Received October 31, 2021, accepted November 15, 2021, date of publication November 22, 2021,
date of current version December 27, 2021.

Digital Object Identifier 10.1109/ACCESS.2021.3129878

EnTiered-ReRAM: An Enhanced Low Latency and Energy Efficient TLC Crossbar ReRAM Architecture

YANG ZHANG[1,2,3], ZHIBIN YU[1], (Member, IEEE), LIANG GU[2], CHENGNING WANG[3], AND DAN FENG[3], (Senior Member, IEEE)
[1]Shenzhen Institute of Advanced Technology, Chinese Academy of Sciences, Shenzhen 518055, China
[2]Sangfor Technologies Inc., Shenzhen 518055, China
[3]Wuhan National Laboratory for Optoelectronics, School of Computer Science and Technology, Huazhong University of Science and Technology, Wuhan 430074, China

Corresponding author: Zhibin Yu (zb.yu@siat.ac.cn)

ABSTRACT Resistive Random Access Memory (ReRAM) is promising to be employed as high density storage-class memory due to its crossbar array and Triple-Level Cell (TLC) structures. However, TLC crossbar ReRAM suffers from high write latency and energy due to three unique challenges: (1) The crossbar array structure incurs IR drop issues. (2) The TLC structure requires iterative program-and-verify (P&V) procedure. (3) The resistance drift problem needs short interval scrub to avoid uncorrectable soft errors. In this article, to overcome the challenges of TLC crossbar ReRAM, we propose an enhanced low latency and energy efficient TLC crossbar ReRAM architecture, called EnTiered-ReRAM. The proposed EnTiered-ReRAM is composed of four components, including EnTiered-crossbar design, Compression-based Incomplete Data Mapping (CIDM), Compression-based Flip Scheme (CFS), and Compression-based Error Correction Code (CECC). Specifically, based on the observation that our previously proposed Tiered-crossbar design still suffers from large IR drops along bitlines in the far segments due to the long length of bitlines, EnTiered-crossbar partitions each crossbar array into two halves along bitlines, and then splits each bitline of the half crossbar array into the near and far segments by an isolation transistor, which thoroughly mitigates the IR drop issues. Then we use our previously proposed CIDM and CFS in the near and far segments of EnTiered-crossbar arrays to further decreases the write latency and energy. In addition, CECC is deeply coupled with CIDM and CFS. CECC dynamically employs the most appropriate ECC capability according to the remaining space of each cache line after CIDM or CFS encoding, which effectively improves the scrub interval and performance/energy with insignificant space overhead. The evaluation results show that, compared with an aggressive baseline, EnTiered-ReRAM can improve the system performance by 56.3% and reduce the energy consumption by 60.6% on average.

INDEX TERMS TLC crossbar ReRAM, IR drop, compression, IDM, flip scheme, ECC.

I. INTRODUCTION

There is a growing demand for large capacity memory in modern data-intensive applications, e.g., video streaming, big data analytics and graphical games. However, the conventional main memory, DRAM, faces low density, scalability, and short refresh interval challenges. ITRS has indicated that the scaling path of DRAM beyond 16nm is not clear [1]. Recently, Non-Volatile Memories (NVMs) such as Spin-Transfer Torque Magnetic Random Access Memory (STT-MRAM), Phase Change Memory (PCM) and Resistive Random Access Memory (ReRAM) have been actively explored as potential candidates for storage-class memory due to their high density, good scalability and non-volatility [2]–[6]. Comparing to other NVM technologies, ReRAM has much better performance than PCM, and much higher density than STT-MRAM. Therefore, among these candidates, ReRAM has become the most promising one [4]–[7].

The associate editor coordinating the review of this manuscript and approving it for publication was Christian Pilato.

ReRAM cells can be built into the unique crossbar array structure, which can achieve the smallest planar cell size ($4F^2$) and much high density [8]–[10]. Moreover, each ReRAM cell can store three bits to further improve the density by employing Triple-Level Cell (TLC) structure. The extremely high density makes TLC crossbar ReRAM suitable for storage-class memory applications. However, TLC crossbar ReRAM also suffers from many severe challenges in terms of performance and energy consumption: (1) The high-density crossbar structure incurs IR drop issues due to wire resistance and sneak currents, resulting in high leakage energy and non-uniform access latency in crossbar arrays. Unfortunately, the worst-case access latency of all cells is conservatively employed in conventional ReRAM writes, leading to significant performance degradation and energy waste. (2) TLC structure requires the iterative program-and-verify (P&V) procedure to program the cell into a certain state, which also causes high write latency and energy. (3) ReRAM's resistance drift problem needs short interval scrub, which consists of reading the data, correcting errors with Error Correction Code (ECC) and rewriting data into ReRAM. The scrub operation also incurs high performance and energy penalties. Recent research has experimentally demonstrated that a write operation on a 4Mb Single-Level Cell (SLC) ReRAM only takes $7.2ns$, while the same operation on Multi-Level Cell (MLC) ReRAM takes $160ns$ and TLC ReRAM achieves much higher write latency [11]. Moreover, TLC ReRAM also has seven times higher write energy than SLC ReRAM [12], [13]. Therefore, for TLC crossbar ReRAM memory systems, the high write latency and energy are the greatest design concerns.

There are many techniques proposed to optimize TLC crossbar ReRAM. Double-Sided Ground Biasing (DSGB) design [4] applies another ground on the other side of the selected wordline to decrease IR drops along wordlines. Our previously proposed Tiered-crossbar design [14] enhances DSGB design and effectively mitigates IR drops of near segments, however the far segments still suffers from the same IR drops as DSGB design does. Incomplete Data Mapping (IDM) [12] technique eliminates certain high-latency and high-energy cell states during the iterative P&V procedure to reduce the write latency and energy of TLC ReRAM. 0-Dominated Flip Scheme (0-DFS) [15] flips the written data with the additional flip flag bits to increase the number of high resistance cells in crossbar arrays, which can effectively decrease the leakage energy. Stronger ECC such as Double Error Correction and Double Error Detection (DECDED) has also been proposed to improve the resilience toward resistance drift problem and enlarge the periodical scrub interval [16]. However, IDM, 0-DFS and stronger ECC techniques are limited by their space overheads. Our previously proposed Compression-based IDM (CIDM) and Compression-based Flip Scheme (CFS) [14] combine the compression technique with IDM and 0-DFS to leverage the saved space by compression, respectively. However, the saved space is not fully

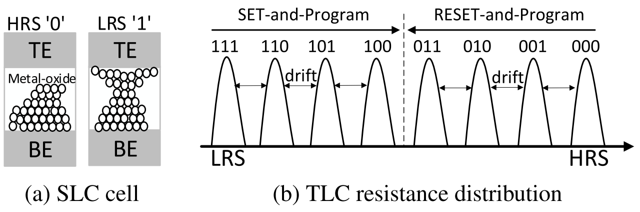


FIGURE 1. Overview of ReRAM cell structures: (a) HRS and LRS for a SLC ReRAM cell; (b) TLC resistance distribution and resistance drift problem.

utilized by CIDM and CFS and there is considerable remaining space unused after CIDM and CFS encoding.

In this article, to overcome the challenges of TLC crossbar ReRAM, we propose an enhanced low latency and energy efficient TLC crossbar ReRAM architecture, called **EnTiered-ReRAM**. The proposed EnTiered-ReRAM consists of four components, namely the EnTiered-crossbar design, CIDM, CFS and Compression-based ECC (CECC). Based on the observation that our previously proposed Tiered-crossbar design [14] still suffers from large IR drops along bitlines in the far segments due to the long length of bitlines, EnTiered-crossbar partitions each crossbar array into two halves, and then splits each bitline of the half crossbar array into the near and far segments by an isolation transistor, which thoroughly mitigates the IR drop issues. Besides, we use our previously proposed CIDM and CFS techniques [14] in the near and far segments to further decrease the write latency and energy, respectively. In addition, we observe that there is considerable remaining space unused after CIDM and CFS encoding and the remaining space varies greatly. On the other hand, we also observe that different ECCs have tradeoffs between error correction capabilities and space overhead. The ECC that can correct more errors has higher space overhead. Therefore, to make full use of the saved space by compression for stronger ECC capability and smaller scrub penalty, CECC dynamically selects the most appropriate ECC for each cache line according to the remaining space after CIDM or CFS encoding. For each cache line, the proposed encoding techniques are employed on the condition that the total encoded data size never exceeds the original cache line size. The contributions of this article include:

- We propose an enhanced microarchitectural design called *EnTiered-crossbar* to partition each crossbar array into two halves along bitlines and split each bitline of the half crossbar array into the near and far segments by an isolation transistor, which thoroughly mitigates the IR drop issue and enables low latency/energy TLC crossbar ReRAM.
- Based on the observation that there is various remaining space unused after CIDM and CFS encoding and different ECCs have tradeoffs between error correction capabilities and space overhead, we subtly integrate CECC with CIDM and CFS techniques. We implement CECC in both near and far segments by dynamically selecting the most appropriate ECC for each cache line

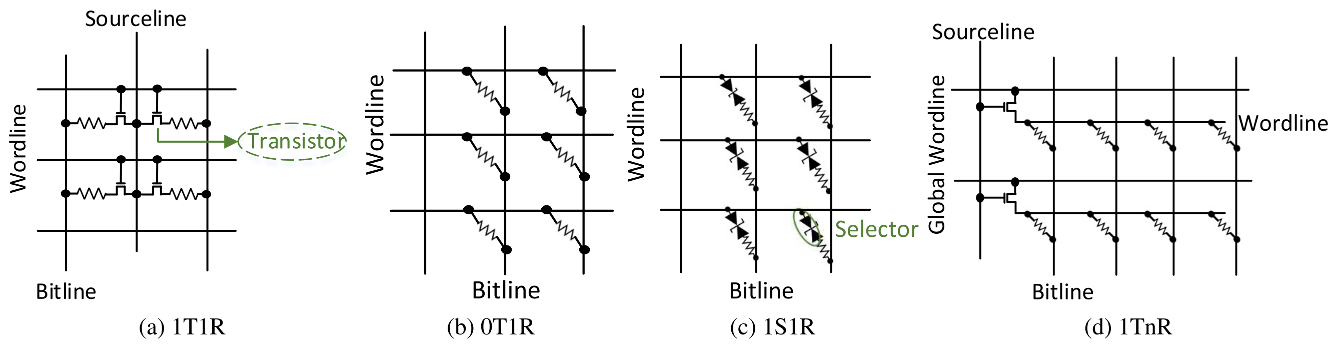


FIGURE 2. **Overview of ReRAM array structures: (a) One-Transistor-One-ReRAM (1T1R) structure. Each cell owns a dedicated access transistor; (b) Zero-Transistor-One-ReRAM (0T1R) crossbar structure. ReRAM cells are interconnected to each other without access transistors; (c) One-Selector-One-ReRAM (1S1R) crossbar structure. Each cell has no access transistor, but integrates a dedicated selector; (d) One-Transistor-n-ReRAM (1TnR) structure (n is 4 in the figure). Four ReRAM cells share an access transistor.**

according to the remaining space after CIDM or CFS encoding, which effectively improves the scrub interval and penalties of TLC crossbar ReRAM.

- We evaluate the proposed EnTiered-ReRAM comparing with an aggressive baseline. The evaluation results demonstrate that EnTiered-ReRAM improves the system performance by 56.3% and reduces the energy consumption by 60.6% on average.

II. BACKGROUND

A. RERAM CELL STRUCTURE

As depicted in Figure 1a, a ReRAM cell is composed of three layers, namely a metal-oxide layer sandwiched between a top electrode (TE) and a bottom electrode (BE). The resistance range is used to represent the state of a ReRAM cell. The high resistance state (HRS) and low resistance state (LRS) denote logic 0 and 1 respectively for a SLC ReRAM cell. To switch the resistance state of a ReRAM cell, the cell should apply an external voltage with specified polarity, magnitude and duration across its terminals. The switching process from LRS to HRS is referred to as a RESET operation and the switching process from HRS to LRS is referred to as a SET operation. ReRAM has the TLC feature by storing three bits into a single cell due to the huge resistance range between HRS and LRS (Resistance ratio of HRS to LRS can exceed 1000) [12], [17]. TLC ReRAM divides the wide range resistance into eight levels, as shown in Figure 1b. Compared to SLC ReRAM, TLC ReRAM can offer higher data density.

B. RERAM ARRAY STRUCTURE

ReRAM array structure can be categorized into three types, including One-Transistor-One-ReRAM (1T1R), crossbar and One-Transistor-n-ReRAM (1TnR). In 1T1R structure, each cell owns a dedicated access transistor, as depicted in Figure 2a. The sourceline is used to control the access transistors. When a sourceline is activated, the access transistors in the selected line provide exclusive access to the cells in that line. Hence, each cell can be accessed independently without disturbing other cells in the array. Among all ReRAM

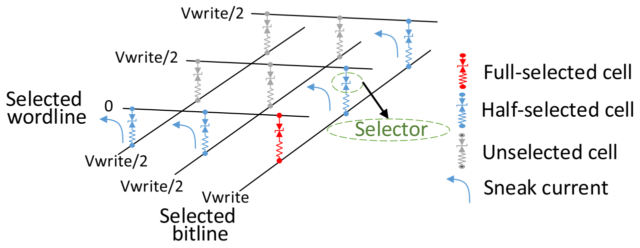


FIGURE 3. **RESET operation in 1S1R crossbar structure. Sneak currents across half-selected cells and wire resistance result in IR drop issues.**

array structures, 1T1R structure can offer the best access performance due to the dedicated access transistor. However, 1T1R structure significantly reduces the area efficiency because the size of an access transistor is typically much larger than that of a ReRAM cell. For example, Sato *et al.* [18] have constructed a 1T1R ReRAM prototype with a cell size of $15F^2$ and Sheu *et al.* [11] have also built a HfOx-based ReRAM prototype with a cell size of $9.5F^2$.

In crossbar structure, all ReRAM cells are interconnected to each other without access transistors, where a cell only occupies an area of $4F^2$ (The smallest planar cell size in theory) [8]. The crossbar structure can be categorized into Zero-Transistor-One-ReRAM (0T1R) and One-Selector-One-ReRAM (1S1R) structures. In 0T1R crossbar structure, the access transistors are eliminated and the smallest theoretical size can be achieved, as shown in Figure 2b. In 1S1R structure, a dedicated selector is integrated into each cell based on 0T1R structure, as shown in Figure 2c. The selector is effective to improve the nonlinearity of ReRAM cells, where the nonlinearity is referred to as the ratio of the amount of current flowing through a selected ReRAM cell with voltage V to the cell with voltage $V/2$ (V is the applied voltage for write or read operation). The higher the nonlinearity, the higher the feasibility to construct a large crossbar array. On the other hand, since the selector can be constructed on top of the switching material, the selector incurs no extra area overhead. Therefore, compared with 0T1R cells, 1S1R cells have higher nonlinearity with the same cell size. In other words, 1S1R crossbar structure enables the fabrication of large crossbar arrays.

One-Transistor-n-ReRAM (1TnR) structure is a tradeoff between 0T1R and 1T1R structures, where n ReRAM cells share one access transistor. Figure 2d presents the One-Transistor-Four-ReRAM (1T4R) structure, where there is one ReRAM cell located at each crosspoint of bitline and wordline, and each wordline connects four ReRAM cells to the access transistor. Similar with 1T1R structure, the sourceline is used to control the connecting access transistors. Each global word line can connect multiple sourcelines and wordlines. Compared with 1T1R structure, 1TnR has better area efficiency and worse access performance. Compared with crossbar structure, 1TnR has worse area efficiency and better access performance.

Considering the better scalability and lower fabrication cost, 1S1R crossbar structure is apparently more suitable for constructing the high density storage-class memory. In 1S1R structure, the selectors can be constructed using many different materials with various operating voltage, current densities and endurance. In this article, we model a selector referring to the previous work [19]. We select the 512 × 512 1S1R crossbar ReRAM as the baseline, which has been widely used in community [4], [5], [14].

C. IR DROP ISSUE OF CROSSBAR

In order to perform a write operation in the crossbar array, the bitline and wordline connected to the target cell should be activated with the proper potential ($\pm V_{write}$). Moreover, the unselected bitlines and wordlines of the array are half biased at $V_{write}/2$ to avoid write disturbance. Figure 3 depicts the RESET operation in the crossbar array. In this case, the target cell is applied with full voltage (V_{write}), called *full-selected cell*. Other cells on the selected bitline and wordline are set to $V_{write}/2$, called *half-selected cells*. The remaining cells of the array are referred to as *unselected cells*. Due to $V_{write}/2$ voltage drop across the half-selected cells, there are currents flowing across these cells. The currents are commonly referred to as *sneak currents*. Sneak currents and wire resistance result in large voltage reduction along both bitlines and wordlines, referred to as *IR drop issue*.

The IR drop issue can reduce the voltage drop across the target cell. Unfortunately, the RESET latency of a ReRAM cell is exponentially inverse to the voltage drop across the cell [4], [20]. Therefore, the IR drop issue significantly enlarges the RESET latency. Moreover, ReRAM cells at different locations of the crossbar array have various IR drops, leading to non-uniform access latency in crossbar arrays. However, ReRAM writes conservatively use the worst-case access latency of all cells, resulting in significant performance degradation and energy waste. In addition, due to the sneak currents of LRS half-selected cells, the IR drop issue also causes high leakage energy. As demonstrated in the recent study [21], for a 100 × 100 crossbar array, accessing the target cell only consumes about 1% of the total energy and about 97% of the total energy is dissipated by the sneak currents of LRS half-selected cells.

TABLE 1. Write parameters of TLC crossbar ReRAM.

Target states	111	110	101	100	011	010	001	000
Iterations	1.21	5.27	10.1	15	14.3	9.83	4.68	1.52
Latency (ns)	14.2	95.4	192	290	383	338.3	286.8	255.2
Energy (pJ)	1.8	13.4	24.3	46.8	94	66.4	41.1	33.6

D. ITERATIVE PROGRAM-AND-VERIFY OF TLC RERAM

For a TLC ReRAM cell, it is difficult to employ a generic scheme to precisely program the cell into a certain resistance range due to the statistical characteristics and process variation. Instead, program-and-verify (P&V) is the commonly used mechanism for TLC ReRAM programming. The P&V operation starts from a RESET or SET operation, which initializes the TLC state to the highest or lowest resistance range. Then series of smaller I_{SET} or V_{RESET} pulses follow. Each I_{SET} or V_{RESET} pulse is followed by a read operation to verify whether the state of the cell reaches the programming target. Once the resistance range of the cell reaches the programming target, the write operation terminates. The iterative P&V procedure leads to high write latency and energy. In order to reach the target states in fewer iterations for improving the write latency and energy, SET-and-Program (SAP) and RESET-and-Program (RAP) schemes [12] have been proposed. SAP and RAP are the P&V mechanisms starting from SET and RESET operations, respectively. For both SAP and RAP schemes, the most significant bit (MSB) of the target state is first checked. If the MSB of the target state is '0', RAP scheme is applied. Otherwise, SAP is used, as shown in Figure 1b.

However, the write latency and energy of TLC crossbar ReRAM are still high even with RAP and SAP schemes. The number of P&V iterations is highly dependent on the data written into TLC cells. Programming some states requires more iterations, such as states '011' and '100', leading to high write latency and energy. Moreover, since the RESET operation is more sensitive to the IR drop issue, TLC writes with V_{RESET} (such as '000' and '001') result in higher latency/energy. The worst-case iterations, latency and energy for programming different TLC states in DSGB crossbar arrays are presented in Table 1, where the parameters are mainly obtained from Xu *et al.* [4], [17].

E. RESISTANCE DRIFT PROBLEM

The resistance of a programmed ReRAM cell tends to continuously drift due to the operational disturbance and ambient temperature. Once the resistance moves to a different resistance range (as shown in Figure 1b), the data are corrupted, referred to as *resistance drift problem*. The time for drifting to a different resistance range is referred to as *drift time*. Periodically scrub operations are commonly used in memory systems to avoid such data corruption by reading the data, correcting errors with ECC and rewriting data into arrays. Unlike DRAM which has a very long scrub interval (usually 3 hours for a 4GB DRAM memory system) [22], ReRAM has much shorter scrub interval due to its high soft error rates resulting from the resistance drift problem. As demonstrated in the previous work [16], the worst-case drift time of MLC NVMs is only 1.8 seconds and TLC NVMs has smaller drift time. Palangappa *et al.* [13] have also pointed out the drift time for TLC ReRAM is no more than 2 seconds. The short drift time of ReRAM requires short interval scrub, which incurs high performance and energy penalties.

Generally, conventional memory systems support Single Error Correction and Double Error Detection (SECDED) and execute a scrub process to trigger error correction before the happening of a second error. For a cache line, if the interval between the first and the second errors is shorter than the given scrub interval, the errors may not be corrected, resulting in an uncorrectable error. In this case, an expensive system-level memory error exceptions occurs, leading to serious consequences, e.g., program termination and program rollbacks. Therefore, with SECDED, the scrub interval must be shorter than the interval between the first and second errors. It is reasonable for DRAM systems due to the extremely low soft error rate in DRAM. However, for ReRAM memory systems, the resistance drift problem incurs relatively high soft error rate. If a weak ECC which can correct few errors for a given data block size, is applied in ReRAM, the scrub interval must be very short to avoid uncorrectable soft errors and the scrub penalties will be much high. While if a stronger ECC which can correct more errors for a given data block size, is used in ReRAM, more bit errors can be tolerated and the scrub interval can also be enlarged. Therefore, the scrub interval of ReRAM is closely related to the ECC capability.

III. MOTIVATION

A. IR DROPS IN DSGB DESIGN

Conventional DSGB design has been widely adopted in community to reduce the IR drops in crossbar arrays [4], [5]. By applying another ground on the other side of the selected wordline, DSGB effectively reduces the length of the worst-case IR drop path and significantly mitigates the IR drops along wordlines. However, the magnitude of IR drops is determined by both wordlines and bitlines [6]. In DSGB crossbar arrays, the IR drops along bitlines are still large due to the long length and large wire resistance of bitlines, leading to significant performance degradation and energy waste. Most prior studies [5], [23], [24] focus on optimizing the non-uniform access latency in crossbar arrays resulting from IR drops, failing to optimize the IR drops from the source. Although our previously proposed Tiered-crossbar design [14] effectively mitigates IR drops of the near segments, however the far segments still suffer from large IR drops as DSGB design does due to the long length of bitlines. Therefore, an enhanced crossbar design is required to thoroughly mitigate IR drops.

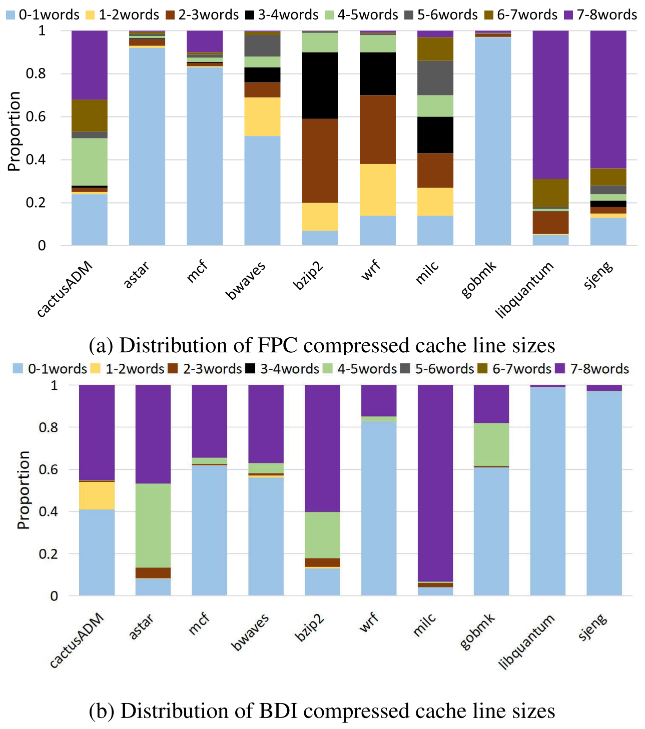


FIGURE 4. Distribution of compressed cache line sizes using SPEC CPU2006 benchmarks with (a) FPC technique; (b) BDI technique. Each 512-bit cache line is composed of eight 64-bit words. The compressed cache line sizes are classified into eight types by 64-bit word. Y-axis represents the proportion of each type.

B. SAVED SPACE BY COMPRESSION VARIES

Compression is an effective technique to save the storage space [25]–[28]. Two pattern-based compression techniques, e.g., Frequent pattern compression (FPC) [29] and base-delta-immediate compression (BDI) [30], are commonly used due to their high compressibility, low overhead, low latency, and low complexity.

FPC. FPC is a typical pattern-based compression technique and it can successfully compress a wide range of data by leveraging program data statistics. Actually, FPC was originally used in L2 caches with 32-bit word for improving the memory capacity [29]. Recently, extended FPC has been proposed to reduce bit writes in NVMs [31]. In this work, FPC is used for a 64-bit word, as tabulated in Table 2. Each compressed word has a 3-bit prefix to denote the encoded patterns, as presented by column 1 of the table. With FPC technique, a 64-bit word can be compressed to 3, 11, 19, or 35 bits, so that 61, 53, 45 or 29 bit storage space can be saved. Each word is compressed separately for a 512-bit cache line that includes eight 64-bit words. Thus, the saved space of a cache line varies and may range from 0 to 488 bits.

BDI. BDI is another typical pattern-based compression technique that was originally applied for data compression in on-chip cache [30] by using a 'base' (B) and 'delta' (D) to store the compressed data. D is composed of a series of offsets related to B. By employing BDI, a cache line CL = (P0, P1, ..., Pn-1) can be compressed as CLcom = (B, D0, D1, ..., Dn-1), where B denotes P0, Di denotes Pi - B, n represents

TABLE 2. The 64-bit FPC patterns with 3-bit prefix (indicated in red).

Prefix	Pattern Encoded	Example	Compressed Example	Encoded Size	Saved Space
000	Zero run	0x0000000000000000	0x0	3 bits	61 bits
001	8-bit sign extended	0x000000000000007F	0x17F	11 bits	53 bits
010	16-bit sign extended	0xFFFFFFFFFFFFB6B6	0x2B6B6	19 bits	45 bits
011	Half-word sign extended	0x0000000076543210	0x376543210	35 bits	29 bits
100	Half-word, padded with a zero half-word	0x7654321000000000	0x476543210	35 bits	29 bits
101	Two half-words, each a byte sign extended	0xFFFFBEEF00003CAB	0x5BEEF3CAB	35 bits	29 bits
110	Word consisting of four repeated double bytes	0xCAFECAFECAFECAFE	0x6CAFE	19 bits	45 bits

TABLE 3. The 64-byte BDI patterns with 3-bit prefix (indicated in red).

Prefix	Pattern Encoded	Example	Compressed Example	Encoded Size
000	Zeros	0x000000000000...000000000000	0x0	0 byte
001	Repetition	0xBEEFCAFEEEC2FF3E ... BEEFCAFEEEC2FF3E	0x1BEEFCAFEEEC2FF3E	8 bytes
010	Base8-D1	0xCDEFCDEFCDEFCD00 ... 0xCDEFCDEFCDEFCD07	0x2CDEFCDEFCDEFCD00, ..., 07	15 bytes
011	Base8-D2	0xCDEFCDEFCDEFCD00 ... 0xCDEFCDEFCDEFAB07	0x3CDEFCDEFCDEFCD00, ..., AB07	22 bytes
100	Base8-D4	0xCDEFCDEFABCDAB00 ... 0xCDEFCDEFEEFFAB07	0x4CDEFCDEFABCDAB00, ..., EEFFAB07	36 bytes
101	Base4-D1	0xCDEFAB00 CDEFAB01 ... CDEFABFF	0x5CDEFAB00, 01, ..., FF	19 bytes
110	Base4-D2	0xCDEFAB00 CDEFCD01 ... CDEFADFF	0x6CDEFAB00, CD01, ..., ADFF	34 bytes
111	Base2-D1	0xCD00 CD1A CDED ... CDFF	0x7CD00, 1A, ED, ..., FF	35 bytes

the number of D in a cache line. For simplicity, we use BDI without the implicit second base in this work. Different from FPC, BDI is used with 64-byte patterns to make full use of data regularity to compress cache lines, as shown in Table 3. Similar to FPC, BDI also employs a 3-bit prefix to indicate the BDI compression pattern. With BDI technique, a 64-byte cache line can be compressed to 0, 8, 15, 19, 22, 34, 35 or 36 bytes (Prefix is not included), and thus 64, 56, 49, 45, 42, 30, 29, or 28 byte storage space can be saved. Therefore, the saved space of a cache line varies and may also range from 0 to 509 bits.

In order to quantitatively show the distribution of compressed cache line sizes, we evaluate FPC and BDI techniques with the SPEC CPU2006 benchmarks in our architectural simulator (Detailed description in Section V). Figure 4a and 4b present the distribution of FPC and BDI compressed cache line sizes, respectively. The results show that most cache lines can be compressed with both FPC and BDI techniques and the compressed cache line sizes vary greatly. Lots of cache lines can be compressed to smaller than one word. In this case, more than seven word storage space can be saved. While some cache lines still have more than seven words after compression (Including the uncompressible cache lines) and the saved space of these cache lines is less than one word. In this work, the proposed techniques take FPC as an example and can also achieve the similar effects based on BDI technique because both FPC and BDI can effectively compress cache lines.

C. REMAINING SPACE VARIES AFTER CIDM AND CFS ENCODING

With the FPC technique [29], the saved space of a cache line may range from 0 to 488 bits. Our previously proposed CIDM and CFS techniques [14] select the most appropriate IDM and flip scheme for each cache line according to the saved space by compression, as shown in Table 4 and 5, respectively.

TABLE 4. Remaining space after CIDM encoding.

Saved space (bit)	Encoding method	Remaining space (bit)
[341, 488]	IDM((8,2),1)	[0, 147]
[170, 341)	IDM((8,4),1)	[0, 171)
[85, 170)	IDM((8,6),2)	[0, 85)
[0, 85)	CDM	[0, 85)

TABLE 5. Remaining space after CFS encoding.

Saved space (bit)	Encoding method	Remaining space (bit)
[74, 488]	2-bit word-size 0-DFS	[0, 414]
[40, 74)	4-bit word-size 0-DFS	[0, 34)
[21, 40)	8-bit word-size 0-DFS	[0, 19)
[11, 21)	16-bit word-size 0-DFS	[0, 10)
[0, 11)	Without 0-DFS	[0, 11)

However, the saved space by compression is not fully utilized by CIDM and CFS. We observe that there is considerable remaining space unused after CIDM and CFS encoding and the remaining space varies greatly. For example, for a compressed cache line with CIDM encoding, if the cache line saves 341 to 488 bits, the IDM((8,2),1) is applied. In this case, there are 0 to 147 bits remaining unused. Even if the cache line only saves 0 to 84 bits, (Complete Data Mapping) CDM encoding is employed and there are 0 to 84 bits unused. The remaining space after CIDM encoding is presented in Table 4. Similarly, for a compressed cache line with CFS encoding, if the cache line saves 74 to 488 bits, the ((8,2),1) is applied. In this scenario, there are 0 to 414 bits remaining unused. While if the cache line saves 0 to 10 bits, there are only 0 to 10 bits remaining. The remaining space after CFS encoding is shown in Table 5.

D. TRADEOFFS IN DIFFERENT ECCS

As described in section II-E, stronger ECC can improve the immunity toward resistance drift and enlarge the scrub

TABLE 6. Tradeoffs in capabilities and space overhead.

BCH-n	Capability for each 64-bit word	Additional ECC bits	Space overhead
BCH-1	Correct 1 error	8	12.5%
BCH-2	Correct 2 errors	14	21.9%
BCH-3	Correct 3 errors	21	32.8%
BCH-4	Correct 4 errors	28	43.8%
BCH-5	Correct 5 errors	35	54.7%
BCH-6	Correct 6 errors	42	65.6%

interval. In order to theoretically analyse the effects of ECC capabilities on the scrub intervals, we build a ECC model about the Uncorrectable Block Error Rate (UBER). Assume an k-bit cache line is composed of the data block and the corresponding ECC field. We define the probability that a ReRAM cell changes its originally programmed state (due to the resistance drift problem) as f(t), where t denotes the time elapsed since the last programming. In fact, f(t) is referred to as the soft error rate. Assume an ECC can correct m soft errors and the soft errors are all independent [16]. Based on the principles of binomial distribution, the UBER can be expressed as follows, where k denotes the total bits of the data block and the corresponding ECC field in a cache line:

$$UBER(t) = 1 - \sum_{i=0}^{m} \binom{k}{i} f(t)^{i}(1 - f(t))^{(k-i)} \tag{1}$$

For any f(t), the larger the m, the smaller the UBER(t). Note that a stronger ECC can correct more soft errors and corresponds to a larger value of m. On the other hand, since the probability of soft errors in a programmed ReRAM cell continuously increases as time, UBER(t) and f(t) are both monotonically increasing function about t. Therefore, given a UBER value, larger m achieves longer t, where t corresponds to the scrub interval. In other words, a stronger ECC can effectively enlarge the scrub interval and improve the performance/energy.

However, different ECCs have tradeoffs between error correction capabilities and space overhead. We take BCH-n code [32] as an example, where n denotes that the selected ECC can correct n errors in each 64-bit word. Table 6 shows the tradeoffs of different ECCs in capabilities and space overhead. As n increases, BCH-n have stronger error correction capability, but the space overhead also becomes larger. Although BCH-1 achieves the lowest space overhead (12.5%), it can only correct 1 error in a 64-bit word. BCH-6 has the strongest error correcting capability which can correct up to 6 errors in each 64-bit word, however it incurs 65.6% space overhead. In a word, stronger ECC can correct more errors and achieve larger scrub interval and smaller performance/energy overhead, but it incurs larger space overhead.

IV. ENTIERED-RERAM ARCHITECTURE

A. OVERVIEW

In this article, to reduce the write latency and energy of TLC crossbar ReRAM, we propose the EnTiered-ReRAM

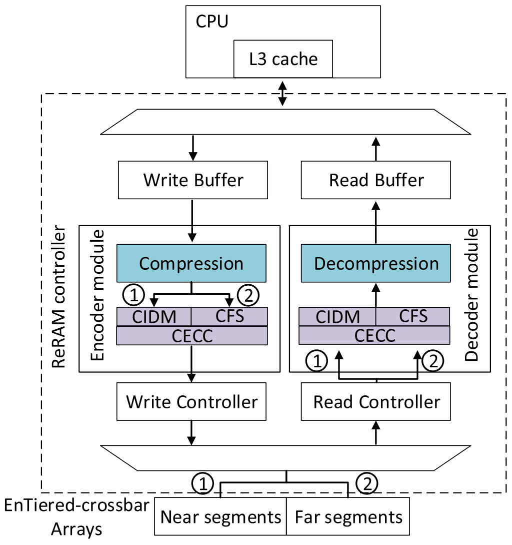


FIGURE 5. Overview of EnTiered-ReRAM architecture. Near segment access is performed as path ① and far segment access is performed as path ②.

architecture. Figure 5 depicts the overview of the proposed design. EnTiered-ReRAM is composed of four components, namely the EnTiered-crossbar design, CIDM, CFS and CECC. The EnTiered-crossbar design is implemented in the ReRAM array level, with the purpose of fundamentally optimizing the IR drops along bitlines based on DSGB design. Similar to our previous work [14], CIDM and CFS are performed in the ReRAM controller as Path ① and Path ②, respectively, which dexterously combine the compression technique with IDM and flip scheme to decreases the write latency/energy with insignificant space overhead. CECC is also performed in the ReRAM controller as both Path ① and Path ②, which dynamically selects the most appropriate ECC for each cache line according to the remaining space after CIDM or CFS encoding and achieves stronger ECC capability and smaller scrub penalty with insignificant space overhead. Next, we elaborate the details of the proposed design.

B. ENTIERED-CROSSBAR DESIGN

As described in III-A, DSGB design [4] still suffers large IR drops along bitlines due to the long length and large wire resistance of bitlines (Figure 6a), and the write latency/energy is still high. Latency optimized crossbar arrays have shorter bitlines and smaller IR drops, resulting in reduced write latency/energy, as shown in Figure 6b. However, for a given ReRAM capacity, additional write drivers (WD) and sense amplifiers (SA) are required, which results in high peripheral circuit overhead. To shorten the bitlines with smaller peripheral circuit overhead, our previously proposed Tiered-crossbar design [14] splits each long bitline into two shorter segments using an isolation transistor, as shown in Figure 6c. Although Tiered-crossbar effectively mitigates IR drops of near segments (Near segments only occupy 1/4 of the whole array), the far segments still suffer from the same IR drops as DSGB design does.

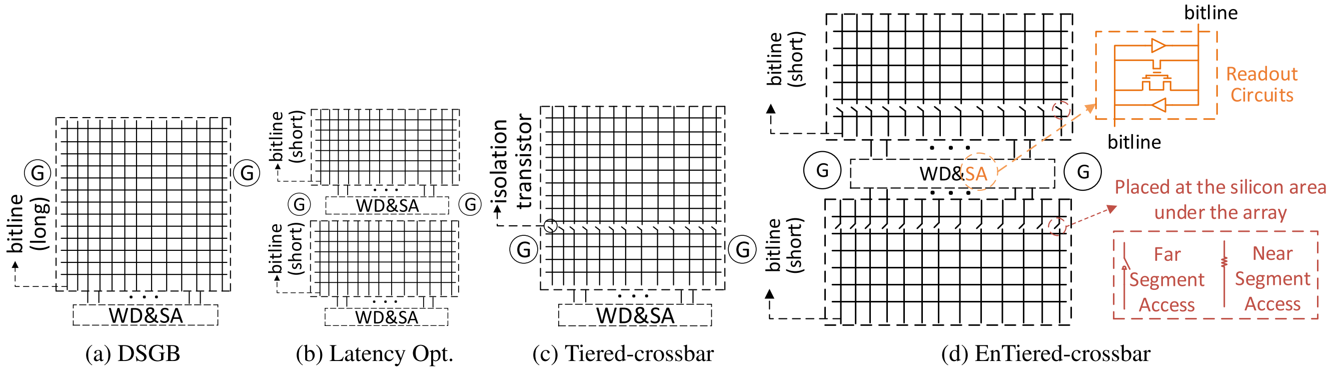


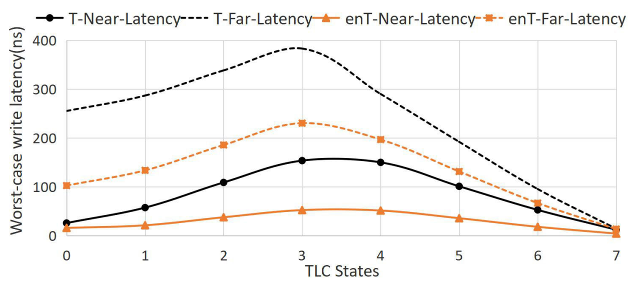
FIGURE 6. Comparison among different crossbar designs: (a) DSGB applies ground at double sides of wordlines. G represents the applying ground; (b) latency optimized design halves the crossbar array and has smaller write latency, but requires additional write drivers (WD) and sense amplifiers (SA); (c) the previous proposed Tiered-crossbar still suffers from large IR drops in the far segments; (d) the target EnTiered-crossbar combines the advantages of latency optimized and Tiered-crossbar arrays. EnTiered-crossbar halves the crossbar array and shares WD/SA between adjacent crossbar arrays, and then splits each bitline of the half crossbar array into near and far segments by an isolation transistor.

TABLE 7. Parameters in our ReRAM circuit model.

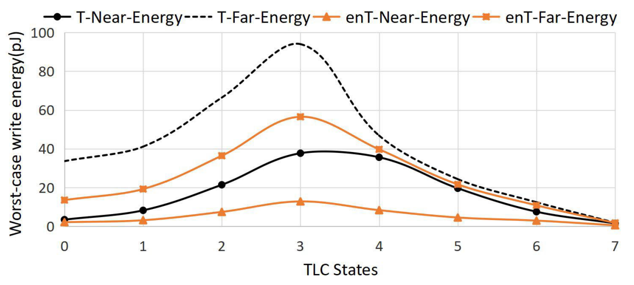
Metric	Description	Value
A	Crossbar array size: A $wordlines\times$ A $bitlines$	512
R_{wire}	Wire resistance between adjacent cells	2.82Ω
R_{LRS}	Resistance of a LRS cell	$100K\Omega$
K_r	Nonlinearity of a LRS ReRAM device	10
K_s	Nonlinearity of the selector	3000
V_W	Full selected voltage during write	3.2V
C_R	Capacity ratio of near segments to far segments	1:3
–	Voltage biasing scheme	DSGB

To thoroughly mitigate IR drops along bitlines based on DSGB design, we propose an enhanced microarchitectural design called *EnTiered-crossbar*. Similar to latency optimized crossbar design, EnTiered-crossbar also halves the length of bitlines and the whole crossbar array is partitioned into two halves, as shown in Figure 6d. However, EnTiered-crossbar shares peripheral circuit between adjacent crossbar arrays and incurs no additional WD/SA overheads. The data parallelism can be maintained by alternately activating the half crossbar arrays. The readout circuits are shown in Figure 6d. In addition, similar to Tiered-crossbar design, EnTiered-crossbar splits each bitline of the half crossbar array into near and far segments by an isolation transistor, where the isolation transistor is placed at the silicon area under the array. The near segment is directly connected to the WD/SA and the far segment is far away from the WD/SA. When a ReRAM cell in the near segment is accessed, the isolation transistor on the selected bitline is turned off, so that only the bitline in the near segment incurs IR drops. Therefore, cells in the near segment have smaller IR drops and lower write latency/energy. On the other hand, when a ReRAM cell in the far segment is accessed, the isolation transistor on the selected bitline is turned on. In this scenario, although the entire bitline incurs IR drops, EnTiered-crossbar has shortened the bitline by half and IR drops along bitlines have been significantly decreased. Compared with latency optimized crossbar array, EnTiered-crossbar has smaller IR drops due to the tiered structure and can reduce additional transistors by 81.8% (WD and SA require 11 transistors per nanowire) due to the shared peripheral circuit. Compared with Tiered-crossbar design, EnTiered-crossbar doubles the area of near segments and reduces the IR drops by halving the length of bitlines with only 8.3% more transistors.

To make a fair comparison with Tiered-crossbar, the capacity ratio of the near segments to the far segments in each half of EnTiered-crossbar is also 1:3. To quantitatively show the effectiveness of EnTiered-crossbar design, we construct a detailed circuit model for the 512 $\times$ 512 TLC crossbar array based on Kirchhoff's Current Law [6], [15], [33]. Table 7 presents the key parameters derived from IBM's MIEC device [19] and the HfOx-based cells [20]. The worst-case voltage drops of the near and far segments are obtained from the circuit simulation. According to the relationship between the voltage drop and write latency/energy [20], the worst-case write latency/energy is also achieved. In this article, Tiered-crossbar and EnTiered-crossbar have the same technology node (22nm) and the same boundary simulation conditions. Figures 7a and 7b present the worst-case write latency and energy comparison between Tiered-crossbar and EnTiered-crossbar designs, respectively. In the two figures, T-Near-Latency/Energy represents the write latency/Energy of the near segments in Tiered-crossbar design and enT-Near-Latency/Energy denotes the write latency/Energy of the near segments in EnTiered-crossbar design. T-Far-Latency/Energy and enT-Far-Latency/Energy can be explained in the same way. Since DSGB design has the same write latency and energy as the far segments of Tiered-crossbar (namely T-Far-Latency and T-Far-Energy), we don't draw the curve of DSGB design repeatedly. The results show that compared with Tiered-crossbar design, EnTiered-crossbar can achieve 64.3% and 43% write latency reduction in the near and far segments, respectively. Besides, EnTiered-crossbar decreases 69.2% and 38.3% write energy in the near and far segments, respectively. Therefore, EnTiered-crossbar design allows the near and far segments to be accessed with lower latency and energy. Similar to many prior



(a) Worst-case write latency of Tiered-ReRAM and EnTiered-ReRAM.



(b) Worst-case write energy of Tiered-ReRAM and EnTiered-ReRAM.

FIGURE 7. Comparisons between Tiered-crossbar and EnTiered-crossbar designs in terms of (a) write latency; (b) write energy. The lower write latency and energy indicate the better access performance of the design. enT-Near-Latency, enT-Far-Latency, enT-Near-Energy and enT-Far-Energy respectively represent the write latency and energy in the near and far segments of EnTiered-crossbar, which is the target design. DSGB design has the same write latency and energy as the far segments of Tiered-crossbar (namely T-Far-Latency and T-Far-Energy). Tiered-crossbar and EnTiered-crossbar have the same technology node (22nm) and the same boundary simulation conditions.

TABLE 8. The most appropriate ECC configuration.

Remaining space (bit)	Encoding method	3-bit ECC flag
[272, 488]	BCH-6	000
[216, 272)	BCH-5	001
[160, 216)	BCH-4	010
[104, 160)	BCH-3	011
[48, 104)	BCH-2	100
[0, 48)	Default SECDED	101

works [14], [23], [24], [34]–[39], EnTiered-crossbar design remaps cold data to the far segments and hot data to the near segments, which further improves the access performance. The dynamic mapping method [23] is adopted in this work to improve the access performance.

C. CIDM AND CFS

Considering that hot data in the near segments of EnTiered-crossbar arrays are sensitive to the access performance and cold data in the far segments of EnTiered-crossbar arrays are not, EnTiered-ReRAM employs our previously proposed CIDM and CFS techniques in the near and far segments, respectively. CIDM and CFS delicately apply the compression technique in conjunction with IDM and flip scheme by dynamically selecting the most appropriate IDM and flip scheme for each cache line according to the saved space by compression. CIDM and CFS techniques have been presented in our previous work [14] in detail.

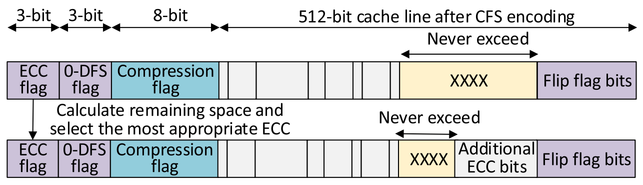


FIGURE 8. CECC encoding procedure in far segments.

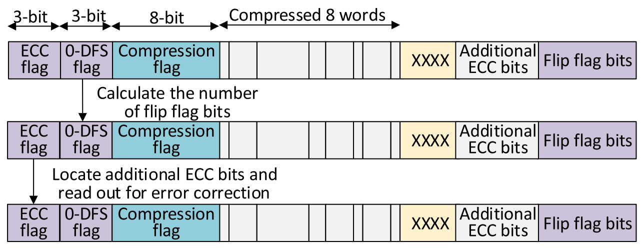


FIGURE 9. CECC decoding procedure in far segments.

D. COMPRESSION-BASED ECC

The resistance drift problem of ReRAM results in high uncorrectable error rates. Although scrub operation can effectively solve the problem, the short interval scrub incurs high performance and energy penalties. Stronger ECC can be used to improve the scrub interval and performance/energy. However, different ECCs have tradeoffs in error correction capabilities and space overhead. The ECC that can correct more errors has higher space overhead. Note that the default SECDED is the commonly used ECC in conventional memory systems, which applies 8 ECC bits every 64-bit word and has additional 64 ECC bits for each cache line. In order to maintain fairness, we argue that the additional 64 ECC bits always exist in this work.

As analyzed in section III-B, the saved space of a 521-bit cache line ranges from 0 to 488 bits under FPC technique. After CIDM or CFS encoding, there may be remaining space unused in a compressed cache line. According to the remaining space of each cache line, there exists the most appropriate ECC for the cache line. For example, when the remaining space of a cache line is smaller than 48 bits after CIDM or CFS encoding, the default SECDED is the most appropriate because the remaining space is too small to support stronger ECC. If stronger ECCs are applied, the total encoded data size will exceed the original cache line size, resulting in high space overhead. However, when the remaining space of a cache line is larger than 272 bits after CIDM or CFS encoding, the stronger ECC that can correct 6 errors for each 64-bit word is the most appropriate ECC. In this case, the stronger ECC can correct more errors with no additional space overhead. To make full use of the saved space by compression for higher ECC capability and smaller scrub penalty, we should dynamically select the most appropriate ECC for each cache line according to the remaining space after CIDM or CFS technique.

To achieve the goal above, we propose *Compression-based ECC* (*CECC*). For each cache line, CECC dynamically selects the most appropriate ECC according to the remaining

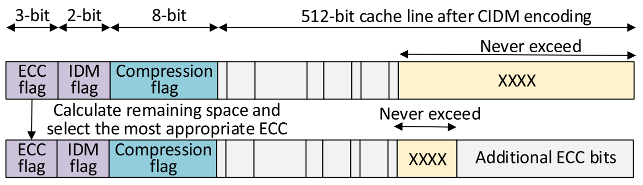

3-bit
2-bit
8-bit
512-bit cache line after CIDM encoding
Never exceed
ECC flag
IDM flag
Compression flag
XXXX
Calculate remaining space and select the most appropriate ECC
Never exceed
XXXX
Additional ECC bits

FIGURE 10. **CECC encoding procedure in near segments.**

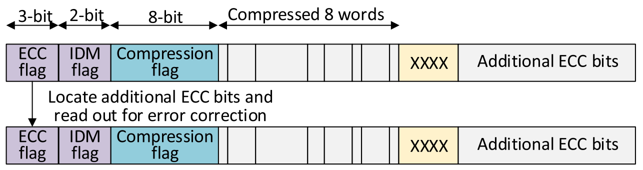


FIGURE 11. **CECC decoding procedure in near segments.**

space after CIDM or CFS encoding. Similar to CIDM and CFS techniques, CECC is also performed at the cache line granularity. CECC calculates the remaining space after CIDM or CFS encoding in each cache line, and then selects the ECC that has as strong error correction capability as possible on the condition that the total encoded data size never exceeds the original cache line size. Therefore, cache lines after CIDM or CFS encoding are encoded with different ECCs. Table 8 presents the most appropriate ECC for each cache line according to the remaining space after CIDM or CFS technique. For each cache line, a 3-bit ECC flag is employed to denote the ECC encoding method, e.g., '101' for the default SECDED, '100' for BCH-2, '011' for BCH-3, '010' for BCH-4, '001' for BCH-5, '000' for BCH-6, where BCH-n denotes that the ECC can correct n errors in every 64-bit word.

CECC is composed of the encoder and decoder modules, which are embedded inside the ReRAM controller, as shown in Figure 5. Due to the different encoding techniques (CIDM and CFS) in near and fast segments, CECC encoder and decoder also varies in near and far segments.

1) CECC ENCODER IN FAR SEGMENTS

When the far segments receive a write request from the processor, CFS encoding technique is first applied. After that, CECC encoder calculates the remaining space and dynamically selects the most appropriate ECC according to the remaining space, as shown in Figure 8. In order to rapidly locate the additional ECC bits for the ECC decoding, these bits are placed next to the flip flag bits of CFS technique. Each cache line is encoded with the most appropriate ECC through the additional ECC bits. A 3-bit ECC flag is applied to represent the ECC encoding method. For the cache line whose remaining space is smaller than 48 bits after CFS encoding (including the uncompressible cache line), the default SECDED is used as the conventional memory systems do.

2) CECC DECODER IN FAR SEGMENTS

When performing an error correction operation in the far segments, the CECC decoder functions as Figure 9 depicts. CECC decoder first calculates the number of flip flag bits of CFS encoding according to the 0-DFS flag, and then locates the additional ECC bits according to the ECC flag. Then the additional ECC bits and the 64-bit inherent ECC bits are read out for error correction. Note CECC decoding procedure for the far segments has no impact on CFS decoding.

3) CECC ENCODER IN NEAR SEGMENTS

When the near segments receive a write request from the processor, cache lines are first encoded through CIDM technique. After that, CECC encoder for the near segments also calculates the remaining space and dynamically selects the most appropriate ECC according to the remaining space, as shown in Figure 10. Different from the CECC encoder for the far segments, the CECC encoder for the near segments places the additional ECC bits at the end of each cache line. A 3-bit ECC flag is also set to denote the ECC encoding method. The default SECDED is applied to the cache line whose remaining space is smaller than 48 bits after CIDM encoding.

4) CECC DECODER IN NEAR SEGMENTS

When performing an error correction operation in the near segments, the CECC decoder functions as Figure 11 depicts. CECC decoder only needs to locate the additional ECC bits according to the ECC flag and read out these bits for error correction. Similarly, CECC decoding procedure for the near segments has no influence on CIDM decoding.

V. EVALUATION METHODOLOGIES

At the circuit level, the write latency and energy parameters of the near and far segments are obtained from our ReRAM circuit model, as presented in Figures 7a and 7b. The power, latency and area parameters of additional circuits is achieved through NVsim [40]. After that, these parameters are added to our architectural simulator.

At the architecture level, to evaluate the proposed designs, we use GEM5 [41] as our simulation platform with the integration of NVMain [42], which is a cycle accurate memory simulator for NVMs. Table 9 shows the detailed simulation configurations. The write latency (tWR) parameters of the near and far segments are obtained from the circuit simulation (Refer to Figure 7a). Other ReRAM-related memory timing parameters are derived from previous works [4], [23]. We get the scrub intervals of different ECCs in CECC technique according to the ECC model in section III-D (UBER is 10^{-10} in this work) and add the scrub overheads into our simulator. Table 10 presents the selected 10 benchmarks from SPEC CPU2006 with different memory Read Per Kilo Instructions (RPKI) and memory Write Per Kilo Instructions (WPKI) rates. For each selected benchmark, we run 500 million instructions to warm up caches and then run 1 billion instructions to evaluate the proposed techniques. DSGB [4] with the integration of IDM((8,6),2) [12] is chosen as the aggressive baseline, which uses IDM((8,6),2) encoding technique in DSGB-based crossbar arrays to reduce write

TABLE 9. Simulation configurations.

Parameter	Value
CPU	4-Core, out of order, 3GHz, 192-entry recoder buffer, 8 issue width
L1 Cache	Private, 16KB I-cache, 16KB D-cache, 2-way assoc, 2-cycle access latency
L2 Cache	Private, 1MB, 64B cache line, 8-way assoc, 20-cycle access latency
L3 cache	Shared, 16MB, 64B cache line, 16-way assoc, 50-cycle access latency
Main memory	8GB, DDR3-1333, 4 channel, 2 ranks/channel, 32 banks/rank, 1024 crossbar arrays/bank
ReRAM Timing(ns)	tRCD(18), tCL(15), tCWD(13), tFAW(30), tWTR(7.5), tWR(refer to Figure 7a)

TABLE 10. The selected SPEC CPU2006 benchmarks.

Benchmark	Description	RPKI	WPKI
cactusADM	Four copies of cactusADM	6.82	6.61
astar	Four copies of astar	2.04	1.05
mcf	Four copies of mcf	2.24	1.23
bwaves	Four copies of bwaves	10.14	9.62
bzip2	Four copies of bzip2	2.32	1.17
wrf	Four copies of wrf	8.18	7.88
milc	Four copies of milc	1.28	1.12
gobmk	Four copies of gobmk	1.65	1.44
libquantum	Four copies of libquantum	7.31	7.06
sjeng	Four copies of sjeng	8.32	8.13

latency/energy. The comparison configurations are listed as follows:

- *baseline*: Apply IDM((8,6),2) in DSGB-based crossbar arrays to reduce write latency/energy.
- *TC*: Apply the Tiered-crossbar design.
- *enTC*: Apply the EnTiered-crossbar design.
- *Tiered-ReRAM*: Our previous work [14] which applies CIDM and CFS techniques in the near and far segments based on Tiered-crossbar design, respectively.
- *Tiered-ReRAM + CECC*: Apply CECC based on *Tiered-ReRAM* to evaluate the effectiveness of CECC.
- *enCIDM*: Apply CIDM and CECC techniques in the whole crossbar array based on *enTC*.
- *enTiered-ReRAM*: Apply all proposed techniques, including the EnTiered-crossbar design, CIDM in the near segments, CFS in the far segments and CECC.

VI. EVALUATION RESULTS

A. OVERHEAD ANALYSIS

1) ADDITIONAL CIRCUIT OVERHEADS

The power, latency and area overheads of EnTiered-ReRAM are evaluated through NVsim [40], which mainly come from the additional isolation transistors, encoders, decoders and multiplexers. We obtain the transistor device characteristics (scaled down to 22nm technology) from Narasimha's work [43]. The results show that the isolation transistors only incur 19.8pW power, 142ps latency and 0.74% area overheads in a 512×512 crossbar array, which are acceptable. Compared with the conventional ECC in ReRAM, CECC only takes more time to calculate the remaining space in the CECC encoder and locate additional ECC bits in the CECC decoder, which is negligible. The overheads of CIDM and CFS as well as the additional control logic to determine the near/far segment access in EnTiered-ReRAM are the same as our previously proposed work [14]. Similar to our previous work [14], the write latency table lookup incurs no additional latency overhead because EnTiered-ReRAM can look up the write latency table in parallel to the write operation.

2) STORAGE OVERHEAD

The storage overheads of CIDM and CFS are the same as our previously proposed work [14]. Besides, CECC in the near segments requires additional 3-bit flag to denote the selected ECC encoding method. Therefore, CIDM (10 bits for each 512-bit cache line) and CECC techniques incur 2.5% storage overhead in the near segments. CECC in the far segments also requires additional 3-bit flag. Therefore, CFS (11 bits for each 512-bit cache line) and CECC techniques in the far segments result in 2.7% storage overhead. Similar to our previously proposed work [14], the address remapping table in the memory controller leads to 256KB storage overhead for the 8GB TLC ReRAM.

B. SYSTEM PERFORMANCE

We evaluate the system performance through the IPC (Instructions Per Cycle) speedup. Figure 12 illustrates the average IPC speedup of different design configurations with the results normalized to *baseline*. The results show that the proposed *enTiered-ReRAM* can improve the system performance by 56.3% on average. Compared to *TC*, *enTC* obtains 22.6% more performance improvements because the EnTiered-crossbar design reduces more IR drops and achieves lower write latency in both near and far segments. Compared to *Tiered-ReRAM*, *Tiered-ReRAM + CECC* achieves 4.7% more performance improvements due to the effectiveness of CECC technique. *enTiered-ReRAM* gets 25.1% more performance improvements than *Tiered-ReRAM* owing to the effective EnTiered-crossbar design and CECC technique. Since both EnTiered-crossbar design and CIDM technique in the whole crossbar array can significantly reduce the write latency, *enCIDM* achieves the best performance. *enTiered-ReRAM* has 2.4% fewer performance improvements than *enCIDM* because CFS technique programs more high resistance cells in the far segments, which slightly increases the write latency. However, *enTiered-ReRAM* can effectively decrease the sneak currents and leakage energy through these high resistance cells.

C. WRITE LATENCY

Figure 13 presents the average write latency of different design configurations with the results normalized to *baseline*. It can be observed that the proposed techniques can significantly reduce the write latency. *enTiered-ReRAM* achieves 59.4% write latency reduction over *baseline* on average. *enTC* gets 23.4% more write latency reduction than *TC* due to the effectiveness of EnTiered-crossbar design. Since CECC

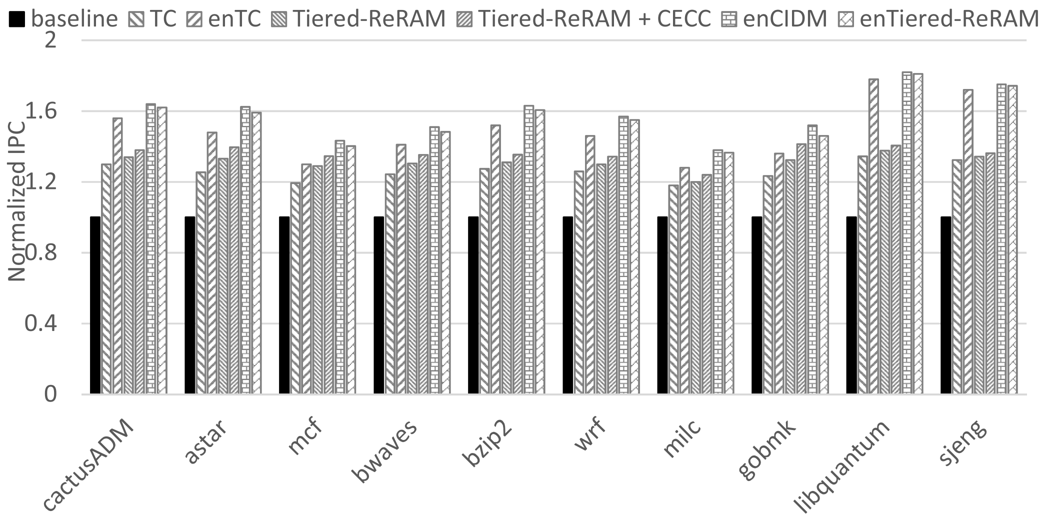


FIGURE 12. The average IPC speedup.

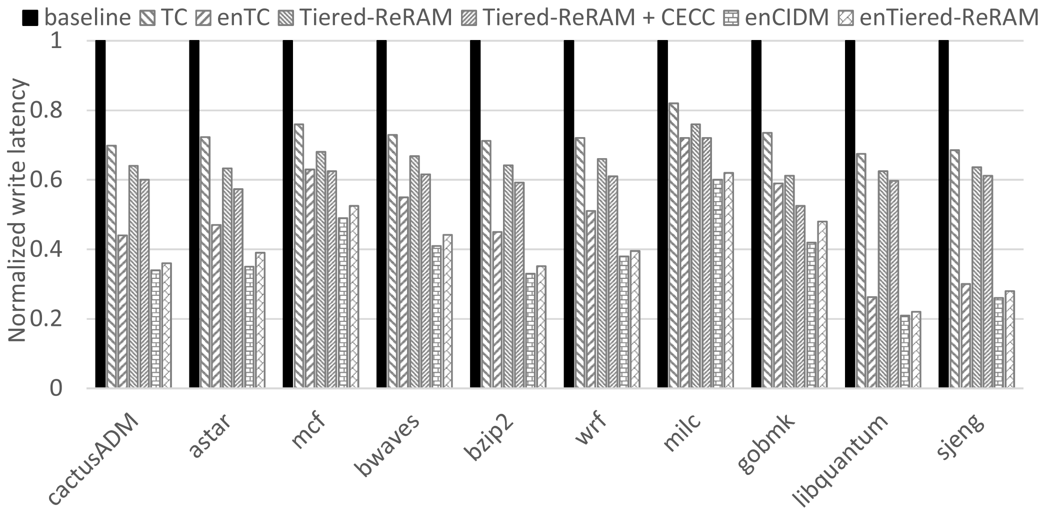


FIGURE 13. The average memory write latency.

technique can enlarge the scrub interval and reduce addition write operations, *Tiered-ReRAM + CECC* achieves 4.9% more write latency reduction over *Tiered-ReRAM*. Compared to *enTiered-ReRAM*, *enCIDM* obtains 2.7% more write latency reduction because the CFS technique in *enTiered-ReRAM* slightly increases the write latency of far segments. However, *enTiered-ReRAM* can effectively reduce the sneak currents and leakage energy through the CFS technique.

D. READ LATENCY

Since the proposed techniques, including the EnTiered-crossbar design, CIDM and CECC, can significantly reduce the write latency, the write service time can be significantly reduced. As the write service time is shortened, read requests significantly benefit from the waiting time reduction. Therefore, the overall read latency is also decreased. Figure 14 illustrates the average read latency of different design configurations. The results are normalized to *baseline* and show that, on average, the proposed *enTiered-ReRAM* can reduce the read latency by 52.4% on average. Compared to *TC*, enTC gets 20.3% more read latency reduction. Due to the effectiveness of CECC, *Tiered-ReRAM + CECC* has 5.3% more read latency reduction than *Tiered-ReRAM*. Although *enCIDM* achieves 2% more read latency reduction than *enTiered-ReRAM*, it enlarges the leakage energy.

E. ENERGY CONSUMPTION

The energy consumption of EnTiered-ReRAM mainly comes from fix sources: read operations, write operations, scrub operations, isolation transistors, encoders and decoders. Due to the significantly reduced read and write latency, EnTiered-ReRAM can effectively reduce the energy consumption even if the isolation transistors, encoders and decoders consume additional energy. Besides, the proposed

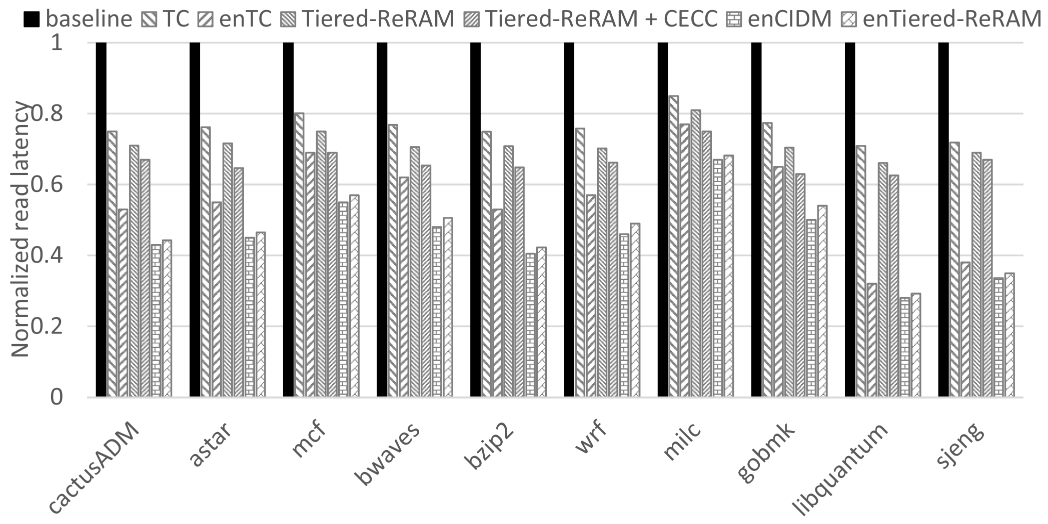


FIGURE 14. The average memory read latency.

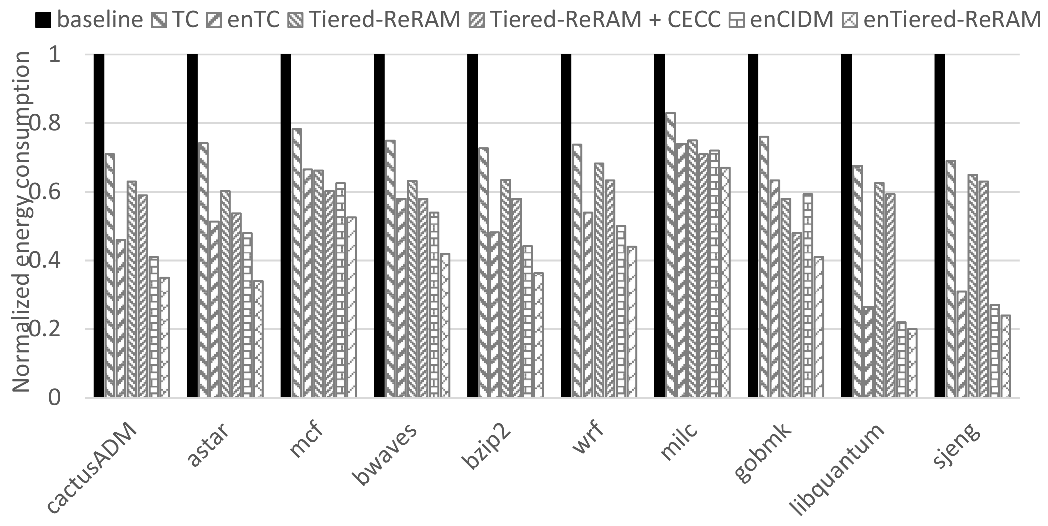


FIGURE 15. The average memory energy consumption.

CECC technique can effectively improve the scrub interval compared with the conventional design, which decreases the number of additional read and write operations for scrubbing and also saves the energy consumption. In addition, the proposed CFS technique in the far segments of EnTiered-ReRAM effectively reduces the leakage energy by programming more high resistance ReRAM cells into crossbar arrays.

Figure 15 depicts the average energy consumption of different configurations with the results normalized to *baseline*. The results show that, *enTC*, *enCIDM*, *enTiered-ReRAM* can reduce the energy consumption by 48.1%, 52% and 60.6%, respectively. Compared to *TC*, *enTC* achieves 22.2% more energy reduction. *Tiered-ReRAM + CECC* achieves 5.8% more energy reduction than *Tiered-ReRAM* owing to the effectiveness of the proposed CECC technique. *enTiered-ReRAM* can achieve 8.6% more energy reduction over *enCIDM* because the CFS technique in *enTiered-ReRAM* programs more high resistance ReRAM cells into crossbar arrays and significantly reduces leakage energy.

VII. RELATED WORK

A. ECC TECHNIQUES

Since NVMs are susceptible to both soft and hard errors, where soft error represents the error that can be corrected by ECC and hard error denotes the error that can't be corrected by ECC, many researches focus on Error Detection And Correction (EDAC) techniques. Awasthi *et al.* [16] analysed the resistance drift problem on NVMs and used ECC to scrub NVMs for high reliability. They also found that stronger ECC support led to increased scrub intervals and decreased scrub energy. Schechter *et al.* [44] proposed to use ECP to tolerate hard errors in resistive memories by permanently encoding the locations of failed cells into a table and assigning new cells to replace them. Xu *et al.* [45] proposed an error-resilient ReRAM architecture by applying

ECC to address the retention failure and using hard error tolerating scheme (e.g., ECP) to solve the stuck-at-fault problem. Zheng *et al.* [46] pointed out the pseudo-hard error on ReRAM and presented a detection method through the multiple-program-and-verify process and a recovery scheme by increasing the voltage amplitude or the write pulse width. These mechanisms can all be applied in conjunction with our techniques.

Other memory technologies also face the reliability problem and lots of EDAC techniques are proposed. For STT-MRAM, Wen *et al.* [47] have proposed a content-dependent ECC (CD-ECC) technique to achieve balanced error correction at different bit-flipping directions. For PCM, Seong *et al.* [48] have proposed SAFER to recover multi-bit stuck-at-fault error with single bit error correction techniques by dynamically partitioning a data block and ensuring at most one failed bit in each partition. Ipek *et al.* [49] have proposed DRM mechanism at operation system level by allowing continued operation when hard faults occur. Besides, Seong *et al.* [50] proposed to apply conventional Hamming code to correct a ternary cell error and synchronously maintain the information density for TLC PCM. Stanisavljevic *et al.* [51] proposed the stronger binary encoded BCH code for TLC PCM. Long *et al.* [52] and Palframan *et al.* [53] have proposed Free ECC and COP techniques respectively to apply compression for improved reliability, but both of them could not improve the ECC capability. These works primarily focus on EDAC techniques in other memory technologies, and also can be used in ReRAM.

B. MITIGATING IR DROP ISSUE

The IR drop issue of crossbar arrays impedes the development of ReRAM-based memory systems, which significantly increases the write latency and energy of ReRAM. There are numerous works focusing on mitigating the IR drop issue of crossbar arrays. Xu *et al.* [4] proposed Double Sided-Ground Biasing (DSGB) design, which applies another ground on the other side of the selected wordline to reduce the IR drops. However, DSGB fails to take the IR drops of the long bitlines into account, leading to significant performance degradation and energy waste. Different from DSGB design, the proposed EnTiered-crossbar design first partitions the whole crossbar array into two halves along bitlines, and then splits each long bitline of the half crossbar array into the near and far segments by an isolation transistor, which fundamentally reduces the IR drops. Moreover, EnTiered-crossbar design only incurs 0.74% area overhead based on DSGB design. Zhao *et al.* [54] proposed the 1TnR V-ReRAM design, which changes the directions of access lines and reorganizes the peripheral circuitry to reduce the IR drops. Shevgoor *et al.* [55] proposed a novel sample and hold circuit for mitigating the influence of IR drops on read operations. Zhang *et al.* [15] and Wen *et al.* [5] proposed to optimize the data patterns and write more 0s into SLC crossbar arrays for decrease the IR drops. Different from these works, our proposed techniques aim to optimize the write operation of TLC crossbar arrays.

C. LEVERAGING NON-UNIFORM ACCESS LATENCY

Since ReRAM cells at different locations of the crossbar array suffer from various IR drops, this characteristic results in non-uniform access latency in crossbar arrays. Conventional ReRAM writes conservatively employs the worst-case access latency of all cells, leading to significant performance degradation and energy waste. Lots of works pay attention to leveraging the non-uniform access latency in crossbar arrays for optimizing the access performance. Zhang *et al.* [23] proposed the *Leader* design to partition each crossbar array into fast and slow regions by rows based on the observation that the access latency of crossbar arrays is related to the distance between selected row and write drivers. They also proposed the static and dynamic mapping methods to remap hot data to fast regions and cold data to slow regions, which effectively improves the access performance. In EnTiered-crossbar design, we also adopt the dynamic mapping method to further improve access performance. Compared with *Leader*, EnTiered-crossbar fundamentally mitigates the IR drop issue of both the near and far segments, offering much better performance for the whole array. Zhang *et al.* [24] observed that the access latency of crossbar arrays varies even in the same row, and then proposed a fine-grained region partition and address remapping scheme to improve the access performance. Other works [34]–[36] leverage the non-uniform access latency of DRAM to improve access performance and can also be applied in ReRAM.

D. DATA ENCODING TECHNIQUES

Compression technique is effective to save the storage space. FPC [29] can successfully compress a wide range of data by leveraging program data statistics and can be extended for a 64-bit word. BDI [30] employs a ‘base’ (B) and ‘delta’ (D) to store the compressed data. Different from FPC, BDI is used with 64-byte patterns to make full use of data regularity to compress cache lines. Both FPC and BDI can effectively compress cache lines and we choose FPC as the baseline due to its low overhead and low complexity.

IDM can effectively decrease the write latency and energy of TLC ReRAM by mapping only part of TLC states into binary digits. Niu *et al.* [12] observed that programming different TLC states costs different latency and energy, and then proposed IDM((8,6),2) method which eliminates two latency/energy critical states. Two 6-state cells are employed to represent 5 digit bits in IDM((8,6),2). That’s because $log_2 6^2 \approx 5$. IDM((8,6),2) reduces the write latency and energy of TLC ReRAM, however it incurs 20% space overhead. Palangappa *et al.* [13] proposed to integrate compression techniques with the expansion coding for write latency and energy reduction. However, the space overhead of the expansion code is fixed and the saved space by compression can’t be fully utilized. Different from these techniques, our proposed encoding technique makes full use of the saved space by compression for more write latency/energy reduction.

In general, flip schemes are applied in memory technologies to reduce the bit flips. Flip-N-Write [56] and FlipMin [57] are commonly used to reduce the bit flips of PCM writes. If the number of different bits is more than half with comparison to the old data, Flip-N-Write will flip the new data. FlipMin employs the coset code to encode each possible input data vector into 256 different vectors, and selects the vector with the minimum bit flips to write. There exist other flip schemes, which can achieve specific effects. Zhang *et al.* [15] proposed 0-DFS to program more 0s into SLC crossbar arrays for decreasing the sneak currents and energy consumption. However, due to the flip flag bits, all the flip schemes suffer from high storage overhead.

VIII. CONCLUSION

In this article, we propose EnTiered-ReRAM architecture to overcome the challenges of TLC crossbar ReRAM in terms of write latency and energy. The proposed EnTiered-ReRAM is composed of EnTiered-crossbar, CIDM, CFS and CECC components. Specifically, EnTiered-crossbar partitions each crossbar array into two halves along bitlines and splits each bitline of the half crossbar array into the near and far segments by an isolation transistor, which thoroughly mitigates the IR drops. Our previously proposed CIDM and CFS are used in the near and far segments of EnTiered-crossbar arrays to further reduce the write latency and energy, respectively. CECC leverages the remaining space of each cache line after CIDM or CFS encoding and dynamically employs the most appropriate ECC capability, which effectively improves the scrub interval and performance/energy. The evaluation results show that, compared with an aggressive baseline, EnTiered-ReRAM can improve the system performance and energy consumption by 56.3% and 60.6%, respectively.

Actually, modern data-intensive applications have exhibited a growing demand for large capacity memory, such as video streaming, big data analytics, graphical games, artificial intelligence algorithms and so on. Our proposed EnTiered-ReRAM is well suited for these use-case scenarios by offering storage-class memory. Although there are no commercial ReRAM chips at present, according to the developments of other NVMs, e.g., PCM and STT-MRAM which have been commercially used, we believe ReRAM will also be commercially used in several years and then our proposed design can be used in these real world use-case scenarios.

ACKNOWLEDGMENT

The authors would like to express their thanks to the reviewers and the editors for their constructive comments and suggestions. The authors would also like to thank anyone who helped them improve this work.

REFERENCES

[1] I. Roadmap, *International Technology Roadmap for Semiconductors*. Washington, DC, USA: Semiconductor Industry Association, 2013.

[2] B. C. Lee, E. Ipek, O. Mutlu, and D. Burger, "Architecting phase change memory as a scalable dram alternative," *ACM SIGARCH Comput. Archit. News*, vol. 37, no. 3, pp. 2–13, Jun. 2009.

[3] E. Kultursay, M. Kandemir, A. Sivasubramaniam, and O. Mutlu, "Evaluating STT-RAM as an energy-efficient main memory alternative," in *Proc. IEEE Int. Symp. Perform. Anal. Syst. Softw. (ISPASS)*, Apr. 2013, pp. 256–267.

[4] C. Xu, D. Niu, N. Muralimanohar, R. Balasubramonian, T. Zhang, S. Yu, and Y. Xie, "Overcoming the challenges of crossbar resistive memory architectures," in *Proc. IEEE 21st Int. Symp. High Perform. Comput. Archit. (HPCA)*, Feb. 2015, pp. 476–488.

[5] W. Wen, L. Zhao, Y. Zhang, and J. Yang, "Speeding up crossbar resistive memory by exploiting in-memory data patterns," in *Proc. IEEE/ACM Int. Conf. Comput.-Aided Design (ICCAD)*, Nov. 2017, pp. 261–267.

[6] C. Wang, D. Feng, J. Liu, W. Tong, B. Wu, and Y. Zhang, "DAWS: Exploiting crossbar characteristics for improving write performance of high density resistive memory," in *Proc. IEEE Int. Conf. Comput. Design (ICCD)*, Nov. 2017, pp. 281–288.

[7] C. Wang, D. Feng, W. Tong, J. Liu, B. Wu, W. Zhao, Y. Zhang, and Y. Chen, "Improving write performance on cross-point RRAM arrays by leveraging multidimensional non-uniformity of cell effective voltage," *IEEE Trans. Comput.*, vol. 70, no. 4, pp. 566–580, Apr. 2021.

[8] Y. Deng, P. Huang, B. Chen, X. Yang, B. Gao, J. Wang, L. Zeng, G. Du, J. Kang, and X. Liu, "RRAM crossbar array with cell selection device: A device and circuit interaction study," *IEEE Trans. Electron Devices*, vol. 60, no. 2, pp. 719–726, Feb. 2013.

[9] C. Wang, D. Feng, W. Tong, J. Liu, B. Wu, W. Zhao, and Y. Zhang, "Design and analysis of address-adaptive read reference settings for multilevel cell cross-point memory arrays," *IEEE Trans. Electron Devices*, vol. 66, no. 12, pp. 5347–5352, Dec. 2019.

[10] C. Wang, D. Feng, W. Tong, Y. Hua, J. Liu, B. Wu, W. Zhao, L. Song, Y. Zhang, J. Xu, X. Wei, and Y. Chen, "Improving multilevel writes on vertical 3-D cross-point resistive memory," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 40, no. 4, pp. 762–775, Apr. 2021.

[11] S.-S. Sheu, M. F. Chang, K. F. Lin, C. W. Wu, Y. S. Chen, P. F. Chiu, C. C. Kuo, Y. S. Yang, P. C. Chiang, W. P. Lin, and C. H. Lin, "A 4Mb embedded SLC resistive-RAM macro with 7.2ns read-write random-access time and 160ns MLC-access capability," in *IEEE Int. Solid-State Circuits Conf. (ISSCC) Dig. Tech. Papers*, Feb. 2011, pp. 200–202.

[12] D. Niu, Q. Zou, C. Xu, and Y. Xie, "Low power multi-level-cell resistive memory design with incomplete data mapping," in *Proc. IEEE 31st Int. Conf. Comput. Design (ICCD)*, Oct. 2013, pp. 131–137.

[13] P. M. Palangappa and K. Mohanram, "CompEx: Compression-expansion coding for energy, latency, and lifetime improvements in MLC/TLC NVM," in *Proc. IEEE Int. Symp. High Perform. Comput. Archit. (HPCA)*, Mar. 2016, pp. 90–101.

[14] Y. Zhang, D. Feng, W. Tong, J. Liu, C. Wang, and J. Xu, "Tiered-ReRAM: A low latency and energy efficient TLC crossbar ReRAM architecture," in *Proc. 35th Symp. Mass Storage Syst. Technol. (MSST)*, May 2019, pp. 92–102.

[15] Y. Zhang, D. Feng, W. Tong, J. Liu, C. Wang, and J. Xu, "CACF: A novel circuit architecture co-optimization framework for improving performance, reliability and energy of reram-based main memory system," *ACM Trans. Archit. Code Optim.*, vol. 15, no. 2, pp. 1–26, 2018.

[16] M. Awasthi, M. Shevgoor, K. Sudan, B. Rajendran, R. Balasubramonian, and V. Srinivasan, "Efficient scrub mechanisms for error-prone emerging memories," in *Proc. IEEE Int. Symp. High-Perform. Comp Archit.*, Feb. 2012, pp. 1–12.

[17] C. Xu, D. Niu, N. Muralimanohar, N. P. Jouppi, and Y. Xie, "Understanding the trade-offs in multi-level cell ReRAM memory design," in *Proc. 50th Annu. Design Autom. Conf. (DAC)*, 2013, pp. 1–6.

[18] Y. Sato, K. Tsunoda, K. Kinoshita, H. Noshiro, M. Aoki, and Y. Sugiyama, "Sub-100-μA reset current of nickel oxide resistive memory through control of filamentary conductance by current limit of MOSFET," *IEEE Trans. Electron Devices*, vol. 55, no. 5, pp. 1185–1191, May 2008.

[19] G. W. Burr, K. Virwani, R. S. Shenoy, A. Padilla, M. BrightSky, E. A. Joseph, M. Lofaro, A. J. Kellock, R. S. King, K. Nguyen, and A. N. Bowers, "Large-scale (512kbit) integration of multilayer-ready access-devices based on mixed-ionic-electronic-conduction (MIEC) at 100% yield," in *Proc. Symp. VLSI Technol. (VLSIT)*, Jun. 2012, pp. 41–42.

[20] H. Y. Lee, Y. S. Chen, P. S. Chen, P. Y. Gu, Y. Y. Hsu, S. M. Wang, W. H. Liu, C. H. Tsai, S. S. Sheu, P. C. Chiang, W. P. Lin, C. H. Lin, W. S. Chen, F. T. Chen, C. H. Lien, and M.-J. Tsai, "Evidence and solution of over-RESET problem for HfOX based resistive memory with sub-ns switching speed and high endurance," in *IEDM Tech. Dig.*, Dec. 2010, pp. 7–19.

[21] M. A. Lastras-Montano, A. Ghofrani, and K.-T. Cheng, "A low-power hybrid reconfigurable architecture for resistive random-access memories," in *Proc. IEEE Int. Symp. High Perform. Comput. Archit. (HPCA)*, Mar. 2016, pp. 102–113.

[22] B. Schroeder, E. Pinheiro, and W.-D. Weber, "DRAM errors in the wild: A large-scale field study," in *Proc. ACM SIGMETRICS*, Seattle, WA, USA, vol. 37, no. 1, Jun. 2009, pp. 193–204.

[23] H. Zhang, N. Xiao, F. Liu, and Z. Chen, "Leader: Accelerating ReRAM-based main memory by leveraging access latency discrepancy in crossbar arrays," in *Proc. Design, Autom. Test Eur. Conf. Exhib. (DATE)*, 2016, pp. 756–761.

[24] Y. Zhang, D. Feng, Z. Tan, J. Liu, W. Tong, and C. Wang, "Asymmetric-ReRAM: A low latency and high reliability crossbar resistive memory architecture," in *Proc. IEEE Int. Conf. Parallel Distrib. Process. With Appl., Ubiquitous Comput. Commun., Big Data Cloud Comput., Social Comput. Netw., Sustain. Comput. Commun. (ISPA/IUCC/BDCloud/SocialCom/SustainCom)*, Dec. 2018, pp. 330–337.

[25] G. Pekhimenko, V. Seshadri, Y. Kim, H. Xin, O. Mutlu, P. B. Gibbons, M. A. Kozuch, and T. C. Mowry, "Linearly compressed pages: A low-complexity, low-latency main memory compression framework," in *Proc. MICRO*, New York, NY, USA: ACM, Dec. 2013, pp. 172–184.

[26] G. Pekhimenko, E. Bolotin, N. Vijaykumar, O. Mutlu, T. C. Mowry, and S. W. Keckler, "A case for toggle-aware compression for GPU systems," in *Proc. IEEE Int. Symp. High Perform. Comput. Archit. (HPCA)*, Mar. 2016, pp. 188–200.

[27] J. Kim, M. Sullivan, E. Choukse, and M. Erez, "Bit-plane compression: Transforming data for better compression in many-core architectures," in *Proc. ACM/IEEE 43rd Annu. Int. Symp. Comput. Archit. (ISCA)*, Jun. 2016, pp. 329–340.

[28] J. Gaur, A. R. Alameldeen, and S. Subramoney, "Base-victim compression: An opportunistic cache compression architecture," in *Proc. ACM/IEEE 43rd Annu. Int. Symp. Comput. Archit. (ISCA)*, Jun. 2016, pp. 317–328.

[29] A. Alameldeen and D. Wood, "Frequent pattern compression: A significance-based compression scheme for L2 caches," Dept. Comp. Scie., Univ. Wisconsin-Madison, Madison, WI, USA, Tech. Rep. 1500, 2004.

[30] G. Pekhimenko, V. Seshadri, O. Mutlu, P. B. Gibbons, M. A. Kozuch, and T. C. Mowry, "Base-delta-immediate compression: Practical data compression for on-chip caches," in *Proc. 21st Int. Conf. Parallel Archit. Compilation Techn. (PACT)*, Sep. 2012, pp. 377–388.

[31] D. B. Dgien, P. M. Palangappa, N. A. Hunter, J. Li, and K. Mohanram, "Compression architecture for bit-write reduction in non-volatile memory technologies," in *Proc. IEEE/ACM Int. Symp. Nanosc. Archit. (NANOARCH)*, Jul. 2014, pp. 51–56.

[32] D. Niu, Y. Xiao, and Y. Xie, "Low power memristor-based ReRAM design with error correcting code," in *Proc. 17th Asia South Pacific Design Autom. Conf.*, Jan. 2012, pp. 79–84.

[33] Y. Zhang, D. Feng, J. Liu, W. Tong, B. Wu, and C. Fang, "A novel ReRAM-based main memory structure for optimizing access latency and reliability," in *Proc. 54th Annu. Design Autom. Conf.*, Jun. 2017, pp. 1–6.

[34] Y. H. Son, O. Seongil, Y. Ro, J. W. Lee, and J. H. Ahn, "Reducing memory access latency with asymmetric DRAM bank organizations," *ACM SIGARCH Comput. Archit. News*, vol. 41, no. 3, pp. 380–391, Jun. 2013.

[35] D. Lee, Y. Kim, V. Seshadri, J. Liu, L. Subramanian, and O. Mutlu, "Tiered-latency DRAM: A low latency and low cost DRAM architecture," in *Proc. IEEE 19th Int. Symp. High Perform. Comput. Archit. (HPCA)*, Feb. 2013, pp. 615–626.

[36] K. K. Chang, A. Kashyap, H. Hassan, S. Ghose, K. Hsieh, D. Lee, T. Li, G. Pekhimenko, S. Khan, and O. Mutlu, "Understanding latency variation in modern DRAM chips: Experimental characterization, analysis, and optimization," *SIGMETRICS Perform. Eval. Rev.*, vol. 44, no. 1, pp. 323–336, Jun. 2016.

[37] Y. Kim, V. Seshadri, D. Lee, J. Liu, and O. Mutlu, "A case for exploiting subarray-level parallelism (SALP) in DRAM," *ACM SIGARCH Comput. Archit. News*, vol. 40, no. 3, pp. 368–379, Jun. 2012.

[38] D. Lee, Y. Kim, G. Pekhimenko, S. Khan, V. Seshadri, K. Chang, and O. Mutlu, "Adaptive-latency DRAM: Optimizing DRAM timing for the common-case," in *Proc. IEEE 21st Int. Symp. High Perform. Comput. Archit. (HPCA)*, Feb. 2015, pp. 489–501.

[39] D. Lee, Y. Kim, G. Pekhimenko, S. Khan, V. Seshadri, K. Chang, and O. Mutlu, "Design-induced latency variation in modern dram chips: Characterization, analysis, and latency reduction mechanisms," *Proc. ACM Meas. Anal. Comput. Syst.*, vol. 1, no. 1, p. 26, 2017.

[40] X. Dong, C. Xu, Y. Xie, and N. P. Jouppi, "NVSim: A circuit-level performance, energy, and area model for emerging non-volatile memory," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 31, no. 7, pp. 994–1007, Jul. 2012.

[41] N. Binkert, B. Beckmann, G. Black, S. K. Reinhardt, A. Saidi, A. Basu, J. Hestness, D. R. Hower, T. Krishna, S. Sardashti, R. Sen, K. Sewell, M. Shoaib, N. Vaish, M. D. Hill, and D. A. Wood, "The gem5 simulator," *ACM SIGARCH Comput. Archit. News*, vol. 39, no. 2, pp. 1–7, May 2011.

[42] M. Poremba and Y. Xie, "NVMain: An architectural-level main memory simulator for emerging non-volatile memories," in *Proc. IEEE Comput. Soc. Annu. Symp. VLSI*, Aug. 2012, pp. 392–397.

[43] S. Narasimha, T. Ivers, C. Warm, R. Wise, R. Wachnik, D. Schepis, S. Sankaran, J. Norum, S. Luning, Y. Li, and M. Khare, "High performance 45-nm SOI technology with enhanced strain, porous low-k BEOL, and immersion lithography," in *IEDM Tech. Dig.*, Dec. 2006, pp. 1–4.

[44] S. Schechter, G. H. Loh, K. Straus, and D. Burger, "Use ECP, not ECC, for hard failures in resistive memories," *ACM SIGARCH Comput. Archit.*, vol. 38, no. 3, pp. 141–152, Jun. 2010.

[45] C. Xu, D. Niu, Y. Zheng, S. Yu, and Y. Xie, "Impact of cell failure on reliable cross-point resistive memory design," *ACM Trans. Design Autom. Electron. Syst. (TODAES)*, vol. 20, no. 4, p. 63, 2015.

[46] Y. Zheng, C. Xu, and Y. Xie, "Modeling framework for cross-point resistive memory design emphasizing reliability and variability issues," in *Proc. 20th Asia South Pacific Design Autom. Conf.*, Jan. 2015, pp. 112–117.

[47] W. Wen, M. Mao, X. Zhu, S. H. Kang, D. Wang, and Y. Chen, "CD-ECC: Content-dependent error correction codes for combating asymmetric nonvolatile memory operation errors," in *Proc. IEEE/ACM Int. Conf. Comput.-Aided Design (ICCAD)*, Nov. 2013, pp. 1–8.

[48] 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 *Proc. 43rd Annu. IEEE/ACM Int. Symp. Microarchitecture*, Dec. 2010, pp. 115–124.

[49] E. Ipek, J. Condit, E. B. Nightingale, D. Burger, and T. Moscibroda, "Dynamically replicated memory: Building reliable systems from nanoscale resistive memories," *ACM SIGARCH Comput. Archit. News*, vol. 38, no. 1, pp. 3–14, Mar. 2010.

[50] N. H. Seong, S. Yeo, and H.-H.-S. Lee, "Tri-level-cell phase change memory: Toward an efficient and reliable memory system," in *Proc. 40th Annu. Int. Symp. Comput. Archit.*, Jun. 2013, pp. 440–451.

[51] M. Stanisavljevic, H. Pozidis, A. Athmanathan, N. Papandreou, T. Mittelholzer, and E. Eleftheriou, "Demonstration of reliable triple-level-cell (TLC) phase-change memory," in *Proc. IEEE 8th Int. Memory Workshop (IMW)*, May 2016, pp. 1–4.

[52] L. Chen, Y. Cao, and Z. Zhang, "Free ECC: An efficient error protection for compressed last-level caches," in *Proc. IEEE 31st Int. Conf. Comput. Design (ICCD)*, Oct. 2013, pp. 278–285.

[53] D. J. Palframan, N. S. Kim, and M. H. Lipasti, "COP: To compress and protect main memory," in *Proc. 42nd Annu. Int. Symp. Comput. Archit.*, Jun. 2015, pp. 682–693.

[54] L. Zhao, L. Jiang, Y. Zhang, N. Xiao, and J. Yang, "Constructing fast and energy efficient 1TnR based ReRAM crossbar memory," in *Proc. 18th Int. Symp. Quality Electron. Design (ISQED)*, Mar. 2017, pp. 58–64.

[55] M. Shevgoor, N. Muralimanohar, R. Balasubramonian, and Y. Jeon, "Improving memristor memory with sneak current sharing," in *Proc. 33rd IEEE Int. Conf. Comput. Design (ICCD)*, Oct. 2015, pp. 549–556.

[56] S. Cho and H. Lee, "Flip-N-write: A simple deterministic technique to improve PRAM write performance, energy and endurance," in *Proc. 42nd Annu. IEEE/ACM Int. Symp. Microarchitecture (Micro)*, 2009, pp. 347–357.

[57] A. N. Jacobvitz, R. Calderbank, and D. J. Sorin, "Coset coding to extend the lifetime of memory," in *Proc. IEEE 19th Int. Symp. High Perform. Comput. Archit. (HPCA)*, Feb. 2013, pp. 222–233.

YANG ZHANG received the B.E. and Ph.D. degrees in computer science and technology from the Huazhong University of Science and Technology, in 2014 and 2019, respectively. He is currently a Postdoctoral Researcher with the Shenzhen Institute of Advanced Technology, Chinese Academy of Sciences, Shenzhen, China. He is also a Research and Development Expert of cloud computing technology at Sangfor Technologies Inc., Shenzhen. His research interests include computer architecture, cloud computing, distributed storage systems, non-volatile memory, and intelligent storage technology. His research works have been published in DAC, MSST, ICCAD, ICCD, *ACM TACO*, *ACM TODAES*, IEEE Transactions on Computers, IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on Electron Devices, and *FGCS*. He has served as a Reviewer for IEEE International Symposium on Circuit and Systems.

ZHIBIN YU (Member, IEEE) received the M.S. degree in mechanical and electronical engineering, and the Ph.D. degree in computer architecture from the Huazhong University of Science and Technology, Wuhan, China, in 2000 and 2008, respectively. He is a Professor with the Shenzhen Institute of Advanced Technology, Chinese Academy of Sciences. His current research interests include computer architecture, big data, and fog computing.

LIANG GU received the Ph.D. degree in computer software and theory from Peking University, in 2010. He worked as an Associate Research Fellow at Yale University, from 2010 to 2015. He is currently the Chief Scientist and the Director of the Sangfor Research Institute, Sangfor Technologies Inc. As the person in charge of research and development technology at Sangfor, he is responsible for the technical framework improvement of a series of core products, including NGAF, AC, a Cloud HCI, and aSAN.

CHENGNING WANG received the Ph.D. degree from the Wuhan National Laboratory for Optoelectronics, Huazhong University of Science and Technology, China, in June 2021. He is currently a Postdoctoral Researcher with the Wuhan National Laboratory for Optoelectronics, Huazhong University of Science and Technology. His research interests include modeling, design, analysis, and co-optimization of high-density memristive nanodevices and arrays, three-dimensional integrated memristive systems, and analog parallel computation-in-memory for novel applications. His research works have been published in IEEE Transactions on Electron Devices, IEEE Transactions on Computers, IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, *ACM TODAES* (review article), ICCD, and NVMW. He served as a Technical Program Committee Member for the Emerging Device Technologies of the 59th Design Automation Conference (DAC). He was invited and served as a Reviewer for the *Journal of Applied Physics* (AIP), IEEE Transactions on Very Large Scale Integration (VLSI) Systems, IEEE Transactions on Computers, IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, and *Frontiers of Information Technology & Electronic Engineering*.

DAN FENG (Senior Member, IEEE) received the B.E., M.E., and Ph.D. degrees in computer science and technology from the Huazhong University of Science and Technology (HUST), Wuhan, China, in 1991, 1994, and 1997, respectively. She is a Professor and the Dean of the School of Computer Science and Technology, HUST. She has more than 100 publications in major journals and international conferences, including IEEE Transactions on Computers, IEEE Transactions on Parallel and Distributed Systems, IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, *ACM TOS*, FAST, USENIX ATC, EuroSys, ICDCS, HPDC, SC, ICS, IPDPS, DAC, and DATE. Her research interests include computer architecture, non-volatile memory technology, distributed and parallel file systems, and massive storage systems. She is a member of ACM and the Chair of Information Storage Technology Committee of Chinese Computer Academy. She has served on the program committees of multiple international conferences, including SC 2011 and 2013, and MSST 2012 and 2015.