TLC STT-MRAM Aware LLC for Multicore Processor

Taejin Park¹, Jae Young Hur², and Wooyoung Jang¹a

Abstract Since state-of-the-art multicore processors that execute complicated applications demand a large last-level cache (LLC) for reducing memory latency, next-generation memories have been recently attracted great attentions. Triple-level cell (TLC) spin-transfer torque (STT)-magnetic random access memories (MRAMs) provide high storage density, but degrade latency and power consumption due to three-step resistance state transition and detection processes for write and read operations, respectively. In this paper, we propose a TLC STT-MRAMs aware LLC that limit such penalties for multicore processors. Our LLC minimizes the occurrence of three-step resistance state transition and detection processes via the proposed cell division mapping and conditional block swapping techniques. Experimental results show that the proposed LLC achieves on average 17.2% higher performance and 17.9% lower power consumption than conventional LLCs comprised of TLC STT-MRAMs.

key words: Memory, cache, multicore processor
Classification: Integrated circuits (memory, logic, analog, RF, sensor)

1. Introduction

With the advent of a variety of artificial intelligence, virtual/augmented reality, and ultra high-quality multimedia applications, there is an increasing demand for a last-level cache (LLC) that provides high capacity, high performance, and low power consumption for state-of-the-art multicore processors [1, 2, 3, 4]. On the contrary, static random access memories (SRAMs) used for implementing caches are approaching their scaling limit [5, 6]. Thus, next-generation non-volatile memories such as a phase-change random access memory (PRAM) [7, 25], a resistive random access memory (ReRAM) [9, 10], and a spin-transfer torque (STT)-magnetic random access memory (MRAM) [11, 12] are being researched for multicore processors. Among them, spin-transfer torque (STT)-MRAMs are taken into consideration as the most promising SRAM alternative for designing such LLCs since they show attractive results towards performance, endurance and power consumption [13]. However, STT-MRAMs have lower storage density than other non-volatile memories. Recently, triple-level cell (TLC) technologies were introduced in STT-MRAMs, and thus, STT-MRAMs enable a large LLC to be implemented at a lower cost [14, 15, 16]. TLCs can be structured in parallel or in series. Series TLC STT-MRAMs require a larger cell area, and a higher switching current, but they are easier to be fabricated and more tolerant of errors than parallel TLC STT-MRAMs. In this paper, our ideas are described in series STT-MRAMs.

Fig. 1(a) shows the cell structure of series STT-MRAMs. Each magnetic tunneling junction (MTJ) consists of two ferromagnetic layers separated by an oxide barrier (MgO) [11, 12]. The magnetization direction of a free layer is altered by passing a current whereas the magnetization direction of a reference layer is fixed. When two ferromagnetic layers of an MTJ have the same magnetization direction, the MTJ has a high resistance, and it represents logic ‘1’. On the contrary, when two ferromagnetic layers of an MTJ have the opposite magnetization directions, the MTJ has a low resistance, and it represents logic ‘0’. On a write operation, a voltage inducing a current across an MTJ is applied between source and bit lines to switch the magnetization direction of a free layer. On a read operation, a small sensing current is applied to induce a bit line voltage. Then, the bit line voltage is compared to a reference voltage to decide whether logic ‘0’ or logic ‘1’ is stored in an MTJ.

Each series TLC MTJ can store a 3-bit data. Fig. 1(b) shows the state transition of the series TLC MTJ for write operation. The series TLC MTJ has eight resistance states: $[hms]$, where $h$, $m$, and $s$ denote the state of hard, medium, and soft domains, respectively. A switching current for hard domains ($I_{\text{high}}$) is higher than a switching current for medium domains ($I_{\text{medium}}$), and $I_{\text{medium}}$ is higher than a switching current for soft domains ($I_{\text{low}}$). Thus, $I_{\text{high}}$ can switch all domains, $I_{\text{medium}}$ can switch the medium and soft domains,
and \( I_{\text{low}} \) can switch only soft domains.

Such a cell structure makes the writing of 3-bit data considerably complicated. For example, in a series TLC MTJ, state \([010]\) can be switched into state \([101]\) by a three-step resistance state transition process as shown in Fig. 1(b). That is, state \([010]\) is first switched into state \([111]\) by \( I_{\text{high}} \), then state \([111]\) is switched into state \([100]\) by \( I_{\text{medium}} \), and finally state \([100]\) is switched into state \([101]\) by \( I_{\text{low}} \). In addition, a series TLC MTJ has a long read latency due to a three-step resistance state detection process. The state of a soft domain is first detected in comparison with the reference voltage of the soft domain (Ref-S), then the state of a medium domain is detected in comparison with the reference voltages of the medium domain (Ref-M0 and M1), and finally the state of a hard domain is detected in comparison with the reference voltages of the hard domain (Ref-H0, H1, H2, and H3), as shown in Fig. 1(c). Such complex resistance state transition and detection processes for write and read operations, respectively, seriously degrade performance, power consumption, and endurance. Previous works have mitigated the degradation of multi-level cell (MLC) STT-MRAMs \([12, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27]\), but they are not effective for a large LLC comprised of series TLC STT-MRAMs.

In this paper, we propose the architecture and operation of a novel LLC comprised of series TLC STT-MRAMs. Our LLC minimizes the occurrence of three-step resistance state transition and detection processes for write and read operations, respectively, via proposed cell division mapping and conditional block swapping techniques. The cell division mapping technique composes LLC blocks with the same domain of series TLC MTJs, and the conditional block swapping technique stores more used data in LLC blocks comprised of soft domains and less used data in LLC blocks comprised of hard domains. Since the proposed techniques make three-step resistance state transition and detection processes less required, as a result, the performance, power, and endurance penalties of series TLC STT-MRAMs can be considerably mitigated in an LLC.

2. Cell Division Mapping Technique

Our LLC is structured by the proposed cell division mapping technique. Fig. 2 shows 3-way set-associative LLCs with series TLC STT-MRAMs in two different mapping manners. In Fig. 2(a), flags, tags, and data (or instructions) are mapped into TLC MTJs by the conventional sequential mapping technique which makes LLC blocks comprised of the hard, medium, and soft domains of series TLC MTJs. Whenever the resultant LLC blocks are accessed for writing and reading operations, thus, all domains of the TLC MTJs must be always accessed by three-step resistance state transition and detection processes, respectively. Furthermore, in the case that just data in hard domains need to be updated, the original data in medium and soft domains must be read via a two-step resistance state detection process, and then rewritten together with data in hard domains via a three-step resistance state transition process. The reason is that a switching current for hard domains is high enough to remove data in medium and soft domains. Similarly, just data in medium domains can be updated via one-step resistance state detection and two-step resistance state transition processes. Thus, LLCs to which TLC STT-MRAMs are mapped via such a sequential mapping technique can significantly degrade performance, power consumption, and endurance.

Fig. 2(b) shows the proposed cell division mapping technique that makes LLC blocks comprised of the same domain of series TLC MTJs. If data are written to LLC block \( s \) mapped to soft domains, the writing operation can be simply performed by a one-step resistance state transition process. The reason is that a switching current for LLC block \( s \) is too low to change data in LLC blocks \( m \) and \( h \) just comprised of medium and hard domains, respectively. In addition, LLC block \( s \) can be accessed for a read operation by a one-step resistance state detection process since the state of soft domains is detected earliest in TLC MTJs as shown in Fig. 1(c).

On the contrary, our cell division mapping technique makes LLC blocks \( m \) and \( h \) accessed via multi-step resistance state transition and detection processes. For example, data are written to LLC block \( m \) via one-step resistance detection and then two-step resistance state transition processes in Fig. 2(b). Since a switching current for LLC block \( m \) is high enough to remove data in LLC block \( s \), the data in LLC block \( s \) should be read, and then rewritten together with data in medium domains. In addition, data in LLC block \( m \) can be read after reading data in LLC block \( s \) via a two-step resistance state detection process. Similarly, data are be written to LLC block \( h \) by two-step resistance state detection and then three-step resistance state transition processes. This is because a switching current for LLC block \( h \) removes data in LLC blocks \( s \) and \( m \), and thus the data in LLC blocks \( m \) and \( s \) should be restored. In addition, data in LLC block \( h \) can be read after reading data in LLC blocks \( s \) and \( m \) via a
three-step resistance state detection process. Our LLC blocks $s$ and $m$ are accessed with shorter latency than LLC blocks constructed by the sequential mapping technique. On the contrary, our LLC block $h$ may be accessed with similar latency to LLC blocks constructed by the sequential mapping technique. If there are additional techniques that make LLC blocks $s$ and $m$ more accessed than LLC blocks $h$, the performance, power consumption, and endurance gains by the proposed cell division mapping technique can greatly increase. Therefore, in the next section, we present our conditional block swapping technique that makes LLC blocks $s$ and $m$ more accessed, and makes LLC block $h$ less accessed.

3. Conditional Block Swapping Technique

When a pseudo least recently used (LRU) algorithm evicts data (or instructions) with low temporal locality from LLC block $h$ data in an upper-level cache need to be updated to LLC block $h$, data (or instructions) in corresponding LLC blocks $m$ and $s$ should be restored. That is, (or instructions) in LLC blocks $s$ and $m$ that were constructed with the same TLC MTJs as LLC block $h$ are read, and then incoming data (or instructions) and read data (or instructions) are written in the LLC blocks $h$, $m$, and $s$. However, in such a restoration process, it is not always required to write all the data to the original LLC blocks. If data likely to be updated from a upper-level cache to an LLC migrate to LLC block $s$ in advance, they can be rapidly updated via a one-step resistance state transition process when that actually happens. Our conditional block swapping technique ensures that such data have a high priority and are written in the order of LLC blocks comprised of soft, medium, and hard domains. Similarly, when data (or instructions) in LLC block $m$ are selected as a victim or data are updated from a upper-level cache, data that come into an LLC and that are originally in LLC block $s$ can be written to LLC block $s$ or $m$, depending on their priorities. On the contrary, if data (or instructions) in LLC block $s$ are evicted from an upper-level cache, incoming data are written just to LLC block $s$.

Table I shows the priority of data and instructions in our LLC. Data in an LLC can be copied or updated from (or to) a upper-level cache and a main memory whereas instructions in an LLC are copied from a main memory and to a upper-level cache. The priorities of data in our LLC are determined by whether the copies of data and instructions are in a upper-level cache, and whether dirty and reference bits in a upper-level cache are set. A dirty bit is set when data corresponding to the dirty bit in a upper-level cache are updated, and a reference bit is set when data corresponding to the reference bit in a upper-level cache is accessed within a given period. In Table I, data which are in not only an LLC, but also a upper-level cache and of which dirty and reference bits are set and reset, respectively have the highest priority. The reason is that data with the set dirty and reset reference bits are likely to be soon evicted from a upper-level cache and updated to an LLC. Next, data which are in both an LLC and a upper-level cache and of which the dirty and reference bits are set have the second highest priority. Since data with the set reference bit are likely to continue to be used in a upper-level cache, they have lower possibility to be evicted from a upper-level cache and updated to an LLC. In the case that data which are both an LLC and a upper-level cache and of which the dirty bit is reset, the data with set and reset reference bits have the third and fourth priorities, respectively. Since data with a reset dirty bit are not required to be updated to an LLC and data with a set reference bit are more read, the data with a set reference bit have a higher priority than the data with a reset reference bit. When data in an LLC are not in a upper-level cache, lastly, they have the lowest priorities regardless of dirty and reference bits.

On the contrary, a cache for instructions does not need a dirty bit since instructions in the cache are not modified during execution. Thus, the priority of instructions starts at 3, and depends on whether instructions in an LLC are in a upper-level cache and whether the reference bit of instructions is set. Unlike the priorities 3 and 4 of data, when data which are in both an LLC and a upper-level cache, the data with reset and set reference bits have the third and fourth priorities. The reason is that instructions that are not recently used may be evicted from a upper-level cache and read from an LLC to a upper-level cache again. On the contrary, since instructions that are recently used are unlikely to be evicted from a upper-level cache, it is not necessary to read them from an LLC. Data (or instructions) with a high priority are written in the order of LLC blocks comprised of soft, medium, and hard domain. Therefore, the proposed conditional block swapping technique greatly reduces the occurrence of complex multi-step resistance state transition and detection processes in our LLC.

4. Experimental Results

We use MARSSx86 [28] and DRAMSim2 [29] simulators for evaluating our TLC STT-MRAM aware LLC for multicore Processors. MARSSx86 is a cycle-accurate full system simulator for multicore x86 CPUs, and DRAMSim2 is a cycle-accurate memory system simulator for double data rate (DDR) 2/3/4 SDRAMs. They are configured as shown in Table II. A level 2 (L2) cache with 6-way associativity is used as an LLC comprised of a series TLC STT-MRAM. Table III shows the parameters of a series TLC STT-MRAM calculated with an NVSim [30] simulator that is a circuit-
Table II. Configuration of processor, cache, and memory

| Processor | 2.4GHz, quad-core, out-of-order, 4-issue per core |
|-----------|-----------------------------------------------|
| L1 I- and D-cache | 16KB SRAM, 64B block size, 4-way associativity, write-back, LRU |
| L2 shared cache | 3MB series TLC STT-MRAM, 64B block size, 6-way associativity, write-back, pseudo-LRU |
| Main Memory | DDR3-1600 SDRAM, 1 channel, 2 ranks, 8 banks per rank, 32K rows, 2K columns |

Table III. Simulation parameters of series STT-MRAM (*a* soft domain, *b* medium domain, *c* hard domain).

| Parameter                  | S*a* | M*b* | H*c* |
|---------------------------|------|------|------|
| Cell size                 | 36F  | 4.5F |      |
| Read latency (cycle)      | 8    | 5    | 3    |
| Write latency (cycle)     | 19   | 41   | 66   |
| Read energy (pJ/bit)      | 0.828| 1.125| 1.422|
| Write energy (pJ/bit)     | 2.111| 5.176| 8.662|
| Leakage (mW)              | 121.2|      |      |

level performance, energy, and area model for emerging non-volatile memory. We use PARSEC benchmarks [31] to conduct evaluations for 100 million clock cycles, and compare LLCs adopting our cell division mapping (CDM) technique alone and our CDM and conditional block swapping (CBS) techniques with an LLC adopting a sequential mapping (SM) technique.

Fig. 3 shows instructions per cycle (IPC) comparison that is normalized by the IPC of a processor with an LLC adopting the SM technique. An LLC adopting the proposed CDM technique alone achieves, on average, 6%, 10.1%, and 13.3% higher IPC than an LLC adopting the SM technique when one, two, and four benchmarks, respectively are executed simultaneously. This improvement results from our LLC blocks comprised just of soft or medium domains. The LLC blocks are accessed via one-step and two-step resistance state transition and detection processes whereas LLC blocks constructed by the SM technique are always accessed via three-step resistance state transition and detection processes. Moreover, our LLC adopting both CDM and CBS techniques achieves, on average, 8%, 13.3%, and 17.2% higher IPC than an LLC adopting the SM technique in the case that one, two, and four benchmarks, respectively are executed at the same time. Our CBS technique migrates data that are less accessed in an LLC into LLC blocks comprised of hard domains in advance, and thus reduces the occurrence of three-step resistance state transition and detection processes for LLC blocks comprised of hard domains. In addition, our LLC shows higher performance as more applications are concurrently executed in a multicore processor. The reason is that a long LLC access for one application resulting from three-step resistance state transition and detection processes delays subsequent LLC accesses for other applications.

Fig. 4 shows power consumption comparison that is normalized by power consumed by an LLC adopting the SM technique. An LLC adopting our CDM technique alone consumes, on average, 6.7%, 10.5%, and 14.1% lower power than an LLC adopting the SM technique when one, two, and four benchmarks, respectively are executed simultaneously. In addition, the proposed LLC adopting both CDM and CBS techniques consumes 8.9%, 13.9%, and 17.9% lower power when one, two, and four benchmarks, respectively are executed at the same time. The reason is that low and medium currents for switching soft and medium domains, respec-

![Fig. 3. Performance comparison. (a) One benchmark. (b) Two benchmarks. (c) Four benchmarks.](image1)

![Fig. 4. Power consumption comparison. (a) One benchmark. (b) Two benchmarks. (c) Four benchmarks.](image2)
tively are more used in our LLC. On the contrary, LLC blocks constructed by the SM technique always need a high current for switching hard domains. Furthermore, we expect our LLC consumes much lower power than an LLC adopting the SM technique in the case that more applications are concurrently executed in a multicore processor.

5. Conclusion

As the demand of artificial intelligence and high quality services has dramatically increased, recently, LLCs with large capacity have been actively researched. A series TLC STT-MRAM that is one of the alternatives to SRAM needs long and complex three-step resistance state transition and detection processes for write and read operations due to its cell structure. The proposed cell division mapping and conditional selection block swapping techniques make such a three-step process less occur in an LLC comprised of series TLC STT-MRAMs, and thus achieve higher performance and lower power consumption. In addition, our techniques will be easily extended for quad-level cell STT-MRAMS and beyond.

Acknowledgments

The present research was conducted by the research fund of Dankook University in 2019.

References

[1] Y. Kwon, et al.: ‘A Disaggregated Memory System for Deep Learning,’ IEEE Micro, 2019, 39, (5), pp. 82-90, doi: 10.1109/MM.2019.2929165.
[2] D. Chatzopoulos, et al.: ‘Mobile Augmented Reality Survey: From Where We Are to Where We Go,’ IEEE Access, 2017, 5, pp. 6917-6950, doi: 10.1109/ACCESS.2017.2698164.
[3] G. Pastuszak and A. Abramowski: ‘Algorithm and Architecture Design of the H.265/HEVC Intra Encoder,’ IEEE Transactions on Circuits and Systems for Video Technology, 2016, 26, (1), pp. 210-222, doi: 10.1109/TCSVT.2015.2428571.
[4] W. Jang: ‘Screen Orientation Aware DRAM Architecture for Mobile Video and Graphic Applications,’ IEEE Transactions on Circuits and Systems for Video Technology, 2018, 28, (12), pp. 3527-3538, doi: 10.1109/TCSVT.2017.2715062.
[5] B. M. Rogers, et al.: ‘ Scaling the bandwidth wall: challenges in and avenues for CMP scaling,’ ACM SIGARCH Computer Architecture News, 2009, 37, (30), pp. 371-382, doi: 10.1145/1555815.1555801.
[6] ITRS: ‘ International technology roadmap for semiconductors,’ http://www.itrs2.net, accessed July 2019.
[7] M. Wuttig, et al.: ‘Phase-change materials for rewritable data storage,’ Nature materials, 2007, 6, (11), pp. 824-838, doi: 10.1038/nmat2009.
[8] W. Jang: ‘Enhancing lifetime of phase-change memory for video processor,’ IEICE Electronics Express, 2017, 14, (11), pp. 20170402, doi: 10.1587/elecf.14.20170402.
[9] H. Akinaga and H. Shima: ‘Resistive Random Access Memory (ReRAM) Based on Metal Oxides,’ Proceedings of the IEEE, 2010, 98, (12), pp. 2237-2251, doi: 10.1109/JPROC.2010.2070830.
[10] H.-S. P. Wong, et al.: ‘Metal–oxide rram’, Proceedings of the IEEE, 2012, 100, (6), pp. 1951-1970, doi: 10.1109/JPROC.2012.2190369.
[11] C. Chappert, et al.: ‘The emergence of spin electronics in data storage’, Nature materials, 2007, 6, (11), pp. 813-823, doi: 10.1038/nmat2024.
[12] W. Jang: ‘MLC STT-MRAM aware memory sub-system for smart image applications’, IEEE Transactions on Multimedia, 2020, 22, (3), pp. 717-729, doi: 10.1109/TMM.2019.2930342.
[13] M. Chang, et al.: ‘ Technology comparison for large last-level caches (L3C): Low-leakage SRAM, low write-energy STT-RAM, and refresh-optimized eDRAM,’ IEEE International Symposium on High Performance Computer Architecture (HPCA), 2013, pp. 143-154, doi: 10.1109/HPCA.2013.6522314.
[14] Y. Xu, et al.: ‘Write-Efficient STT/SOT Hybrid Triple-Level Cell for High-Density MRAM,’ IEEE Transactions on Electron Devices, 2020, 67, (4), pp. 1460-1465, doi: 10.1109/TED.2019.2963421.
[15] S. Swami and K. Mohanram: ‘E3R: Energy Efficient Error Recovery for Multi/Triple-Level Cell Non-volatile Memories,’ International Conference on VLSI Design and International Conference on Embedded Systems (VLSID), 2016, pp. 373-378, doi: 10.1109/VLSID.2016.33.
[16] A. Alsuwaiyan, and K. Mohanram: ‘An offline frequent value encoding for energy-efficient MLC/TLC non-volatile memories,’ Proceedings of the 26th edition on Great Lakes Symposium on VLSI, 2016, pp. 403-408, doi: 10.1145/2902961.2902979.
[17] T. Ishigaki, et al.: ‘A multi-level-cell spin-transfer torque memory with series-stacked magnetotunnel junctions,’ Symposium on VLSI Technology, 2010, pp. 47-48, doi: 10.1109/ISVLSI.2010.5556126.
[18] Y. Chen, et al.: ‘Processor caches built using multi-level spin-transfer torque RAM cells,’ IEEE/ACM International Symposium on Low Power Electronics and Design, 2011, pp. 73-78, doi: 10.1109/ISLPED.2011.5993610.
[19] L. Jiang, et al.: ‘Constructing large and fast multi-level cell STT-MRAM based cache for embedded processors,’ ACM/IEEE Design Automation Conference, 2012, pp. 907-912, doi: 10.1145/2228360.2228521.
[20] P. Chi, et al.: ‘ Building energy-efficient multi-level cell STT-MRAM based cache through dynamic data-resistance encoding,’ International Symposium on Quality Electronic Design, 2014, pp. 639-644, doi: 10.1109/ISQED.2014.6783387.
[21] J. Wang, et al.: ‘Optimizing MLC-based STT-RAM caches by dynamic block size reconfiguration,’ IEEE International Conference on Computer Design (ICCD), 2014, pp. 133-138, doi: 10.1109/ICCD.2014.6974672.
[22] S. Hong, et al.: ‘Ternary cache: Three-valued MLC STT-RAM caches’, IEEE International Conference on Computer Design, Seoul, Oct. 2014, pp. 83-89, doi: 10.1109/ICCD.2014.6974666.
[23] Z. Li, et al.: ‘A new self-reference sensing scheme for TLC MRAM’, IEEE International Symposium on Circuits and Systems, Lisbon, Portugal, May 2015, pp. 593-596, doi: 10.1109/ISCAS.2015.7168703.
[24] H. Luo, et al.: ‘Two-step state transition minimization for lifetime and performance improvement on MLC STT-RAM,’ ACM/IEEE Design Automation Conference (DAC), 2016, pp. 1-6, doi: 10.1145/2897937.2898106.
[25] W. Jang: ‘Multi-level cell STT-RAM controller for multimedia applications,’ IET Electronics Letters, 2017, 53, (1), pp. 12-14, doi: 10.1049/el.2016.2435.
[26] X. Chen, et al.: ‘Energy-aware adaptive restore schemes for MLC STT-RAM cache,’ IEEE Transactions on Computers, 2017, 66, (5), pp. 786-798, doi: 10.1109/TC.2016.2625245.
[27] M. A. Qureshi, et al. : ‘A restore-free mode for MLC STT-RAM caches,” IEEE Transactions on Very Large Scale Integration (VLSI) Systems, 2019, 27, (6), pp. 1465-1469, doi: 10.1109/TVLSI.2019.2899894.

[28] A. Patel, et al. : ‘MARSS: a full system simulator for multicore x86 CPUs’, ACM/IEEE Design Automation Conference, San Diego, CA, USA, June 2011, pp. 1050-1055, doi: 10.1145/2024724.2024954.

[29] P. Rosenfeld, et al. : ‘DRAMSim2: A Cycle Accurate Memory System Simulator’, Computer Architecture Letters, 2011, 10, (1), pp. 16-19, doi: 10.1109/L-CA.2011.4.

[30] X. Dong, et al. : ‘NVSim: A Circuit-Level Performance, Energy, and Area Model for Emerging Nonvolatile Memory’, IEEE Trans. Computer-Aided Design of Integrated Circuits and Systems, 2012, 31, (7), pp. 994-1007, doi: 10.1109/TCAD.2012.2185930.

[31] C. Bienia, et al. , “The PARSEC benchmark suite; Characterization and architectural implications,” IEEE International Conference on Parallel Architectures and Compilation Techniques, Toronto, Ontario, Canada, Oct. 2008, pp. 72-81, doi: 10.1145/1454115.1454128.