A Holistic Tri-region MLC STT-RAM Design with Combined Performance, Energy, and Reliability Optimizations

Wujie Wen †, Mengjie Mao ‡, Hai Li ‡, Yiran Chen ‡, Yukui Pei ∗, Ning Ge ∗
† Florida International University, Miami, FL 33174, USA
‡ University of Pittsburgh, Pittsburgh, PA 15261, USA
∗ Tsinghua University, Beijing 100084, CHINA
†wwen@fiu.edu, ‡{mem231,hal66,yic52}@pitt.edu, ∗{peiyk,gening}@tsinghua.edu.cn

***Abstract*—Multi-level cell spin-transfer torque random access memory (MLC STT-RAM) demonstrates great potentials in on-chip cache design for its high storage density and non-volatility but also suffers from the degraded access time, reliability and energy efficiency. The existing MLC STT-RAM cache designs primarily focus on the performance and energy optimizations, however, often ignore the crucial demand for reliability. In this work, we propose a tri-region MLC STT-RAM cache design (TMSC) to simultaneously meet the requirements of performance, energy, and reliability. The tri-region MLC STT-RAM cache is optimized partitioned into fast, mixed, and slow ways according to different access performance, energy and reliability. A new error correction code (ECC) scheme, namely, non-uniform strength ECC (NUS-ECC), is also developed to tolerate the different bit failure rates in these ways. Compared to the latest performance-driven MLC STT-RAM cache design with pessimistic ECC scheme, our TMSC technique can improve the system performance and energy by averagely 9.3% and 9.4%, respectively, for various applications. The additional area cost associated with NUS-ECC is limited by 3.2% compared to the pessimistic ECC scheme.**

I. Introduction

Exponential growth of data processing capability of modern computer systems produces sharply-increasing demand on memory capacity. As conventional memories, e.g, SRAM and DRAM, are facing severe scaling challenges in manufacturing and designs, tremendous resources have been invested in some emerging memory technologies. In particulary, *Spin-transfer torque random access memory* (STT-RAM) has demonstrated great potentials in on-chip cache [1] and main memory applications [2], offering nanosecond access time comparable to SRAM, high integration density close to DRAM, and zero standby power like Flash memory.

Very recently, multi-level cell (MLC) technique, which allows for two digital bits being stored in one storage element, is also introduced to STT-RAM designs [3], [4] for storage density boosting. However, the performance degradation of MLC STT-RAM has been identified as one of the most crucial problems. Both read and write operations of MLC STT-RAM are more costly than single-level cell (SLC) design due to the unique two-step access procedure: the logic detection in a read operation requires two sensing stages and two bits of a MLC may need to be programmed separately. Hence, the prolonged access latency of such native MLC STT-RAM cache may offset the advantages of the increased storage capacity. Extensive works are conducted to mitigate the drawback of two-step access [5], [3]. All these performance-driven designs, named as *cell-split-mapping* (CSM) here, logically divide a MLC into soft-bit and hard-bit, and assemble an entire cache line with either soft-bits or hard-bits, respectively. The cache lines composed of only soft-bits (namely, soft-lines), can operate at a faster read/write speed and lower energy cost than those composed of hard-bits (namely, hard-lines). By allocating majority of the cache accesses to the soft-lines, CSM is able to improve the cache access performance.

CSM leverages the *asymmetric access latency* of soft and hard bit to greatly enhance performance, however, it incurs several disadvantages ignored before. First, CSM fails to consider *the asymmetric reliability* property of those two bits, i.e. the average error rate of hard-bits can be up to five orders of magnitude higher than that of soft-bits. As we shall show later, the reliability of hard-lines in CSM degrades severely from the level that the cache lines composed of mixed soft-bits and hard-bits (namely, mixed-lines) in native MLC can offer. Tolerating increased multi-bit errors in hard-lines, with the traditional long-latency error correction codes (ECCs) like Bose-Chaudhuri-Hocquenghem (BCH) code [6], will totally offset the performance advantage of CSM; Second, CSM increases the average programming delay and energy over all cache lines due to additional read and restore operations in soft-lines, jeopardizing the performance improvement at system-level.

In this work, we propose a holistic design–*tri-region MLC STT-RAM caches* (TMSC) to simultaneously satisfy the requirements of performance, energy, and reliability (PER). To the best of our knowledge, this is the first work quantitatively revealing the reliability concern of the widely adopted CSM design. Three types of cache lines – soft-lines, hard-lines, and mixed-lines are constructed based on the performance needs. The data, hence, are managed to be stored at different types of cache lines in the corresponding TMSC design based on their different PER requirements. A novel ECC scheme, namely, *non-uniform strength ECC* (NUS-ECC), is proposed to protect the data integrity of different types of cache lines by taking into account their reliability variance. Our simulation results show that TMSC can significantly improve system performance and energy while still offering the same level reliability of the conventional performance-driven CSM design with pessimistic ECC scheme under very low cost.

II. Background and Motivation

A. SLC and MLC STT-RAM

An STT-RAM cell uses the different resistance states of a magnetic tunneling junction (MTJ) to store data. As shown in Figure 1(a), an MTJ is composed of two magnetic layers (i.e., reference layer and free layer) and an oxide insulator. Figure 1(b) illustrates the popular 1T1J structure of a SLC STT-RAM, in which the MTJ is selected through a NMOS transistor. The magnetization direction of the reference layer is fixed while

978-3-9815370-6-2/DATE16/

that of the free layer is changeable [7]. Applying a current from BL to SL (SL to BL) will flip the magnetization direction of the free layer to be parallel (anti-parallel) to that of the reference layer, resulting in a low (high) resistance or logic '0' ('1').

The 1T1J structure can be extended to MLC STT-RAM designs which stack two MTJs in serial [8], as shown in Figure 1(c). The two MTJs, which have different sizes, can construct four resistance levels to represent a 2-bit data. Figure 1(d) shows that sensing a MLC STT-RAM cell requires three references. Due to the difference in the required amplitudes of the programming current to these two MTJs, a two-step write scheme is usually needed. Figure 1(e) shows the different resistance state transitions of an MLC cell, including both one-step and two-step transitions. Based on the programming difficulty of the two data bits in a MLC, we name the first bit that is decided by the resistance state of the small MTJ as *soft-bit* while the second bit that is decided by the resistance state of the large MTJ as *hard-bit*. In general, the soft-bit has a higher resistance but a lower switching current I_c than the hard-bit.

B. Cell-split-mapping MLC STT-RAM

Figure 2(a) shows conventional 2-bit MLC design storing a data block of *N* bits in *N/2* MLC cells. Half of the data (i.e., $A_0A_2...A_{N-2}$ of Data Block *0* in Figure 2) are stored in soft-bits while the other half ($A_1A_3...A_{N-1}$) are saved in hard-bits. We name such a storage structure as native MLC STT-RAM design. When accessing a data block containing both soft- and hard-bit (mixed-line), the costly two-step read or write in Figure 1(d,e) are always needed. Hence, many prior works propose CSM MLC STT-RAM design [3], [5]. Figure 2(b) illustrates a CSM design that maps two data blocks ($A_0A_1...A_{N-1}$ and $B_0B_1...B_{N-1}$) into the entire soft-bits (soft-line) and hard-bits (hard-line) of *N* MLC cells, respectively. As such, only one-step read and write are needed when accessing the soft-line. Note that two-step read and write are still required when accessing the hard-line. CSM logically divides the cache into two parts – the soft- and the hard-lines with different access speeds. Since the access to the soft-lines is faster than the hard-lines, many studies focused on managing the data allocation to the soft-lines as much as possible to enhance the overall performance of CSM MLC STT-RAM cache [3], [5].

C. Drawbacks of CSM MLC STT-RAM

Reliability Concern: CSM STT-RAM is motivated by the asymmetry in the access speed of soft-bits and hard-bits. However, high asymmetry in the write reliability of these two bits is often neglected in the relevant works. There are two kinds of write failures in MLC STT-RAM: *incomplete write* and *overwrite*. Incomplete write happens when the programming current is removed before the MTJ switching process completes [9],

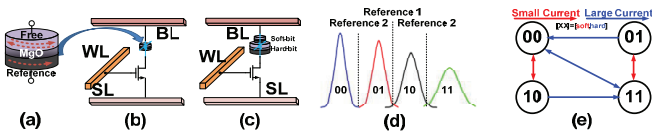


Fig. 1: STT-RAM technology. (a) MTJ; (b) SLC STT-RAM cell; (c) MLC STT-RAM cell; (d) Two-step read of MLC STT-RAM; (e) Two-step write of MLC STT-RAM.

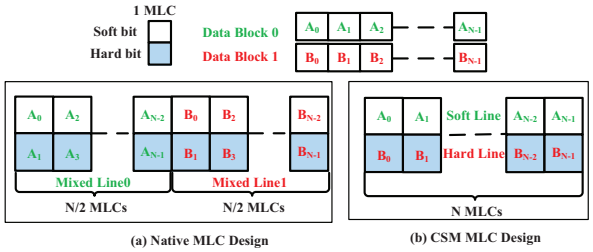


Fig. 2: Illustration of native MLC vs. CSM MLC design.

[10], [11]. Raising the amplitude of the programming current can effectively reduce the MTJ switching time and hence, suppress the error rates associated with incomplete write for a fixed write pulse width. Note that incomplete write can occur at both soft-bits and hard-bits. On the contrary, overwrite happens only in a soft transition when the amplitude of programming current is large enough to accidentally flip the resistance state of the big MTJ. As discussed in section II-A, hard-bit flipping requires a larger switching current and hence, often suffers from a higher incomplete write error rate than soft-bit flipping. Considering overwrite happens only in soft-bit flipping, the overall write error rate of hard-bits is much higher than that of soft-bits in MLC STT-RAM designs.

We adopt the MLC design from [8] with similar parameter configurations as [3] at 32nm technology node. Figure 3(a) demonstrates the high bit error rate asymmetry of the two bits under all write pulse widths. For example, the asymmetric ratio of the error rates of those two bits could reach five orders of magnitude around 10ns. Hence, the reliability of hard-line degrades severely from the level offered by the the mixed-line in native design. We assume the length of cache line is 64Byte and choose the 10ns as working condition for aforementioned STT-RAM designs. The bit error rates are adopted from Figure 3(a). Table I summarizes the calculated probabilities of 0-bit (failure free), 1-bit and 2-bit errors for the three types of cache lines. Compared to the mixed-line, the probabilities of 1-bit and 2-bit errors in the hard-line are increased by 71% and 244%, respectively. To ensure a target cache line failure rate below 1E-3 [12], single-bit ECCs (e.g., SEC-DED for every 64 bits sub-block) that can be applied to the mixed-line are no longer sufficient for the hard-line. BCH code with double-error correction capability could be a choice for the hard-line. However, the classical decoding scheme of BCH code is based on Berkeamo Massey algorithm and Chien search and often requires a long decoding latency [12], [6]. If these reliability-related design factors are considered, the performance advantage offered by the CSM MLC STT-RAM will be gone.

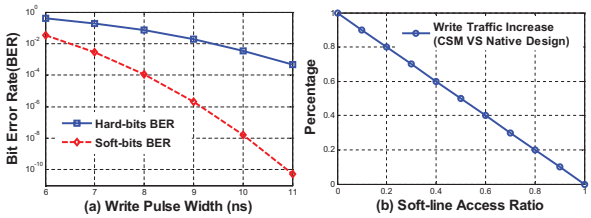


Fig. 3: Design motivations: (a) BER of soft/hard bit v.s. write time (b) Increased write traffic percentage of CSM v.s. access probability

TABLE I: Reliability comparison of mixed/hard/soft-line.

Design	Native MLC	CSM MLC	
64B Cache Line mode	Mixed-line	Hard-line	Soft-line
0 bit ER	8.57133E-01	7.34683E-01	9.99999E-01
1 bit ER	1.32177E-01	2.26583E-01	7.76799E-06
2 bit ER	1.01516E-02	3.48720E-02	2.94333E-11

Energy/Delay Concerns: CSM design also increases the dynamic energy consumption of STT-RAM accesses and aggravates the access traffic. Since programming a hard-bit is associated with a soft-bit switching to the same value, programming a hard-line in CSM design always disturbs the corresponding soft-line. To avoid losing data, we have to backup the entire soft-line before programming the hard-line and restore it after the hard-line programming completes. On the contrary, no data backup is required in mixed-line programming as all the bits are updated. Figure 3(b) shows an example of increased write traffic of CSM MLC normalized to the native MLC over different soft-line access ratios. The percentage grows quickly as the soft-lines access ratios decreases. Given the same access probabilities to both lines, the CSM design will introduce 50% more writes to the soft-lines compared to the native design, leading to more dynamic energy cost. Meanwhile, at least one read latency will be saved in native design compared to hard-line programming in CSM though both of the cases may involve two-step write.

In summary, compared to the hard-lines in CSM MLC, the mixed-lines in native MLC have advantages in reliability, energy and access latency.

III. Proposed TMSC Design

The different performance, energy and reliability (PER) tradeoffs of CSM and native MLC designs motivated us to propose a tri-region MLC STT-RAM cache (TMSC), which is composed of three types of cache lines simultaneously – soft-lines, hard-lines, and mixed-lines. Our TMSC design includes two techniques: 1) a novel nonuniform strength ECC (NUS-ECC) scheme that offers variable decoding delay adapting to the reliability requirement of different types of cache lines; 2) a subtle data migration policy to maximize the utilization of the soft-ways and also suppress the accesses to the hard-ways with optimal way partition.

A. Reliability–NUS-ECC

1) Hard-Lines Protection–HECC: Inspired by the low occurrence probability of 2-bit error in a 64B hard-line (e.g., $\sim 3.49\%$ (2-bit error) vs. $\sim 96.5\%$ (others) in Table I), we introduce a *hierarchical ECC* (HECC) design to particularly speedup the access latency of 0- or 1-bit error rather than the 2-bit error [13]. By redesigning its parity check matrix based on single-bit ECC–SEC-DED and multi-bit ECC–BCH, HECC can dynamically switch between two different modes: Mode 1 is a high-speed mode offering 0 and 1-bit error correcting capability. Its access latency and implementation cost are as low as SEC-DED; Mode 2 is dedicated to correct 2-bit error at cost of a longer latency (but still shorter than that of BCH after hardware optimization). Apparently, most accesses fit into mode 1. In the rare case that a 2-bit error happens, mode 2 will take place immediately. Hence, the average decoding latency of HECC during system execution is largely reduced from that of the BCH. HECC's storage overhead, however, is similar to the BCH.

Assume α is a primitive element in Galois Field GF(2^m), the parity check matrix for Hamming ECC could be expressed as:

$$H_h = \left[\alpha^0 \alpha \cdots \alpha^{n-1}\right]. \tag{1}$$

Here m is the number of redundancy bits. n and $k = n - m$ are the post-coding length and pre-coding length, respectively. Similarly, the double error correcting BCH code in GF(2^m) can be obtained by:

$$B^{\Delta} = \begin{bmatrix} H^{\Delta} \\ L \end{bmatrix} \begin{bmatrix} 0 & \alpha^1 \cdots & \alpha^{n-1+m} \\ \alpha^0 & \alpha^3 \cdots & \alpha^{3(n-1-m)} \end{bmatrix}. \tag{2}$$

Here $H^{\Delta} = \left[H_h | \alpha^n \cdots \alpha^{n-1+m}\right]$. The number of redundance bits can be doubled as $2m$ compared to that of Hamming code.

Based on Eq.(1) and (2), the parity check matrix B^* for HECC (pre-coding length 64) can be derived as:

$$B^* = \begin{bmatrix} H^* \\ L^* \end{bmatrix} = \left[\begin{array}{cccc|c} 0 & \alpha^0 & \cdots & \alpha^{70} & \mathbf{0} \\ 1 & 1 & \cdots & 1 & \\ \hline \delta & \beta_0 & \cdots & \beta_{70} & \mathbf{I} \end{array}\right] = \left[\begin{array}{cccccc|c} 0 & \alpha^0 & \alpha^1 & \alpha^2 & \cdots & \alpha^{70} & \mathbf{0} \\ 1 & 1 & 1 & 1 & \cdots & 1 & \\ \hline 0 & \alpha^0 & \alpha^3 & \alpha^6 & \cdots & \alpha^{83} & \mathbf{I} \end{array}\right], \tag{3}$$

and

$$H = \begin{bmatrix} 0 & \alpha^0 & \cdots & \alpha^{70} \\ 1 & 1 & \cdots & 1 \end{bmatrix}, \alpha \in GF\left(2^7\right). \tag{4}$$

Here H is the parity check matrix of SEC-DED with $n_1 = 72$, $k_1 = 64$, $r_1 = 8$, $m = 7$. The primitive polynomial is $p(x) = 1 + x^3 + x^7$. $\delta = 0$, $\beta_i = \alpha^{3i}, 0 \le i \le 70$. The dimensions of B^* and identity matrix $\mathbf{I}$ are $(r_1 + r_2) \times (n_1 + r_2)$ and $r_2 \times r_2$, respectively, where $r_2 = r_1 = m$. Note that HECC adds 15 redundancy bits for the 64 bits data block. The generator matrix G^* can be simply derived from the equation $G^*(B^*)^T = \mathbf{0}$. Hence, the encoded data can be calculated as $c = dG^*$. Here the dimensions of d and c are k_1 (64) and $n_1 + r_2$ (79), respectively.

The decoding steps of HECC for a codeword $\widetilde{c}$ can be summarized as follows: **Step 1:** Compute the syndrome for mode 1: $S_1 = H^*\widetilde{c}$; **Step 2:** Predict the number of erroneous bits based on S_1. In the case that only 0 or 1 bit error happens (i.e., the last bit of S_1 is 1 or bitwise of S_1 is 0), the decode procedure of mode 1, which is similar to SEC-DED, will be performed; Otherwise, go to **Step 3**; **Step 3:** If one uncorrectable error is detected by S_1, compute the syndrome for mode 2: $S_2 = L^*\widetilde{c}$. The error locator polynomial generation and error locator solving will be also conducted to correct the data.

The decoding of mode 1 can be implemented with fast XOR logics with low latency comparable to that of SEC-DED. The decoding procedure of mode 2 can also be substantially accelerated compared to the iterative Chien search in BCH based on similar hardware optimizations as [13].

2) Soft (Mixed) Lines Protections–SEC-DED: HECC is designed to protect hard-lines that are vulnerable to write errors. Since both soft-lines and mix-lines are more reliable than hard-lines, low-cost SEC-DEDs with different protection granularity could be applied. Note soft-lines have the best reliability. We name this scheme as *nonuniform strength ECC* (NUS-ECC). Here the "nonuniform" denotes two properties of this scheme:

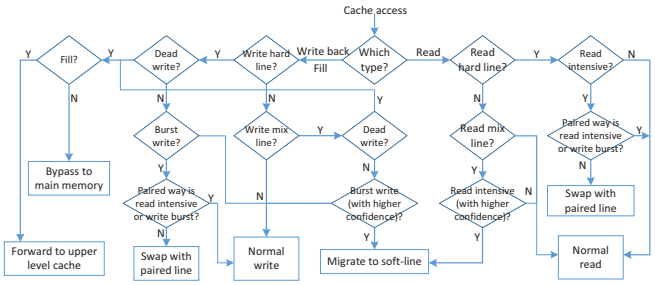


Fig. 4: Flow chart of cache access considering data migration.

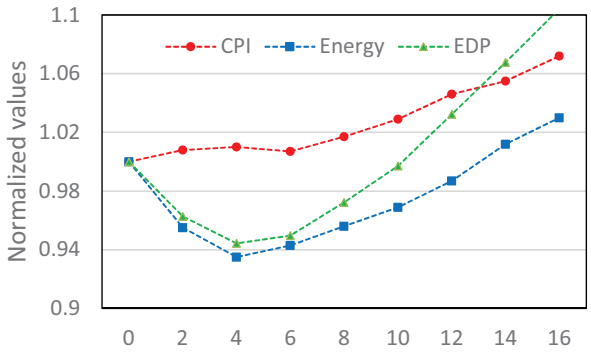


Fig. 5: The CPI, energy and EDP of TMSC as the number of mixed-way is varied. The horizontal axis shows the number of mixed-way allocated from a 16-way 2MB cache.

1) the nonuniform error correction strength of the different ECC schemes applied to the different types of cache lines and 2) the nonuniform coding/decoding of HECC for the hard-lines with different number of erroneous bits.

3) Evaluation of NUS-ECC: To keep the 64B cache line failure rates below 1E-3 [12], we leverage the following three ECCs: HECC (79,64), *medium ECC*–SEC-DED (72,64) and *simple ECC*–SEC-DED(523,512), to protect the hard-lines, mixed-lines and soft-lines, respectively. We synthesized the worst-case ECC–BCH (78,64) and above three different ECC schemes in Synopsys Design Compiler with 32nm technology. The latency results are represented as the numbers of the system clock cycles (4GHz). For HECC, the decoding latencies of mode 1 and mode 2 are $\sim$ 1 cycle and $\sim$ 15 cycles, respectively. As a comparison, the decoding latencies of *medium ECC* and *simple ECC* are $\sim$ 1 cycle and $\sim$ 2 cycles, respectively, while the one of BCH is $\sim$ 24 cycles. Moreover, the storage overhead for those four ECC schemes are 23.4%, 21.9%, 12.5% and 2.15%, respectively. Note that the first two ECCs can correct up to 2 bit errors while the others can only correct a single bit error. Our synthesized results also show that HECC only consumes about 70% of dynamic power of the BCH code.

B. TMSC Architecture Optimizations

In TMSC, the whole cache is partitioned into three types of cache ways, namely, soft-way, hard-way, and mixed-way. Compared to many previous CSM designs, the purpose of introducing mixed-ways is to reallocate some costly but unreliable write accesses from hard-ways to mixed-ways. Increasing the size of mixed-ways, however, also implies the reduction of soft-ways which show the best PER.

Data Management: In order to maximize the utilization of the soft-ways and also suppress the accesses to the hard-ways in TMSC, a subtle data migration policy must be introduced. The cache access flow with the data migration is shown in Figure 4. Every hard-line write needs to be first predicted if it is a dead write and/or burst write before being performed. If the hard-line write is categorized as a dead write by the dead block predictor, it will be either bypassed to the memory or forwarded to the upper-level cache depending on where it comes from; A hard-line write predicted as a burst write by write burst predictor always swaps with its paired soft-line. By doing so, the swapping operation is executed together with the hard-line programming and no extra delay is introduced atop the programming latency of the hard-line [3]. We also move the writes from mixed-line to soft-line within a set, but only when the confidence of the prediction on write burst is high[1]. The hard-line and mixed-line reads are also treated similarly to the corresponding hard-line and mixed-line writes, except that a hard-line read is swapped with the paired line only when it is predicted as intensive read by the read intensive predictor.

We design a versatile access predictor covering dead/burst write and intensive read prediction. The predictor is extended from the one proposed in [14] augmented with read intensive prediction. A read hitting a sampled set is used to update the pattern simulator in the access predictor; an evicted read from the pattern simulator uses its partial PC to update the skewed read intensive predictor [15]. A hard-line or mixed-line read indexes the read intensive predictor to determine if the hard-line or mixed-line needs to swap with or migrate to soft-line.

Cache Partitioning: Figure 5 shows the average CPI, energy consumption and *energy-delay-production* (EDP) of 20 benchmarks of SPEC CPU 2006 [16] when varying the number of mixed-ways in a 16-way set-associated TMSC. Zero mixed-way represents CSM design, in which soft-lines and hard-lines are protected by simple ECC and HECC, respectively. Sixteen mixed-ways represents native MLC design where mixed-lines are protected by medium ECC. Note that data migration is employed for all configurations except for native MLC design. The performance becomes worse as the number of mixed-way increases because the access latency of mixed-ways is much longer than that of soft-lines. On the other hand, the energy consumption of the benchmarks reaches the lowest level when the number of mixed-way is around 4.

The result of EDP shows the tradeoff between performance and energy. At the partition where the best EDP is achieved, TMSC retains sufficient number of soft-way that can satisfy performance requirement, and moves a certain volume of write operations from hard-ways to mixed-ways for energy saving. As Figure 5 shows, the benchmarks achieve the best average EDP if the ratio of the soft-, the hard-, and the mixed-ways are 6:6:4. We adopt such partition in our TMSC design.

Support for NUS-ECC and Tri-way set: We use a stand-alone ECC array attached to each TMSC set to support NUS-ECC, which means that the ECC bits are separated from its

[1]The prediction of write burst is represented by a two-bit saturate counter. If a counter value is greater than one (two), the hard-line (mixed-line) write indexing the counter is predicted as burst write.

TABLE II: The parameters of L2 caches built with different memory technologies

Type	Native MLC		CSM_1bECC		CSM_2bECC		TMSC	
Capacity	2MB	8MB	2MB	8MB	2MB	8MB	2MB	8MB
Area (mm^2)	0.57	1.74	0.57	1.74	0.57	1.74	0.57	1.74
Read energy (nJ)	0.254	0.297	S: 0.202 H: 0.401	S: 0.241 H: 0.466	S: 0.202 H: 0.401	S: 0.241 H: 0.466	S: 0.202 M: 0.254 H: 0.401	S: 0.241 M: 0.297 H: 0.466
Read delay (Cycles)	13	14	S: 9 H: 13	S: 11 H: 14	S: 10 H: 36	S: 12 H: 38	S: 10 M: 13 H (0/1): 13 H (2): 27	S: 12 M: 14 H (0/1): 14 H (2): 28
Write energy (nJ)	1.571	1.782	S: 0.814 H: 2.442	S: 0.882 H: 2.571	S: 0.814 H: 2.442	S: 0.882 H: 2.571	S: 0.814 M: 1.571 H: 2.442	S: 0.882 M: 1.782 H: 2.571
Write delay (Cycles)	60	61	S: 30 H: 68	S: 32 H: 71	S: 30 H: 68	S: 32 H: 71	S: 30 M: 60 H: 68	S: 32 M: 61 H: 71
Leakage (mW)	5.14	10.71	5.14	10.71	5.14	10.71	5.14	10.71

associated way. Each block in ECC array has 8-bit. Suppose a cache line has 512 bits, 2 ECC blocks are needed to protect a soft-line using Simple ECC (11-bit); 8 ECC blocks are needed to protect a mixed-line using medium ECC; and 15 ECC blocks are need to protect a hard-line using HECC. If a TMSC set has 16 ways, 131 ECC blocks are required (corresponding to the partition of 6 soft-/hard-line pairs and 4 mixed-line per set). To simplify the encoding and decoding logic, we force the first twelve cache lines constructed with six MLC rows are soft/hard lines, and the rest four cache lines constructed with two MLC rows are mix lines.

IV. Results

A. Simulation Platform

We evaluate four types of L2 caches, the parameters of each type are shown in Table II. All caches are modeled by NVSim [17]. We consider two CSM designs with two different protection schemes – CSM_1bECC is equipped with medium ECC securing both soft- and hard-line, and CSM_2bECC employs simple ECC (BCH code) for soft-line (hard-line). BCH code used here is capable of correcting two-bit error and requires 14 ECC blocks. We use two ECC blocks (simple ECC) to protect a soft-line in CSM_2bECC. TMSC is protected by NUS-ECC whose configuration is already explained in Section III-B. The storage overhead of NUS-ECC for a 2MB TMSC cache is 312KB or 15.2%, which is slightly higher than that of the ECCs used in CSM_2bECC (246KB or 12%). All timing overhead of ECCs is already considered in the L2 read/write operations. We use GEM5 [18] as our basic simulation platform for 64-bit Alpha instruction set, the detailed configuration is shown in Table III. A single core CPU is equipped with 2MB L2 cache while a quad-core CMP is equipped with 8MB shared L2 cache.

The configuration of access predictor is similar to [14], except we add an read intensive predictor which requires extra three tables, each of which has 4096-entry with a 2-bit counter in each entry. The total storage overhead of a access predictor is 16.8KB, which is 0.82% of the capacity of L2 cache. According to the modeling using CACTI and NVSim, the energy overhead of a predictor update or access only contributes a small portion, says, less than 2% of the energy consumption of a soft-line read in TMSC. These results are consistent to the results from [14]. Therefore, we believe the access predictor has little impact on the area and energy of the proposed L2 cache designs.

We select 20 benchmarks from SPEC CPU 2006 [16] for evaluation. SimPoints [20] is used to extract a single simulation point of each benchmark. Each simulation point contains 500 million instructions. The memory subsystem is warmed up with 100 million instructions before jumping into a simulation point. We use four-benchmark workloads to evaluate quad-core CMP.

TABLE III: System configuration.

Core	4GHz, OOO 8-width, 64-energy LSQ, 64-entry instruction queue, 192-enrty ROB
Caches	32KB L1I, 2-way, 2-cycle, 64B line
	32 KB L1D, 4-way, 2-cycle, 64B line, write back
	2MB/8MB shared L2 for single-/quad-core, 16-way, 4/16 banks, 1 port, 64B line, write back, 20 MSHRs
Main memory	DDR3-1600, two channels, open-page, FR-FCFS [19]
Access predictor	One sampled set per 32 L2 sets, 12-/4-entry 16-bit partial read/write PC array, 12-/4-entry 4-bit LRU, 12-/4-entry valid bit, 12-entry 16-bit partial tag array, 9 4096-energy 2-bit skewed prediction tables

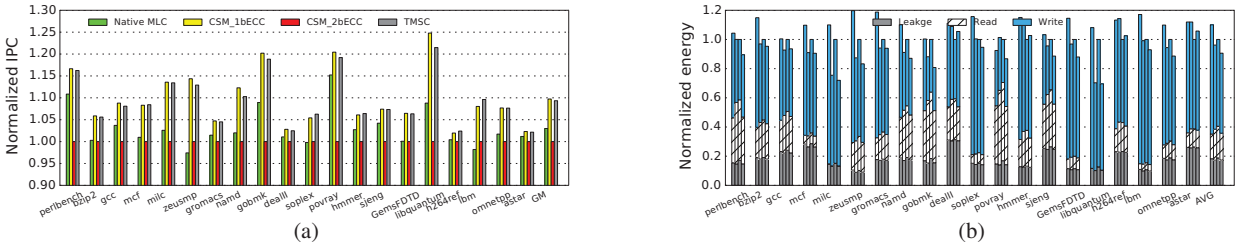
B. Evaluations

Figure 6(a) shows the IPC for different configurations denoted by the their L2 caches when running the workloads, normalized to the IPC of CSM_2bECC. Native MLC outperforms CSM_2bECC by nearly 3%. As shown in Table II, CSM_2bECC employs BCH code to keep a low error rate which is comparable to native MLC. Even the BCH encoding delay of hard-line write can be hidden by paralleling write and encoding, the decoding delay of hard-line read significantly downgrades the speed of read operation by $\sim 3\times$, aggravating the penalty of L1 cache misses in CSM_2bECC. We also show the performance potential of CSM by sacrificing its reliability – CSM_1bECC achieves 6.7% speedup w.r.t. native MLC. The observation of the performance of CSM_1bECC and CSM_2bECC justifies our assertion – the performance merit of CSM will be gone if traditional ECC is used to secure CSM guaranteeing a error rate seen in native MLC.

The performance of TMSC is 9.3% higher than that of CSM_2bECC and comparable to that of CSM_1bECC. TMSC keeps the error rate as low as native MLC or CSM_2bECC, and prevents the performance from being lost: 1) NUS-ECC effectively reduces most decoding overhead when reading a hard-line; 2) The number of soft-line set in tri-way cache partitioning is sufficient to satisfy most read-intensive cache accesses under the context of data migration.

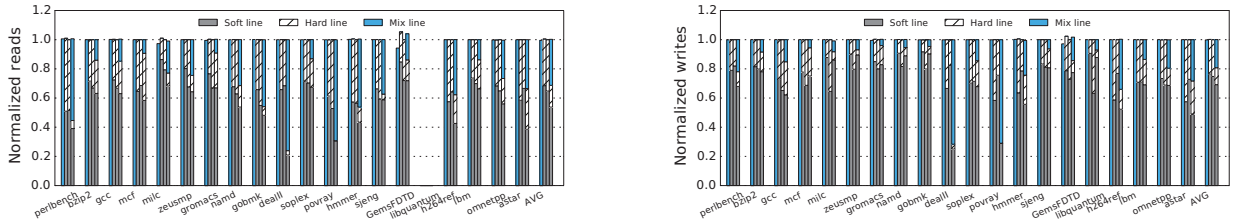
Figure 6(b) shows the energy consumption of L2 caches constructed by different MLC designs. Native MLC consumes the most energy among the MLC STT-RAMs, since it suffers most from the mixed-line writes. The energy consumption of CSM_1bECC is 2.9% lower than that of CSM_2bECC because of its reduced leakage energy. TMSC saves 9.4% energy w.r.t. to CSM_2bECC. Our Tri-way cache partitionings are selected based on the lowest average EDP, and from Figure 5 we can see the best EDP often comes with the lowest energy consumption in each benchmark.

Figure 7(a) and Figure 7(b) show the read and write distribution of L2 caches, normalized to CSM_2b ECC. Most of the L2 reads and writes fall into soft-line in CSM_1bECC, CSM_2bECC and TMSC because of the data migration initiated by the access predictor. A considerable part of reads and writes which originally fall into hard-line in CSM_1bECC and CSM_2bECC are shifted to less-cost mixed-line in TMSC,



(a) (b)

Fig. 6: The performance and L2 energy consumption of single-threaded workloads. In each bar cluster of (b): 1^{st} bar – Native MLC, 2^{nd} bar – CSM_1bECC, 3^{rd} bar – CSM_2bECC, 4^{th} bar – TMSC.



(a) (b)

Fig. 7: The read and write distribution of L2 cache. The representation of each bar in a bar cluster is the same as that in Figure 6(b). All L2 reads/writes in STT-RAM are plotted as soft-read/write for convenience. Note that no L2 read happened when running the simulation point of `libquantum`.

which makes lower energy consumption in TMSC. The distribution of reads and writes of L2 cache also implicates the coverage of access predictor (e.g., $\sim$55% for hard-line writes and $\sim$28% for hard-line reads in TMSC_2bECC).

V. Conclusion

In this work we propose a Tri-region MLC STT-RAM (TMSC) cache design to offer comprehensively optimized trade-off between performance, energy, and reliability. TMSC is based on three-fold innovations at different levels: 1) nonuniform strength ECC (NUS-ECC) that offers variable decoding delay adapting to the reliability requirement of cache lines; 2) tri-way MLC STT-RAM cache design that is composed of three types of cache lines and able to fulfill different PER requirements; and 3) Versatile access predictor that guides the data migration among the different types of cache lines. Our experimental results show that TMSC is as reliable as the performance-driven MLC STT-RAM cache that is employed with very pessimistic and costly ECC scheme, while offering significantly better performance and energy efficiency.

VI. Acknowledgement

This work was partially supported by NSF CNS-1311706, NSF CNS-1116171, National Natural Science Foundation of China (NSFC) under No. 61301079 and No. 91338103, Beijing Higher Education Young Elite Teacher Project and Tsinghua University Initiative Scientific Research Program.

References

[1] G. Sun *et al.*, "A Novel Architecture of the 3D Stacked MRAM L2 Cache for CMPs," in *IEEE 15th International Symposium on High Performance Computer Architecture*, 2009, pp. 239–249.

[2] E. Kultursay *et al.*, "Evaluating STT-RAM as an energy-efficient main memory alternative," in *IEEE International Symposium on Performance Analysis of Systems and Software (ISPASS)*, 2013, pp. 256–267.

[3] X. Bi *et al.*, "Unleashing the potential of MLC STT-RAM caches," in *IEEE International Conference on Computer-Aided Design*, 2013, pp. 429–436.

[4] Y. Chen *et al.*, "On-chip Caches Built on Multilevel Spin-Transfer Torque RAM Cells and its Optimizations," *ACM Journal on Emerging Technologies in Computing Systems (JETC)*, vol. 9, no. 2, p. 16, 2013.

[5] L. Jiang *et al.*, "Constructing Large and Fast Multi-level Cell STT-MRAM based Cache for Embedded Processors," in *IEEE/ACM 49th Design Automation Conference*, 2012, pp. 907–912.

[6] J. Wiley and Sons, *Error Correction Coding: Mathematical Methods and Algorithms*, 2005.

[7] M. Hosomi *et al.*, "A Novel Nonvolatile Memory with Spin Torque Transfer Magnetization Switching: Spin-RAM," in *IEEE International Electron Devices Meeting*, 2005, pp. 459–462.

[8] T. Ishigaki *et al.*, "A Multi-level-cell Spin-transfer Torque Memory with Series-stacked Magnetotunnel Junctions," in *IEEE Symposium on VLSI Technology*, 2010, pp. 47–48.

[9] W. Wen *et al.*, "State-Restrict MLC STT-RAM Designs for High-Reliable High-Performance Memory System," in *IEEE/ACM 51st Design Automation Conference*, 2014, pp. 35:1–35:6.

[10] Y. Zhang *et al.*, "Multi-level cell STT-RAM: Is it realistic or just a dream?" in *IEEE International Conference on Computer-Aided Design*, 2012, pp. 526–532.

[11] W. Wen *et al.*, "PS3-RAM: a Fast Portable and Scalable Statistical STT-RAM Reliability Analysis Method," in *IEEE/ACM 49th Design Automation Conference*, 2012, pp. 1191–1196.

[12] A. R. Alameldeen *et al.*, "Energy-efficient Cache Design Using Variable-strength Error-correcting Codes," in *ISCA*, 2011, pp. 461–472.

[13] Z. Wang, "Hierarchical decoding of double error correcting codes for high speed reliable memories," in *DAC*, 2013, pp. 1–7.

[14] Z. Wang *et al.*, "Adaptive Placement and Migration Policy for an STT-RAM-based hybrid cache," in *IEEE 20th International Symposium on High Performance Computer Architecture*, 2014, pp. 13–24.

[15] P. Michaud *et al.*, "Trading Conflict and Capacity Aliasing in Conditional Branch Predictors," in *ACM SIGARCH Computer Architecture News*, vol. 25, no. 2, 1997, pp. 292–303.

[16] J. L. Henning, "SPEC CPU2006 benchmark descriptions," *ACM SIGARCH Computer Architecture News*, vol. 34, no. 4, pp. 1–17, 2006.

[17] X. Dong *et al.*, "Nvsim: A Circuit-level Performance, Energy, and Area Model for Emerging Nonvolatile Memory," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 31, no. 7, pp. 994–1007, 2012.

[18] N. Binkert *et al.*, "The Gem5 Simulator," *ACM SIGARCH Computer Architecture News*, vol. 39, no. 2, pp. 1–7, 2011.

[19] S. Rixner *et al.*, "Memory Access Scheduling," *ACM SIGARCH Computer Architecture News*, vol. 28, no. 2, pp. 128–138, 2000.

[20] T. Sherwood *et al.*, "Automatically characterizing large scale program behavior," *ACM SIGARCH Computer Architecture News*, vol. 30, no. 5, pp. 45–57, 2002.