Wear-out Analysis of Error Correction Techniques in Phase-Change Memory

Caio Hoffman, Luiz Ramos, Rodolfo Azevedo, and Guido Araújo
Institute of Computing, University of Campinas
{caio.hoffman, luiz.ramos, rodolfo, guido}@ic.unicamp.br

***Abstract*—Phase-Change Memory (PCM) is new memory technology and a possible replacement for DRAM, whose scaling limitations require new lithography technologies. Despite being promising, PCM has limited endurance (its cells withstand roughly 10^8 bit-flips before failing), which prompted the adoption of Error Correction Techniques (ECTs). However, previous lifetime analyses of ECTs did not consider the difference between the bit-flip frequencies of data and code bits, which may lead to inaccurate wear-out analyses for the ECTs. In this work, we improve the wear-out analysis of PCM by modeling and analyzing the bit-flip probabilities of five ECTs. Our models also enable an accurate estimation of energy consumption and analysis of the endurance-energy trade-off for each ECT.**

I. Introduction

Phase-Change Memory (PCM) is a new memory technology considered a possible replacement for DRAM. PCM is byte-addressable, non-volatile, has multi-level cell (MLC) capability, and has demonstrated scalability in the manufacturing process. A concern about PCM that hinders its use as main memory is its low endurance in number of memory writes.

Currently, a PCM cell can withstand roughly 10^8 bit-flips (modification of stored bit values) [1] before failing, which falls short of DRAM's virtually unlimited endurance [2]. In fact, due to the variability in the manufacturing process, PCM cells may withstand even less than 10^8 bit-flips. Thus, in a baseline memory system that is unable to handle cell failures, a single cell failure may invalidate an entire PCM chip.

To extend the average lifetime of PCM chips, Error Correction Techniques (ECTs) have been used to ensure that a few cell failures will not cause a chip failure. Recently, some ECTs for PCM were proposed in the literature: DRM [3], ECP [2], SAFER [4], and FREE-p [5].

However, the works that propose those ECTs have simplifying assumptions that overlook important characteristics of memory writes. First, they assume a **Bit-Flip Probability** (BFP) of 50%, i.e., on a write to a PCM memory block, every bit flips with a 50% probability. Second, they assume that the bit-flip behavior in data and code bits (which can correct errors in data bits) is identical. We show that those assumptions can mislead the evaluation of PCM's lifetime and do not enable an accurate analysis of energy consumption. Moreover, Schechter *et al.* argue that Error-Correcting Codes (ECCs) speed up cell wear-out because any modification to the data bits requires a rewrite of the code bits [2]. Nonetheless, the authors do not confirm their hypothesis with experimental results or theoretical analyses.

In this work, we carefully evaluate those hypotheses by mathematically modeling the bit-flip behavior in DRM, ECP, SAFER, FREE-p, and also in SECDED. Specifically, our models consider that data and code bits have different bit-flip frequencies, which enables a more accurate evaluation of PCM's endurance, and provides the means for computing energy consumption, for each ECT. This kind of analysis and comparison had not been done in the original ECT research. In our study, for each ECT, we evaluate the corresponding PCM wear-out, the energy consumption, and the energy-endurance trade-off. We compare our results to those in the literature, and find that they support Schechter *et al.*'s hypothesis.

II. Background and Related Work

We briefly describe recent ECTs proposed for PCM (DRM, ECP, SAFER, and FREE-p) and an ECC that is traditionally used in DRAM (SECDED) [2], [3], [5]. In conformity with the literature, for all ECTs, we adopt a memory block composed of one 512-bit data block plus an overhead of up to 64 code bits (error detection/correction meta data), which corresponds to 12.5% of the data block size.

DRM uses one parity bit per byte to detect up to one error in that byte. Thus, the DRM memory block has 64 parity bits of overhead. **SECDED** uses blocks of 72 bits containing a non-standard (71,64) Hamming code plus 1 parity bit. A memory block fits 8 SECDED blocks, which allows to correct up to 8 errors in the memory block (at most one per SECDED block). **ECP** uses pointers to redirect writes in failed data cells to brand new spare cells. ECP_e corrects up to e errors, using e ECP entries (i.e., spare cells and pointers). The ECP_e overhead is composed of the e entries plus 1 *full bit* to indicate if all pointers are in use. ECP_6 has the largest e that respects our 12.5% overhead limit. **SAFER** uses the failed cells themselves to correctly store the data. It isolates each failed cell in a unique group of bits that are stored inverted or not, depending on a logical matching between failed cells and bits to be stored. SAFER_k can correct up to k errors, and $k = 32$ is the highest value of k that respects the 12.5% overhead. Nonetheless, SAFER_{32} may only be able to correct 7 errors, depending on the position of failed cells in the memory block. **FREE-p** uses a 6EC-7ED ECC that is a (572, 512) BCH code to correct and detect up to 4 hard errors and up to 2 soft errors, plus an extra parity bit that covers the 572 BCH bits.

978-3-9815370-2-4/DATE14/©2014 EDAA
We thank CNPq, CAPES, and FAPESP (2011/05266-3, 2013/05257-0).

III. Simulation Model

Five hypotheses guides our model. **First**, PCM cells may not have the same lifetime. This comes from the known imperfection of any process fabrication. **Second**, writes only modify bits that differ from the bits already stored, using differential writes [6] or read-before-write [5]. **Third**, each cell stores a single bit. **Fourth**, the memory is perfectly wear-leveled. That is, the wear-out is spread over all memory blocks, without hotspots. **Fifth**, there is an intra-row wear leveling (as in the fourth hypothesis, but within every memory block).

A. Mathematical Representation

In our model, we represent memory as a unidimensional array Q of length C_M, where each element contains an integer that represents the remaining lifetime of a memory cell. We define $C_M = N \cdot N_L \cdot N_P$, where N is the data block length, N_L is the number of blocks in a page, and N_P is the number of pages. We initially create Q by ascendingly sorting the elements from an integer matrix that was randomly generated, based on Gaussian distribution with mean μ and standard deviation σ (to account for the effect of process variation, as described in our first hypothesis).

Our simulation handles one element of Q per simulation step. Being that, at the k-th step, all memory cells must have suffered Q_{k-1} writes, except for the $k-1$ memory cells that had already failed; since the elements $Q_0, \ldots, Q_{k-1}$ have a lower remaining lifetime than the elements $Q_k, \ldots, Q_{C_M-1}$.

When the first block in a page becomes unable to recover from a failure, the page is deallocated; and if all pages are deallocated, the simulation stops. Deallocated pages should not be taken into account, for both, let $F_P(k)$ be the number of deallocated pages and let $f_{wr}(k)$ be the wear rate (the percentile reduction of the number of writes) in the k-th simulation step. Also let c_k be the lifetime difference between the elements located in positions k and $k-1$. Given $c_0 = 0$, $f_{wr}(0) = 1$, and $F_P(0) = 0$ before the simulation starts, we can describe a simultaneous write in all memory pages as (1).

$$\begin{aligned} f_{wr}(k) &= f_{wr}(k-1) \cdot \tfrac{N_P - F_P(k)}{N_P - F_P(k-1)} \\ W(k) &= \textstyle\sum_{i=1}^{k} c_i \cdot f_{wr}(i) \end{aligned} \tag{1}$$

The perfect wear-leveling ensures that when we perform c_k writes, all the blocks that do not belong to failed pages will have suffered c_k writes (fourth hypothesis). The wear rate removes unwritable pages from the counting. Finally, N_L and N_P are constants and can be computed off-line, thus (1) is enough for our evaluations.

B. Bit-Flip Probability Modeling

In a memory that stores one bit per cell (third hypothesis), we can model a write in a memory block with a binomial distribution, or a sequence of n (*number of bits*) independent events, with probability p of success (*flip*). The expected number of flipping bits in a write is $n.p$. Thus, a PCM cell should last longer than its original lifetime by a fraction $1/p$, considering the second and the fifth hypothesis.

For the following BFP models, we consider that the probability of success p is unknown. Every memory block has $N + \Delta_X$ bits (X denotes a ECT). Of those bits, $N = 512$ bits have p probability of flipping (data BFP) and Δ_X bits have $P_X(p)$ probability of flipping (code BFP, which is a function p). Thus, we define the **weighted BFP** Υ_X as the weighted arithmetic mean of p and $P_X(p)$. Equation (2) uses Υ_X to compute the total writes in our model for each ECT.

$$\mathbb{W}(k) = \tfrac{W(k)}{\Upsilon_X} \tag{2}$$

1) The DRM BFP model: starts by considering that one parity bit covers n bits. We know that a parity bit is 1 when the covered bits have an odd number of bits with value 1. If the number of covered bits that flip is even, the parity remains unchanged due to compensation. Hence, the BFP of the parity bit depends on the probability of an odd number of flips occurring in n bits. Assuming that the BFP in a memory follows a binomial distribution, the probability of parity change is the probability of occurring 1, 3, 5, $\ldots$, $n-1$ flips. Mathematically,

$$P_{\text{PARITY}}(n,p) = \textstyle\sum_{i=1}^{\frac{n}{2}} B(2 \cdot i - 1; n, p) \tag{3}$$

The notation B(k;n,p) represents the probability mass function for the binomial distribution. In DRM, we have $\Delta = 64$ parity bits with probability $P_{\text{PARITY}}(b,p)$ of flipping, where b is the length of a byte, and N bits with probability p of flipping. The DRM's BFP is:

$$\Upsilon_{\text{DRM}} = p \cdot \tfrac{N}{N+\Delta} + P_{\text{PARITY}}(b,p) \cdot \tfrac{\Delta}{N+\Delta} \tag{4}$$

2) The SECDED BFP model: as in DRM BFP model, the parity in SECDED changes in the presence of an odd number of bit flips in a memory write. However, the model of SECDED is more complex because, in a non-standard Hamming Code, the quantity $q_p(\cdot)$ of bits covered by a parity bit varies. To compute $q_p(j)$ for the i-th parity bit (lying on position $j = 2^{i-1}$ of the codeword) we used an equation described in [7]. Using the $q_p(\cdot)$ for each parity bit, we describe the weighted BFP for (n,k) Hamming code in (5) and for SECDED in (6).

$$P_{\text{HC}}(p) = \tfrac{1}{n} \cdot \left(k \cdot p + \textstyle\sum_{i=1}^{n-k} P_{\text{PARITY}}(q_p(2^{i-1}), p)\right) \tag{5}$$

$$\Upsilon_{\text{SECDED}} = \tfrac{1}{n+1} \cdot \left(n \cdot P_{\text{HC}}(p) + P_{\text{PARITY}}(n, P_{\text{HC}}(p))\right) \tag{6}$$

3) The ECP BFP model: is the simplest one. Essentially, neither the pointer bits nor the full bit will be frequently written. In fact, even the last pointer (the one that maintains the number of active ECP entries) will be updated at most e times in the ECP$_e$, which is negligible compared to the millions of writes that PCM can endure. Thus, we can consider that those bits have a negligible BFP. The N data bits and e spare bits suffer the vast majority of writes. Assuming that e errors will occur and considering our fifth hypothesis. Let $\Delta = e + e \cdot \log_2 N + 1$, thus:

$$\Upsilon_{\text{ECP}_e} = p \cdot \tfrac{N+e}{N+\Delta} \tag{7}$$

4) The SAFER BFP model: is the only model we propose that takes the number of failed bits into account. We do that because, in SAFER, the number of failed bits in a memory block changes the BFP of the data bits. Specifically, in the absence of failures, each bit at a given memory block has a BFP of p upon every new write to the block. However, when a failure occurs, the block is sub-divided into groups of bits, with at most one failed bit (stuck at one or zero) per group. Suppose we have a group G with a failed bit g_i, and G' will overwrite G upon a new block write. Suppose that the bit g'_i in G' will overwrite g_i. Then, consider the probability $q = 1/2$ that the stuck value of g_i matches the value of g'_i. If they match, G' overwrites G with BFP p, and the resulting BFP is $p \cdot q$. If they do not match, SAFER will invert all bits of G' before overwriting G, thus the resulting BFP is $(1-p)\cdot(1-q)$.

In SAFER$_k$ (which can correct up to k errors) there are initially k groups, and they require k spare bits to inform whether the data is stored directly or not. These bits have a BFP of either p or $(1-q)$, depending on if there is a failed cell in the group they are related. Furthermore, SAFER needs bits to organize the groups and keep them with only one error. Like the pointers in ECP these bits are written very few times; thus, we assume their BFP is zero. For SAFER, Δ is $k + \lceil \log_2 k \rceil \cdot \lceil \log_2 \lceil \log_2 N \rceil \rceil + \lceil \log_2 (\lceil \log_2 k \rceil + 1) \rceil$ bits.

Being k and N powers of 2, the number of bits in a group is N/k. Therefore, with e errors in a memory block, e groups of (N/k) bits have a BFP of either $p \cdot q$ or $(1-p)\cdot(1-q)$, which results in an overall BFP of $(N + (k-e) - (\frac{N}{k} \cdot e)) \cdot p + (\frac{N}{k} \cdot e) \cdot q \cdot p + (\frac{N}{k} \cdot e) \cdot (1-q) \cdot (1-p) + e \cdot (1-q)$. We simplify that expression into (8).

$$\Upsilon_{\text{SAFER}}(e) = \frac{(N+k-e)\cdot p + \frac{N}{k}\cdot e \cdot \left(\frac{1}{2} - p\right) + e \cdot \frac{1}{2}}{N+\Delta} \tag{8}$$

Our simulation model cannot handle this bit-flip model, since every memory block has its own number of errors. So we resort to a simplification. Specifically, we calculate the average number of errors $e_\mu(i)$ across all memory blocks at the $i-th$ simulation step. Then, we computer the number of writes at each simulation step using $\Upsilon_{\text{SAFER}}(e_\mu(i))$, which effectively replaces (2) with (9).

$$\mathbb{W}(k) = \sum_{i=1}^{k} \frac{c_i \cdot f_{wr}(i)}{\Upsilon_{\text{SAFER}}(e_\mu(i))} \tag{9}$$

5) The FREE-p BFP model: is based on the binary division of the data word by a word called generator polynomial. For a (n,k) BCH code, we define $D = \{d_0, d_1, \ldots, d_{k-1}\}$ as data word with k bits, $G = \{g_0, g_1, \ldots, g_m\}$ as the generator polynomial, where $m = n - k$. Finally, we define $V = \{v_0, v_1, \ldots, v_{m-1}\}$ as the verification code.

To illustrate our model, we take a generic (7,4) BCH code showed in the Fig. 1. V results from additions modulo 2 of elements of $L = \{\mathfrak{L}_1, \mathfrak{L}_2, \mathfrak{L}_3, \mathfrak{L}_4\}$. Formally, V is an element of $\mathcal{P}(L)$ over the application of $f_\oplus$ (defined in [7]), that is, an element belonging to $\mathcal{L} = \bigcup_{P \in \mathcal{P}(L)} f_\oplus(P) = \{0, \mathfrak{L}_1, \mathfrak{L}_2, \mathfrak{L}_3, \mathfrak{L}_4, \mathfrak{L}_1 \oplus \mathfrak{L}_2, \mathfrak{L}_1 \oplus \mathfrak{L}_3, \ldots, \mathfrak{L}_1 \oplus \mathfrak{L}_2 \oplus \mathfrak{L}_3 \oplus \mathfrak{L}_4\}$. For example, suppose that some division is represented by $\mathfrak{L}_2 \oplus \mathfrak{L}_3$, thus $V = \{0, g_0, g_0 \oplus g_1\}$.

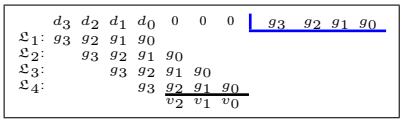

Figure 1. Calculation of a generic verification code for the (7,4) BCH code.

We know that $|\mathcal{L}| = 16$ and, for example, all the possible values for v_2 are $\{0, g_0, g_1, g_2, g_0 \oplus g_1, g_0 \oplus g_2, g_1 \oplus g_2, g_0 \oplus g_1 \oplus g_2\}$. However, note that there are 16 possible values of v_2, from whose only 8 are linearly independent values, since every combination with $\mathfrak{L}_1$ results in a combination that already exists.

Now suppose that there is a data word D and its code V stored in a memory block. If a new data word D' and its code V' will be stored, the probability of occurring $d_3 = d'_3$, $d_2 = d'_2$, $d_1 = d'_1$ and $d_0 = d'_0$ (i.e., no bit flips) is $(1-p)^4$.

Consider the new code bit v'_2. There are 16 possible results among which exactly half shall be 1 and the other half 0 (details in [7]). Regardless of v'_2 being 1 or 0, when v'_2 overwrites v_2, there would be a flip in 8 out of 16 chances. Disregarding the probability of $D = D'$, there is 8/15 chances of occurring a BFP of $1-(1-p)^4$, for every code bit.

Generalizing for a data word $D = \{d_0, d_1, \ldots, d_{k-1}\}$ and its code $V = \{v_0, v_1, \ldots, v_{m-1}\}$ stored in a memory block, the new data word D' and its code V' to be stored will incur a BFP of $1-(1-p)^k$ for every verification code bit, which may happen with probability $2^{k-1}/(2^k-1)$. Therefore,

$$P_{\text{code}}(p,k) = (1-(1-p)^k) \cdot \frac{2^{(k-1)}}{2^k - 1} \tag{10}$$

In the (n,k) BCH code of FREE-p, the memory parameters are $k = N$ and $n - k = \Delta - 1$. Δ is the sum of the code bits from BCH and the parity bit. The BFP of (n,k) BCH code is given by (11) and for FREE-p is given by (12).

$$P_{\text{BCH}}(p) = \frac{1}{n} \cdot \left(k \cdot p + (n-k) \cdot P_{\text{code}}(p,k)\right) \tag{11}$$

$$\Upsilon_{\text{FREE-p}} = \frac{1}{n+1} \cdot \left(P_{\text{PARITY}}(n, P_{\text{BCH}}(p)) + n \cdot P_{\text{BCH}}(p)\right) \tag{12}$$

IV. Results and Discussion

A. Weighted BFP

Table I shows the weighted BFPs (Υ_X) for every technique (X) computed using our models for each value of p (the BFP of data bits). The higher the value of Υ_X, the worst the technique performs in terms of PCM wear-out. For SAFER, because its BFP depends on the number of error, Υ is the average BFP across 1000 simulations.

We observe that when p is below 50%, the ECC-based techniques (DRM, SECDEC, and FREE-p) have a higher Υ_X than the other techniques. The reason is that, in ECC-based techniques, the BFP of code bits is higher than that of data bits, which accelerates the wear-out. For $p \leq 30\%$, SAFER performs as poorly as the ECC-based techniques. This happens because SAFER's bit-inversion mechanism increases the BFP of bit groups in the presence of errors. The only technique that shows a consistently low weighted BFP is ECP, which benefits from infrequent updates to its code bits. For $p \geq 50\%$, all techniques exhibit a higher BFP in data bits than in code bits, which lowers Υ.

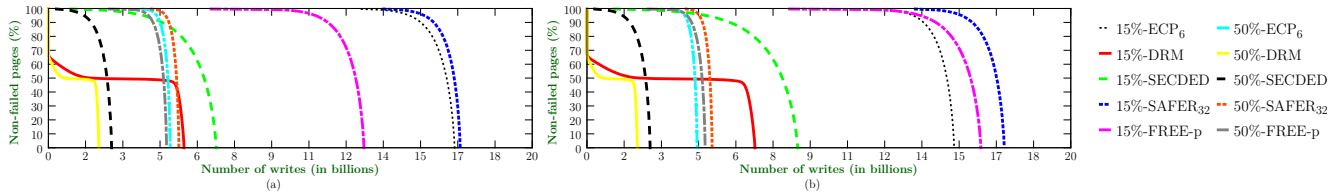

Figure 2. Memory lifetime with our BFP models for (a); and without our BFP models for (b).

Table I
WEIGHTED BFPS (Υ_X) FOR EACH TECHNIQUE (X) CONSIDERING DIFFERENT BFPS OF DATA BITS (p).

p	10%	15%	20%	30%	40%	50%	60%	70%	80%	90%	100%
Υ_{ECP_6}	0.09	0.14	0.18	0.27	0.36	0.45	0.54	0.63	0.72	0.81	0.90
Υ_{DRM}	0.14	0.19	0.23	0.31	0.41	0.50	0.59	0.68	0.77	0.85	0.89
Υ_{SECDED}	0.14	0.19	0.23	0.32	0.41	0.50	0.59	0.68	0.77	0.86	0.90
$\mu(\Upsilon_{\text{SAFER}_{32}})$	0.13	0.17	0.22	0.30	0.39	0.48	0.57	0.66	0.75	0.83	0.92
$\Upsilon_{\text{FREE-p}}$	0.14	0.19	0.23	0.32	0.41	0.50	0.59	0.68	0.77	0.86	0.95

B. Empirical and Theoretical BFP

For our wear-out analysis of PCM, we feed two parameters into our BFP models: an **empirical bit-flip rate** ($p = 15\%$) and the theoretical BFP assumed in the literature ($p = 50\%$). We obtained $p = 15\%$ by instrumenting SPEC CPU2006 execution in our cache memory simulator (for details, see [7]). Our experimental BFP corroborates with the value in [6].

Using the empirical and theoretical parameters, we study the impact of ECTs on memory lifetime in two ways. First, we simulate the different ECTs using our BFP models (Section III), which differentiate between the BFPs of data and code bits. Fig. 2 (a) shows these results. Second, we study the ECTs without differentiating between the BFPs of data and code bits as it was done in previous works (Fig. 2 (b)). In both studies, we run 1000 simulations of each ECT to account for the randomized lifetimes of PCM cells (Section III-A). We also set $N = 512$, $N_L = 64$, $N_P = 256$, the average cell endurance $\mu = 10^8$ with a standard deviation $\sigma = 2.5{\cdot}10^7$. The curves in all figures represent memory lifetime, as the number of non-failed pages, versus the number of writes to PCM. The memory lifetime ends when reaching zero non-failed pages. The curves with 50% in Fig 2 (b) represent the results as they were originally published in [2], [4], [5]. Although in those works there was no direct comparison between FREE-p and SAFER, both had been found superior to ECP. The results are similar for the BFP of 15% in Fig. 2 (b).

FREE-p, in Fig. 2 (a), exhibits degradation compared to the previous results, performing worse than ECP, which contradicts the previous result. In addition to that, ECP improves its memory lifetime, whereas SAFER maintains approximately the same lifetime. These results highlight our claim that differentiating between the bit-flip rates of data and code bits enables more fine-grained and accurate studies. They also provide support for the hypothesis by Schechter *et al.* [2].

C. Energy Consumption and Bit-Flip Probability

Since each ECT has a unique BFP for every value of p, their expected write energy should be different. To study the write energy we adopt the same cell write energy for all ECT, i.e., $E_{reset} = 19.2$ pJ and $E_{set} = 13.5$ pJ (from Lee *et al.* [1]). To evaluate the trade-off between endurance and energy consumption, we combine them into a single metric in (13).

Table II
Λ OF THE ECTS NORMALIZED TO Λ_{ECP_6}. OUR EMPIRICAL BIT-FLIP RATE IS $p = 15\%$ AND THE THEORETICAL BFP IS $p = 50\%$.

$\Lambda_X / \Lambda_{\text{ECP}_6}$	Υ_{DRM}	Υ_{SECDED}	$\mu(\Upsilon_{\text{SAFER}_{32}})$	$\Upsilon_{\text{FREE-p}}$
$p = 15\%$	0.24	0.30	0.81	0.56
$p = 50\%$	0.37	0.47	1.02	0.88

$$\Lambda_X = \frac{\mathbb{W}_X(\omega)}{(N+\Delta_X)\cdot\Upsilon_X\cdot(E_{set}+E_{reset})\cdot 0.5} \quad (13)$$

The denominator is the average number of bit-flips in a write $(N + \Delta_X) \cdot \Upsilon_X$ multiplied by the average write energy per bit. The numerator $\mathbb{W}(\omega)$ is the total number of writes that the memory withstood until the last simulation step denotated by ω. In Table II, we have the Λ of all ECTs normalized to Λ_{ECP_6}. From the definition of Λ, the higher the memory endurance and the lower the write energy imposed by an ECT, the better the ECT's Λ. Based on this table, we conclude that ECP and SAFER exhibit the best trade-off between endurance and write energy, for both empirical bit-flip rate and theoretical BFP.

V. CONCLUSIONS

In this work, we introduced a more accurate and fine-grained approach to analyze the impact of ECTs on PCM's lifetime. Our approach also enables a meaningful computation of PCM's write energy and its trade-off with memory endurance. This kind of analysis was not possible with previous ECT models, but it is still critical for modern computer systems.

We evaluated five state-of-the-art ECTs using our approach and compared our results to those in the literature. Our results extend and shed light on previous works that used simplifying assumptions; and support the argument that ECC-based techniques speed up the wear-out of PCM.

REFERENCES

[1] B. C. Lee, E. Ipek, O. Mutlu, and D. Burger, "Architecting phase change memory as a scalable dram alternative," ser. ISCA '09, pp. 2–13.

[2] S. Schechter, G. H. Loh, K. Straus, and D. Burger, "Use ecp, not ecc, for hard failures in resistive memories," ser. ISCA '10, pp. 141–152.

[3] E. Ipek, J. Condit, E. B. Nightingale, D. Burger, and T. Moscibroda, "Dynamically replicated memory: building reliable systems from nanoscale resistive memories," in *ASPLOS '10*, 2010.

[4] 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 '43*. IEEE Computer Society, 2010, pp. 115–124.

[5] D. H. Yoon, N. Muralimanohar, J. Chang, P. Ranganathan, N. Jouppi, and M. Erez, "Free-p: Protecting non-volatile memory against both hard and soft errors," in *HPCA 2011*, feb. 2011, pp. 466–477.

[6] L. Jiang, B. Zhao, Y. Zhang, J. Yang, and B. Childers, "Improving write operations in mlc phase change memory," in *HPCA*, 2012, pp. 1–10.

[7] C. Hoffman, L. Ramos, R. Azevedo, and G. Araújo, "Improving the Modeling and Analysis of Error Correction Techniques for Phase-Change Memory," Unicamp, Tech. Rep. IC-13-34, December 2013, in English.