM makes the entire main memory system non-volatile. Thus, expensive slow storage based checkpointing can now be easily replaced by lightweight main memory based checkpointing. Since, the main memory now falls within the persistence domain, the data in the CPU caches and the write queues in the memory controller need to be flushed to the main memory. This dramatically reduces the checkpointing overhead.

VII. Conclusion

Phase Change Memory (PCM) is considered one of the most promising scalable non-volatile main memory alternatives to DRAM. It provides $\sim$4x-5x cost per bit advantage over DRAM. However, the PCM has more than4x higher read latency, which leads to significantly poorer overall system performance (up to 80% for memory-intensive applications based on our analysis). To use PCM as a viable DRAM replacement, the performance gap between the two memory technologies has to be bridged, primarily by improving PCM read latency. In this work we propose an optimized PCM architecture, PCM-Duplicate, that trades off capacity to improve PCM read latency. In PCM-Duplicate, every row in the PCM subarray has a duplicate row. During a memory read, both the rows are activated simultaneously. As a result, the bitline discharges through two PCM cells. This reduces the discharge time significantly, bringing down the overall sensing latency by $>$3x compared to baseline PCM. While the overall PCM density benefit over DRAM halves, it still provides 2x more capacity than DRAM while having almost comparable read latency. PCM-Duplicate can either be used as a low-cost DRAM main memory alternative or can be used to replace DRAM-based last level cache for slower PCM main memory. Both these system options allow main memory-based persistence by replacing DRAM

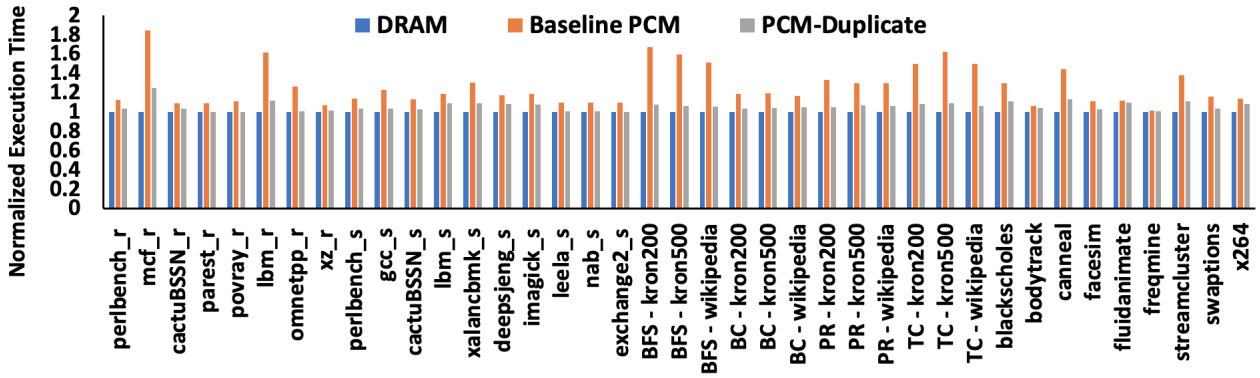

Fig. 7: Normalized Execution Time of SPEC-2017, GAP and Parsec workloads comparing DRAM (System-1), Baseline-PCM (System-2) and PCM-ECC (System-3) based main memory systems. The execution times are normalized against the System-1.

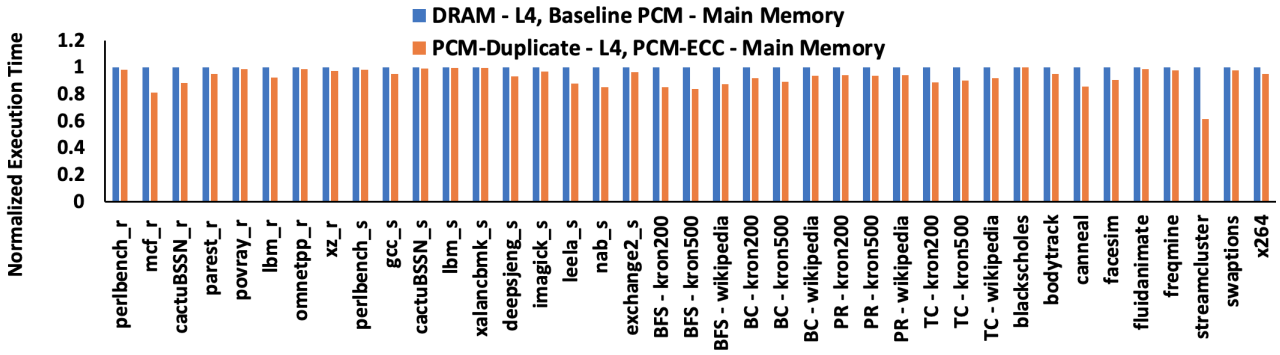

Fig. 8: Normalized Execution Time of SPEC-2017, Parsec and GAP workloads comparing DRAM and PCM-Duplicate as last level caches (System 4 vs. System 5) for slower PCM main memories. The execution times are normalized against the system using DRAM-based cache (system 4).

and we evaluate both options in this work. The first system using PCM-Duplicate-based main memory has 2x more memory capacity than DRAM, provides non-volatility while having only 6% worse overall system performance (average). The second system provides a fully non-volatile hybrid system with a PCM-Duplicate cache for slower PCM-ECC main memory. This system has 2x more cache capacity and 75% main memory capacity as compared to today's hybrid system with DRAM cache and baseline PCM. Our proposed hybrid system can provide up to 38.7% (average 7.87%) better overall performance than the baseline hybrid system. Thus, PCM-Duplicate-based memory systems provide two significant benefits over DRAM - (1) Higher capacity at lower cost (2) Lightweight main memory-based persistence.

References

[1] D. Waddington, M. Kunitomi, C. Dickey, S. Rao, A. Abboud, and J. Tran, "Evaluation of intel 3d-xpoint nvdimm technology for memory-intensive genomic workloads," in *Proceedings of the International Symposium on Memory Systems*, 2019, pp. 277–287.

[2] O. Patil, L. Ionkov, J. Lee, F. Mueller, and M. Lang, "Performance characterization of a dram-nvm hybrid memory architecture for hpc applications using intel optane dc persistent memory modules," in *Proceedings of the International Symposium on Memory Systems*, 2019, pp. 288–303.

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

[4] A. Kokolis, D. Skarlatos, and J. Torrellas, "Pageseer: Using page walks to trigger page swaps in hybrid memory systems," in *2019 IEEE International Symposium on High Performance Computer Architecture (HPCA)*, 2019, pp. 596–608.

[5] Intel, "3D XPoint™: A Breakthrough in Non-Volatile Memory Technology," https://www.intel.com/content/www/us/en/architecture-and-technology/intel-micron-3d-xpoint-webcast.html.

[6] TECHENABLEMENT, "3D XPoint Memory Poised to Revolutionize System Memory and Storage," https://www.intel.com/content/www/us/en/architecture-and-technology/intel-micron-3d-xpoint-webcast.html.

[7] P. J. Nair, C. Chou, B. Rajendran, and M. K. Qureshi, "Reducing read latency of phase change memory via early read and turbo read," in *2015 IEEE 21st International Symposium on High Performance Computer Architecture (HPCA)*, 2015, pp. 309–319.

[8] D. Tiwari, S. Gupta, and S. S. Vazhkudai, "Lazy Checkpointing: Exploiting Temporal Locality in Failures to Mitigate Checkpointing Overheads on Extreme-Scale Systems," in *Proceedings of the IEEE/IFIP International Conference on Dependable Systems and Networks (DSN)*, 2014.

[9] Y. Choi, I. Song, M.-H. Park, H. Chung, S. Chang, B. Cho, J. Kim, Y. Oh, D. Kwon, J. Sunwoo, J. Shin, Y. Rho, C. Lee, M. G. Kang, J. Lee, Y. Kwon, S. Kim, J. Kim, Y.-J. Lee, Q. Wang, S. Cha, S. Ahn, H. Horii, J. Lee, K. Kim, H. Joo, K. Lee, Y.-T. Lee, J. Yoo, and G. Jeong, "A 20nm 1.8v 8gb pram with 40mb/s program bandwidth," in *2012 IEEE International Solid-State Circuits Conference*, 2012, pp. 46–48.

[10] G. F. Close, U. Frey, J. Morrish, R. Jordan, S. C. Lewis, T. Maffitt, M. J. BrightSky, C. Hagleitner, C. H. Lam, and E. Eleftheriou, "A 256-mcell phase-change memory chip operating at 2+ bit/cell," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 60, no. 6, pp. 1521–1533, 2013.

[11] B. C. Lee, E. Ipek, O. Mutlu, and D. Burger, "Architecting phase change memory as a scalable dram alternative," in *Proceedings of the 36th annual international symposium on Computer architecture*, 2009, pp. 2–13.

[12] 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 *HPCA - 16 2010 The Sixteenth International Symposium on High-Performance Computer Architecture*, 2010, pp. 1–11.

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

[14] S. Lavizzari, D. Ielmini, D. Sharma, and A. L. Lacaita, "Transient effects of delay, switching and recovery in phase change memory (pcm) devices," in *2008 IEEE International Electron Devices Meeting*, 2008, pp. 1–4.

[15] D. Mantegazza, D. Ielmini, A. Pirovano, A. Lacaita, E. Varesi, F. Pellizzer, and R. Bez, "Explanation of programming distributions in phase-change memory arrays based on crystallization time statistics," *Solid-State Electronics*, vol. 52, no. 4, pp. 584–590, 2008, special Issue: Papers Selected from the Ultimate Integration on Silicon Conference 2007 - ULIS 2007. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0038110108000087

[16] D. H. Yoon, N. Muralimanohar, J. Chang, P. Ranganathan, N. P. Jouppi, and M. Erez, "Free-p: Protecting non-volatile memory against both hard and soft errors," in *2011 IEEE 17th International Symposium on High Performance Computer Architecture*. IEEE, 2011, pp. 466–477.

[17] C. Wilkerson, A. R. Alameldeen, Z. Chishti, W. Wu, D. Somasekhar, and S.-l. Lu, "Reducing cache power with low-cost, multi-bit error-correcting codes," in *Proceedings of the 37th annual international symposium on Computer architecture*, 2010, pp. 83–93.

[18] I. Alam, S. Pal, and P. Gupta, "Compression with multi-ecc: Enhanced error resiliency for magnetic memories," in *Proceedings of the International Symposium on Memory Systems*, 2019, pp. 85–100.

[19] N. Binkert, B. Beckmann, G. Black, S. K. Reinhardt, A. Saidi, A. Basu, J. Hestness, D. R. Hower, T. Krishna, S. Sardashti, R. Sen, K. Sewell, M. Shoaib, N. Vaish, M. D. Hill, and D. A. Wood, "The gem5 simulator," *SIGARCH Comput. Archit. News*, vol. 39, no. 2, p. 1–7, Aug. 2011. [Online]. Available: https://doi.org/10.1145/2024716.2024718

[20] "SPEC releases major new CPU benchmark suite." [Online]. Available: https://www.spec.org/cpu2017/press/release.html

[21] S. Beamer, K. Asanovic, and D. A. Patterson, "The GAP benchmark suite," *CoRR*, vol. abs/1508.03619, 2015. [Online]. Available: http://arxiv.org/abs/1508.03619

[22] W. Zhang and T. Li, "Helmet: A resistance drift resilient architecture for multi-level cell phase change memory system," in *2011 IEEE/IFIP 41st International Conference on Dependable Systems Networks (DSN)*, 2011, pp. 197–208.

[23] "How 3D XPoint Phase-Change Memory Works," https://pcper.com/2017/06/how-3d-xpoint-phase-change-memory-works/2/.

[24] C. Bienia, "Benchmarking modern multiprocessors," Ph.D. dissertation, Princeton University, January 2011.

[25] "gem5-20 Working Status of Benchmarks." [Online]. Available: https://www.gem5.org/documentation/benchmark_status/gem5-20