Open access • Proceedings Article • DOI:10.1109/ISLPED.2011.5993610

Processor caches built using multi-level spin-transfer torque RAM cells

Yi Chen, Weng-Fai Wong, Hai Li, Cheng-Kok Koh

Institutions: University of Pittsburgh, National University of Singapore, New York University, Purdue University

Published on: 22 Aug 2011 - International Symposium on Low Power Electronics and Design

Topics: Tag RAM, Cache and Static random-access memory

Processor Caches Built Using Multi-Level Spin-Transfer Torque RAM Cells

Yiran Chen
Electrical and Computer Engineering Department
University of Pittsburgh
Pittsburgh, PA 15216 USA
Email: yic52@pitt.edu

Weng-Fai Wong
School of Computing
National University of Singapore
Singapore 117417
Republic of Singapore
Email: wongwf@nus.edu.sg

Hai (Helen) Li
Department of Electrical and Computer Engineering
Polytechnic Institute of NYU
Brooklyn, NY 11201 USA
Email: hli@poly.edu

Cheng-Kok Koh
School of Electrical and Computer Engineering
Purdue University
West Lafayette, IN 47907 USA
Email: chengkok@ecn.purdue.edu

***Abstract*—It has been predicted that a processor's caches could occupy as much as 90% of chip area for technology nodes from the current. In this paper, we study the use of multi-level spin-transfer torque RAM (STT-RAM) cells in the design of processor caches. Compared to the traditional SRAM caches, a multi-level cell (MLC) STT-RAM cache design is denser, fast, and consumes less energy. However, a number of critical issues remains to be solved before MLC STT-RAM technology can be deployed in processor caches. In this paper, we shall offer solutions to the issue of bit encoding as well as tackle the write endurance problem. The latter has been neglected in previous works on STT-RAM caches. We propose a set remapping scheme that can potentially prolong the lifetime of a MLC STT-RAM cache by $80\times$ on average. Furthermore, a method for recovering the performance that may be lost in some applications due to set remapping is introduced.**

***Index Terms*—STT-RAM, spintronic, MLC.**

I. Introduction

Fast on-chip cache, traditionally implemented by SRAM technology, is predicted to occupy as much as 90% of the die's real-estate in the scaled technologies [1]. However, the large leakage power and the degraded reliability due to the process variations severely limits the scalability of SRAM. Therefore, there are active attempts to find technological alternatives to SRAM.

Spin-transfer torque RAM (STT-RAM), which belongs to the class of magneto-resistive RAM (MRAM), is an emerging memory technology that has received increasing attention from the solid state device and circuit communities [2]. Unlike SRAM or DRAM, STT-RAM is a nonvolatile memory technology that consumes no standby power, and has nanosecond access times [3]. Furthermore, its cell size is less than one tenth that of SRAM's, making it ideal especially for embedded applications [4][5][6].

Compared to SRAM caches, STT-RAM ones have relatively long write latency and large write energy. However, significant power saving can still be achieved by eliminating the leakage power consumption of the memory array [5]. Also, because of a smaller cell area, STT-RAM caches may have a shorter read latency at the same capacity, or a larger capacity with the same die area. Thus, integrating STT-RAM atop of microprocessors by 3D stacking technology becomes a promising solution to achieve both large cache capacity and low power simultaneously [5][6].

The use of *multi-level cell* (MLC) STT-RAM technology to maximize the density benefits of STT-RAM caches in microprocessors was first introduced in [7]. With a cell size of about $12.9F^2$ for two bits of data, where F is the feature size of the technology node, MLC promises a $1.5\times$ increase in data density compared to *single level cell* (SLC) STT-RAM. Both the energy efficiency of the cache as well as the performance of the microprocessor can be further improved by the proper application of MLC STT-RAM technology.

A number of circuit and architectural challenges were left unsolved in [7], including the bit encoding scheme, the architecture of a MLC STT-RAM cache, and the issue of *write endurance*. In particular, the latter has been largely ignored by the earlier works on SLC STT-RAM caches. Although a prediction of up to 10^{15} programming cycles is often cited by many papers, the best endurance test result for STT-RAM devices so far is less than 4×10^{12} cycles [8]. In MLC STT-RAM, the largest write current is about 23% higher than that of SLC STT-RAM, and exponentially degrades the lifetime of memory cells as dielectric breakdown [9]. This problem is further aggravated by the unbalanced write accesses to caches. In this paper, we proposed an architectural wear-leveling technique that significantly prolongs the lifetime of STT-RAM. This is further backed up by a refinement that partially recovers performance lost in the wear leveling.

This paper makes three key contributions: 1) we propose an architecture for using MLC STT-RAM technology in on-chip caches and demonstrate its great potentials in energy efficiency and die area; 2) we discussed several critical circuit issues of MLC STT-RAM cache design; 3) we proposed an architectural solution that improves the endurance of MLC STT-RAM cache by $80\times$ on average, with minimized performance overheads.

II. Multi-level cell MTJ

A. SLC MTJ vs. MLC MTJ

The key component of STT-RAM is *magnetic tunnel junction* (MTJ), as shown in Fig. 1(a). An MTJ is composed of two ferromagnetic layers that are separated by an oxide barrier layer (e.g., MgO). The magnetization direction of one

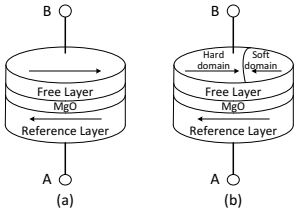

Fig. 1. MTJ structure. (a) SLC MTJ (b) MLC MTJ.

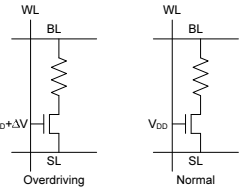

Fig. 2. Write operations of 1T1J MLC STT-RAM cell.

ferromagnetic layer (*reference layer*) is fixed while that of the other ferromagnetic layer (*free layer*) can be changed by passing a current that is polarized by the magnetization of the reference layer. In an SLC MTJ, when the magnetization directions of the free layer and reference layer are parallel (anti-parallel), MTJ is in low (high) resistance state.

Fig. 2 shows the most popular one-transistor-one-MTJ (or 1T1J) STT-RAM cell structure where MTJ is selected by turning on the wordline (WL). When writing "1" (high-resistance state) to STT-RAM cell, a high voltage is applied to sourceline (SL) while bitline (BL) is connected to ground. When writing "0" (low resistance state), a high voltage is applied to BL while SL is connected to ground. In a read operation of the STT-RAM cell, a read current is injected and generate the corresponding BL voltage $V_{\rm BL}$. The resistance state of the MTJ can be readout by comparing $V_{\rm BL}$ to an reference voltage.

Fig. 1(b) shows a recently proposed four-level (2-bit) MTJ device [10]. The free layer has two magnetic domains with different magnetic properties. The magnetization direction of one domain (*soft domain*) can be switched by a small current while that of the other domain (*hard domain*) can be switched only by a larger current. Four combinations of the magnetization directions of the two domains correspond to the four resistance states of MTJ. In particular, the first and second bits of a 2-bit data can be defined by the magnetization directions of the hard and soft domain, respectively.

B. Operation of MLC STT-RAM

The read and write schemes of MLC STT-RAM cells were studied in [7]. "Binary-search" read, which sequentially reads out the first and then the second bit of a 2-bit data using two comparisons, was shown to be a good tradeoff between area and performance.

Writes to MLC STT-RAM cell are relatively more complex compared to SLC STT-RAM cells. Table I shows the switching currents required of a typical MLC MTJ within 10 ns at the 45nm technology node. This data was obtained by scaling that for a MTJ implemented with the 90nm process [10] considering that the switching current is proportional to the MTJ area.

TABLE I
SWITCHING CURRENTS FOR MLC STT-RAM STATE TRANSITIONS AT 45NM TECHNOLOGY NODE (IN μA).

From \ To	R00	R01	R10	R11
R00	0	-38.3	X	-56.7
R01	26.3	0	X	-56.7
R10	66.4	X	0	-9.1
R11	66.4	X	39.3	0

TABLE II
PARAMETERS OF STT-RAM CELLS.

		SLC	MLC
Normal	Min. NMOS width (nm)	174.4	715.0
$V_{\rm DD}$ = 1.0V, $V_{\rm WL}$ = 1.0V	Memory cell area (F^2)	14.6	50.7
Overdriving	Min. NMOS width (nm)	88.1	148.6
$V_{\rm DD}$ = 1.0V, $V_{\rm WL}$ = 1.2V	Memory cell area (F^2)	9.0	12.9

What is special about the write mechanism for MLC STT-RAM cells is that the magnetization directions of hard and soft domains cannot be flipped to two opposite directions simultaneously. Some state transactions, e.g., from state $R11$ to $R01$, has to be completed in two steps. First by going from $R11$ to $R00$ by applying a switching current of 66.4 μA. This is followed by a $R00$ to $R01$ transition through the application of a switching current of -38.3μA. Here the positive current direction denotes applying the current from the free layer to the reference layer. A transition that cannot be completed in one step is shown as "X".

III. MLC STT-RAM CACHES

A. Design of MLC STT-RAM Cells

In a typical 1T1J STT-RAM cell, the cell area is mainly constrained by the size of the NMOS transistor, which need to provide sufficient switching current to the MTJ. In a MLC STT-RAM cell, the largest switching current is 66.4μA, which happens at the transition from $R11$ (or $R10$) to $R00$. By comparison, the switching current of an SLC STT-RAM at 45nm technology node is 54μA after scaling from a 0.18μm STT-RAM design [5][6].

Interestingly, our simulation shows that the state transition that determines the size of the NMOS transistor is actually from $R01$ to $R11$. The voltage drop across the MTJ in the MLC design weakens the driving strength of NMOS transistor by reducing the voltage $V_{\rm GS}$ difference between the gate (G) and the source (S) when driving current from SL to BL. Because $R01 > R00$, the driving strength of NMOS transistor decreases more when driving the MTJ with $R01$ state than the case of $R00$. Table II shows the minimal size of NMOS and the corresponding memory cell area at 45nm technology node in the rows labeled "Normal". Here, the PTM model is used in the simulation [11] with power supply $V_{\rm DD}$=1.0V.

Overdriving the gate voltage of NMOS transistor can increase $V_{\rm GS}$ and consequently, enhance its driving strength [12]. In the switching step to flip the magnetization direction of both hard and soft domains, an overdriving voltage $V_{\rm DD}$+ΔV is applied. as shown in Fig. 2. The parameters of STT-RAM cells when the overdriving scheme is applied are shown in the row labeled "Overdriving" in Table II. The areas of both SLC and MLC STT-RAM cells are significantly reduced. Also, a

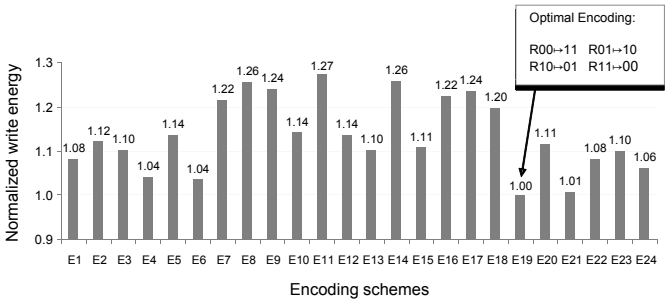

Fig. 3. Write energy comparison of a 16MB MLC STT-RAM cache under different encoding schemes on average.

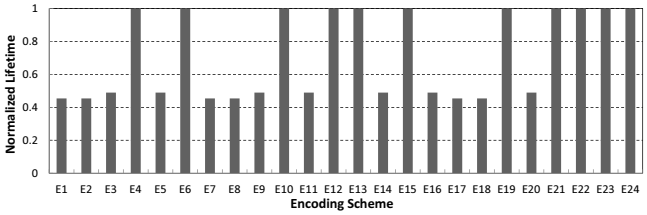

Fig. 4. Relative lifetime of a 16MB MLC STT-RAM cache with different encoding schemes for benchmark `434.zeusmp`.

MLC STT-RAM cell is about 43% larger than an SLC STT-RAM cell with 2× data density.

B. Resistance-Logic State Encoding

The four resistance states of a MLC MTJ – $R00$, $R01$, $R10$, and $R11$ from low to high, are not necessarily assigned to the combinations of two data bits – 00, 01, 10 and 11, respectively. In fact, there are total of $4! = 24$ encoding schemes to map the 4 resistance states to the four two-digits data – 00, 01, 10, and 11. Similar as the conclusion in [7], our simulation of a 16MB MLC STT-RAM shows that without considering the cases of staying at the same value, writes of data 00 contribute to about 37% of the total write operations. As shown in Table I, the transitions to the resistance state $R11$ requires smaller energy consumption than other states. Therefore, assigning $R11$ to 00 will be power efficient. Fig. 3 compares the normalized write energy under 24 different encoding schemes ($E_1 \sim E_{24}$) of a 16MB MLC cache on average for all benchmarks. We found that the optimal encoding is $\{R00 \mapsto 11,\ R01 \mapsto 10,\ R10 \mapsto 01,\ R11 \mapsto 00\}$. The write energy consumption can differ by as much as 27.5% among various encoding scheme. In the rest of the paper, we assumed this encoding scheme.

C. Endurance of MLC STT-RAM Cells

Although there is no known endurance issues of the magnetic materials, the lifetime of a MTJ is limited by the time dependent dielectric breakdown (TDDB) of the MgO layer. Based on the $1/E$-model, the TDDB time to failure (TTF) can be modeled as: ln(TTF) $\approx 1/E$, where E is the electric field applied. Compared to SLC STT-RAM, the increased switching current of the transitions from $R10$/$R11$ to $R00$ exponentially degrade the lifetime of the MTJ though the smaller switching current of other transitions alleviate the TDDB issues. Fig. 4 shows the simulated normalized lifetime of a 16MB MLC STT-RAM cache with different encoding schemes for benchmark `434.zeusmp`. The large lifetime improvement at

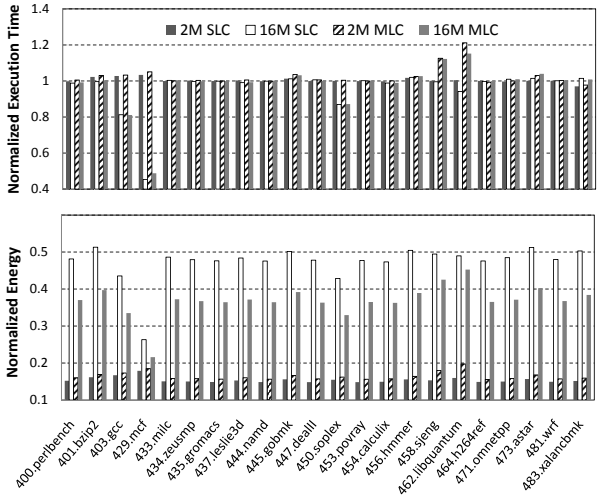

Fig. 5. Average execution time and energy normalized against 2MB SRAM cache operating at a temperature of 300K.

encoding schemes E19 etc. is due to the minimization of the state transitions that require large switching current, which is also the primary reason for the improvement in energy consumption of the same encoding scheme in Fig. 3.

The overdriving scheme does not affect the hot carrier effect of NMOS transistor because the V_{DS} remains the same as that in normal operation. However, the higher V_{GB} increases the potential on the gate oxide of NMOS transistor and may incur potential reliability issues, i.e., oxide TDDB. Considering that the probability of write accesses in memory cells are much lower than the switching probabilities of the transistors in logic block, the overall lifetime of NMOS in STT-RAM caches should not be the limiting factors of the system.

IV. Energy Savings of a MLC STT-RAM Cache

We performed extensive simulations to quantitatively analyze the benefits of using MLC STT-RAM caches. Using the x86 simulator `ptlsim` [13], we ran 22 of the SPEC 2006 benchmarks[1]. The simulations skip the initialization phases of every benchmarks and thereafter simulate 2 billion instructions. The simulated typical Intel Core 2 architecture includes a 32KByte, 8-way set associative, 64Byte block, write-back L1 I-cache and D-cache with a hit latency of 2 cycles. Without loss of generality, we choose a 2MByte, 32-way set associative, 64Byte block, write-through unified L2 cache as our SRAM cache baseline. The hit and miss latencies of the L2 SRAM cache is 14 and 140 cycles, respectively.

For comparison, we chose 2MByte L2 SLC and MLC STT-RAM caches (we call them SLC caches and MLC caches, respectively in the later sections) as well as 16MByte L2 SLC and MLC caches as the latter occupies the area close to the 2MB SRAM L2 cache. Except for possibly the number of sets, the SLC and MLC caches have the same set associativity and block size. The power and performance results of each cache configurations are calculated by a modified CACTI [14] at 45nm technology and shown in Table III. Here 'WL' and 'LB'

[1]The remaining ones fail even on vanilla `ptlsim`.

TABLE III
POWER AND PERFORMANCE OF SIMULATED CACHE CONFIGURATIONS.

	2M SRAM	2M SLC	16M SLC	2M MLC	16M MLC	16M MLC + ECC
Read lat. (cycles)	14	11	14	11	14	15
Write lat. (cycles)	N/A	41	43	71	74	75
Read lat. + LB (cycles)	N/A	12	16	13	16	17
Write lat. (WL) (cycles)	N/A	42	45	72	76	77
Read dyn. eng. (nJ)	0.753	0.243	0.593	0.240	0.476	0.651
Write dyn. eng. - periph. ckt. (nJ)	0.531	0.093	0.440	0.074	0.356	0.603
Standby leakage power (W)	1.699	0.252	0.807	0.265	0.617	0.733
Area (mm^2)	17.616	3.538	14.506	3.401	10.553	11.429

denote the data of wear-leveling scheme and lookback technique, which shall be introduced in Section V), respectively. For small caches, SLC has a slight edge over MLC, mainly due to the complexity in the MLC's read and write mechanisms. For larger cache sizes, other factors start to dominate.

Fig. 5 summarizes the energy reductions obtained by using STT-RAM caches compared to a 2MB SRAM cache operating at 300K. At the same 2MByte capacity, both SLC and MLC STT-RAM's consume only about 15-17% of the SRAM's energy. The difference in the energy savings between SLC and MLC caches becomes even more pronounced as capacity scales: the energy of a 16MB MLC cache is only 78% of that of a 16MB SLC cache.

We also studied the energy impact in case error correcting code (ECC) is needed in the cache. For example, a (12,8) Hamming code ECC scheme would require 50% more memory array area. However, our simulation shows that the 16MB MLC cache we simulated, adding ECC consumes on the average only 5% more energy, although a worst case of 19% was also observed.

Although the energy consumption reduction by MLC STT-RAM is significant, the performance penalty is very marginal. When the sizes of both MLC cache and SRAM cache are both 2MByte, the SRAM cache shows slightly better performance because the long write latencies of the STT-RAM caches increases the probability of write buffers being full. However, except for some write intensive benchmarks such as `462.libquantum`, the processor performance may be improved by increasing the capacity of MLC caches due to the reduced miss rate. We also observed that some benchmarks having large working sets are able to take advantage of a larger L2 with significant speedup: `429.mcf` runs more than twice as fast if a 16MB cache is used instead of a 2MB one. Fig. 5 shows the normalized performances of MLC STT-RAM caches at the capacities of 2MByte and 16MByte against the 2MB SRAM cache under different benchmarks. Compared to the microprocessor with a 2MB SRAM cache, the average normalized execution times of the 2MB and 16MB MLC cache designs are 1.025 and 0.979 while that of the 2MB and 16MB SLC cache designs are 1.004 and 0.96, respectively.

V. ENHANCING WRITE ENDURANCE

Write endurance is an important challenge in the deployment of MLC STT-RAM caches. Moreover, this problem is further aggravated by the extremely skewed nature of cache write accesses. When we simulated the SPEC2006 benchmark `429.mcf` for a 16MB MLC cache, we found that the distribution of writes was very unbalanced: the write access number varies from 0 to 102,000. By assuming a 3 GHz processor clock and a write endurance of 4×10^{12} write cycles [8], the STT-RAM cache will last only 513 days.

A. Wear Leveling via Set Remapping

The basic idea of our proposed wear leveling technique is to remap all set indices after a certain length of time (an 'epoch' or δ). At the beginning of every epoch, a *remap register* is updated based on Gray's code. During an access, the bits in the address that normally form the set index is XOR'ed with this remap register's value to obtain the actual set index. In our experiments, we selected three different epoch δ settings: 3 million, 12 million and 30 million cycles, which can be simply recorded by a global counter. Also, in a normal cache design, the set index bits are not stored as part of the tag. Two distinct addresses mapping to different sets can have the same tags. However, due to remapping, the set index bits must now be included as part of the tag.

B. Performance Recovering via Lookback

Set remapping distributes write operations across the cache by translating the same memory address into different cache sets at different epochs. However, after remap register is reconfigured, a L2 cache access that is a hit under the previous remapping scheme may become a miss because the access now is redirected to a different set under the new

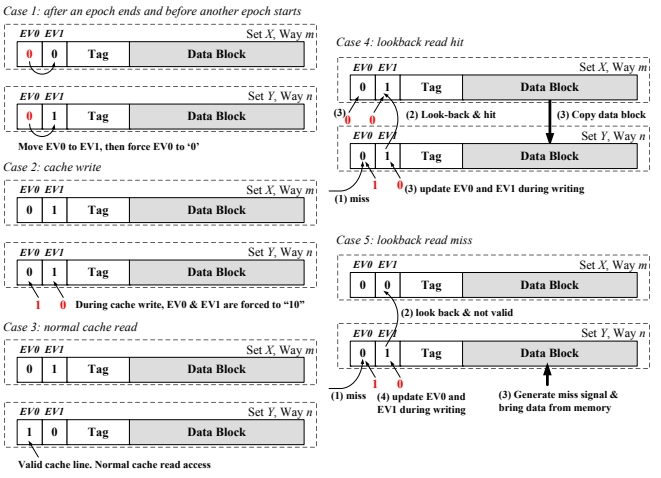

Fig. 6. Operation of the proposed lookback scheme.

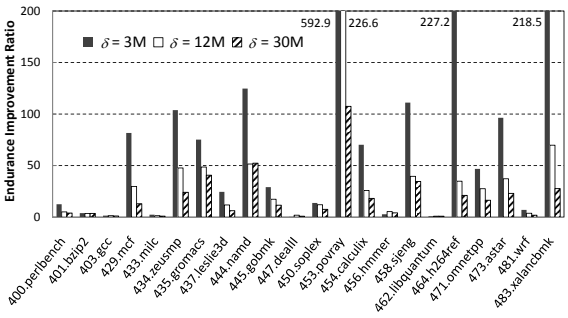

Fig. 7. Endurance improvement ratio of 16MB MLC caches w.r.t. 16MB baseline cache.

remapping scheme. In such cases, we can recover some of the corresponding performance lost through an access lookback scheme:

In a sequential L2 cache where tag check is done before data is accessed, we implement two valid bits, EV0 and EV1 instead of one bit normally. Here EV0 and EV1 are respectively the valid bits for the current and immediate previous epoch: EV0 = 1 means that the block is valid in the current epoch; EV1 = 1 means that it is valid in the previous epoch; EV0 = 0 and EV1 = 1 means that the block was brought in during the previous epoch and in the current epoch has not yet been evicted; EV0 = 1 and EV1 = 0 means the block is brought into the cache during the current epoch; EV0 = EV1 = 0 means no valid block was brought in during the current as well as the previous epoch. EV0 and EV1 should never be both '1'. Fig 6 illustrates an example of our proposed access lookback technique. Assuming that a memory address is translated into cache block A (Set X Way m) and block B (Set Y Way n) in previous epoch $t-1$ and current epoch t, respectively, we have:

- *Case 1: In-between two epochs.* During the switch of two epoches $t-1$ and t, The EV0's of all blocks must be copied to the corresponding EV1's, followed by reseting the EV0's of all blocks to '0'.
- *Case 2: Cache write.* When a new block is brought into the cache, the physical memory address is mapped to block B, say. The EV0 and EV1 of block B are set to '1' and '0', respectively. We assume a write-through cache, and hence a write to the next level of the memory hierarchy is also performed, thereby ensuring consistency.
- *Case 3: Normal cache read.* Suppose the content of block B has been updated in current epoch t, we have EV0=1 and EV1=0 for block B. During subsequent read accesses to block B, since EV0 = 1, a normal cache read procedure is followed: if the tag matches, you have a hit. Otherwise, it is a miss.
- *Case 4: Lookback read hit.* During a read access to block B with EV0 = 0, a ***lookback*** of this memory address's set mapping in the previous epoch $t-1$ is performed by using the previous epoch's remap register: Suppose that remap yields block A, we then check EV1 of block A. if the EV1 = 1, we know that block A is holding a block brought in during the previous epoch and has not yet been evicted. We continue the access with a tag check against block A's tag. If it is a cache hit, block A will be copied to block B. At the same time, we set block A's EV0 = 0 and EV1 = 0, and block B's EV0 = 1 and EV1 = 0. In our simulations, such a hit has a latency of 16 cycles, two more than the normal L2 cache hit latency of 14 cycles to account for the additional sequential tag lookup.
- *Case 5: Lookback read miss.* If a lookback is performed from block B to A and meet block A's EV1 = 0, we have a cache miss. The new block is brought into B from memory. Then block B's EV0 and EV1 are set to 1 and 0, respectively.

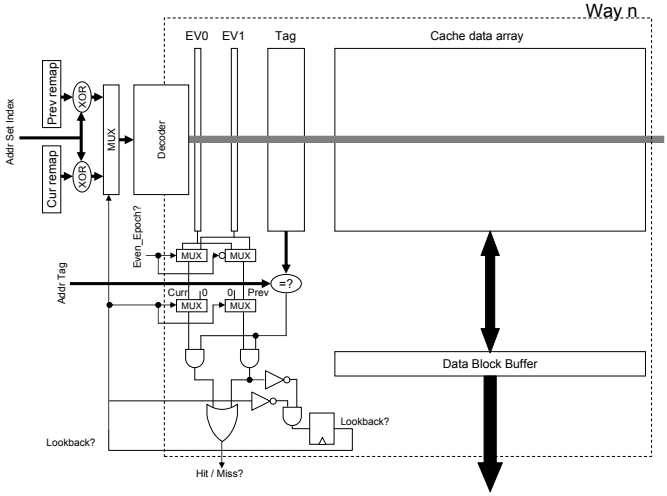

Fig. 8. Block diagram of a cache with lookback scheme.

Fig. 7 shows the endurance improvement ratio of a 16MB MLC cache against the 16MB MLC cache *without* wear-leveling technique (we refer it to "16MB baseline cache") at different epoch length. It is measured as the reduction of the maximum per-set writes. In some extreme cases, the endurance lifetime of MLC caches is improved up to 593×. Also, smaller values of δ re-distribute cache accesses more evenly, thereby leads to longer lifetime of MLC caches.

In the actual design of valid bits EV0 and EV1, each of them represent *even* and *odd* epochs t, respectively. For example, let's assume t is an odd number. Then in epoch $t-1$, EV0 is used for $t-1$ and EV1 is for $t-2$. When epoch $t-1$ ends and the new epoch t starts, EV0 and EV1's roles are swapped: EV0 is still used for even epoch $t-1$, which becomes the previous epoch now, whereas EV1 is re-used for the current odd epoch t. To speedup the erasing of all valid bits during the epoch switch, these valid bits can be implemented with some nonvolatile memories, e.g., SRAM. A short power gating period can simply erase all the valid bits.

The implementation of our set-remapping-based wear-leveling techniques is shown in Fig. 8. Two global remap registers, XOR gates for set remapping, and a multiplexer for look back are introduced. Some additional registers are also needed to latch the input address for lookback. These circuitry are shared globally by all cache sets and incur negligible hardware overhead. At the end of every epoch, L2 cache has to idle for some cycles to reset all EV0's. However, because epochs last for millions of cycles, this overhead is negligible.

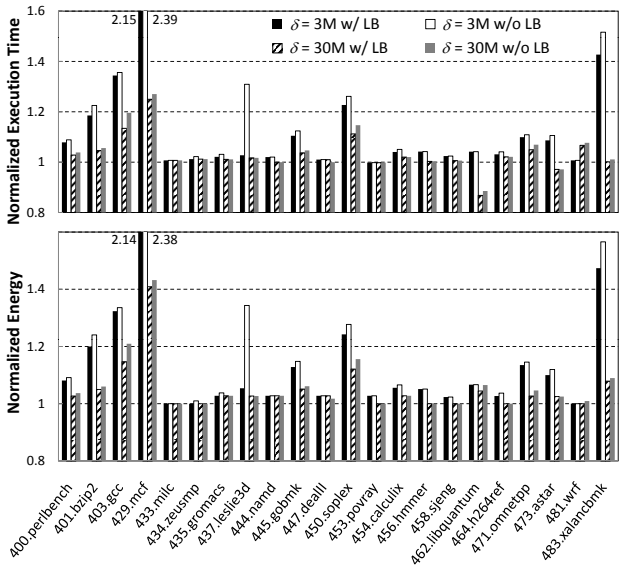

Fig. 9. Summary of execution time and energy for 16MB MLC caches w.r.t the 16MB baseline cache.

The timing and energy overhead of lookback scheme was calculated using SPICE simulation based on the design implementation in Fig. 8, and included in our simulation.

C. Performance and Energy Simulations

The performance and energy results of the 16MB MLC caches with wear-leveling techniques are shown in Fig. 9, including those with and without our lookback scheme. All the results are normalized against the 16MB baseline cache. Comparing Fig.7 and Fig. 9, for benchmarks with large working sets and very unbalanced cache accesses, i.e., `429.mcf` with δ=3M, set remapping significantly improves the cache lifetime ($> 80\times$) but at the expense of a significant overall performance and energy overheads ($> 2\times$). For such applications, our lookback mechanism can recover both the performance and energy loss. For the most applications with small working sets, they will most likely have most of their data needs fulfilled within δ, even though data accesses are unbalanced. Set remapping significantly improves the MLC cache lifetime with very marginal performance and energy overheads. When the L2 cache is accessed infrequently, e.g., in `447.dealII` and `462.libquantum`, the impact is less predictable. The maximum per-set writes after applying set remapping can be even bigger. However, it does not have any real impact on the system's endurance due to the small number of writes.

On the average, our proposed set remapping scheme together with lookback can achieve up to a $84\times$ improvement in endurance, compared to our 16MB baseline cache. the incurred performance and energy overheads (at δ=3M) are less than 13.6% and 18.3%, respectively, (except for `429.mcf`). The results of different epoches are summarized in Table IV. We note that this improvement is orthogonal to the one received from resistance-logic state encoding.

VI. Conclusion

We studied several critical architectural and design issues in using MLC STT-RAM for the secondary caches of processors.

TABLE IV
Parameters of a 16MB MLC cache with wear-leveling w.r.t. 16MB baseline cache.

	δ=3M	δ=12M	δ=30M
Lifetime impr. (best)	592.9×	226.6×	107.5×
Lifetime impr. (average)	84.1×	32.1×	19.5×
Ave. slowdown (w/o LB)	11.3%	5.53%	2.82%
Ave. slowdown (w/ LB)	8.7%	4.44%	2.00%
Ave. energy overhead (w/o LB)	12.6%	6.23%	4.22 %
Ave. energy overhead (w/ LB)	9.87%	5.14%	3.38 %

Our results showed that by carefully mapping the resistance states of MLC MTJ to the logic states, a MLC STT-RAM cache can yield up to $8\times$ more capacity than a SRAM cache of the same footprint, while consuming about 37% of the energy. Compared to SLC STT-RAM caches, MLC STT-RAM caches use about 30% less area, and 21% less energy.

We also raised a concern about the write endurance of MLC STT-RAM caches, and proposed an architectural solution that uses set remapping to even out writes in the cache. Combined with a performance recovering technique called "lookback" and the encoding scheme, our proposed scheme can prolong the cache's lifetime on the average by $19.5\times$ to $84.1\times$ with a performance degradation as low as 3.38% of MLC STT-RAM caches without applying the wear-leveling scheme.

References

[1] B. R. et. al., "Scaling the bandwidth wall: challenges in and avenues for cmp scaling," in *36th ISCA*, Jun. 2009, pp. 371–382.

[2] "The International Technology Roadmap for Semiconductors," http://www.itrs.net, 2008.

[3] X. Wang, Y. Chen, H. Li, H. Liu, and D. Dimitrov, "Spin torque random access memory down to 22nm technology," *IEEE Transaction on Magnetics*, vol. 44, no. 11, pp. 2479–2482, 2008.

[4] R. D. et. al., "On-chip MRAM as a high-bandwidth low-latency replacement for DRAM physical memories," http://www.cs.utexas.edu/ftp/pub/techreports/tr02-47.pdf.

[5] X. D. et. al., "Circuit and microarchitecture evaluation of 3D stacking magnetic RAM (MRAM) as a universal memory replacement," in *DAC*, 2008, pp. 554–559.

[6] G. S. et. al., "A novel architecture of the 3D stacked MRAM L2 cache for CMPs," in *14th HPCA*, 2009, pp. 239–249.

[7] Y. C. et al., "Access scheme of multi-level cell spin-transfer torque random access memory and its optimization," in *53rd IEEE Int'l Midwest Symp. on Ckt. and Syst.*, 2010, pp. 1109 –1112.

[8] Z. D. et al., "Spin-transfer torque switching in magnetic tunnel junctions and spin-transfer torque random access memory," *Jour. of Physics: Condensed Matter*, vol. 19, no. 16, p. 165209, 2007.

[9] K. Croes and Z. Tokei, "$e-$ and $\sqrt{E}-$model too conservative to describe low field time dependent dielectric breakdown," in *2010 Int'l Reliability Physics Symp.*, 2010, pp. 543 –548.

[10] X. Lou, Z. Gao, D. V. Dimitrov, and M. X. Tang, "Demonstration of multilevel cell spin transfer switching in MgO magnetic tunnel junctions," *Applied Physics Letter*, vol. 93, p. 242502, 2008.

[11] Y. C. et al., "New paradigm of predictive mosfet and interconnect modeling for early circuit design," in *IEEE Custom Integrated Ckt. Conf.*, 2000, pp. 201–204, http://www-device.eecs.berkeley.edu/ ptm.

[12] H. Li and Y. Chen, "An overview of non-volatile memory technology and the implication for tools and architectures," in *DATE*, 2009, pp. 731–736.

[13] M. T. Yourst, "PTLsim: A cycle accurate full system x86-64 microarchitectural simulator," in *ISPASS '07*, pp. 23–34.

[14] D. Tarjan, S. Thoziyoor, and N. P. Jouppi, "CACTI 4.0," HP Western Research Labs, Tech. Rep., 2006.