ER: Elastic RESET for Low Power and Long Endurance MLC based Phase Change Memory *

Lei Jiang †, Youtao Zhang ‡, Jun Yang †
† Electrical and Computer Engineering Department
‡ Computer Science Department
University of Pittsburgh, Pittsburgh, PA 15261, USA
†{lej16,juy9}@pitt.edu ‡zhangyt@cs.pitt.edu

ABSTRACT

Phase Change Memory (PCM) has recently emerged as a promising nonvolatile memory technology. To effectively increase memory capacity and reduce per bit fabrication cost, multi-level cell (MLC) PCM stores more than one bit per cell by differentiating multiple intermediate resistance levels. However, MLC PCM suffers from significantly shortened endurance due to its large RESET current that initiates the cell state. In this paper, we propose elastic RESET (ER) to construct non-2^n-state MLC PCM, e.g., 3-state MLC PCM instead of 4-state one for 2-bit MLC. We then adopt data compression and propose fraction encoding to store compressed data using non-2^n-state MLC. By reducing RESET energy, ER significantly reduces write power and prolongs PCM lifetime. On average, we observed 17% RESET power reduction and 32× endurance improvement for 2-bit MLC.

Categories and Subject Descriptors

B.3.4 [**Memory Structures**]: Reliability, Testing, and Fault-Tolerance

General Terms

Design, Reliability

Keywords

Phase Change Memory, Multi-Level Cell

1. INTRODUCTION

Phase Change Memory (PCM) has emerged as a promising nonvolatile memory to alleviate the large leakage and poor scalability problems of traditional DRAM [26]. With advantages such as near zero cell leakage, better scalability and comparable read speed as DRAM, PCM is projected to replace a significant portion of DRAM in main memory system [26]. A PCM cell uses its resistance level to represent logic bits. A single-level cell (SLC) PCM stores one bit (logic '0' or '1'). Given the large resistance contrast between '0' and '1', it is possible to take advantage of intermediate resistance levels such that one cell can store more than one bit, referred to as multi-level cell (MLC) PCM. MLC PCM can effectively increase memory capacity while reducing per bit fabrication cost.

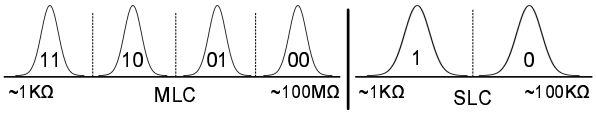


Figure 1: MLC has tighter resistance ranges.

PCM is known for long write latency and short endurance problems. Recently, many architectural techniques have been proposed to attack these two problems. *Differential-write* [26] extends PCM lifetime by removing redundant writes to cells. Wear leveling techniques [18] evenly distribute unbalanced write traffic among the entire chip. Salvaging schemes [22] were proposed to extend chip lifetime when non-negligible number of cells fail. For MLC based PCM, *write pausing* prioritizes read operations to improve system performance [19]. *Write truncation* uses error correction code (ECC) to reduce the number of MLC write iterations [12]. Low power data encodings for MLC PCM were studied in [23]. By transforming MLC to SLC [20, 2, 12], both a longer chip lifetime and a shorter read latency can be obtained on MLC PCM system.

Comparing to SLC, MLC PCM suffers more severely from large RESET power and long read/write latency problems. In order to reliably represent the stored logic value, a PCM cell needs to have its resistance programmed within a tight resistance range. Since MLC has more resistance levels to represent, it often has tighter resistance range per level and/or higher maximum resistance, as shown in Figure 1. Given a PCM write circuit that has certain programming precision, MLC PCM requires a larger RESET current than SLC PCM to initialize its maximum resistance and contain more resistance levels.

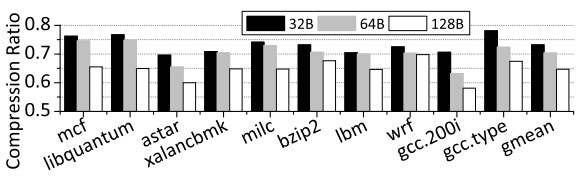


Figure 2: Compression ratio of all line-level writes.

*This work was supported in part by NSF CSR #1012070 and NSF CAREER #0747242.

ISLPED'12, July 30–August 1, 2012, Redondo Beach, CA, USA.

However, larger RESET energy significantly shortens cell endurance. As revealed by device studies [9, 14], 2× RESET energy results in 100× endurance degradation. To address this problem, in this paper, we propose elastic RESET (ER) to construct non-2^n-state MLC PCM. By reducing the RESET energy, we effectively reduce RESET power and prolong PCM lifetime. The existing work that is most close to our design is *MLC-to-SLC transformation* [2, 12] that compresses and stores highly compressible PCM lines in SLC format. *MLC-to-SLC transformation* is only applicable if a line is highly compressible, i.e., the compressed size is $\leq$ half of the original size. Unfortunately, as shown in Figure 2, many applications have 60%∼75% compression ratio and thus cannot benefit much from it.

Our contributions are summarized as follows.

- We propose Elastic RESET (ER) that reduces RESET current to construct non-2^n-state MLC PCM. ER effectively reduces the RESET energy such that it can extend PCM endurance exponentially.
- We propose Fraction Encoding (FE) to store compressed data using non-2^n-state MLC PCM cells. Instead of storing multiple bits in one cell, FE combines multiple cells to store multiple bits, e.g., 2 cells to store 3 bits, and thus can adaptively store compressed data with relaxed compression ratio.
- We evaluate our proposed designs and compare them to related works in the literature. Our experimental results show that ER improves PCM chip lifetime by ×32 and reduces RESET power by 17% on average.

The rest of this paper is organized as follows. Section 2 introduces the models of MLC PCM write. Section 3 elaborates the motivation and the design details. Section 4 presents our experimental methodology. Section 5 analyzes our experimental results. We conclude in Section 6.

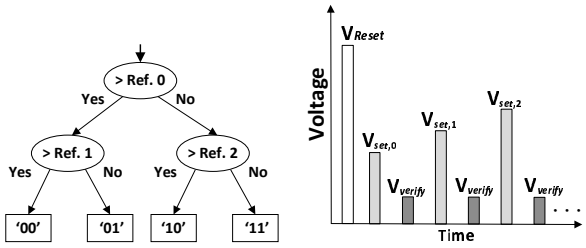
2. MODELING MLC PCM WRITE

2.1 MLC PCM Read and Write Operations

Phase Change Memory (PCM) exploits the phase-change behavior of chalcogenide glass (a.k.a GST, $Ge_2Sb_2Te_5$). By injecting electrical pulses, GST can be switched between large resistance state (amorphous state) and small resistance state (crystalline state). MLC PCM exploits intermediate resistance levels between these two states to store multiple logic bits per cell.

Due to process variations and material composition fluctuation, different PCM cells within one memory line respond differently to the programming pulses; and even the same cell responds differently at different times [15]. PCM uses tight resistance ranges to represent the information stored in the cell (and a small unused guardband is often left between consecutive ranges to prevent resistance drift [13]). As we discuss later, all 2-bit MLC cells with resistance between $10^{5.4}\Omega$ and $10^{6.5}\Omega$ are considered to store '01'.

To read a MLC PCM cell, a short current with small amplitude is injected to compare the cell resistance with reference values. It often takes n iterations to read n-bit MLC cells [7] (as shown in Figure 3(a)). To write a MLC PCM cell, a programming and verify (P&V) strategy [4] is often adopted. We first inject a large-amplitude RESET current to initialize cells to amporhous state with large enough resistance, and then inject a sequence of lower-amplitude SET currents to program each cell to the target resistance range. After each SET iteration, a verify (i.e., read) operation is employed to check if the write operation can be terminated (see Figure 3(b)).



(a) MLC PCM read (b) MLC PCM write

Figure 3: PCM read and write operations.

Recently an alternative P&V programming strategy was proposed [6]. The write starts with a SET pulse and then a sequence of RESET pulses (one SET and multi-RESET). While this strategy tolerates better to process variations, it requires larger write energy and longer time to finish. Cell endurance is also reduced, since more RESETs are performed. In this paper, we adopt the traditional P&V programming strategy (one RESET and multi-SET).

2.2 MLC PCM Resistance Ranges

The width of resistance range is often determined by the programming circuit. MLC PCM tends to have tighter ranges and thus requires more precise control. On the other hand, with a given programming precision, MLC PCM has a larger maximum resistance than that of SLC PCM.

Device studies [21] built an accurate RESET current model, whose inputs include the radius of heater (A), the heights of top and bottom electrodes (H_T and H_E), the height of GST (H) and the thickness of amorphous GST fraction (X_0), as shown in Figure 4(a). The resistance of PCM cell is decided by the thickness of amorphous GST fraction. The resistance model [17] described the relation between thickness of amorphous GST and cell resistance, as shown in Figure 4(b).

In this paper, we adopt both two former models with cell parameters in [21]. When X_0 is set to 50nm, the maximum resistance after RESET operation can be 79.4 $M\Omega$. Table 1 shows the resistance ranges and mean values for 2-bit MLC PCM [16].

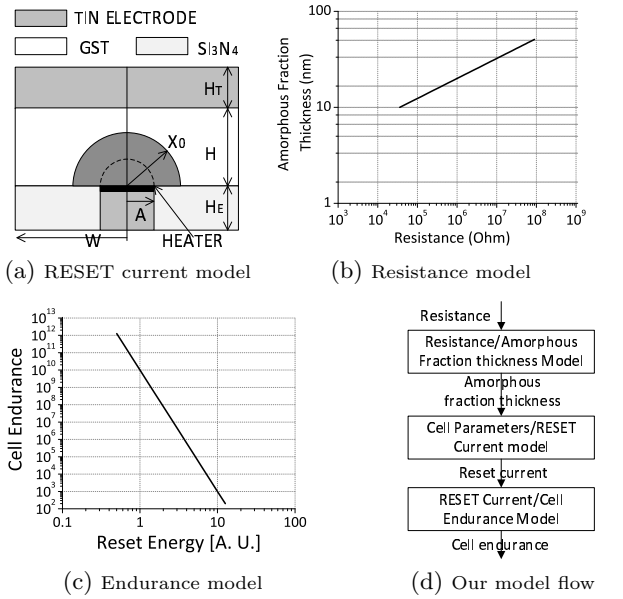
2-bit MLC	Resistance Range(Ω)	Mean(Ω)
'11'	$[10^3, 10^{4.1}]$	$10^{3.6}$
'10'	$[10^{4.2}, 10^{5.3}]$	$10^{4.75}$
'01'	$[10^{5.4}, 10^{6.5}]$	$10^{5.9}$
'00'	$[10^{6.6}, 10^{7.9}]$	$10^{7.2}$

Table 1: The resistance range of 2-bit MLC.

2.3 The PCM Endurance Model

In this paper, we focus on the hard faults determining PCM endurance. MLC PCM is also vlunerable to soft errors introduced by resistance drift, which can be prevented by guardbands, ECC [24], and/or refresh [3].

PCM cell lifetime has been tested at the device level. In [14], both SET-cycling and RESET-cycling tests were performed to indentify which operation is more critical to cell endurance. The SET-cycling experiment has no failure



(a) RESET current model (b) Resistance model

(c) Endurance model (d) Our model flow

Figure 4: PCM cell, resistance, endurance models.

up to 10^{12} cycles, which is seen as the upper bound of PCM cell lifetime. In contrast, hard errors appear after 10^9 cycles in RESET-cycling test. RESET operation energy has been indentified as the most important factor deciding the PCM cell endurance. In this paper, we adopt the RESET-endurance model from [14], with the cell lifetime as a function of RESET pulse energy plotted in Figure 4(c). The power law relationship between the cell lifetime and RESET energy can be summarized in Equation 1, while the conversion between RESET current and RESET energy can be obtained using Equation 2.

$$log_{10}(lifetime) = -7 \times log_{10}(\frac{E_{actual\_reset}}{E_{optimal\_reset}}) + 10 \quad (1)$$

$$E_{reset} = I_{reset}^2 \times R \times T \quad (2)$$

where, I_{reset}, E_{reset} denote the RESET current and energy respectively; R represents PCM cell heater resistance; and T is the RESET pulse duration measured in time.

From Figure 4(c), the ideal PCM cell lifetime can be 10^{10} cyclings. But due to process variation and unavoidable over-programming, chip level PCM cell endurance only achieves 10^8 writes for SLC [25, 11] and 10^5 writes for MLC [4]. Through Equation 1, SLC only needs about 40% RESET energy (or power) of MLC.

2.4 Our Work Flow

Our model flow can be found in Figure 4(d). The resistance value is fed into resistance model to get the amorphous fraction thickness. The RESET current model calculates the RESET current based on amorphous fraction thickness. With the RESET current input, the cell endurance model gets the lifetime for each PCM cell. Through our model flow, we can evaluate the impact of the reduced RESET current on PCM lifetime.

3. THE DETAILS OF ELASTIC RESET

In this section, we first discuss the benefits of constructing non-2^n-state MLC cells and then elaborate the design details of ER.

3.1 Constructing Non-2^n-state MLC Cells

An implicit assumption of using n-bit MLC cell is that we need to distinguish 2^n states and thus 2^n resistance ranges, e.g., 4 states for 2-bit MLC PCM. However, this is often not necessary. For example, we need to store a 64B memory line using 2-bit MLC and the line can be compressed to 48B (75% of the original size). There are three alternative approaches to save the compressed line.

- Approach 1: we can store 48B using 4-state 2-bit MLC, which uses 192 cells out of 256 cells in the memory line. By conducting intra-line perfect wear leveling, we can extend the chip lifetime to 133.3%.
- Approach 2: we can store 32B using 4-state 2-bit MLC and 16B using 2-state SLC, as proposed in [2]. Let us optimistically assume SLC lifetime is infinite. By again conducting intra-line perfect wear leveling, we can extend the chip lifetime to 200%.
- Approach 3: we can store 48B using 3-state 2-bit MLC, which uses all 256 cells of the memory line. As one cell has 3 states, two consecutive cells can have 9 combinations, which is enough to record 3 logic bits(i.e., 8 states). Therefore 256 cells can be split into 128 pairs to record 48 bytes (=128 pairs ×3 bits/pair).

Although approach 3 uses all 256 cells, we only need to distinguish 3 states per cell, which requires a smaller RESET current to initialize each cell. According to the write model in Section 2, we reduce memory line level RESET power to 67%, and increase memory line lifetime to ×58 (with an assumption of all bits being changed).

3.2 Elastic RESET

Given the large potential gain from utilizing non-2^n-state MLC cells, we propose elastic RESET (ER) in this paper. ER is a technique initializing a PCM line according to the storage need at runtime. ER is applicable to n-bit MLC. The following illustrates how it works using 2-bit MLC.

To write a PCM line to 2-bit MLC, we first compress the data using *frequent pattern compression (FPC)* [1]. And then (1) if the compressed data (together with FPC meta information) is $\leq$ half of the original size, we save the data in the line and treat each cell as 2-state SLC; (2) if the compressed ratio is $>$ 50% but $\leq$ 75%, we treat each cell as 3-state 2-bit MLC; (3) if the compressed ratio is $>$ 75%, we treat each cell as 4-state 2-bit MLC. To differentiate these cases, a 2-bit SLC cell flag is attached to each line. The flag gets written, when the memory line changes from one format to another.

When reading a memory line written by ER, the data line and the flag are accessed in parallel. Since the flag is SLC cell, its value is known at the end of the first read iteration, which can guide if we should continue the second read iteration for the line.

Recent device study [5] found RESET resistance variation becomes larger when smaller RESET current is applied. The largest variation (σ) can reach $\sim$ 20%. Therefore, we conservatively increase the initial RESET resistance of the non-2^n-state to tolerate variations. To make a n-state cell, we adopt the mean resistance of the original $n+1$ resistance range as the new initial RESET resistance. For example, for 3-state 2-bit MLC cell, we adopt $10^{7.2}\Omega$ (Table 1) as the new initial RESET resistance.

ER, while reducing RESET current, still initializes cells to be written into similar states. It does not require more precise programming circuit and only slightly increases control complexity. ER sets an initial state with lower resitance, which may take less iterations to finish MLC write. However, in this paper, we conservatively assume that writing non-2^n-state MLC takes the same number of write iterations as the original.

3.3 Fraction Encoding

With elastic RESET, we need to store multiple bits in multiple cells. Figure 5 illustrates a 2-bit MLC with three resistance states: '00', '01' and '1X', referred to as 3-state cell. Two 3-state cells have 9 state combinations, which are enough for representing 3 bits (or 8 states). The read latency for 3-state cell is the same as 2-bit MLC cell, since both require two read iterations.

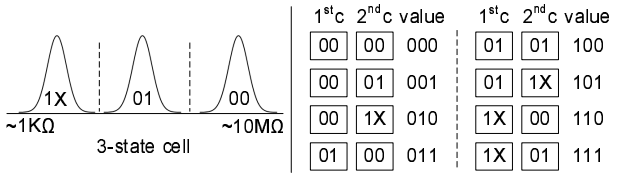


Figure 5: 4/3 fraction encoding.

Figure 5 shows a naive data encoding with two 3-state cells. With naive encoding, one bit flip ('010' to '011') may trigger changes of both two cells. Figure 6 shows the intuitive guess that the rightmost bit is more likely to flip. To minimize cell changes and thus extend chip lifetime, we develop a new encoding that ensures only one cell change if the rightmost bit flips (Figure 7(a)).

We further optimize this encoding by taking write energy into consideration. Since '01' is the intermediate state, it is slow to write and requires more write energy. So, fraction encoding should have as less '01's as possible. Figure 7(a) has six '00's, six '01's and four '1X's. Figure 7(b) shows the improved fraction encoding with only four '01's. Our technique is orthogonal to the energy-efficient data encoding in [23].

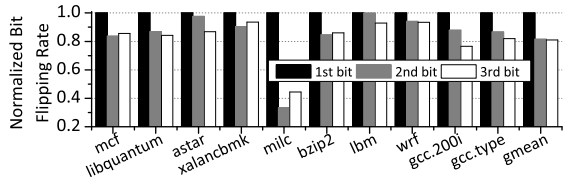


Figure 6: Flipping rate comparison among 3 bits.

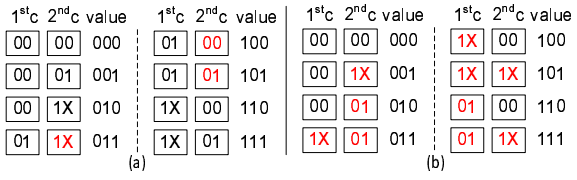


Figure 7: Optimized encoding for cell flipping (a) and energy (b).

3.4 Architectural Designs

Integrating ER in PCM chip faces the following two challenges. First, existing PCM chips usually adopt *differential-write* [26] or flip-n-write [8] to reduce PCM writes at the cell level. Although recent study [12] found that the overall cell changing rate is comparable before and after compression, ER exhibits different pattern of bit changes at the cell level. As Figure 8 shows, ER has larger cell flipping rate than compressed 2-bit MLC. In the worse case, ER may shorten the chip lifetime due to increased cell writes.

Second, when compression ratio changes, a PCM line may switch between different modes. For example, assume the old data in a memory line has a 70% compression ratio such that each cell was treated as 3-state MLC. If the new data has a 78% compression ratio, we need to treat each cell as 4-state MLC, which requires to reset **many cells** due to mode switch. If the compression ratio swings above and below 75%, we may see a large increase of cell changes, resulting in significantly shortened chip lifetime.

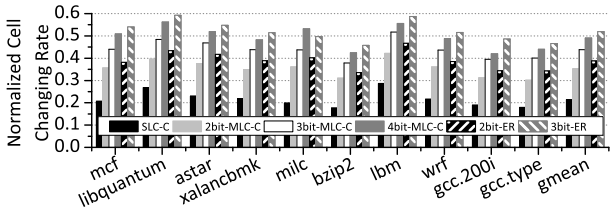


Figure 8: Cell changes for different schemes with compression.

To address these challenges, we propose to dynamically monitor bit flips to prevent ER-introduced cell change increase. The intuitive idea is to compare the number of cell changes with and without ER, and disable ER if the increase is above a preset threshold.

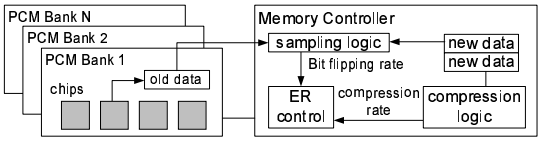


Figure 9: Dynamic sampling for ER.

Figure 9 illustrates our sampling based dynamic monitoring design. A two-entry sample buffer is added to the onchip memory controller while a one-entry buffer is added to each memory bank. We choose to sample one write per 100 offchip writes. We first save the new data in the onchip sample buffer and then read the old data into the PCM chip buffer. The old data is not sent back to the memory controller directly due to the fact that it takes time to switch from write mode to read mode for modern offchip buses. Instead, the old data is buffered in the PCM chip buffer, and is sent back when the bus is in read mode. The memory controller explicitly switches the bus mode to get the old data if the onchip sample buffer is full.

Once the memory controller has both the new and the old data, it computes and compares the number of cell changes (1) if using compressed format; and (2) if using the ER operation. If the compression ratio triggers a mode switch, we pessimistically assume all cells need to be changed. The memory controller disables ER if the increased cell change is above a preset threshold.

4. EXPERIMENTAL METHODOLOGY

To evaluate the effectiveness of our proposed design, we simulated and compared ER with related works using a

PIN-based trace-driven simulator. Table 2 lists our baseline configuration. We strictly follow the baseline configuration from [10].The 4GB PCM memory has 32 banks, 8-entry read queue and 16-entry write queue.

For the baseline configuration, we modeled cache and memory bank conflicts, cache and memory bus contention and memory bus scheduling constraints in our simulator. In the case that the write queue is completely full, the memory controller launches a write burst, where only writes are issued and all reads will be held until the write queue is empty. We adopted flip-n-write [8] and power management policy in [10], and MLC-to-SLC form switching [2, 12]. We adopted Frequent Pattern Compression (FPC) [1] and had highly compressible lines saved in SLC in the baseline.

We picked up 5 integer and 5 floating point benchmarks from SPEC2006 with the compression ratio shown in Figure 2. `gcc-200i` and `gcc-type` are `gcc` with two different inputs. The average compression ratio is about 71%. For fair discussion, when using 2-bit MLC PCM, if the compression ratio is $\leq 50\%$, then our scheme is close to FPC-based compression design; if the compression ratio is $> 75\%$, then our scheme is close to the baseline as few opportunities can be exploited. While we did not find a benchmark whose compression ratio fluctuates below and above 75%, we did observe that some memory lines change from compressible to uncompressible at runtime. We evaluated the entire execution for compression/lifetime evaluation. For performance results, we skipped the warmup phase and ran 50 billion instructions to get the IPC.

Processor	4-core, 2GHz, Intel atom-like cores
I/D-L1	private, 32KB/32KB, 4-way, LRU 64B line size, 1-cycle access latency
L2	private, 8MB, 16-way, LRU 64B line size, 10-cycle access latency
PCM Main Memory	4GB, 64B line size, 32 banks Frequent pattern compression frontend 8/16-entry read/write queues Read first, write burst when full 1000-cycle write, 250-cycle read If compressed data size $\leq 50\%$, write in SLC, otherwise write in MLC flip-n-write at line level

Table 2: Baseline Configuration

5. EVALUATION

In this paper, we studied following schemes.

- `n-bit MLC/C` — n-bit MLC with compression. Highly compressible lines are stored as SLC.
- `n-bit MLC/2` — In addition to `n-bit MLC/C`, for compression ratio (50%, 75%], `2-bit (3-bit) MLC/2` stores 25% data line in SLC (2-bit MLC) while the other in 2-bit (3-bit) MLC [2].
- `n-bit ER` — n-bit MLC using ER.

Hardware Overhead. We first evaluated the hardware overhead. FPC overhead is negligible [12]. For sampling-based cell change monitoring, we added 192B onchip SRAM buffer and 96B per bank offchip SRAM buffer. It has negligible timing overhead as it is not on the critical path. The area and energy overhead is also negligible due to its simple control and low activity.

The relatively large hardware overhead comes from 2 SLC bits per PCM line (0.8%) added to indicate the mode of each line. It has no timing overhead due to its parallel access, with the result telling if reading of data cells should continue.

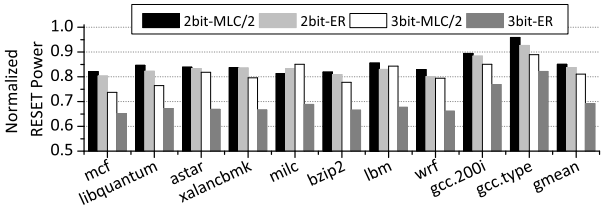


Figure 10: RESET power comparison (normalized to `n-bit MLC/C`)

RESET Power Reduction. Figure 10 compares the RESET power (donimating write power) of different schemes. On average, `ER` reduces 17% and 31% RESET power for 2-bit and 3-bit MLC PCM over `n-bit MLC/C` respectively. Compared to `MLC/2`, `ER` reduces RESET power by 2% and 15% for 2-bit and 3-bit MLC respectively. For 2-bit MLC, `MCL/2` and `ER` have a similar RESET power, as `MCL/2` loses from reseting MLC cells but gains from reseting some cells in SLC mode. `MCL/2` has less lower RESET opportunity to exploit for 3-bit MLC.

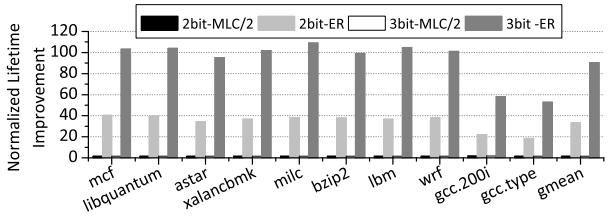
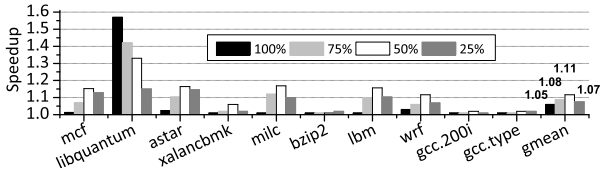
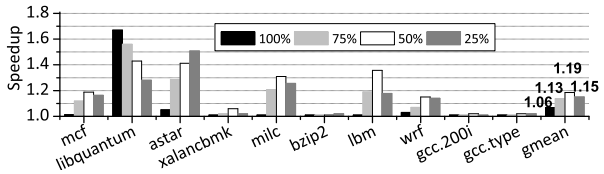


Figure 11: Lifetime improvement (normalized to `n-bit MLC/C`).

Lifetime Improvement. As Figure 11 shows, `ER` significantly improves the PCM chip lifetime by 32× (89×) for 2-bit (3-bit) MLC. Smaller RESET energy per cell produces longer PCM chip endurance. `gcc-200i` and `gcc-type` have less lifetime improvements. We found that they have execution phases in which written PCM lines are less ($> 75\%$) or highly ($\leq 50\%$) compressible (as shown in compression ratio study Figure 13).

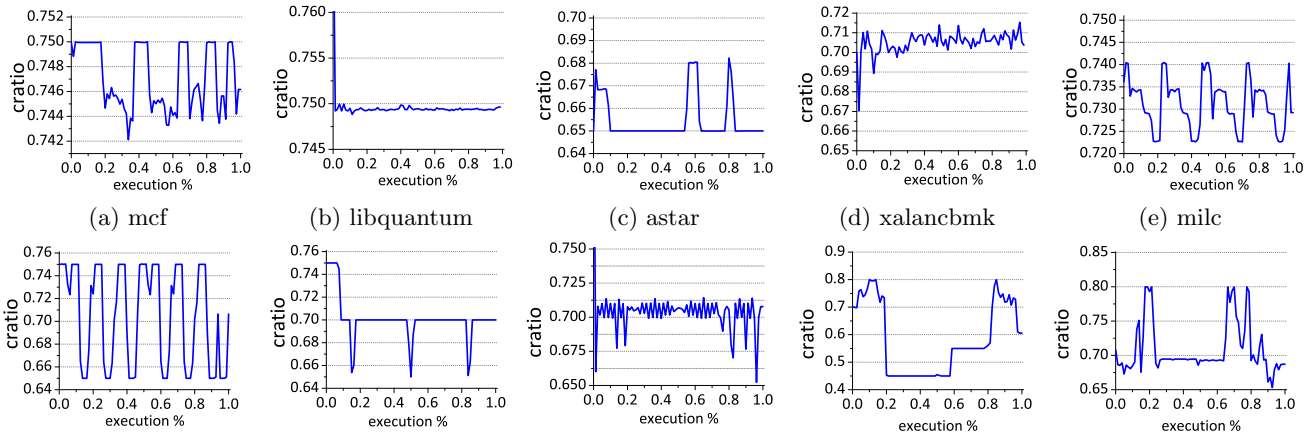


(a) 2-bit MLC



(b) 3-bit MLC

Figure 12: Performance improvement under different power budgets



(a) mcf (b) libquantum (c) astar (d) xalancbmk (e) milc

(f) bzip2 (g) lbm (h) wrf (i) gcc-200i (j) gcc-type

Figure 13: Compression ratio of the whole execution (X-axis is normalized execution percentage).

Exploiting Low-Power Write for Performance. By reducing write power per PCM write, ER opens the door for either low-power or high-performance designs. For the latter, PCM system usually has tight power budget such that significantly reducing per write power enables more concurrent writes. Figure 12(a) illustrates the potential of improving system performance when using 2-bit MLC PCM. We evaluated with different power budgets. For the power budget of current DDR2 standard, `ER` gets only 5% performance improvement indicating that the current power is sufficient. With lower power budget, `ER` shows significant benefits over `n-bit MLC/C`. With 75% and 50% of the original power budget, `ER` obtains 8% and 11% performance improvements over `n-bit MLC/C`, respectively. However, if we further reduce the power budget, `ER` achieves less (i.e., 7%) improvement as the extreme low power budget has become a global bottleneck for the whole system. Figure 12(b) compares the performance improvements when using 3-bit MLC PCM. The results showed that `ER` has the potential to gain better performance improvements for more-bit MLC.

Compression Study. We also studied the compression ratio during the entire execution. Figure 13 reports the dynamic compression ratio at runtime, i.e., the averaged compression ratio of all write instances for the entire program execution. The results show that the compression ratios are relatively stable such that `ER` can benefit for the entire execution. Note while the averaged compression ratio may be below 75%, some memory lines are still not compressible.

6. CONCLUSIONS

In this paper, we developed elastic RESET (ER) to construct non-2^n-state MLC cells. Non-2^n-state cells store less information but significantly extend cell lifetime due to using less RESET initialization energy. By adopting compression and fraction encoding, ER can store the compressed line using non-2^n-state cells for PCM lines with different compression ratios. Our experimental results showed that on average, ER gains 17%/31% write power reduction, 32×/89× lifetime improvement, for 2-bit/3-bit MLC PCM respectively.

7. REFERENCES

[1] A. R. Alameldeen and D. A. Wood, "Adaptive Cache Compression for High-Performance Processors," in *ISCA*, 2004.

[2] M. Arjomand, *et al.*, "A Morphable PCM Architecture Considering Frequent Zero Values," in *ICCD*, 2011.

[3] M. Awasthi, *et al.*, "Efficient Scrub Mechanisms for Error-Prone Emerging Memories," in *HPCA*, 2012.

[4] F. Bedeschi, *et al.*, "A Bipolar-Selected Phase Change Memory Featuring Multi-Level Cell Storage," *JSSC*, 2009.

[5] S. Braga, *et al.*, "Modeling of Partial-RESET Dynamics in Phase Change Memories," in *ESSDERC*, 2010.

[6] S. Braga, *et al.*, "Voltage-Driven Partial-RESET Multilevel Programming in Phase-Change Memories," in *ESSDERC*, 2010.

[7] C. Calligaro, *et al.*, "Comparative Analysis of Sensing Schemes for Multilevel Non-volatile Memories," in *ICISS*, 1997.

[8] S. Cho and H. Lee, "Flip-N-Write: A Simple Deterministic Technique to Improve PRAM Write Performance, Energy and Endurance," in *MICRO*, 2009.

[9] B. Gleixner, *et al.*, "Reliability Characterization of Phase Change Memory," in *EPCOS*, 2009.

[10] A. Hay, *et al.*, "Preventing PCM Banks from Seizing Too Much Power," in *MICRO*, 2011.

[11] L. Jiang, *et al.*, "Enhancing Phase Change Memory Lifetime through Fine-Grained Current Regulation and Voltage Upscaling," in *ISLPED*, 2011.

[12] L. Jiang, *et al.*, "Improving Write Operations in MLC Phase Change Memory," in *HPCA*, 2012.

[13] M. Joshi, *et al.*, "Mercury: A Fast and Energy-Efficient Multi-level Cell based Phase Change Memory System," in *HPCA*, 2011.

[14] K. Kim and S. J. Ahn, "Reliability Investigations for Manufacturable High Density PRAM," in *IRPS*, 2005.

[15] D. Mantegazza, *et al.*, "Statistical Analysis andModeling of Programming and Retention in PCM Arrays," in *IEDM*, 2007.

[16] T. Nirschl, *et al.*, "Write Strategies for 2 and 4-bit Multi-Level Phase-Change Memory," in *IEDM*, 2007.

[17] N. Papandreou, *et al.*, "Estimation of Amorphous Fraction in Multilevel Phase Change Memory Cells," in *ESSDERC*, 2009.

[18] M. K. Qureshi, *et al.*, "Enhancing Lifetime and Security of PCM-based Main Memory with Start-Gap Wear Leveling," in *MICRO*, 2009.

[19] M. K. Qureshi, *et al.*, "Improving Read Performance of Phase Change Memories via Write Cancellation and Write Pausing," in *HPCA*, 2010.

[20] M. K. Qureshi, *et al.*, "Morphable Memory System: A Robust Architecture for Exploiting Multi-Level Phase Change Memories," in *ISCA*, 2010.

[21] B. Rajendran, *et al.*, "Analytical Model for Reset Operation of Phase Change Memory," in *IEDM*, 2008.

[22] S. Schechter, *et al.*, "Use ECP, not ECC, for Hard Failures in Resistive Memories," in *ISCA*, 2010.

[23] J. Wang, *et al.*, "Energy-efficient Multi-Level Cell PCM System with Data Encoding," in *ICCD*, 2011.

[24] D. H. Yoon, *et al.*, "FREE-p: Protecting Non-volatile Memory against Both Hard and Soft Errors," in *HPCA*, 2011.

[25] W. Zhang and T. Li, "Characterizing and Mitigating the Impact of Process Variations on Phase Change based Memory Systems," in *MICRO*, 2009.

[26] P. Zhou, *et al.*, "A Durable and Energy Efficient Main Memory Using Phase Change Memory Technology," in *ISCA*, 2009.