Architecture Design with STT-RAM: Opportunities and Challenges

Ping Chi†, Shuangchen Li†, Yuanqing Cheng†, Yu Lu‡, Seung H. Kang‡, Yuan Xie†
†Department of Electrical and Computer Engineering, University of California, Santa Barbara, USA
‡Qualcomm Incorporated, San Diego, USA
†{pingchi, shuangchenli, yuanqing, yuanxie}@ece.ucsb.edu, ‡yu.lu@qualcomm.com

Abstract— The emerging spin-transfer torque magnetic random-access memory (STT-RAM) has attracted a lot of interest from both academia and industry in recent years. It has been considered as a promising replacement of SRAM and DRAM in the cache and memory system design thanks to many advantages, including non-volatility, low leakage power, SRAM comparable read performance and read energy consumption, higher density than SRAM, better scalability than conventional CMOS technologies, and good CMOS compatibility. However, the disadvantages of STT-RAM, such as higher write energy and longer write latency than SRAM, also bring design challenges. This paper introduces state-of-the-art architectural approaches to adopt STT-RAM in the cache and memory system design by taking advantage of the opportunities brought by STT-RAM as well as overcoming the challenges.

I. Introduction

In the conventional memory hierarchy, SRAM and DRAM have played important roles during the evolvement of modern computer systems. However, with the coming of multi-core and many-core processor era and the continuous technology node shrinking, there are several severe design challenges. First, the power consumption of the cache and memory is a big concern in contemporary computer systems, especially for high performance and big data computing applications [1, 2]. As technology scales, SRAM and DRAM both suffer from severe leakage power dissipation. Moreover, the emerging many-core processor design demands a large capacity of cache to facilitate the data transfer among the cores. For example, Intel Iris Pro Graphics is featured with a 128MB L4 cache [3]. The large area overhead of SRAM prevents it from being adopted in large-scale caches. Furthermore, as DRAM density advances, the increasing refresh penalty results in significant performance degradation and high refresh power consumption [4].

In recent years, some emerging non-volatile memory (NVM) technologies, such as phase change memory (PCM) [5] and spin-transfer torque magnetic random access memory (STT-RAM) [6], have been developed and studied to replace SRAM and DRAM in cache and memory system design. Among them, STT-RAM has been considered as the most promising SRAM replacement to build on-chip large-scale caches and also a potential DRAM alternative to build energy-efficient main memory, thanks to its unique characteristics. The STT-RAM stores data in a magnetic tunneling junction (MTJ), which is a multi-layer structure consisting of two ferromagnetic layers and an insulating barrier sandwiched between them. This data storage mechanism has zero leakage power, and can retain data for ten years without power supply. In addition, STT-RAM has a comparable read speed with that of SRAM, and its single cell structure presents much higher density than SRAM. Therefore, as a SRAM replacement, it can enlarge the cache capacity without area overhead. Moreover, the endurance test result showed that STT-RAM cells can sustain more than 10^{12} writes, and the estimated endurance value of STT-RAM is up to 10^{15} [7]. STT-RAM is also inherently immune to high energy particle radiation from space, which is a promising property for mission critical systems such as servers for aerospace and aeronautical applications.

Although STT-RAM has many nice features, it also has some disadvantages that may be a hindrance for its adoption. First of all, the write operation of STT-RAM is generally slower and more energy-consuming than its read operation. In addition, as the technology generation advances, the write current decreases quickly, whereas the read current does not scale well, and therefore the read disturbance error increases [8]. Moreover, when the MTJ is over-stressed by the write voltage, it may not achieve the required endurance for on-chip caches [9].

In this paper, we survey the state-of-the-art research work on STT-RAM based cache and memory system design. By investigating the proposed approaches to leverage the opportunities and overcome the challenges of STT-RAM, we provide an overview of this research area from an architectural perspective.

II. Background

STT-RAM is the second generation of Magnetoresistive RAM (MRAM). The key component of MRAM to store

This work is supported in part by NSF 1461698 and 1500848, and a grant from Qualcomm.

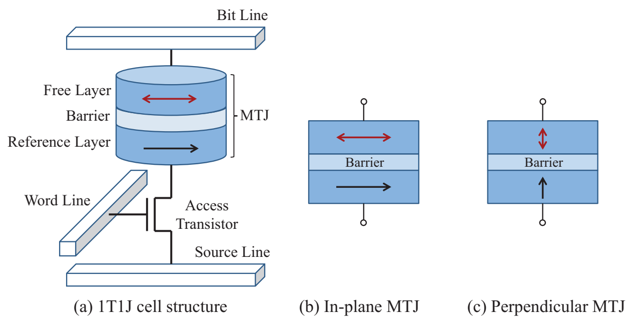

Fig. 1. (a) 1T1J STT-RAM cell structure; (b) in-plane MTJ; (c) perpendicular MTJ.

TABLE I
Comparison of different memory technologies [7]

	SRAM	DRAM	STT-RAM	PCM
Cell size (F^2)	120∼200	4∼6	6∼50	4∼12
Multi-level cell	No	No	Yes	Yes
Read speed	Very fast	Slow	Fast	Slow
Write speed	Very fast	Slow	Slow	Very slow
Read energy	Low	Medium	Low	Medium
Write energy	Low	Medium	High	High
Leakage	High	Medium	Low	Low
Throughput	Very high	Medium	High	Low
Write endurance	10^{16}	10^{16}	$> 10^{12}$	10^8∼10^9
Soft error	Low	High	No	No

the bit information is magnetic tunnel junction (MTJ), as shown in Fig. 1. An MTJ consists of two ferromagnetic layers and one barrier layer in the middle. The magnetization direction of one ferromagnetic layer is fixed (called the reference layer or pinned layer); while that of the other can be changed by applying an external magnetic field or a spin polarized current through the MTJ (called the free layer). When the magnetization direction of the free layer is in parallel with that of the reference layer (P), the MTJ has a low resistance state (LRS) and represents logical '0'; when the magnetization directions of the two ferromagnetic layers are anti-parallel (AP), the MTJ has a high resistance state (HRS) and denotes logical '1'.

The first generation of MRAM uses an external magnetic field to switch the MTJ by applying currents through two orthogonal metal lines, which incurs a large area overhead and a slow access speed. STT-RAM utilizes a spin-polarized current to switch the MTJ, which reduces the cell area and the switching current significantly. Fig. 1 (a) shows the most popular one-transistor-one-MTJ (1T1J) cell structure of STT-RAM. In Fig. 1 (a), the free layer of the MTJ is connected to the bit line (BL), and the access transistor is used to select the MTJ by activating the word line. To write the LRS to a cell, a positive voltage difference is established between the BL and the source line (SL), and an $AP \rightarrow P$ switching current is applied. In contrast, to write the HRS, a negative voltage difference is established between the BL and the SL, and a $P \rightarrow AP$ switching current is applied. To read the data from a cell, a small sensing current or voltage is applied across the MTJ, and the data value is identified by comparing the sensing result with a reference signal. A simple way to generate the reference signal is to average the sensing results of two cells which have one in the LRS and the other in the HRS.

The magnetization directions of the ferromagnetic layers of an MTJ can be made either in-plane or perpendicular, as shown in Fig. 1 (b) and (c). Compared with in-plane MTJ based STT-RAM, emerging perpendicular MTJ based STT-RAM demonstrates its superior characteristics as the technology node scales. It requires a lower switching current while maintaining a high thermal stability for a long retention time.

Table I compares different memory technologies, including conventional SRAM and DRAM, and emerging STT-RAM and PCM. Compared with SRAM, STT-RAM has a much smaller cell size, lower leakage power, and no radiation-induced soft errors; however, it suffers from longer write latency and higher write energy. Compared with DRAM, STT-RAM has many advantages, except its cell size and high write energy. STT-RAM has the multi-level cell (MLC) technology, which can represent two bits in one cell, promising twice the density of single-bit cells. MLC STT-RAM can be implemented by stacking two MTJs of different sizes serially to achieve four distinguishing resistance states in a single cell [10]. PCM also has the MLC property, and it can achieve a higher density than DRAM. However, its write latency is longer and its write energy is higher than DRAM. It also suffers from the write endurance issue because its endurance value is only 10^8∼10^9 [7]. Compared with PCM, STT-RAM has much better read and write performance, and much higher write endurance.

III. Using STT-RAM in Architecture Design

With its unique characteristics, STT-RAM has been studied as a promising candidate to build on-chip caches and off-chip memories.

A. On-chip Caches

Most of the existing work studies STT-RAM as a promising candidate in low level caches, *e.g.*, L2 or L3 cache, as shown in Fig. 2(a) and (b) [11–13]. First, it has a higher density and lower leakage power than SRAM, providing an opportunity to build large-scale and low-power low level caches, especially last level caches. Second, despite of relatively long write latency and high write energy, it has comparable read latency and energy to SRAM, satisfying the performance and energy demand for low level caches. Moreover, it has a higher write endurance than other emerging NVMs, such as PCM, making it capable of frequent write accesses as on-chip caches.

Some studies also explore the adoption of STT-RAM in L1 cache for a large capacity from its high density and also for its energy efficiency [12, 14, 15]. The major challenge for STT-RAM to be used in L1 cache is its write latency, which may degrade the system performance significantly. To overcome this challenge, there

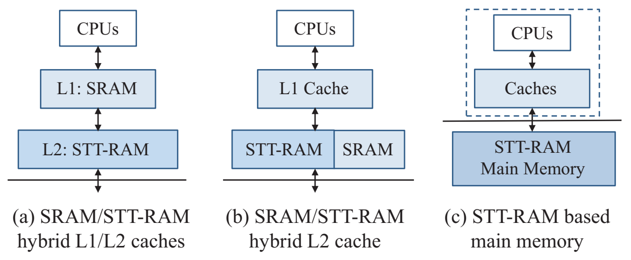
CPUs
L1: SRAM
L2: STT-RAM
(a) SRAM/STT-RAM hybrid L1/L2 caches
CPUs
L1 Cache
STT-RAM
SRAM
(b) SRAM/STT-RAM hybrid L2 cache
CPUs
Caches
STT-RAM Main Memory
(c) STT-RAM based main memory

Fig. 2. (a) SRAM/STT-RAM hybrid L1/L2 caches; (b) SRAM/STT-RAM hybrid L2 cache; (c) STT-RAM based main memory.

are two major categories of solutions. One is to reduce the retention time of STT-RAM so as to reduce its write latency [12,14]. This requires refresh operations for STT-RAM cells to maintain their data. The other is to exploit the SRAM/STT-RAM hybrid cache architecture, leveraging the nice write features of SRAM to mitigate the performance degradation [15]. They usually use SRAM as a buffer of the STT-RAM L1 cache.

B. Main Memory

STT-RAM has also been studied as a DRAM replacement in main memory [16–18]. Although it cannot offer density benefits over DRAM, its low leakage power and non-volatility are attractive to build energy-efficient and non-volatile main memory. Kültürsay *et al.* first explored using STT-RAM to completely replace DRAM in main memory, as shown in Fig. 2(c) [16]. By taking advantage of the decoupled sense amplifier and row buffer structure of STT-RAM main memory, they proposed selective and partial write and row buffer write bypassing techniques to optimize the system performance and energy. Their experiment results show that, equal-capacity STT-RAM main memory can provide comparable performance to DRAM main memory, and reduce the main memory energy consumption by 60%. Recent work also exploited its non-volatility, utilizing STT-RAM main memory to serve as a local checkpoint storage [18].

IV. Approaches to Overcome The Challenges

This section introduces the architectural approaches to address STT-RAM's long write latency, high write energy, and reliability issues.

A. Hybrid Cache and Memory Architecture

To overcome the shortcomings of one memory technology, an effective approach is using hybrid architecture, which makes use of the complementary characteristics of different memory technologies. To mitigate the impact of STT-RAM's long write latency and high write energy, there are many hybrid cache and memory designs with SRAM, DRAM, and other memory technologies. For cache design, the STT-RAM and SRAM hybrid designs are most commonly studied, to combine the high density and low power benefits from STT-RAM as well as the fast read and write speeds of SRAM. The hybrid cache architectures (HCAs) can be vertical or horizontal, *i.e.* inter-level or intra-level [11], and these two types are orthogonal.

In the vertical or inter-level HCA, higher level caches (such as L1 cache), usually employ fast SRAM, because they are latency-sensitive and optimized for performance; while lower level caches (such as LLC), usually utilize dense STT-RAM, because they are capacity-sensitive and optimized for a high hit rate to reduce off-chip memory accesses. Fig. 2(a) illustrates a vertical HCA, with the SRAM L1 cache and the STT-RAM L2 cache. Wu *et al.* evaluated a 3-level HCA which uses SRAM for L1 and L2 caches and STT-RAM for L3 cache, and compared it with a baseline 3-level SRAM cache design [11]. The experiment results show that, under the same area constraint, the HCA can improve the instruction per cycle (IPC) by $\sim$4% on average across a collection of 30 workloads. This indicates that the capacity benefit overwhelms the impact of its long write latency. The HCA also achieves a 63% reduction in power consumption.

In the horizontal or intra-level HCA, the cache level is usually partitioned into two regions in a way-based manner, a small SRAM region, and a large STT-RAM region, as shown in Fig. 2(b). This HCA requires proper cache placement and migration policies. From one point of view, the SRAM region is fast and the STT-RAM region is slow. Therefore, the cache blocks are classified according to their access frequencies, and some work allocates or migrates frequently accessed blocks to the SRAM region [11]. From another point of view, the SRAM region is write-friendly, whereas the STT-RAM region is not. Hence, the cache blocks are classified based on their write frequencies, and some other work allocates or migrates write-intensive blocks to the SRAM region [19,20]. One recent study categorized the write accesses to the hybrid LLC into three distinct classes, and adapted each class to different block placement and migration policies [20]. It also proposed an access pattern predictor to direct the block placement and migration. This study is based on the observation that block placement is often initiated by a write access, and the write accesses are categorized into: core-write, prefetch-write, and demand-write. Prefetch-write is a write from the LLC replacement due to a prefetch miss. For a prefetch-write request, the prefetched data are placed into an SRAM line; when the data block is evicted from the SRAM line, if it is predicted dead, it is evicted from the LLC, otherwise it is migrated to an STT-RAM line. Core-write is a write from the core, and it can be a dirty block eviction from the upper level cache. For a core-write request, if it is an LLC miss, the data are written to the main memory directly; if it is a hit in an STT-RAM line, the data will be migrated to an SRAM line if it is predicted to be a write burst access. Demand-write is a write from the LLC replacement due to a demand miss. For a demand-write request, the data block is fetched from the main memory. If it is predicted to be dead-on-arrival, it bypasses the LLC; otherwise, it is placed in an STT-RAM line. With the proposed technique, the hybrid LLC can improve the IPC by 20.5% and reduce the power consumption by 19.3% for multicore workloads, compared with the SRAM-based LLC in the same area footprint.

There are also HCA designs with other memory technologies besides SRAM, such as eDRAM, and PCM [11, 21]. For STT-RAM main memory, a DRAM cache can be used to reduce the write accesses to STT-RAM so as to reduce the latency and energy overheads from STT-RAM write operations [16].

B. Reducing High-Cost Writes

To minimize the latency and energy overhead of write operations, a lot of work has been proposed to reduce the high-cost writes at different granularities, from the bit level to the block level. A conventional write access updates all the bits of a data block. However, a large portion of bit writes are redundant because they write the same values that are already stored in the memory cells. A previous study evaluated a 16MB STT-RAM L2 cache, and found that about 88% of bit writes are redundant across SPEC CPU2006 benchmarks [22]. Since reads are much less costly than writes in terms of both latency and energy in STT-RAM, we can first read the original values of the data block, and then compare them with the new values to identify the bits whose values are changed, and finally update only those bits. This is data comparison write scheme, which has been used in other NVMs that also have asymmetric read and write features, such as PCM. It can eliminate the redundant bit writes by adding a read-before-write operation with additional supporting circuits. Zhou *et al.* proposed early write termination, *EWT*, which performed a read operation during the write operation, and terminate the redundant bit writes at their early stages [22]. With a simple circuit design, it can reduce write energy significantly with no performance penalty. The experiment results show that, *EWT* reduces the write energy by up to 80%, and reduce the total memory energy consumption by 33%, for a 16MB STT-RAM L2 cache.

Besides smart designs of write schemes, there has been some work that reduces writes at the sub-block level by partitioning the data block of a cache line into several sub-blocks. The all-zero-data flag scheme exploits the fact that a large portion of written data are zero-valued [23]. The authors observed that, on average across SPEC CPU2006 benchmarks, 68.40% of the bytes and 54.02% of the words written to a 1MB L2 cache are zero-valued data. Therefore, for an STT-RAM cache, they added several all-zero-data flag bits in the SRAM tag arrays for each cache line, at a byte or word granularity. To execute the write operations to STT-RAM, if some bytes or words of the written data are zero-valued, the proposed scheme sets the corresponding all-zero-data flag bits in the tag array, and only updates the non-zero bytes or words in the STT-RAM array. Their experiment results show that, the all-zero-data flag scheme at the word granularity can reduce the write energy by 69.30% and the total memory energy consumption by 42.51%, and improve the processor performance by 5.44% on average. Similarly, the partial line update scheme divides a data block into multiple sub-blocks (partial lines) [24]. Instead of tracking zero-valued data in STT-RAM L2 cache, it tracks the dirty sub-blocks of each cache line in SRAM L1 cache by adding some dirty bits in its tag array. When a data block is evicted from L1 cache, according to the dirty bits, only the dirty sub-blocks are updated in L2 cache. Therefore, the partial line update scheme eliminates the redundant writes at the sub-block level.

There has been also some work that reduces writes at the block level. One common solution is to add a write cache or buffer in front of an STT-RAM cache or memory [6, 25]. Rasquinha *et al.* designed a fully-associative write cache that sits between SRAM L1 cache and STT-RAM L2 cache [25]. It is mutually exclusive with L2 cache and stores the dirty lines evicted from L1 cache. It can reduce most of the L1 writebacks to STT-RAM. In their work, they also proposed the write biasing technique to reduce the number of L1 writebacks by modifying the cache replacement policy and keeping the dirty blocks in L1 cache for a longer time.

C. Addressing The High Write Latency

Since the high write latency of STT-RAM may degrade the system performance, there have been several classes of solutions to address it. One class relaxes the non-volatility (data retention time) of STT-RAM to reduce the write latency [12, 14]. By decreasing the planar area of the memory cells (the area of the free layer of the MTJ), the required write currents become smaller, and hence the write latency and energy are reduced; and at the same time, the data retention time becomes shorter. When the data retention time is significantly reduced, from 10 years to several *ms* or *μs*, STT-RAM can achieve similar or even better write performance compared with SRAM. However, to prevent the loss of data, refresh schemes are required. Smullen *et al.* designed a hybrid cache with SRAM L1 cache and retention-reduced STT-RAM L2 and L3 caches, and found that it reduced the energy-delay product by more than 70% without performance loss compared with a 3-level SRAM design of the same capacities [12]. In this work, they adopted a simple DRAM-style refresh scheme, in which the refresh operations themselves incur performance and energy overhead depending on the refresh interval. To reduce the refresh overhead, a counter-controlled refresh scheme is used to refresh only those blocks that need a fresh operation before they lose data [14]. This dynamic refresh scheme saves more than

80% of the refresh energy than the DRAM-styled refresh scheme.

Another class of solutions mitigates the impact of long write latency through novel cache management policies. One work addressed the obstruction issue in STT-RAM LLC [26]. Since a large-capacity STT-RAM LLC usually adopts single port, a write operation may occupy the port for a long time and block the following read operations to the same cache bank, and hence degrade the system performance. In this work, an obstruction-aware cache management policy, *OAP*, is proposed. It monitors each process and labels it as LLC-obstruction if the miss rate in a sample period is larger than the obstruction threshold. When a request to the LLC results in a write operation in LLC, *OAP* first checks whether the requester is LLC-obstruction. If it is LLC-obstruction, *OAP* bypasses the LLC and accesses to the main memory directly, therefore improving the system performance. The experiment results on a 4-core architecture with an 8MB STT-RAM L3 cache show that, *OAP* improves the IPC by 14% and reduces the energy consumption of L3 cache by 64% on average across duplicated and mixed workloads from SPEC CPU2006. Another work addresses the increased write burden on STT-RAM LLCs due to aggressive prefetching schemes [13]. It prioritizes different types of LLC requests, including load, store, prefetch and writebacks, based on their criticality, and serves them according to their priorities. Like *OAP*, it also differentiate applications with intensive accesses from those with non-intensive accesses, and prioritizes the applications with non-intensive accesses. The experiment results on a 4-core architecture with an 8MB STT-RAM L3 cache show that, the proposed technique improves the system performance by 8.3% on average across the SPEC CPU2000 and CPU2006 benchmarks.

A third class of solutions reduces the long write latency by addressing the asymmetry in writing '0' and '1' [27]. Switching from '0' to '1' ($P \rightarrow AP$) requires longer time and higher energy than switching from '1' to '0'($AP \rightarrow P$). The cache write latency is determined by slow SET operations. If a line is pre-SET, the latency to write this line can be reduced since only fast switches from '1' to '0' are needed. One proposed STT-RAM cache design [27] added one more line in each cache set for pre-SET, and the data to be write to a cache set is written to the pre-SETed line in priority, and hence the write performance of the STT-RAM cache is improved.

D. Addressing The Read/Write Reliability Issues

With further technology scaling, the read current of STT-RAM will get closer to the write current, and then the read operations will become destructive [8]. To tackle this issue, we can apply the restore-after-read approach to ensure the data reliability. To reduce the large energy overhead of frequent restores, Wang *et al.* proposed an energy-efficient scheme, selective restore (*SR*), for STT-RAM based low level caches [28]. With *SR*, a read-disturbed data block in the STT-RAM cache will not be restored until it is evicted from the upper level cache. Additionally, during the restore operation, *SR* only restores the disturbed cells. The experiment results on an 8MB STT-RAM L2 cache show that, *SR* improves the system performance by 5% and reduces the dynamic energy consumption by 62%, compared with an optimized restore-after-(every-)read design. Moreover, although STT-RAM has a high write endurance value, it may not meet the endurance requirement for on-chip caches under some conditions, for example, when the MTJ is over-stressed by the write voltage [9]. Jadidi proposed a wear-leveling mechanism at line granularity for STT-RAM caches [19].

V. Approaches to Leverage The Non-volatility

Although STT-RAM is always introduced as an emerging NVM technology, there has not been much work that leverages its non-volatile benefits. Zhao *et al.* proposed an efficient persistent memory design, *Kiln*, which constructs a persistent memory hierarchy by using a non-volatile LLC and a non-volatile main memory to provide persistence support [17]. They implemented *Kiln* with STT-RAM LLC and also STT-RAM main memory in their evaluation. This persistent memory architecture provides multiple persistent data versions naturally, one in the non-volatile LLC and the other in the non-volatile main memory. Therefore, with a set of light-weight software and hardware support, it enables atomic in-place updates, which means direct updates to the real in-memory data structures rather than performing logging or copy-on-write. The experiment results show that, compared with NVM based persistent memory using write-ahead logging, *Kiln* can achieve 2× performance improvement, which is 91% of native system performance with no persistence support.

Another study exploited the inherent multiversioning opportunity provided by MLC, since each cell can store two persistent versions of data simultaneously [18]. It leverages MLC STT-RAM main memory for efficient local checkpointing, by using one bit of each cell to store the working data, and the other bit to save the checkpoint data. It also takes advantage of the unique features of MLC STT-RAM write operations. MLC STT-RAM has different write operations from other NVMs, like PCM. MLC PCM adopts the program-and-verify (P&V) technique for write operations, which achieves an intermediate resistance level by iteratively applying partial set or reset pulses and verifying whether a specified precision criterion has been met. Its complex process makes the write latency much longer and the write energy much higher than writing single-bit cells. However, writing MLC STT-RAM is simple, taking at most two steps. In the proposed local checkpointing scheme, only one-step write operations are required. During the error-free execution, it works only on the soft bits (the operated memory) that can be written by small write currents in one step; during a checkpoint, it first senses the values of the soft bits, and then updates the hard bits (the checkpoint data) according to their soft-bit values by large write currents in one step; during a recovery, contrary to a checkpoint, it first senses the values of the hard bits, and then updates the soft bits according to their hard-bit values also in one step. The experiment results show that, compared with a previous local checkpointing scheme which uses DRAM main memory and PCM local checkpoint storage, the proposed MLC STT-RAM solution is much more efficient, reducing the checkpoint overhead from 7.79% to 0.46% and the memory energy consumption by more than three quarters in a multiprogrammed four-core process node with a $5s$ local checkpoint interval.

VI. Conclusion

STT-RAM is an emerging non-volatile memory technology that has many attractive characteristics, such as non-volatility, low leakage power, fast read speed, good CMOS compatibility, and so on. As a promising alternative to conventional SRAM and DRAM technologies, it provides new opportunities in cache and memory system design. However, its shortcomings, like long write latency and high write energy, also bring great challenges for its widespread adoption. This paper presents an overview of the existing work that facilitates STT-RAM in on-chip cache and off-chip memory applications by utilizing its advantages and overcoming its disadvantages.

References

[1] S. Mittal, "A survey of architectural techniques for improving cache power efficiency," *Sustainable Computing: Inf. Syst.*, vol. 4, no. 1, pp. 33–43, Mar 2014.

[2] S. Mittal, "A survey of architectural techniques for DRAM power management," *Int. J. High Perform. Syst. Archit.*, vol. 4, no. 2, pp. 110–119, Dec. 2012.

[3] P. Hammarlund, A. Martinez, A. Bajwa *et al.*, "Haswell: The fourth-generation intel core processor," *Micro, IEEE*, vol. 34, no. 2, pp. 6–20, Mar 2014.

[4] T. Zhang, M. Poremba, C. Xu *et al.*, "Cream: A concurrent-refresh-aware DRAM memory architecture," in *HPCA'14*, Feb 2014, pp. 368–379.

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

[6] G. Sun, X. Dong, Y. Xie *et al.*, "A novel architecture of the 3D stacked MRAM L2 cache for CMPs," in *HPCA'09*, Feb 2009, pp. 239–249.

[7] S. Mittal, J. Vetter, and D. Li, "A survey of architectural approaches for managing embedded DRAM and non-volatile on-chip caches," *Parallel and Distributed Systems, IEEE Transactions on*, vol. 26, no. 6, pp. 1524–1537, June 2015.

[8] H. Noguchi, K. Ikegami, K. Kushida *et al.*, "7.5 A 3.3ns-access-time 71.2 μW/MHz 1Mb embedded STT-MRAM using physically eliminated read-disturb scheme and normally-off memory architecture," in *ISSCC'15*, Feb 2015, pp. 1–3.

[9] H.-C. Yu, K.-C. Lin, K.-F. Lin *et al.*, "Cycling endurance optimization scheme for 1Mb STT-MRAM in 40nm technology," in *ISSCC'13*, Feb 2013, pp. 224–225.

[10] M. Aoki, H. Noshiro, K. Tsunoda *et al.*, "Novel highly scalable multi-level cell for STT-MRAM with stacked perpendicular MTJs," in *VLSIT'2013*, June 2013, pp. T134–T135.

[11] X. Wu, J. Li, L. Zhang *et al.*, "Design exploration of hybrid caches with disparate memory technologies," *ACM TACO*, vol. 7, no. 3, pp. 15:1–15:34, Dec. 2010.

[12] C. Smullen, V. Mohan, A. Nigam *et al.*, "Relaxing non-volatility for fast and energy-efficient STT-RAM caches," in *HPCA'11*, Feb 2011, pp. 50–61.

[13] M. Mao, H. H. Li, A. K. Jones, and Y. Chen, "Coordinating prefetching and STT-RAM based last-level cache management for multicore systems," in *GLSVLSI'13*, 2013, pp. 55–60.

[14] Z. Sun, X. Bi, H. H. Li *et al.*, "Multi retention level STT-RAM cache designs with a dynamic refresh scheme," in *MICRO'11*, 2011, pp. 329–338.

[15] J. Wang, Y. Tim, W.-F. Wong *et al.*, "A coherent hybrid SRAM and STT-RAM L1 cache architecture for shared memory multicores," in *ASP-DAC'14*, Jan 2014, pp. 610–615.

[16] E. Kultursay, M. Kandemir, A. Sivasubramaniam, and O. Mutlu, "Evaluating STT-RAM as an energy-efficient main memory alternative," in *ISPASS'13*, April 2013, pp. 256–267.

[17] J. Zhao, S. Li, D. H. Yoon *et al.*, "Kiln: Closing the performance gap between systems with and without persistence support," in *MICRO'13*, 2013, pp. 421–432.

[18] P. Chi, C. Xu, T. Zhang *et al.*, "Using multi-level cell stt-ram for fast and energy-efficient local checkpointing," in *ICCAD'14*, 2014, pp. 301–308.

[19] A. Jadidi, M. Arjomand, and H. Sarbazi-Azad, "High-endurance and performance-efficient design of hybrid cache architectures through adaptive line replacement," in *ISLPED'11*, Aug 2011, pp. 79–84.

[20] Z. Wang, D. Jimenez, C. Xu *et al.*, "Adaptive placement and migration policy for an STT-RAM-based hybrid cache," in *HPCA'14*, Feb 2014, pp. 13–24.

[21] J. Zhao, C. Xu, and Y. Xie, "Bandwidth-aware reconfigurable cache design with hybrid memory technologies," in *ICCAD'11*, Nov 2011, pp. 48–55.

[22] P. Zhou, B. Zhao, J. Yang, and Y. Zhang, "Energy reduction for STT-RAM using early write termination," in *ICCAD'09*, Nov 2009, pp. 264–268.

[23] J. Jung, Y. Nakata, M. Yoshimoto, and H. Kawaguchi, "Energy-efficient spin-transfer torque RAM cache exploiting additional all-zero-data flags," in *ISQED'13*, March 2013, pp. 216–222.

[24] S. P. Park, S. Gupta, N. Mojumder *et al.*, "Future cache design using STT MRAMs for improved energy efficiency: Devices, circuits and architecture," in *DAC'12*, June 2012, pp. 492–497.

[25] M. Rasquinha, D. Choudhary, S. Chatterjee *et al.*, "An energy efficient cache design using spin torque transfer (STT) RAM," in *ISLPED'10*, Aug 2010, pp. 389–394.

[26] J. Wang, X. Dong, and Y. Xie, "OAP: An obstruction-aware cache management policy for STT-RAM last-level caches," in *DATE'13*, March 2013, pp. 847–852.

[27] G. Sun, Y. Zhang, Y. Wang, and Y. Chen, "Improving energy efficiency of write-asymmetric memories by log style write," in *ISLPED'12*, 2012, pp. 173–178.

[28] R. Wang, L. Jiang, Y. Zhang *et al.*, "Selective restore: An energy efficient read disturbance mitigation scheme for future STT-MRAM," in *DAC'15*, 2015, pp. 21:1–21:6.