AdaMS: Adaptive MLC/SLC Phase-Change Memory Design for File Storage

Xiangyu Dong and Yuan Xie
Department of Computer Science and Engineering
Pennsylvania State University
e-mail: {xydong,yuanxie}@cse.psu.edu

Abstract— Phase-change memory (PCM) is an emerging memory technology that has made rapid progress in the recent years, and surpasses other technologies such as FeRAM and MRAM in terms of scalability. Recently, the feasibility of multi-level cell (MLC) for PCM, which enables a cell to store more than one bit of digital data, has also been shown. This new property makes PCM more competitive and considered as the successor of the NAND flash technology, which also has the MLC capability but does not have an easy scaling path to reach higher densities. However, the MLC capability of PCM comes with the penalty of longer programming time and shortened cell lifetime compared to its single-level cell (SLC) mode. Therefore, it suggests an adaptive MLC/SLC reconfigurable PCM design that can exploit the fast SLC access speed and the large MLC capacity with the awareness of workload characteristics and lifetime requirements. In this work, a circuit-level adaptive MLC/SLC PCM array is designed at first, the management policy of MLC/SLC mode is proposed, and finally the performance and lifetime of a novel PCM-based SSD with run-time MLC/SLC reconfiguration ability is evaluated[1].

I. Introduction

Phase-change memory (PCM) is an emerging memory technology with many attractive features including fast read access, high density, non-volatility, positive response to increasing temperature, superior scalability, and zero standby leakage. PCM has made rapid progress in the recent years, and surpasses other technologies such as FeRAM and MRAM in terms of scalability. The ultimate goal of PCM is to become a universal memory that works across multiple layers of the memory hierarchy for today's computer systems. Previously, PCM is considered to have the read access latency that is comparable to those of SRAM and DRAM, and the non-volatility like the NAND Flash. Most recently, it is promising that the feasibility of MLC for PCM has been shown [1–3], including programming into two and four bits per cell. Although the MLC doubles or quadruples the PCM bit density, the sophisticated "programming-and-verify" (P&V) write scheme causes much longer SET/RESET latency and reduced cell lifetime. Therefore, an adaptive scheme that leverages both fast SLC access speed and the large MLC capacity is suggested.

Kgil *et al.* [4] studied a NAND flash-based disk cache that dynamically downgrades MLC NAND flash pages to SLC pages when its lifetime approaches and hence increases the overall lifetime of the disk cache system. Lee *et al.* [5] proposed a flexible flash file system that dynamically leverages unused flash storage and uses SLC instead of MLC to improve the access latency. These previous research are focused on NAND flash memory which has the erase-before-write constraint. This enforces the flash-based file system design on the basis of journal, which quickly fills the entire device capacity. Qureshi *et al.* [6] demonstrated a morphable memory system that is a PCM-based main memory space with both MLC and SLC and uses memory monitoring to determine the ratio between the MLC and the SLC regions. However, using PCM as main memory also fills up the device capacity in minutes, which lets their work more focused on how to dynamically determine the optimal partition between SLC and MLC regions.

In this work, we propose *AdaMS*, a circuit-level adaptive MLC/SLC PCM design and the associated MLC/SLC management policy using run-time address re-mapping. The performance and lifetime improvement provided by *AdaMS* is evaluated for a case study of PCM-based file storage.

[1]X. Dong and Y. Xie were supported in part by NSF grants 0702617, 0903432, 0905365, and SRC grants

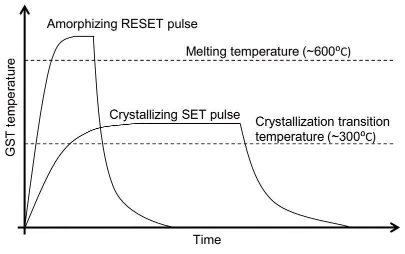

Fig. 1. PCM write operations involve application of electrical power through applied voltage, leading to internal GST temperature changes that either melt and then rapidly quench a volume of amorphous material (RESET), or hold this volume at a slightly lower temperature for sufficient time for recrystallization (SET).

II. MLC PCM Background

PCM exploits the large resistance contrast between the amorphous and crystalline states in the chalcogenide-based material (usually the alloy of germanium, antimony, and tellurium, called GST), which can be switched between amorphous and crystalline phases. The crystalline phase (SET state) shows low electrical resistivity, while the amorphous phase (RESET state) is characterized by high resistivity sometimes 3 or 4 orders of magnitude larger [7]. To SET the cell into its low-resistance state, an electrical pulse is applied to heat a significant portion of the cell above the crystallization temperature. This SET duration mainly depends on the crystallization speed of GST. Although SET pulses shorter than

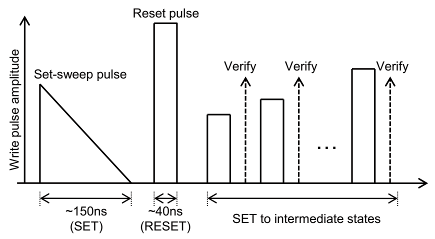

Fig. 2. The basic MLC PCM programming scheme

$10ns$ have been demonstrated [8], the typical value of the SET pulse duration is around $150ns$ [9]. On the other side, in the RESET operation, a larger electrical current is applied in order to melt the central portion of the cell. After this pulse is cut off abruptly, the molten material quenches into the amorphous phase. The RESET operation has shorter duration ($30ns$ [9]) but tends to be current-hungry. Later we also show the RESET operation is the major source of PCM cell wear-out. Fig. 1 shows the conceptual view of PCM RESET and SET operations.

Due to the large resistance contrast between the RESET and SET states, multi-level cell (MLC) PCM is feasible. However, the degree of success of such an MLC write depends on the resistance distributions over a large ensemble of PCM cells. Unlike single-level cell (SLC) write, where the bit write quality can be ensured by over-SET or over-RESET, the intrinsic randomness associated with each write attempt and the inter-cell variability make it infeasible to have a universal pulse shape for writing an intermediate state. In order to deal with this issue, resistance distribution tightening techniques have been developed based on "program-and-verify" (P&V) procedures.

A. Extra Write Overhead

P&V is a common programming technique for multi-bit writing and is widely used in Flash memories. In order to achieve non-overlapping resistance distributions of different bit levels, P&V needs to iteratively apply set pulse and then verify that a specified precision criterion is met, which leads to much longer write latency. As illustrated in Fig. 2, the MLC programming algorithm first programs the cell to its low-resistance (SET) state by means of a SET-sweep pulse. This is followed by a single RESET pulse with a fast quench, whose purpose is to initialize the cell to a totally RESET state before partial SET sequences are applied. In the final PGM step, the SET pulse amplitude is gradually increased under a feedback-loop control, so that the tight resistance distribution can be achieved [1]. It is obvious that, compared to the SLC RESET and SET operations that only require to apply a specific pulse shape, the MLC write scheme has to at least incorporate a SET and a RESET operation in each write operation. Thus, the write latency and the write energy is tremendously larger than the ones in the SLC mode.

B. Extra Read Overhead

Reading data from MLC device is more difficult than reading data from SLC device as it requires distinguishing more precisely between neighboring resistance levels. As we show later in Section IV, reading MLC data needs more comparison steps, which inevitably causes extra read latency overhead.

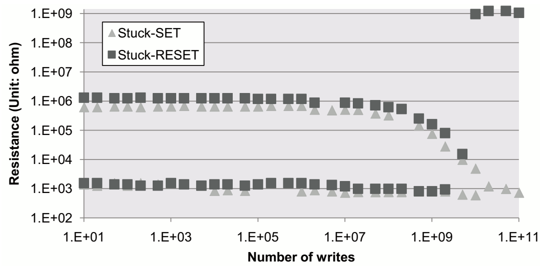

Fig. 3. SET and RESET resistances during cycling, illustrating the difference between failure by "stuck-SET" and by "stuck-RESET" [10].

C. Reduced Cell Lifetime

As shown above, P&V needs to initialize all the target cells to RESET state before each intermediate writing. As we discuss in the next section, RESET is the major source of cell wear-out, hence MLC reduces the cell lifetime compared to the SLC mode.

III. PCM Lifetime Model

Similar to some other types of non-volatile memory technologies, PCM has the limitation on write endurance, which means that every PCM cell can only be overwritten by limited times. Several PCM reliability experiments have shown the PCM write endurance numbers in the range of 10^8-10^{10} cycles [10]. Two types of failure modes have been observed to happen after cycling, called "stuck-RESET" and "stuck-SET", as shown in Fig. 3.

In a stuck-RESET failure, the device resistance suddenly spikes, and the resistance is stuck at the level that is much higher than the normal RESET state. This stuck-RESET failure is typically caused by void formation or delamination that catastrophically severs the electrical path between GST and access device. For example, the contact between heater and GST is disconnected after about 10^{10} cycles.

On the contrary, in a stuck-SET failure, a gradual degradation of RESET-to-SET resistance margin is usually observed as demonstrated in Fig. 3. As PCM cell continues experiencing write cycles, its GST characteristics change, and somehow it becomes more difficult to create an amorphous (RESET) phase in GST than before. Although, a larger amplitude RESET pulse is able to force GST to switch between the RESET and SET states, it causes a larger RESET power consumption and worse it eventually lets stuck-SET occur earlier. Therefore, in this work, stronger RESET pulse is not used to prolong the PCM cell lifetime. Instead, during the stuck-SET degradation, MLC cell is reconfigured to SLC modes as we discuss in the later sections.

Degrading MLC to SLC not only bypasses the issue of decreasing resistance margin, more importantly, writing SLC data does less damage to the PCM cell than writing MLC data

TABLE I
THE LIFETIME MODEL OF PCM CELLS.

RESET cycles	MLC	SLC
0-10^7	Yes	Yes
10^7-10^9	No	Yes
10^9-	No	No

does. According to the study conducted by Goux *et al.*, it is demonstrated that stuck-SET failure is due to a change in the RESET condition that is induced by cycling [11]. Their endurance dta suggests that endurance scales inversely with $t_m^{\frac{3}{2}}$, where t_m is the time-spent-melting during each RESET pulse. Their experiment coincides with other study that observes cycling with only SET pulse greatly extends endurance (more than 10^{12} cycles) over RESET-SET cycling (10^{10}) [10].

According to Fig. 3, in this work, we assume that during the first 10^7 RESET-SET cycles, each PCM cell can be either in MLC mode or in SLC mode depending on the external configuration; between 10^7 and 10^9 cycles, the RESET resistance degradation makes PCM cell to lose the MLC capability and it can only work in SLC mode; all the PCM cells after 10^9 cycles are considered as non-functional. In addition, it is obvious that each MLC write includes a RESET operation and we further assume that the RESET/SET distribution of SLC write is 50%/50%. Therefore, the PCM lifetime model can be concluded as tabulated in Table I.

IV. ADAPTIVE MLC/SLC PCM ARRAY STRUCTURE

In this section, we demonstrate how an MLC PCM array can be configured to support SLC accesses on the fly by incurring only a little hardware overhead. In addition, a density control layer is designed to manage the variable PCM capacity and to track the density mode of PCM cells at a proper granularity.

A. MLC/SLC Write: SET, RESET, and PGM Pulses

An MLC write has to initialize the target cell to the RESET state and then iteratively apply PGM pulses (partial SET pulses with fixed pulse duration but different pulse amplitude or vice versa) until the targeted intermediate resistance level is reached. A full SET pulse (SET-sweep pulse) is also used to program the cell into the SET state. Therefore, SET, RESET, and PGM pulse generators are all required in the MLC PCM chip, as shown in Fig. 4.

In order to degrade an MLC PCM cell work in SLC mode, it is straightforward to use only SET and RESET pulse generators to program the cells. During the SLC writing process, the PGM pulse generator and its associated iteration control logic are bypassed because there is no intermediate resistance levels to program the cells into.

B. MLC/SLC Read: Dual-Mode Sense Amplifier

Since every PCM cell in MLC mode stores more than one bits, during the MLC reading process, each sense amplifier

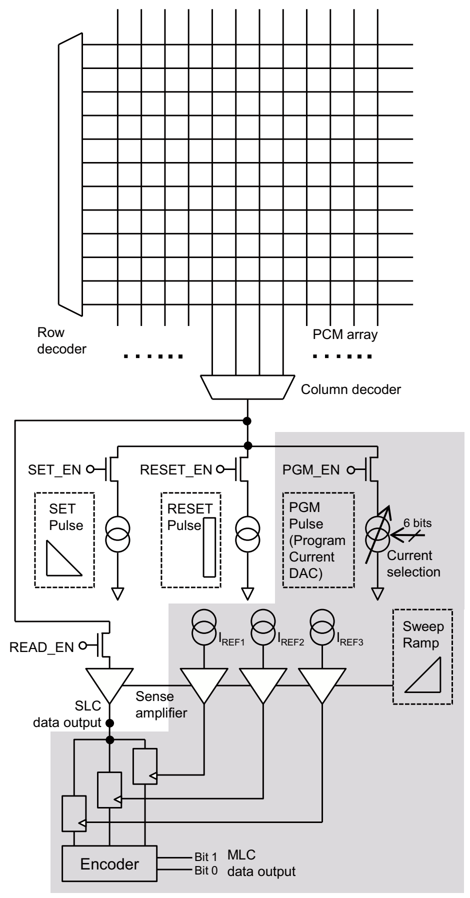

Fig. 4. The block diagram of the PCM array organization that supports both MLC and SLC operations: the components in grey are optional for MLC supporting; the other components are required for basic SLC PCM operations.

output is compared to multiple references. The comparison results are latched separately and then encoded into a multi-bit data. Fig. 4 illustrates an example of the sensing scheme for a 2-bit MLC PCM array. In this scheme, a ramp generator triggers three reference sense amplifiers at different time, the output of each reference sense amplifier triggers a corresponding flip-flop which eventually stores the result of the comparison between the bitline current and one of the reference currents, and finally the value stored in the flip-flops (“000”, “001”, “011”, or “111”) is encoded into a 2-bit data.

However, when an MLC PCM cell degrades to SLC mode, most of the components in the MLC writing scheme become unnecessary. The output node of the bitline sense amplifier can be used directly as the SLC data output. Therefore, switching from MLC to SLC mode does not need to add any significant peripheral circuitry but only some simple control logic. The shaded part of Fig. 4 shows the removable components for SLC PCM read and write operations.

C. Address Re-mapping

With the capability of changing the PCM cells between the MLC mode and the SLC mode on-the-fly, the effective PCM device capacity can vary at runtime. In order to control such a variable device size, an address re-mapping mechanism is

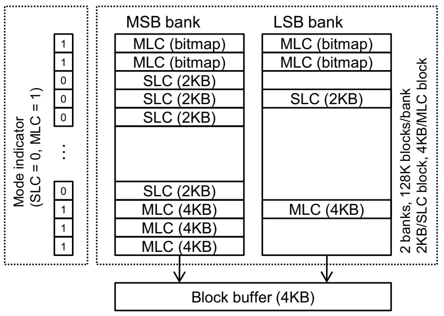
Mode indicator (SLC = 0, MLC = 1)
MSB bank
LSB bank
MLC (bitmap)
SLC (2KB)
MLC (4KB)
2 banks, 128K blocks/bank 2KB/SLC block, 4KB/MLC block
Block buffer (4KB)

Fig. 5. The conceptual view of managing SLC and MLC modes: The PCM chip is divided into two banks. When the accessing address is indicated as MLC block, only the corresponding bank (either MSB or LSB bank) is activated; when the accessing address is indicated as SLC block and the address is mapped to the MSB bank, both SLC and MLC blocks are activated and two SLC blocks are combined into one I/O block; when the accessing address is indicated as SLC block and the address is mapped to the LSB bank, the access operation ends up with a 'bad block' value

introduced as illustrated in Fig. 5. In our proposal, the PCM chip is divided into two banks and each bank has a complete set of I/O path. The PCM cells in each bank are grouped into large blocks. As the optimal block size of the state-of-the-art file system (e.g. Ext2) with large size (more than 1GB) is 4KB, we group each 16K PCM cells into one group. Thereby, if the block is in SLC mode, it has the capacity of 2KB, and if it is in 2-bit MLC mode, it has the capacity of 4KB.

In this design, the blocks having the same row address are always in the same mode. Hence, a bitmap is used to indicate whether a block is in SLC or MLC mode. The general address re-mapping algorithm is described by Algorithm 1. In short, if the bitmap indicates the accessing block is MLC, then only the corresponding bank is activated. However, if the accessing block is SLC, only the address mapped to the first bank (MSB bank) is valid and the 4KB I/O block is accessed by combining two 2KB SLC blocks (one in MSB bank and the other in LSB bank).

In SLC mode, all the accesses to the LSB banks are invalid and end up with a 'bad block' signal. In the case of Linux it is possible to supply a bad block list, which is generally much easier to just run 'badblocks' at the disk formatting time. The state-of-the-art file system (e.g. Ext2) uses a special sort of 'hidden file' to which it allocates all of the bad blocks on the file system. This technique insures that those data blocks will never be accessed or used for any other files. On the other hand, when the block changes its mode from SLC to MLC, it is straightforward to remove this block from the 'bad block' list and thus it becomes available to be allocated by other files. The bitmap indicating the MLC/SLC status is the only hardware overhead to enable our adaptive MLC/SLC proposal. In the case of a PCM device with 4G cells (512MB as pure SLC PCM or 1GB as pure MLC PCM), each bank contains 128K rows and the bitmap size is 16KB. In our design, the first four PCM blocks are always set as MLC, and they store this 16KB bitmap. The bitmap is loaded into the system main memory during the booting time, and it is written back to the PCM device during the unmounting time. Here, we assume the PCM device has sufficient on-device capacitor so that the bitmap data can always be written back even upon emergent events.

Algorithm 1 Address Re-mapping Algorithm

```
bankID = getBankID(blockAddress)
rowID = getRowID(blockAddress)
if Bitmap[rowID] = MLC then
    Active Bank[bankID]
    Access Block[rowID]
else
    if bankID = 0 then
        Active all the banks
        Access Block[rowID] in all the banks
        Combine the SLC blocks into a 4KB block
    else
        Return bad block
    end if
end if
```

D. Reconfigurable PCM-based Solid-State Disk

In the near future, PCM is considered as the direct substitution of the NAND flash. Considering most of the NAND flash-based devices used today have lower capacity utilization (e.g. personal flash device, SD card in digital cameras, etc.), we proposes two ways to partition the MLC and SLC blocks to apply the adaptive MLC/SLC PCM device into real practice,

To optimize the PCM device for performance, it is the rationale to first set all the PCM blocks in SLC mode for fast read/write accesses. In this way, the initial device capacity utilization is 50%. When the required device utilization surpasses 50%, the SLC blocks that belong to the least recently modified file are merged as MLC blocks, and therefore the extra device space is left for the new files. The expansion process continues until the device capacity utilization reaches 100%, in which it means all the PCM blocks are in the MLC mode.

V. Simulation Results

In this section, we evaluate the performance and lifetime improvement after applying the adaptive MLC/SLC technique when the PCM device is under-utilized.

In order to estimate the efficiency of the proposed technique on a real platform, we collected the actual I/O trace on Linux 2.6.32-23 kernel using a 2GB RAMDISK formatted as Ext2 file system with 4KB block size. We created a synthetic file system trace by first filling up the RAMDISK with randomly generated files whose sizes range from 5KB to 10MB and then randomly accessing them 1,500,000 times.

In addition, we used disk traces from Storage Performance Council [12], which were intended to model the disk behavior on enterprise level applications like web servers, database servers, and web search. Later in this section, we call our synthetic trace as *Synthetic* and the traces from SPC as *Financial 1*, *Financial 2*, and *WebSearch*, respectively.

A. PCM MLC/SLC Timing Model

We use a heavily-modified version of *PCRAMsim* [13] to estimate the read and write latencies for SLC and MLC modes.

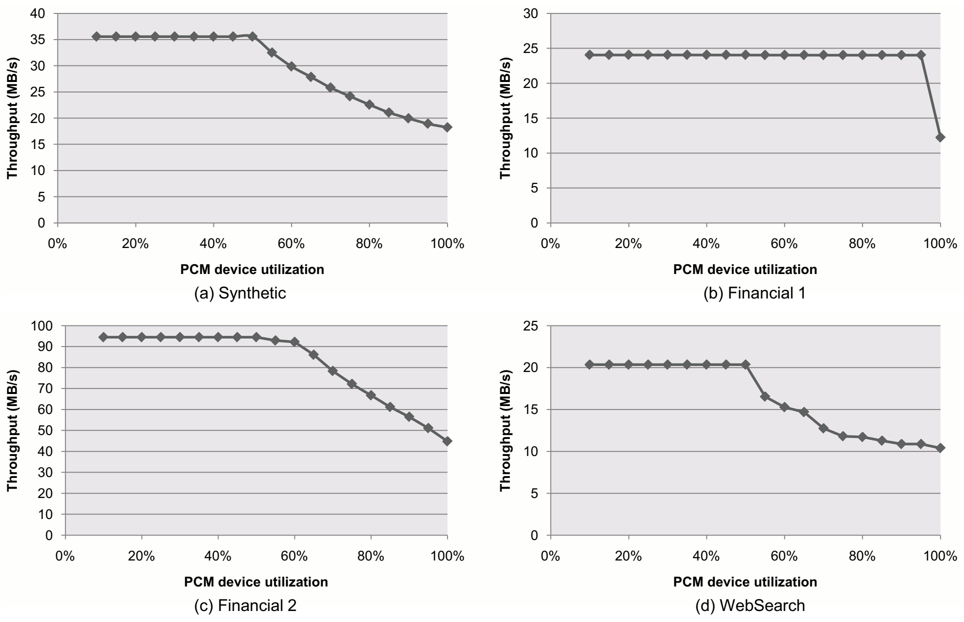

Fig. 6. The performance of the adaptive MLC/SLC solution under different utilizations.

TABLE II
THE TIMING MODEL OF PCM CELLS IN SLC AND MLC MODES.

	SLC	MLC
Read latency	$10ns$	$44ns$
Read width	$64bits$	$128bits$
Read bandwidth	$800MB/s$	$363.6MB/s$
Write latency	$100ns$	$395ns$
Write width	$16bits$	$32bits$
Write bandwidth	$20.0MB/s$	$10.3MB/s$

For simplicity, we average the SET latency and the RESET latency in the SLC write latency calculation, and we set the average P&V steps as 4 to form the MLC write latency. The rounded values are tabulated in Table II. The read width is set to 64 cells (64 bits for SLC and 128 bits for MLC) and the write width is limited to 16 cells (16 bits for SLC and 32 bits for MLC) due to the large amount of current that is required for SET and RESET operations. Under these assumptions, the I/O bandwidth of SLC mode is about twice the MLC bandwidth.

B. Performance-Aware Management Result

Firstly, we demonstrate how the performance-aware partitioning strategy described in Section D can improve the performance. The I/O access distribution has a large impact on the efficiency of the proposed adaptive MLC/SLC PCM. If the accesses are evenly distributed to every portion of the file system, certain amount of data have to be accessed from the MLC regions. On the other hand, if the access pattern is biased and there is a part of file system is not frequently accessed, then that part of the files can be partitioned to the MLC regions and most of the frequently-accessed data can be accomplished by only accessing the SLC regions.

In this analysis, we assume that many PCM chips are connected in an array to form a large PCM device that has sufficient storage capacity to hold the working set (the number of PCM chips is a variable to adjust the device utilization). When the device utilization is lower than 50%, all the PCM blocks are in SLC mode, and when the device utilization is 100%, all the PCM blocks are in MLC mode. When the utilization is between 50% and 100%, the adaptive MLC/SLC technique is applied to supply the required capacity.

Fig. 6 illustrates the relationship between the average throughput and the PCM device utilization under different workloads. It can be observed that *Synthetic* and *Financial 2* has a gradual throughput degradation as the utilization increases. This is due to the high data locality of these two workloads. Furthermore, this phenomenon is exaggerated in *Financial 1*, which has a large portion of the file system that are only accessed once. However, the throughput of *WebSearch* drops abruptly at 50% as this workload has a evenly distributed I/O access pattern.

C. Performance-Cost Analysis

Based on the previous result on the relationship between the performance and the device utilization, we can further derive the relationship between the performance and the cost by assuming each PCM chip has a fixed cost. As an example, under the workload access pattern of *Financial 1*, one extreme configuration is to use SLC-only PCM chips that can supply the bandwidth of 94MB/s, while the other extreme configuration is to use MLC-only PCM chips that only have the bandwidth of 44MB/s but halve the required PCM chip count. To investigate the throughput-per-cost metric, we rephrase the result in Fig. 6 and Fig. 7 shows when the throughput-per-cost reaches the peak value. The average improvement on throughput-per-cost is around 28%.

D. Lifetime Analysis

When a MLC PCM cell has experienced too much RESET operations, its RESET/SET resistance margin decreases.

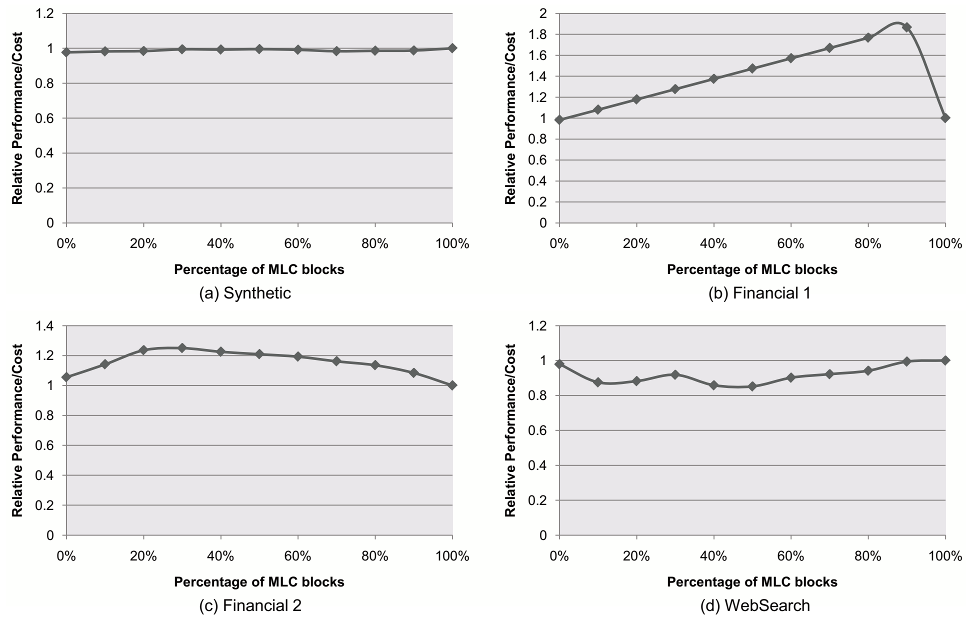

Fig. 7. The performance per cost analysis of the adaptive MLC/SLC solution.

Therefore, it has to be configured as a SLC at that time. In the lifetime-aware partitioning strategy, all the PCM blocks in MSB bank are first set to MLC mode and the ones in LSB bank are left as empty. When the operating system monitor finds that certain blocks have higher accessing probability, these blocks are switched to SLC mode by enabling their associated blocks in the LSB banks. Using the lifetime-aware partitioning strategy, the maximum lifetime improvement can be as high as 100 according to the lifetime model described in Section III. This maximum amount of lifetime improvement comes from the halved device capacity.

VI. Conclusion

PCM is considered as promising memory technology that has several good properties such as fast accesses, high density, and non-volatility. In this work, we focus on the multi-bit storage capability of the PCM technology and propose an adaptive MLC/SLC scheme, called “AdaMS”. *AdaMS* exploits the fast SLC access speed and large MLC capacity with the awareness of workload characteristic and lifetime requirement. In this work, after designing a circuit-level adaptive MLC/SLC PCM array and proposing the management policy of MLC/SLC mode, the simulation result based on four actual I/O traces shows that the *AdaMS* technique can improve the throughput-per-cost of the PCM device by 28% on average or it can extend the PCM device lifetime by 100 if the device utilization is under 50%.

References

[1] F. Bedeschi, R. Fackenthal, C. Resta, E. Donze *et al.*, “A bipolar-selected phase change memory featuring multi-level cell storage,” *IEEE Journal of Solid-State Circuits*, vol. 44, no. 1, pp. 217–227, 2009.

[2] T. Nirschl, J. Phipp, T. Happ, G. Burr *et al.*, “Write strategies for 2 and 4-bit multi-level phase-change memory,” in *Proceedings of the IEEE International Electron Devices Meeting*, 2007, pp. 461–464.

[3] D.-H. Kang, J.-H. Lee, J. Kong, D. Ha *et al.*, “Two-bit cell operation in diode-switch phase change memory cells with 90nm technology,” in *Proceedings of the Symposium on VLSI Technology*, 2008, pp. 98–99.

[4] T. Kgil, D. Roberts, and T. Mudge, “Improving nand flash based disk caches,” in *Proceedings of the International Symposium on Computer Architecture*, 2008, pp. 327–338.

[5] S. Lee, K. Ha, K. Zhang, and J. Kim, “FlexFS: a flexible flash file system for MLC NAND flash memory,” in *USENIX Annual Technical Conference*, 2009.

[6] M. K. Qureshi, M. M. Franceschini, L. A. Lastras-Montano, and J. P. Karidis, “Morphable memory system: a robust architecture for exploiting multi-level phase change memories,” in *Proceedings of the International Symposium on Computer Architecture*, 2010, pp. 153–162.

[7] S. Raoux, G. W. Burr, M. J. Breitwisch, C. T. Rettner *et al.*, “Phase-change random access memory: a scalable technology,” *IBM Journal of Research and Development*, vol. 52, no. 4/5, 2008.

[8] D. Krebs, S. Raoux, C. T. Rettner, G. W. Burr *et al.*, “Characterization of phase change memory materials using phase change bridge devices,” *Journal of Applied Physics*, vol. 106, no. 5, p. 054308, 2009.

[9] F. Bedeschi, E. Bonizzoni, G. Casagrande, R. Gastaldi *et al.*, “SET and RESET pulse characterization in BJT-selected phase-change memories,” in *Proceedings of the IEEE International Symposium on Circuits and Systems*, 2005, pp. 1270–1273.

[10] K. Kim and S. J. Ahn, “Reliability investigations for manufacturable high density PRAM,” in *Proceedings of the IEEE International Reliability Physics Symposium*, 2005, pp. 157–162.

[11] L. Goux, D. Tio Castro, G. Hurkx, J. Lisoni *et al.*, “Degradation of the reset switching during endurance testing of a phase-change line cell,” *IEEE Transactions on Electron Devices*, vol. 56, no. 2, pp. 354–358, 2009.

[12] Storage Performance Council, “I/O Trace Repository,” http://www.storageperformance.org/specs/#traces.

[13] X. Dong, N. P. Jouppi, and Y. Xie, “PCRAMsim: system-level performance, energy, and area modeling for Phase-Change RAM,” in *Proceedings of the International Conference on Computer-Aided Design*, 2009, pp. 269–275.