Enhancing Phase Change Memory Lifetime through Fine-Grained Current Regulation and Voltage Upscaling

Lei Jiang †, Youtao Zhang ‡, Jun Yang †

† Electrical and Computer Engineering Department
University of Pittsburgh
Pittsburgh, PA 15261, USA
†{lej16,juy9}@pitt.edu

‡ Computer Science Department
University of Pittsburgh
Pittsburgh, PA 15260, USA
‡zhangyt@cs.pitt.edu

***Abstract*—Phase Change Memory (PCM) recently has emerged as a promising memory technology. However it suffers from limited write endurance. Recent studies have shown that the lifetime of PCM cells heavily depends on the *RESET* energy. Typically, larger than optimal *RESET* current is employed to accommodate process variation. This leads to over-programming of cells, and dramatically-shortened lifetime. This paper proposes two innovative low power techniques, *Fine-Grained Current Regulation (FGCR)* and *Voltage Upscaling (VU)*, to cut down the *RESET* current, leaving a small number of *difficult-to-reset* cells unused. We then utilize error correction code to rescue those cells. Our experimental results show that FGCR and VU reduce the PCM write power by 33%, and prolong the lifetime of a PCM chip by 71%-102%.**

***Keywords*-Phase Change Memory; Hard Faults; Cell Endurance; Low Voltage;**

I. Introduction

Phase Change Memory (PCM) has recently emerged as one of the most promising memory technologies to alleviate the leakage and scalability problems of traditional DRAM. PCM has many advantages such as non-volatility, comparable read speed as DRAM, near zero cell leakage, and good scalability beyond 22nm [14]. However, PCM also has limitations such as slow write speed, limited write endurance, and resistance drift for multi-level cells. These problems have been partially addressed in literature. Recent work [1] utilizes DRAM write buffer to mitigate long write latency. Several memory line-level techniques [2], [4], [3] have been proposed to eliminate unnecessary writes to PCM cells. Wear leveling schemes [5], [6], [3] balance writes in different memory regions. Salvaging techniques [7], [8] can extend the lifetime of PCM chip even after a non-negligible portion of cells fail.

In this paper, we focus on improving PCM chip lifetime under process variations (PV). As technology enters deep submicron regime, PCM faces difficulties in controlling the device characteristics precisely during fabrication, e.g., transistor length and Joule heater radius [9]. On the one hand, the parameter variation determines the minimal *RESET* current of each PCM cell; on the other hand, PCM cell endurance mainly depends on the *RESET* current [10], [11]. However, existing coarse-grained *RESET* current delivery scheme conflicts with the non-uniform distribution of minimal *RESET* current. As a result, a large portion of cells are over-programmed, which reduces the lifetime of the entire chip significantly. Studies at the device level [12], [11] reported that a single PCM cell with optimal *RESET* current can achieve $> 10^{10}$ write cycles. However, architecture studies adopt $10^6 \sim 10^8$ write cycles at the PCM chip level. The latest industrial prototype [13] only promises 10^6 write cycles.

Since a small excess of *RESET* amplitude results in a dramatic degradation of cell endurance, coarse-grained current regulation, e.g., having the largest *RESET* current for the entire chip or per 4MB block [9], seriously harms cell endurance. In this paper, we propose fine-grained *RESET* current regulation designs to reduce PCM write power and to improve chip lifetime substantially. Our contributions include:

- We propose Fine-Grained Current Regulation (FGCR) to tailor *RESET* current for each memory line, and rescue *difficult-to-reset* cells using error correction technique. Our scheme reduces the *RESET* current below normal working current.
- We propose Voltage Upscaling (VU) to dynamically adjust the coverage of error correction bits between *difficult-to-reset* cells and broken cells. VU helps to extend the lifetime without losing the ability to protect cells from hard errors.
- We evaluate the proposed schemes and compare them to existing current regulation designs. Our experimental results show that FGCR and VU improve PCM chip lifetime by 71%-102% under different process variation settings.

The rest of this paper is organized as follows. Section II introduces background on PCM. Our motivation is presented in Section III. Section IV and Section V elaborate FGCR and VU designs respectively. Section VI presents our experimental methodology. Section VII reports the experimental results and comparisons. Additional related work is discussed in Section VIII. We conclude in Section IX.

II. Background

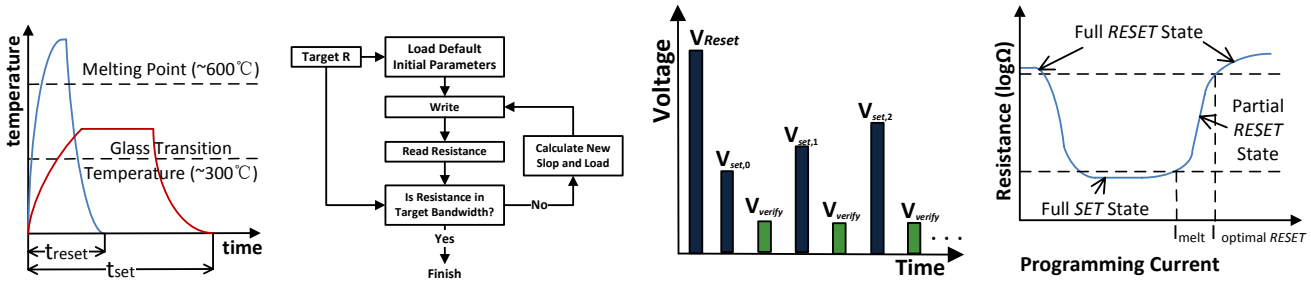
For clarity, we focus our discussion on single-level PCM cells while our designs are also applicable to multi-level PCM cells [15]. The former stores one bit per cell and the latter saves multiple bits per cell.

A. Phase Change Memory and its Write Operation

As shown in Figure 1(a), there are two kinds of PCM write (or programming) operations [14]. A *SET* operation heats the phase change material, GST, above the crystallization temperature (300 °C) but below the melting temperature (600 °C) using a long but small current. A *SET* operation writes the PCM cell into a logic '1' (crystalline state). In contrast, with a short and large current, the GST is melted and quenched quickly by a *RESET* operation, which writes the PCM cell into a logic '0' (amorphous state).

Due to non-deterministic characteristics of GST, different cells, even the same cell at different times, respond differently to the same programming pulse [15], [16], [17]. Therefore, PCM programming widely adopts iteration-based programming and verifying (P&V) write scheme (Figure 1(b)). Particularly, writing a '1' is presented in Figure 1(c). Here, a *RESET* operation is always first conducted to put the cell in an initial state. A series of *SET* and verify (read) operations then follow until the target resistance level is reached.

978-1-61284-660-6/11/$26.00 © 2011 IEEE



(a) *RESET* and *SET* operations (b) Iterative programming algorithm (c) PCM programming scheme (d) *R-I* programming characteristics

Fig. 1. PCM writing operations.

Iterative P&V write strategy is able to precisely control the written resistance in each PCM cell.

The minimal current that heats PCM cell into a full *RESET* state is referred as optimal *RESET* current, which is the most critical factor that decides the cell endurance [11]. Figure 1(d) illustrates the responses of a PCM cell's resistance to different programming current. The initial state is a full *RESET*, which corresponds to the largest PCM resistance value. After several *SET* operations, the cell reaches a full *SET* state, which corresponds to the minimal resistance level. This process is a transition from full *RESET* to full *SET* state. A following write will start with a *RESET* initialization, using a current large enough to melt the GST and help the resistance to reach the full *RESET* state again. During the transition from *SET* to *RESET* state, a larger *RESET* current pulse is preferred, because a smaller one (larger than I_{melt}) would block the PCM cell in a partial *RESET* state, where read operation reliability and large retention time cannot be guaranteed [18].

B. Phase Change Memory Failure Model

The failure mechanisms of PCM cells have been studied at both device [10], [11] and architectural levels [8], [20]. Device level research [10], [11], [19] found that over-programming i.e., applying current larger than optimal *RESET* current, leads to shortened endurance in weak cells. Due to the existence of process variations, optimal *RESET* current varies across memory cells in one chip. A recent architectural work [9] analyzed process variation, including random variation and systematic variation, on gate length of write driver transistors and structural parameters of PCM cell, such as bottom electrode contact diameter, thickness of GST and of heater (a, H and H_e in Figure 2). Other architectural works [7], [8], [20], without modeling the effect of RESET current on endurance in details, directly assume the endurance of PCM cells follows *Normal Distribution*.

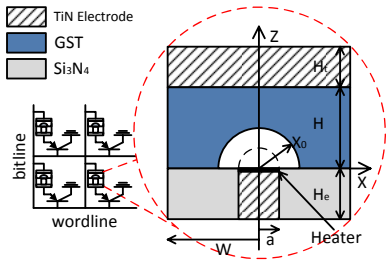


Fig. 2. PCM cell array.

The latest works [7], [8], [20] reported that random variation is the predominant type of process variation in PCM chip, and no large systematic variation is observed within a die. In this paper, we model the variations of structural parameters that affect optimal *RESET* current as random process variation. We adopt a detailed *RESET* current analytical model [21] in this paper. This model considers not only the bottom electrode contact diameter, the thickness of GST and heater, but also several other important parameters including the thickness of top electrode (H_t) and the radius of the whole PCM cell (W), as shown in Figure 2. Moreover, various temperature and heat distributions at different regions inside the memory cell are summarized in this model. Compared to device level experimental data, the result of this analytical model only has 5% error [21].

C. Error Correction for PCM

Since cell over-programming significantly shortens the lifetime of a non-negligible portion of weak cells, it is important to adopt appropriate error correction scheme to tolerate cell failures. Given that hard faults (instead of soft errors) dominate the cell failure, PCM chips prefer error correcting pointers (ECP) [8] over traditional Hamming (72, 64) ECC code. As shown in Figure 3, one ECP consists of one 9-bit pointer (for a 64B memory line) and one replacement cell (for recording the data of the failed cell). Although the storage overhead of 6 ECPs (∼12.5%) is the same as Hamming (72, 64) ECC code for a 64B memory line, experimental results [8] showed that memory lines under the protection of ECP last much longer. In this paper, we adopt 6 ECPs in the baseline configuration.

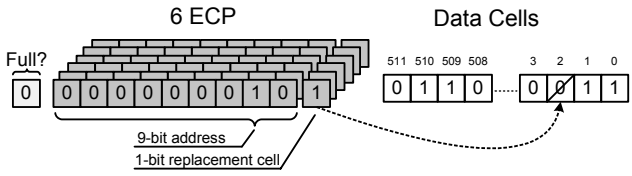


Fig. 3. Error correcting pointer (ECP).

III. Motivation

A simple design, which can ensure reliable programming for all PCM cells in one chip (or 4MB block), has to tailor the *RESET* current to tolerate the weakest cells. Consequently, the write driver must raise the *RESET* voltage to the maximum level, which over-programs most cells, resulting in seriously shortened PCM cell endurance.

Figure 4 plots the cell lifetime as a function of electrical pulse energy applied to the cell during the *RESET* operation [11]. 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.

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

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

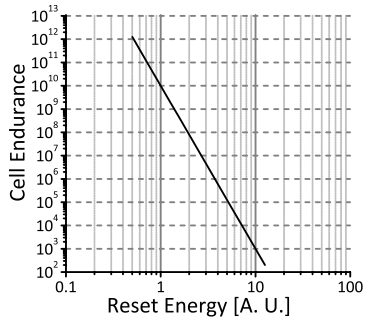


Fig. 4. The cell lifetime as a function of *RESET* energy.

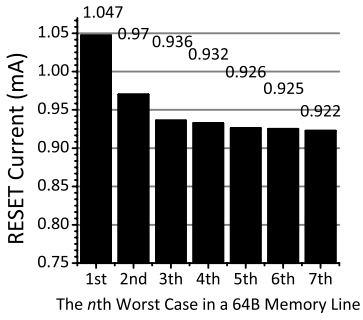


Fig. 5. The worst cases of *RESET* currents in a 64B memory line.

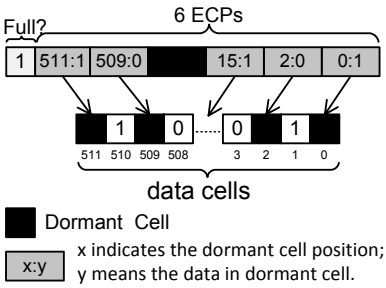


Fig. 6. Fine-Grained Current Regulation (FGCR).

Here, I_{reset}, E_{reset} denote the *RESET* current and energy respectively; R represents cell resistance; and T is the *RESET* pulse width measured in time (We use $50ns$ [15] in this paper).

Assuming the chip has strong systematic PV, previous study proposed to deliver suitable currents to different PCM sub-blocks (e.g. 4MB block) to mitigate over-programming [9]. However, under strong random variations (from recent studies [20], [8], [7]), over-programming is still very common. Figure 5 shows *RESET* current values of the 7 most *difficult-to-reset* cells in a typical 64B memory line (settings shown in Section VI). Even delivering different currents to each line still leaves most cells in a line over-programmed.

Based on the observation that each line contains only few *difficult-to-reset* cells, we utilize the integrated error correction scheme (e.g., ECP) to alleviate over-programming by intentionally reducing the *RESET* below the optimal current to reset these *difficult-to-reset* cells. The latter cells are *temporarily* considered as failed cells which are recovered by ECP. For the particular line in Figure 5, if we mask 6 *difficult-to-reset* cells, the *RESET* current can be reduced from 1.047mA to 0.922mA, such that the line gains extra 493% lifetime.

IV. Fine-Grained Current Regulation

We first propose Fine-Grained Current Regulation (FGCR), a fine-grained *RESET* current tuning design to mitigate over-programming and to extend chip lifetime. Our design is at the line level i.e., each 64B memory line adopts a different *RESET* current. We will compare the trade off between 64B line level and 4KB page level designs in Section VII.

When lowering the *RESET* current for a memory line, several cells may not be able to be fully reset. They are referred as *dormant* cells. Since error correction ECP bits are used to salvage worn-out cells in later stage of the chip's lifetime, these correction bits are free in the early stage. Thus we can utilize ECP bits to recover *dormant* cells in the early stage. Figure 6 shows how ECPs rescue *dormant* cells introduced by low *RESET* current. During post-fabrication calibration, the 6 most *difficult-to-reset* cells are chosen to become *dormant* cells. They are recorded and rescued using ECP. Since ECP adopts priority decoding [8], *dormant* cells can appear in ECP field. Here we assume there are no hard errors yet. Section V discusses how to handle *dormant* cell when hard errors appear.

FGCR consists of two main components: one is the memory line level *RESET* current tuning, the other is the deployment of error correction in the post-fabrication stage for further reducing *RESET* current.

Fine-grained *RESET* current tuning requires modifications to the charge pump for PCM. A charge pump is a DC to DC converter using capacitor as energy storage elements to create either a higher or lower voltage power source [22]. Since various operations on PCM chip require higher voltage than conventional V_{dd}, charge pumps have become the most popular solution for voltage boosting (shown in Figure 7). A conventional charge pump has 4-16 charging stages during the charging process, corresponding to different voltage levels. One charging process begins from the lowest voltage charging stage and ends at the highest voltage stage. Therefore, the charging latencies for higher voltage stages are longer than for lower voltage stages. The charge pump for write driver has 16 charging stages to accommodate multiple intermediate *SET* voltages in iteration-based write scheme [23].

Fine-grained voltage regulation may require more than 16 regulation levels. To enable fine-grained voltage regulation without introducing too much hardware overhead, we propose to divide the original write driver charge pump into two small ones, each of which has 16 charging stages (shown in Figure 8). We do not add new write drivers, since two charge pumps can share the same group of write drivers. The *RESET* current controller decides which charge pump driver to enable by referring to a small *reset current lookup table* that records the *RESET* current for each memory line. The table is initialized at the post-fabrication stage. The latency to access the table and writer driver charge pump incur no performance penalty as they can be overlapped with pre-write read operation of widely adopted differential write [3]. We modeled the charge pump design in [24] with a 1GHz working frequency. The area overhead introduced by the dual charge pump design is $159\mu m \times 42\mu m$ under $32nm$ CMOS technology [24]. The overhead of other peripheral circuit and selection logic are small enough to be neglected.

V. Voltage Upscaling

FGCR allocates all ECP bits to rescue *dormant* cells. It works well when there are no hard errors. However, when weak cells start to wear out, we need to retrieve some or all ECP bits to rescue hard errors. This is done through Voltage Upscaling (VU), which activates *dormant* cells by increasing *RESET* voltage and moving the corresponding ECPs to rescue hard errors.

Figure 9 shows our VU design. Assume the RESET current is α for this line and all ECP bits are allocated to rescue dormant cells. In particular, the first ECP rescues dormant cell #1. When the first hard error occurs at cell #2, we need Voltage Upscaling (VU) since there is no ECP to rescue it. After boosting the RESET current to above α, some dormant cells can now be fully reset, e.g., now dormant cell #1 can be reset reliably. And thus, it does not need the ECP to rescue. Instead, we use the first ECP to recover the hard error. We use 3 bits to record how many ECPs have been used to rescue hard errors.

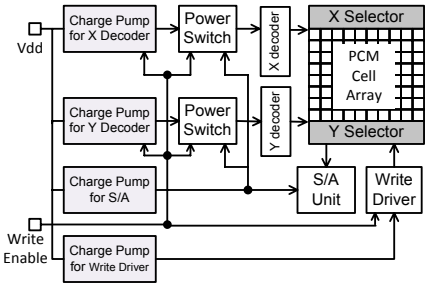


Fig. 7. Charge pump system for PCM array.

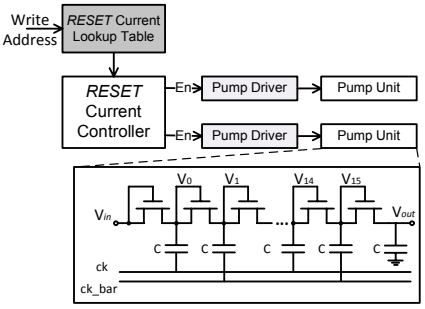


Fig. 8. Charge pump system for FGCR.

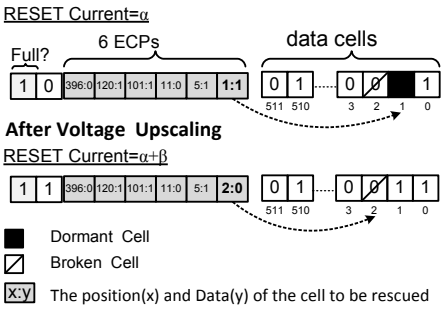


Fig. 9. Activate dormant cells through VU.

Ideally each voltage upscaling converts one ECP from protecting *dormant* cell to rescuing a new hard error. However, depending on the current precision, more than one dormant cells may be activated. For example, each upscaling may activates 3 dormant cells such that all 6 ECPs are converted in two upscalings. Let us assume VU_k finishes the conversion in k upscalings. The larger k is, the less over-programming the memory line experiences.

To determine the new *RESET* voltage at each voltage upscaling, we perform the similar iteration-based testing method as in Figure 1(c). The write driver increases *RESET* voltage gradually such that some dormant cells can be reliably reset. The new *RESET* current value is then recorded in the *reset current lookup table*.

After each voltage upscaling operation, the endurance expectation of each PCM cell in one memory line is re-calculated by Equation 3. In this equation, exp_end_{n+1} indicates the PCM cell endurance expectation after the $(n+1)^{th}$ voltage upscaling operation; m_end_{n+1} represents the PCM cell endurance modelled by methods in Section VI with the *RESET* current of the $(n+1)^{th}$ voltage upscaling operation; and ΔW_n means the cumulative write traffic between the $(n-1)^{th}$and the n^{th} voltage upscaling operations.

$$exp\_end_{n+1} = (1 - \sum_{i=0}^{n} \frac{\Delta W_n}{m\_end_n}) \times m\_end_{n+1} \quad (3)$$

VI. Experimental Methodology

To evaluate the effectiveness of our proposed schemes, we adopted Monte Carlo simulation to conduct 10^5 times for each configuration, and reported the averaged results. The configurations are listed in Table I. *current resolution* indicates the size of memory block sharing one current. For example, $Baseline$ adopts the same RESET current for each 4MB memory block [9]. $Line$ indicates the design at memory line (64 bytes) level. *ECP Use Policy* represents how the ECP bits are used. FGCR and VU utilize ECP bits as we discussed in Section IV and V. In $FGCR_{4KB}$, 256 Layered ECPs [8] can either rescue hard faults in a 4KB page range, or be used to mask *difficult-to-reset* cells in the early stage. 256-Layered ECPs per page has similar storage overhead as 6 ECPs per memory line. VU_k performs k voltage upscaling operations and activates $6/k$ dormant cells per operation.

We use following assumptions in the simulation:

- We adopted the cell structure parameters from [21], modelled its PV using *Normal Distribution*, and chose $variation$ as 0.095, 0.1 and 0.105. We then plugged the structure parameters into the *RESET* current model from [21], and generated the *RESET* current distribution based on the energy-lifetime model (Equation 1) from [11]. The RESET current at memory line level ranges from 0.88mA to 1.15mA. This matches the experimental data from [9], and the endurance distribution at chip-level in [7], [8], [20] (10^6 (weak cells) to 10^8).
- We adopted $0.01mA$ current accuracy in our design. A charge pump design with $0.01mA$ current accuracy was reported in a recent design [23]. Similar current accuracy was used in [9].
- We adopted wear leveling and *differential write* at device level. To focus on studying the impact of *RESET* current on PCM endurance, we assumed perfect wear leveling, and 50% bit changes per write operation. These assumptions were also made in [8], [20].

TABLE I
Simulated Configuration

Name	Current Resolution	ECP Use Policy
$Baseline$	4MB array	6ECPs/line for hard faults
$Page$	4KB page	6ECPs/line for hard faults
$Line$	64B line	6ECPs/line for hard faults
$FGCR_{4KB}$	4KB page	256LECPs/page for FGCR
$FGCR_{64B}$	64B line	6ECPs/line for FGCR
VU_k	64B line	6ECPs/line for FGCR with k VU(s)
iVU_k	64B line	6ECPs/line for FGCR with k VU(s), with ideal current supply

VII. Evaluation

A. Current Regulation Resolution

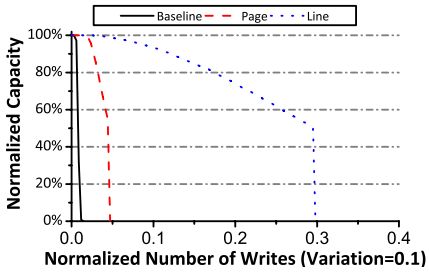


Fig. 10. Comparing current regulation at different levels (Normalized to the cumulative write traffic of iVU_6 with variation 0.095 until half of memory capacity worn out).

Figure 10 compares the lifetime under different current regulation resolutions. We assume a chip fails when half of its capacity wear out. Only variance=0.1 is shown due to space limitation. $Baseline$ fails very quickly, which indicates that adopting one current for 4MB sub-array cannot effectively alleviate over-programming. This is mainly due to the fact that neighboring cells in one 4MB block are short of similarity on optimal *RESET* currents. $Page$ achieves 400% improvement, which motivates us to perform more fine-grained current regulation. $Line$ regulates RESET current at memory line level and achieves 610% longer lifetime over $Page$. $Line$ requires

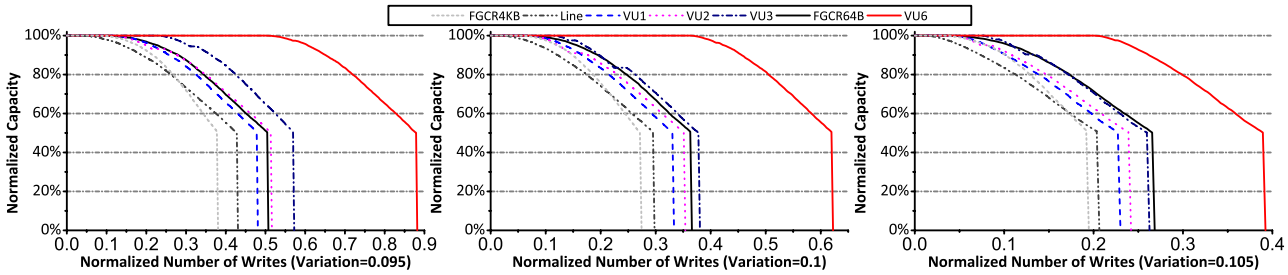


Fig. 11. Memory line lifetime after applying FGCR and VU (normalized to the cumulative write traffic of iVU_6 with variation 0.095 until half of memory capacity worn out).

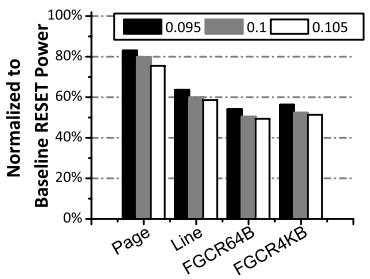


Fig. 12. PCM RESET power savings.

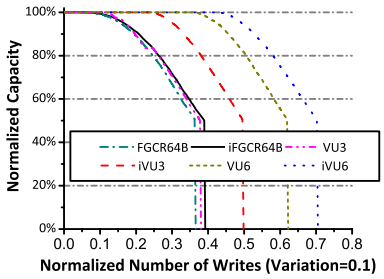


Fig. 13. Lifetime difference between Ideal *RESET* and Discrete *RESET* currents (Normalized to iVU_6 with variation 0.095).

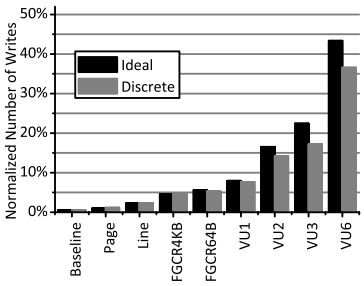


Fig. 14. The first failed PCM line appearing time (Variation=0.1, normalized to iVU_6 with variation 0.095).

extra storage to buffer the current for each line. We will evaluate the overhead in Section VII-F.

B. Lifetime Improved by FGCR and VU

In the preceding experiment, $Line$ still chooses the RESET current such that all cells in one line can be reliably reset at post-fabrication. We then evaluate FGCR and VU. The lifetime improvements are summarized in Figure 11. Since FGCR uses all ECP bits for fine-grained current tuning, it does not perform error correction: a memory line fails when the first hard fault appears in the line. The results show that $FGCR_{64B}$ improves lifetime by 17.8%-30% over $Line$ under different variations. Also from the figure, we observed that FGCR is not preferrable at coarse granularity: $FGCR_{4KB}$ survives shorter than $Line$ by 10%-11%.

Figure 11 also compares the schemes that adopt different numbers of voltage upscaling (VU) operations. VU_k increases voltage by converting $6/k$ ECPs to rescue hard faults during each voltage upscaling operation, and finishes the conversion of all 6 ECPs in k operations. VU_6 achieves the longest PCM cell lifetime in all cases. For all variation settings, VU_6 stays longer than $FGCR_{64B}$ by 46%-73.8% when half capacity is still available. Through more voltage upscaling operations, PCM cells in one memory line gain longer lifetime by staying in less over-programmed states.

While all VU_ks perform better than $Line$, VU_1 and VU_2 increase current too large at a time, and perform even worse than $FGCR_{64B}$, indicating over-programming hurts lifetime more than hard faults. Under large cell structure variation, the *RESET* current differences among cells in one memory line become significant, so the voltage increasing margin has to be large for driving dormant cells originally covered by ECPs. VU_3 fails sooner than $FGCR_{64B}$ by 2% under 0.105 variation at time when half of capacity is available .

C. RESET Power Savings

By reducing the RESET current using FGCR and VU, we reduce the power of RESET operations significantly. Figure 12 shows FGCR_{64B} and FGCR_{4KB} consume 49%–54% and 51%–56% of the baseline RESET power under different variations. A complete write operation includes one large RESET current, and multiple SET iterations. The latter consumes around 50% power of a RESET operation [3]. In summary, our design can reduce the write power to around 2/3 of the baseline.

D. Current Precision

To study the impact of current precision on the effectiveness of FGCR and VU designs, we compared the current charge pump with $0.01mA$ current accuracy with an ideal charge pump with perfect precision. The results are summarized in Figure 13. Imperfect currents do not hurt $FGCR_{64B}$ much: only 6%-7% lifetime losses were observed for $iFGCR_{64B}$ at the time of 50% available capacity.

In contrast, VU_3 and VU_6 are more sensitive — 25% and 12% lifetime degradations on VU_3 and VU_6 are observed. VU_3 and VU_6 boost the voltage gradually and thus require higher current accuracy. For example, with $0.01mA$ current accuracy, VU_6 may not activate just one dormant cell per upscaling. For some lines, after 3 to 4 upscalings, the RESET current has reached the level that activates all dormant cells.

E. First Fail Line Occurrence Time

Figure 14 compares the appearance of the first failed memory line. Although VU_2 and VU_3 achieve smaller lifetime improvement than $FGCR_{64B}$, they delay the appearance of the first failed memory line. This property sometimes is desirable when the OS does not have the ability to mark failed blocks at the device level. From the figure, VU_6 makes the first fail memory line appear later than $FGCR_{64B}$ by 620%. The limited *RESET* current accuracy also thwarts VU_k to reach the same first fail memory line appearing time as iVU_k.

Comparing the appearance time of the first failed line under iVU_k and iVU_k cases, we found that the difference is big: iVU_6 improves the first fail line occurrence time by 15.9%-27.3% over VU_6 design.

F. Storage Overhead

Table II shows the storage overhead when regulating *RESET* current values at different resolutions for a 64MB PCM chip. Since the current accuracy of our design is $0.01mA$, $Baseline$ only needs 2 *RESET* current levels ($1.14mA$ and $1.15mA$), so 1 bit can represent which current a 4MB sub-array needs. Thus $Baseline$ requires only 2B size storage to maintain the *RESET current lookup table* for a 64MB PCM chip. $Page$ needs 6KB for a 64MB chip with the increasing resolution of *RESET* current.

Regulating currents at memory line level provides additional opportunities to lower down the RESET current for some blocks. The RESET currents for all lines range from $0.88mA$ to $1.15mA$ (instead of $1.14mA$ to $1.15mA$ at 4MB block level). That is, some lines can work at 0.88mA after adopting ECP to rescue 6 dormant cells. To differentiate these 28 current levels, we need 5 bits per line, which adds to 640KB extra storage for a 64MB chip for $Line$. Similarly we need to differentiate 4 different currents at 0.01mA at page level, which translates to 4KB storage overhead for $FGCR_{4KB}$.

The current value differences between the most *difficult-to-reset* cells from each memory line are large, while the number of *RESET* current values of the 7th most *difficult-to-reset* cell in each memory line falls into a smaller scope. So, $FGCR_{64B}$ spends less storage overhead in storing *RESET* current values than $Line$. The storage overhead of $FGCR_{64B}$ is 512KB. Since all VU schemes will end in the form of $Line$, the storage overhead for *RESET* current values of all VU schemes is the same as $Line$.

TABLE II
RESET CURRENT VALUES STORAGE OVERHEAD ESTIMATION FOR A 64MB PCM CHIP

Name	Range(mA)	#	Bits	Overhead(Byte)
$Baseline$	1.14-1.15	2	1	2
$Page$	1.11-1.15	5	3	6K
$Line$	0.88-1.15	28	5	640K
$FGCR_{4KB}$	0.98-1.01	4	2	4K
$FGCR_{64B}$	0.85-0.97	13	4	512K
VU_1,VU_2,VU_3,VU_6	0.88-1.15	28	5	640K

VIII. PRIOR ARTS

[9] analyzes the impact of process variation on several parameters, such as gate length and Joule heater's radius, and generates the distribution of *RESET* currents by one dimensional heat model. This work selects the worst case in a 4MB block as the *RESET* current. Several other techniques, like Frequent Pattern Compression and page classification, are also adopted in [9] to increase PCM chip lifetime. However, too large systematic process variation is modeled in [9], while the latest works [7] [8] [20] observe no significant systematic process variation on PCM. Therefore, without the presence of large systematic process variation, employing the worst case *RESET* current in a 4MB size block may not be effective in increasing PCM lifetime.

IX. CONCLUSION

In this paper, we analyzed the necessity of fine-grained *RESET* current control for PCM array. Due to the large impact of *RESET* current on PCM cell endurance, *RESET* current delivery system at memory line level not only reduces the power consumption of write operations, but also significantly prolongs the chip lifetime. We proposed Fine-Grained Current Regulation to further reduce the *RESET* current at memory line level, and leave the most *difficult-to-reset* cells covered by ECP. We also proposed Voltage Upscaling technique to increase *RESET* voltage for transforming inactive cells to normal working cells and dynamically moving ECP protection from inactive cells to hard faulty cells. Compared to existing *RESET* current memory level supply scheme, our techniques reduce PCM write power by 33% and increase PCM chip lifetime by 71%-102%.

X. ACKNOWLEDGMEMNTS

We thank the anonymous referees for their valuable comments and suggestions. The authors acknowledge the help from PCM@pitt research group and Bo Zhao. This research is supported by National Science Foundation grants CNS-CAREER-0747242 and CNS-1012070.

REFERENCES

[1] M. K. Qureshi, V. Srinivasan, and J. A. Rivers, "Scalable high performance main memory system using phase-change memory technology," in *ISCA*, 2009, pp. 24–33.
[2] S. Cho and H. Lee, "Flip-n-write: a simple deterministic technique to improve pram write performance, energy and endurance," in *MICRO*, 2009, pp. 347–357.
[3] P. Zhou, B. Zhao, J. Yang, and Y. Zhang, "A durable and energy efficient main memory using phase change memory technology," in *ISCA*, 2009, pp. 14–23.
[4] B. C. Lee, E. Ipek, O. Mutlu, and D. Burger, "Architecting phase change memory as a scalable dram alternative," in *ISCA*, 2009, pp. 2–13.
[5] M. K. Qureshi, J. Karidis, M. Franceschini, V. Srinivasan, L. Lastras, and B. Abali, "Enhancing lifetime and security of pcm-based main memory with start-gap wear leveling," in *MICRO*, 2009, pp. 14–23.
[6] N. H. Seong, D. H. Woo, and H.-H. S. Lee, "Security refresh: prevent malicious wear-out and increase durability for phase-change memory with dynamically randomized address mapping," in *ISCA*, 2010, pp. 383–394.
[7] E. Ipek, J. Condit, E. B. Nightingale, D. Burger, and T. Moscibroda, "Dynamically replicated memory: building reliable systems from nanoscale resistive memories," in *ASPLOS*, 2010, pp. 3–14.
[8] S. Schechter, G. H. Loh, K. Straus, and D. Burger, "Use ecp, not ecc, for hard failures in resistive memories," in *ISCA*, 2010, pp. 141–152.
[9] W. Zhang and T. Li, "Characterizing and mitigating the impact of process variations on phase change based memory systems," in *MICRO*, 2009.
[10] B. Gleixner, F. Pellizzer, and R. Bez, "Reliability characterization of phase change memory," in *the 10th Annual Non-Volatile Memory Technology Symposium (NVMTS)*, 2009, pp. 7–11.
[11] K. Kim and S. J. Ahn, "Reliability investigations for manufacturable high density pram," in *IEEE Intl. Reliability Physics Symp.*, 2005.
[12] D. Im, J. Lee, and S. Cho, "A unified 7.5nm dash-type confined cell for high performance pram device," in *IEEE IEDM*, 2009, pp. 1–4.
[13] Numonyx Omneo P5Q PCM. http://www.numonyx.com/.
[14] The International Technology Roadmap for Semiconductors Report 2009. http://www.itrs.net/
[15] A. Cabrini, S. Braga, A. Manetto, and G. Torelli, "Voltage-driven multilevel programming in phase change memories," in *IEEE International Workshop on Memory Technology, Design, and Testing*, 2009, pp. 3–6.
[16] M. Qureshi, M. Franceschini, and L. Lastras-Montano, "Improving read performance of phase change memories via write cancellation and write pausing," in *HPCA*, 2010, pp. 1–11.
[17] D. Mantegazza, *et al.*, "Explanation of programming distributions in phase-change memory arrays based on crystallization time statistics," *Solid-State Electronics*, (52)4:584–590, 2008.
[18] E. Yeo, L. Shi, R. Zhao, K. Lim, T. Chong, and I. Adesida, "Parasitic capacitance effect on programming performance of phase change random access memory devices," *Applied Physics Letters*, vol. 96, 2010.
[19] A. Lacaita, "Phase change memories: State-of-the-art, challenges and perspectives," *Solid-State Electronics*, (50)1:24–31, 2006.
[20] N. H. Seong, D. H. Woo, V. Srinivasan, J. A. Rivers, and H.-H. S. Lee, "Safer: Stuck-at-fault error recovery for memories," in *MICRO*, 2010.
[21] B. Rajendran, J. Karidis, M.-H. Lee, M. Breitwisch, G. Burr, Y.-H. Shih, R. Cheek, A. Schrott, H.-L. Lung, and C. Lam, "Analytical model for reset operation of phase change memory," in *IEEE IEDM*, 2008, pp. 1–4.
[22] T. Tanzawa and T. Tanaka, "A dynamic analysis of the dickson charge pump circuit," *IEEE Journal of Solid-State Circuits*, vol. 32, no. 8, pp. 1231–1240, 1997.
[23] S. Gupta, V. Saxena, K. Campbell, and R. Baker, "W-2w current steering dac for programming phase change memory," in *IEEE Workshop on Microelectronics and Electron Devices*, 2009, pp. 1–4.
[24] D. Somasekhar, B. Srinivasan, G. Pandya, F. Hamzaoglu, M. Khellah, T. Karnik, and K. Zhang, "Multi-phase 1 ghz voltage doubler charge pump in 32 nm logic process," *Solid-State Circuits, IEEE Journal of*, vol. 45, no. 4, pp. 751–758, 2010.