Exploring Main Memory Design Based on Racetrack Memory Technology

Qingda Hu[1], Guangyu Sun[2], Jiwu Shu[1*], Chao Zhang[2]
[1]Department of Computer Science and Technology, Tsinghua University, Beijing, China
[2]CECA, Peking University, Beijing, China
hqd13@mails.tsinghua.edu.cn, gsun@pku.edu.cn,
shujw@tsinghua.edu.cn, zhang.chao@pku.edu.cn

ABSTRACT

Emerging non-volatile memories (NVMs), which include PCRAM and STT-RAM, have been proposed to replace DRAM, mainly because they have better scalability and lower standby power. However, previous research has demonstrated that these NVMs cannot completely replace DRAM due to either lifetime/performance (PCRAM) or density (STT-RAM) issues. Recently, a new type of emerging NVM, called Racetrack Memory (RM), has attracted more and more attention of memory researchers because it has ultra-high density and fast access speed without the write cycle issue. However, there lacks research on how to leverage RM for main memory. To this end, we explore main memory design based on RM technology in both circuit and architecture levels. In the circuit level, we propose the structure of the RM based main memory and investigate different design parameters. In the architecture level, we design a simple and efficient shift-sense address mapping policy to reduce 95% shift operations for performance improvement and power saving. At the same time, we analyze the efficiency of existing optimization strategies for NVM main memory. Our experiments show that RM can outperform DRAM for main memory, in respect of density, performance, and energy efficiency.

Keywords

Main Memory; Racetrack; Shifts; Performance; Energy

1. INTRODUCTION

As we move into the era of Big Data, various modern applications (i.e. in-memory computing/database) keep increasing demand for larger memory capacity and better memory performance. However, due to the fact that the progress of DRAM technology is slower than that of CMOS process technology, the gap between on-chip computing logic and off-chip memory subsystem keeps increasing. In addition, the high static power consumption of DRAM further obstacles the development of main memory. Thus, memory researchers have been seeking for alternatives of DRAM for future main memory design [8, 9, 10, 16, 25].

*Jiwu Shu is the corresponding author.

GLSVLSI '16, May 18 - 20, 2016, Boston, MA, USA

ISBN 978-1-4503-4274-2/16/05…$15.00

DOI: http://dx.doi.org/10.1145/2902961.2902967

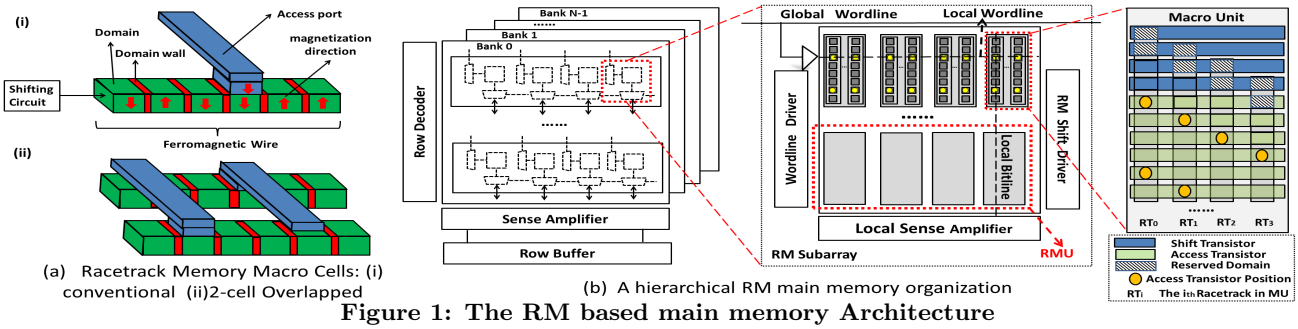
Several emerging non-volatile memory technologies (NVMs) have been extensively investigated for future main memory design. These NVMs include Phase Change Memory (PCRAM), Spin-Transfer Torque RAM (STT-RAM), etc. Prior research has been proposed to improve performance, energy saving, and reliability of these NVMs for main memory [8, 9, 10, 16, 25]. However, it is still difficult to completely replace DRAM with these NVMs due to their own limitations. For example, STT-RAM can achieve better performance than DRAM. But, its storage density is several times smaller than that of DRAM [9]. Similarly, though MLC PCRAM can provide potential higher storage density and lower leakage power consumption, its limited lifetime and poor write performance become major obstacles [8]. Thus, these NVMs are preferred being combined with DRAM for hybrid main memory design as a compromise.

Recently, a new type of non-volatile memory called Racetrack Memory (RM) has attracted more and more attention. Through integrating multiple bits in a tape-like nanowire, RM can provide ultra-high storage density, SRAM-comparable access speed and high write endurance [5, 12, 17, 18, 19, 20, 22, 24]. Previous works focus on using RM as a replacement of SRAM for on-chip memory design, such as cache and registers. There still lacks research on how to leverage RM for main memory. In fact, there exists a difference between on-chip memory and main memory in the design parameters, organization and optimizing strategies. Simply replacing DRAM cells with RMs may result in an inefficient solution.

In this paper, we review the RM based main memory designs in different layers and explore the potential of using RM to completely replace DRAM in main memory. First, we propose the RM based main memory architecture and study design configurations of implementing RM in the chip level, to exploit trade-off among area, performance, and energy. Second, in the architecture level, we design a simple shift-sense address mapping to reduce 95% shift operations to achieve a great performance improvement and energy saving. Third, we analyze the efficiency of applying existing optimization techniques for NVM main memory on RM based design. Our experiments indicate that it is feasible to replace DRAM with RM for improvement of main memory, in respect of storage density, performance, and energy saving.

2. RACETRACK MEMORY

Racetrack memory (RM) is a new generation of spintronic-technology based memory [5, 12, 17, 18, 19, 20, 22, 24]. As shown in Figure 1(a)(i), one RM cell mainly includes a tape-like ferromagnetic wire for data storage, and access ports for data read/write operations. Multiple domains are stored in the wire to share a smaller set of access ports. Each domain can independently store one bit according to its magnetization direction. Thus, RM can achieve ultra-high



(a) Racetrack Memory Macro Cells: (i) conventional (ii)2-cell Overlapped

(b) A hierarchical RM main memory organization

Figure 1: The RM based main memory Architecture

storage density. Because a single domain and an access port form a MTJ (Magnetic Tunnel Junctions) structure similar to that in a STT-RAM cell, it can achieve fast operation speed and low energy consumption like STT-RAM.

RM typically has three basic operations. Read operations are performed by measuring the magneto-resistance of the bit through access ports. Write operations are completed by issuing a large current to change magnetization direction via access ports. Because multiple domains share several ports, the desired bit is required to be aligned with the port before being accessed. Shift operations are implemented by driving a spin-polarized current pulse via shifting circuit through the entire ferromagnetic wire, to shift domains in lock step.

Because a racetrack is longer and narrower compared to an access transistor, it is more area-efficient to overlap RM cells to improve storage density. Figure 1(a)(i) shows that the unused space of one conventional RM cell is dominating. Figure 1(a)(ii) shows that two transistors are stacked on the top of two racetracks, which can reduce the blank space to improve area efficiency. Thus, prior works [18, 20, 24] propose such an organization with overlapped RM cells, called Macro Unit (MU), as a basic building block for RM array.

The layout of MU is determined by three parameters: the number of racetracks in a MU (N_R), the number of bits (domains) in each racetrack (N_B), and the number of access ports in a MU (N_P). As the sample MU shown in Figure 1(a)(ii), N_R=2, N_B=6 and N_P=2. Without increasing the layout area, they are limited by some factors: (1) The maximum N_B is determined by the length of racetrack and domain, (2) The maximum N_R is limited by the width of transistor and racetrack. (3) The maximum N_P is determined by the length of racetrack and transistor. For a detailed introduction to MU, we refer the reader to [24]. In Section 4, we will study how to make choice of MU.

3. THE RM MAIN MEMORY DESIGN

In this section, we propose the RM based main memory architecture with its basic operations and timing restrictions, which differs from traditional DRAM design in the following respects. (1) Its basic building blocks are MUs, (2) varying latency and energy consumption induced by shift operations.

3.1 Main Memory Architecture

A single rank of main memory consists of multiple memory chips, which typically has 4, 8 or 16 data output pins [6, 15, 21]. As shown in Figure 1(b), a RM based chip consists of multiple banks. We use the decoupled sense amplifier and row buffer structure like in prior work [10], which supports multiplexed sense amplifiers, has greater flexibility for row buffer organization. Thus, the data is copied to the row buffer once it is sensed by sense amplifiers.

In order to reduce high latency for driving long bitlines /wordlines and keeping reliable sensing [3, 10], a RM bank is typically divided into a two-dimensional array of subarrays. Each subarray typically includes (1) an array of MUs organized as rows and columns, (2) wordline drivers to strength global wordline signals on the local wordlines, (3) local sense-amplifiers to sense low perturbation from local bitlines and amplify it on the global bitlines, (4) a unique RM shift driver to shift the desired rows to align with access ports.

MUs are the basic building blocks of RM array. The additional domains are reserved on the both ends of MUs to avoid data loss when shifting. Each row of data is stored in a bit-interleaved fashion over a row of MUs [22]. We abstract such a "Row of MUs" as a RMU. When a request arrives, the row decoder selects the RMU which contains the desired row, and shifts the row to align with access ports.

3.2 Basic Operations and Timing Restrictions

The RM based main memory typically has following fundamental operations and timing restrictions.

- Shift operations (ST) shift the desired row to align with access ports before activate operations.
- Activate operations (ACT) sense and latch the desired data into the row buffer.
- Read (RD) and write (WR) operations are issued by reading data from or writing data to the row buffer in a block granularity. The separate sense amplifier and buffer structure enables write-back operations (WB) [9]: the data is written solely on the row buffer, and flushed to the array only when a buffer conflict occurs and the content is dirty.
- Precharge (PRE) operations disconnect the activated row with the bitlines to prepare for another activation row.

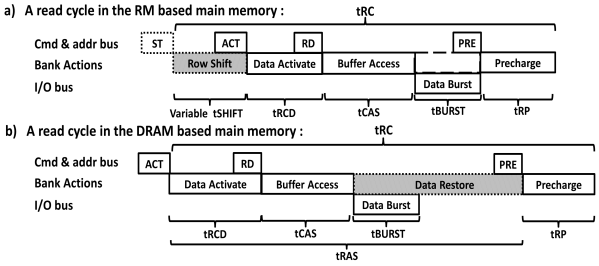
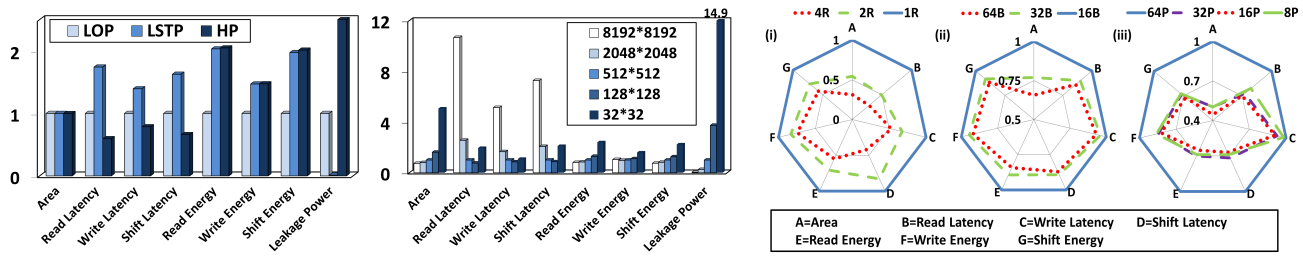


Figure 2: A read cycle of main memory

Using a read cycle as an example, as shown in Figure 2, the RM based main memory differs from DRAM in the following respects. (1) Before activates, if the desired row is not aligned with access ports, the memory controller sends ST operations to the corresponding bank. The time period that it takes to do shift operations is labelled as tSHIFT, which is variable because it is proportional to shift distance. (2) Since DRAM uses charge-based capacitor to store data, its read operations are destructive so that data must be restored back to arrays during each read cycle. The data restoration time (tRAS) is always high [15]. On the contrary, a read cycle of RM is not required to be restored to arrays because it is not read destructive. (3) DRAM must periodically do refresh operations to avoid data lost.



(a) The impact of device type (normalized to the LOP devices)

(b) The impact of sub-array size (normalized to 512*512)

(c) Effect on the area, latency and power induced by MU parameters: (i) N_R, (ii) N_B and (iii) N_P

Figure 3: The Chip-level exploration of RM

4. CHIP LEVEL EXPLORATION

In this section, we address several important configurations in the RM based chip design. In addition, we provide a comparison of RM with other NVMs in the chip level.

4.1 Chip Design Exploration of RM

The previous literature [24] extends a circuit-level simulator - NVSim [3] to develop a RM modeling model, which provides area, performance and power estimation of racetrack memory. In this work, according to Microns data sheet [14], we carefully extend this model to simulate a 128MB RM memory chip with eight banks, eight data output pins, and 256B row buffer in 45nm technology. Typical RM device-level parameters (include physical parameters of single racetrack and peripheral circuit) in this paper are similar to the previous work [24]. We select 45nm technology node because it is the most advanced one we can obtain. We believe that the similar trend is kept for other technology nodes.

4.1.1 Device Type

NVSim supports three types of devices defined by ITRS [3, 25]: low standby power (LSTP), low operating power (LOP) and high performance (HP). These devices have a significant impact on the design target because they are used in the both global support and peripheral circuitry. As shown in Figure 3(a), The HP devices are designed for the fastest access speed but they have highest dynamic/leakage power. The LOP devices are designed for the lowest dynamic power and their performance are between LSTP and HP devices. The LSTP devices are designed for lowest leakage power but they have lowest speed and higher dynamic power. Fortunately, all peripheral circuitry in the RM main memory can be powered down without losing any stored data when it is idle, because RM has non-volatility characteristics. Thus, we use the LOP devices to achieve a best energy-delay product.

4.1.2 Sub-Array Size

The subarray size is a critical parameter for memory chip design. We study the subarray size from 32*32 to 8192*8192. As shown in Figure 3(b), smaller subarrays are designed for latency-optimized target because they decrease bitline (BL) and wordline (WL) latency. However, they result in more complex global interconnects for more sophisticated H-tree organization, increasing energy consumption of the H-tree. Moreover, when the size is further decreased below 32*32, the extra latency caused by more complex H-tree has offset reduction from shorter BL/WL. On the contrary, larger subarrays are designed for area-optimized target as they significantly amortize the peripheral circuitry area. However, they bring higher latency caused by longer BL/WL. Thus, the subarray size should neither be too small or too large.

4.1.3 Macro Unit

As shown in Figure 3(c), we analyze the effect on the area, latency and power induced by MU parameters. We find that increasing N_R and N_B can improve the efficiency of the layout significantly, which gets a lower latency and energy consumption due to its smaller layout. Thus, we use the maximal N_R and N_B to perform further analysis in the rest of this paper. Each macro cell has four racetrack stripes, and each racetrack has 64 data bits [18, 24].

Figure 3(c)(iii) shows the effect of increasing N_P. Considering the length of racetrack, transistor, and the gap distance between two transistor, when N_P is large (> 16), the area increases with the increasing number of ports, dominated by the transistor size, bringing higher latency and power. However, when N_P is small (< 16), the area decreases with the increasing number of ports, because overhead bits reserved on the both ends of MUs decrease greatly. When $N_P = 16$, it is more area efficient to overlap multiple RM stripes as a MU structure. Thus, we use 16 access ports per MU to reduce shifts without increasing latency/power/area significantly. In Section 5, we will provide the sensitivity of the number of shift operations to N_P in the system level.

4.2 Comparison of RM Characteristics with Other Memory Technologies

We compare the characteristics of a RM based memory chip with other memory technologies, including DRAM, PCM and STTRAM. Their device-level parameters come from previous works [3, 21] and are also simulated by modified NVSim. Because DRAM has high leakage power, we use commodity DRAM and LSTP peripheral devices to obtain a best energy-delay product like in related literature [21, 25].

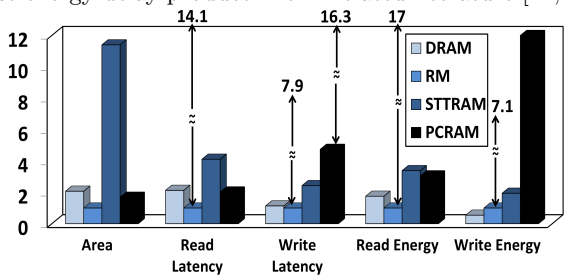
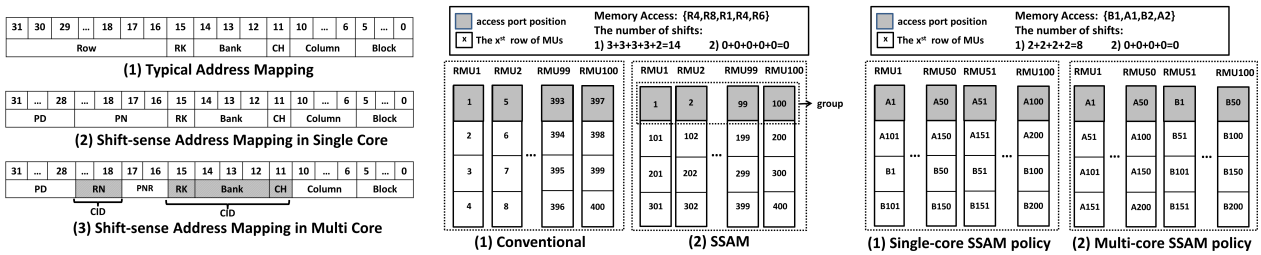


Figure 4: Comparison of a RM based bank with other memory technologies

Figure 4 shows that RM has 2.1X storage density compared with DRAM, which means that a RM chip can store two times capacity than DRAM with the same area size. Also, RM has the lowest read/write latency and read energy due to its smallest layout area. However, due to the effect of variable shifts, the shift plus read latency/energy is 14.1/17 times higher than the baseline read and the shift plus write latency/energy is 7.9/7.1 times higher than the baseline write in the worst case. So optimization for shifts reduction is needed.

5. ARCHITECTURE LEVEL EXPLORATION

In this section, we first propose a simple address mapping technique for shift intensity reduction. Then, we evaluate several existing optimization methods for other NVMs to study their efficiency for the RM based main memory.



(a) Memory Address Mappings (Example system in figure has 2 Channels, 2 Ranks per Channel, 8 Banks and 64B Cache-lines)

(b) One example showing the advantage of replacing the conventional address mapping policy with SSAM in single-core

(c) One example showing the advantage of replacing Single-core SSAM policy with Multi-core SSAM policy in multi-core

Figure 5: Shift-sense Address Mapping in Single-core and Multi-Core

5.1 Experimental Setup

We modify the main memory simulator - NVMain [15] to simulate the basic operations and timing restrictions, and integrate it into gem5 [1] as our simulation platform. Table I shows our major processor and memory parameters. We use the methodology from NVMain to estimate the overall energy consumption via calculating the occurrences of different memory operations. For RM and other NVMs, the timing and energy parameters are output from the extended NVSim. For DRAM, they are obtained from Micron data sheet [14]. Note that although RM can provide higher capacity with the same area constraint, the workloads used in this work with limited memory usage cannot leverage this advantage. Thus, we just use the same capacity for different technologies to simplify the following discussion.

Table 1: Major processor and memory parameters

Processor	4 single Alpha cores, out-of-order, 3GHz,
L1 Caches	32KB I-cache, 32KB D-cache, 4-way, 2-cycle latency
L2 Caches	4MB, 8-way, 10-cycle latency 64B-lines
Memory	DDR3-1600, 4GB, 64bit I/O, FR-FCFS, 8 banks(x8) 1/2 channels(@1-core/4-core), 1/2 ranks per channel(@1-core/4-core)
DRAM	Memory Timing and Energy Parameters from [14]
RM Latency	Read: 5ns Write: 9.3ns Shift: 4.3ns
RM Energy	Read: 10.6nJ Write: 28nJ Shift: 11.3nJ (2KB row)

We use a representative subset of benchmarks from the SPEC CPU2006 suite [4], STREAM [13] and SPLASH [23]. Specific benchmarks in each benchmark suite are selected for their memory intensity. We also randomly choose from these benchmarks to form 10 multiprogrammed workloads, which has four applications and each of which is bound to one processing core. We execute all target benchmarks for five billion instructions to cover enough large working set.

5.2 Shift-Sense Address Mapping

Address mapping has a significant impact on the memory system behavior. As shown in Figure 5(a), conventional memory systems normally use a typical open-page address mapping policy [6, 7, 11, 15], mapping all adjacent columns of the same row to a sequential region, which maximizes the spatial locality to minimize activates. Also, it interleaves memory address among ranks and banks in the granularity of the row size to pipeline memory requests.

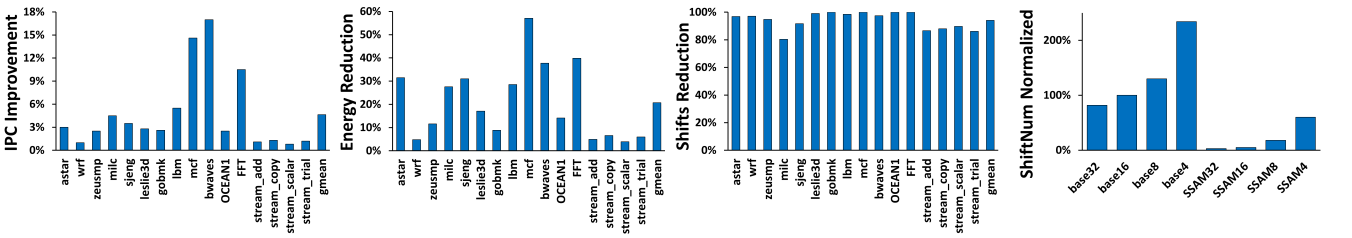
However, the conventional mapping treats every row activation equally without considering the shift issue, which distributes all adjacent rows continuously over all RMUs. As shown in Figure 5(b), 400 rows are distributed sequentially from the first RMU to the last RMU (assuming one bank has 100 RMUs and each RMU has one racetrack, four bits, one port to simplify discussion). However, this mapping may result in frequent shifts among a much small set of RMUs, because memory access always has high spatial locality. Figure 5(b) illustrates an example that one application generates an access sequence {R4,R8,R1,R4,R6}. These requests are mapped to only two RMUs, leading to high frequent shift operations (the total shifts is 14). In consideration of large storage capacity of main memory and strong spatial locality of memory access, we propose the Shift-Sense Address Mapping (SSAM) policy to reduce shifts significantly.

Single Core. We first explain How SSAM works in the single core scenario. SSAM divides row address bits (bit16-bit31) into two parts: PD and PN (Figure 5(a)). PD (Port Distance) indicates the gap distance between the row and its access port. PN (Port Number) indicates the serial number of the port. More specifically, assuming that every x bits share one port, SSAM divides all rows into x groups based on PD, and distributes adjacent row address to the same group. If the entire space of one group is used up, SSAM allocates subsequent rows to the next group.

SSAM reduces shift intensity greatly mainly because of two reasons. (1) Intra-group. Assuming that the capacity of main memory is 2GB and every 16 rows share one wordline, each group is as large as 128MB (2GB/16) to reduce frequent shifts caused by spatial locality of several small regions. (2) Inter-group. Because latency and energy of shift operations are proportional to shift distance, decreasing distance can improve performance and energy. SSAM reduces distance of most shifts to nearly one, because the difference between two adjacent row address in the same RMU is larger (128MB).

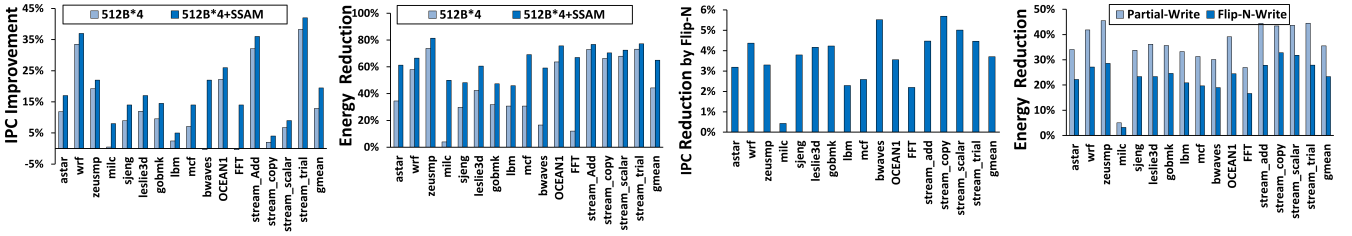
Multi Core. When multiple threads share the same DRAM devices, the single-core SSAM policy may produce inter-thread interference, because memory address of all threads is mapped evenly across all RMUs. As shown in Figure 5(c), when adopting single-core SSAM policy, two threads A and B may be mapped to the same RMUs simultaneously. When they generate an access sequence {B1, A1, B2, A2}, they will conflict with each other, increasing the amount of extra shifts. The probability of the inter-core interference grows with the number of cores. When different cores are not mapped to the same RMUs, this type of interference cannot happen. Thus, we further divide the PN bits into RN and PNR (Figure 5(a)). RN (RMU Number) indicates the serial number of the RMU. PNR (Port Number in RMU) indicates the serial number of the access port within the RMU. The multi-core SSAM policy utilizes RMU partitioning to map different cores to different RMUs, via dividing bit11-bit15 and bit18-bit27, labelled as CID (Color ID). As shown in Figure 5(c), core A and core B are allocated to disjoint RMUs to avoid interference.

Implementation. The operating system can use a buddy system to manage physical pages, and form a hierarchy lists of free pages based on different orders and different colors (based on CIDs), like page-coloring technique used in [7, 11], which allocates different CIDs to specific applications depending on the application's need for memory capacity, can be designed to allocate a sequential region as much as possible. Thus, for single-core SSAM, it can be designed



(a) IPC improvement over baseline after using SSAM (b) Energy reduction over baseline after using SSAM (c) Shift intensity reduction over baseline after using SSAM (d) Effect of access ports per MU on SSAM (normalized to base16)

Figure 6: The IPC improvement, energy reduction and shifts reduction when adopting SSAM in single-core



(a) IPC improvement over baseline after using 512B*4 and 512B*4-SSAM (b) Energy consumption reduction over baseline after using 512B*4 and 512B*4-SSAM (c) IPC degradation over baseline after using Flip-N-Write (d) Energy consumption reduction over baseline after using Flip-N-Write and Partial-Write

Figure 7: Comparison of energy and performance across different existing methods

as a static physical address mapping, which can be implemented in the memory controller or inside RM chips, does not change the existing memory interface, has negligible implementation overhead. For multi-core SSAM, RMU partitioning can be entirely managed via the virtual to physical address mapping of the OS. However, in this paper, we adopt a static partition scheme to allocate an equal number of CID to each core, because it does not need profiling of workloads and we find that it is robust to dynamic workload changes. The strategy of dynamically assigning CIDs to cores can be further explored, but is beyond the scope of this paper.

Results. For single-core, Figure 6 shows that SSAM significantly reduces energy consumption by 21% and improves IPC by 5% over the conventional address mapping (baseline) on average. The reason is that SSAM greatly reduces shifts by 95% (Figure 6(c)). For multi-core, as the resource competition becomes more severe, the multi-core experiments in Figure 8(c) shows that SSAM reduces running time by 18% and energy by 50% compared to the baseline. As shown in Figure 6(d), we also evaluate the sensitivity to the number of shared ports per MU. When adopting the traditional policy, shift operations grow significantly with decreasing ports. When using SSAM, shift operations can still be reduced greatly even using less ports. Note that shifts can be reduced more via SSAM if considering that the RM-based main memory has higher storage density.

Comparison with the existing methods. Numerous works [17, 20, 22] propose to address the shift problem. However, these works need complex control logic and large hardware cost, industrial venders of main memory may have some hesitation in adopting them. In contrast, SSAM reduces shifts with simple control logic and negligible hardware cost. In addition, [19] adjusts the associativity of RM cache to reduce shifts. [12] uses the warp register remapping to place registers around ports to decrease the max shift distance. However, they are designed for targeted on-chip cache or GPU registers, do not consider different characteristics of main memory, such as much greater capacity and stronger spatial locality. SSAM not only uses large capacity of each group to reduce the average shift distance (inter-group), but also to reduce frequent shifts induced by strong spatial locality of several small regions (intra-group). Moreover, SSAM reduces shifts in both single-core and multi-core scenarios.

5.3 Efficiency of Existing Strategies

There exists a lot of prior works about the NVM main memory. They have proposed some novel methods to reduce energy and performance degradation. However, because relatively symmetric read/write operations of the RM based memory are faster than PCRAM/STTRAM, and RM has a unique shift operation, the existing methods may be not appropriate for it. Thus, we analyze the efficiency of the existing popular methods on the RM based main memory.

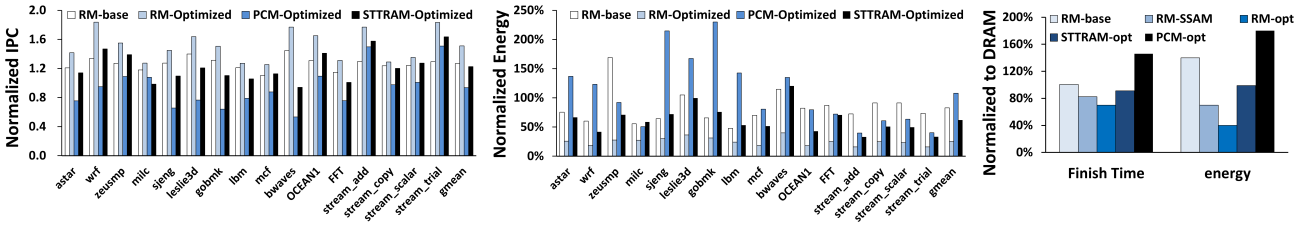
5.3.1 *Performance Optimization*

Prior work [10] proposes the separate sense amplifier and buffer structure to enable the multiplexing of sense amplifiers. Then, it utilizes the saving area to replace a single conventional row buffer with multiple narrower buffers. Narrower buffer can reduce dynamic read/write energy, and multiple buffers can mitigate performance and energy penalties caused by row buffer miss. However, its benefits may be reduced on the RM based main memory for two reasons: (1) Multiple buffers may bring extra shifts. For example, if $(i+1)_{th}$ buffer is required to activate one row, the system shifts access ports with the desired row. But when i_{th} buffer needs to be written back to array, the system may need to shift ports back. (2) Reads and writes of RM are faster, which may decrease the benefits from multiple row buffers.

We replace a single row buffer with four 512B buffer as in Ref.[10]. As shown in Figure 7(a)(b), we find that the buffer organization can still improve IPC by 13% and reduce energy consumption by 44% on average. But we also observe that its benefits for the RM based main memory is smaller than being used in PCRAM/STTRAM. In addition, the combination of the buffer organization and SSAM can greatly reduce shift operations, improves IPC by 20% and reduces energy by 65%, because we find that the buffer organization without SSAM brings 29% extra shifts on average.

5.3.2 *Write Energy Reduction*

Because writes are much slower and consume much more energy than reads for PCRAM and STT-RAM, prior work has proposed various read-modify write based techniques to reduce write energy, such as the representative work Flip-N-Write (FNW) [2]. Before each write, FNW reads the original row of data to make a bit-by-bit comparison, and decides whether to flip or not. FNW aligns each block with a single



(a) The IPC normalized to DRAM in single-core (b) The Energy normalized to DRAM in single-core (c) Finish time, Energy normalized to DRAM in multi-core

Figure 8: Comparison of energy and performance across different memory technologies

bit to indicate whether the data was flipped or not. Therefore, FNW can reduce write bits of dirty blocks by half to save write energy. However, FNW brings severe read latency and energy in the RM based main memory, because both read and write operations of RM are much faster and consume relatively close energy. Thus, we compare FNW with Partial Write (PW) [9, 10]. PW assigns each block with a dirty bit and tracks dirty data from cache to main memory, which aims to write only dirty blocks back to array. Note that PW does not affect system performance.

We consider both PW and FNW at a cache line granularity. Although FNW can reduce more write energy, Figure 7(d) shows that PW reduces more total energy compared than FNW (36% to 23%), because FNW results in extra read energy consumption. In addition, Figure 7(c) shows that FNW reduces IPC by 4% due to the latency of extra read operations.

6. SYSTEM-LEVEL COMPARISON

In this section, we compare the baseline RM main memory (RM-base), the RM main memory with all optimization options (SSAM, PW and buffers organization) (RM-Optimized) with other memory technologies: the DRAM main memory, the PCRAM main memory with all the same optimization options (PCM-Optimized) and the STTRAM main memory with all the same options (STTRAM-Optimized).

In single-core comparison, Figure 8(a)(b) shows that RM-Optimized achieves best performance and lowest energy consumption over all other memory technologies. Compared to DRAM, RM-Optimized improves system IPC by 49% and reduces energy consumption by 75% on average.

In multi-core comparison, Figure 8(c) shows that RM-Optimized also provides significantly best performance and lowest energy consumption over all other technologies. Compared to DRAM, RM-Optimized design reduces the execution time by 31% and reduces energy consumption by 59%.

7. CONCLUSION

RM has potential to replace DRAM for future main memory design. In this work, we provide a holistic research on this topic by investigating some important design factors and issues in the RM based memory. In the chip layer, we explore the design choices to exploit trade-off among area, performance, and energy. In the architecture layer, we design a new shift-sense address mapping policy to reduce 95% shifts. Moreover, we analyze and study the efficiency of applying existing optimization techniques for NVM main memory on RM based design. Our experiments indicate that the optimized RM based main memory can improve IPC by 49% and reduce energy consumption by 75% compared to DRAM.

8. ACKNOWLEDGMENTS

We would like to thank the anonymous GLSVLSI reviewers for their comments that helped improve this paper. This work is supported by the National Natural Science Foundation of China (Grant No.61232003, 61433008), the Beijing Municipal Science and Technology Commission of China (Grant No.D151100000815003), the National High Technology Research and Development Program of China (Grant No.2013AA013201).

9. REFERENCES

[1] N. Binkert et al. The gem5 simulator. *ACM SIGARCH Computer Architecture News*, 39(2), 2011.
[2] S. Cho and H. Lee. Flip-n-write: a simple deterministic technique to improve pram write performance, energy and endurance. In *MICRO*, 2009.
[3] X. Dong et al. Nvsim: A circuit-level performance, energy, and area model for emerging nonvolatile memory. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 31(7), 2012.
[4] J. L. Henning. Spec cpu2006 benchmark descriptions. *ACM SIGARCH Computer Architecture News*, 34(4), 2006.
[5] A. Iyengar and S. Ghosh. Modeling and analysis of domain wall dynamics for robust and low-power embedded memory. In *DAC*, 2014.
[6] B. Jacob et al. *Memory systems: cache, DRAM, disk*. Morgan Kaufmann, 2010.
[7] M. K. Jeong et al. Balancing dram locality and parallelism in shared memory cmp systems. In *HPCA*, 2012.
[8] M. Joshi and T. Li. Mercury: A fast and energy-efficient multi-level cell based phase change memory system. In *HPCA*, 2011.
[9] E. Kultursay et al. Evaluating stt-ram as an energy-efficient main memory alternative. In *ISPASS*, 2013.
[10] B. C. Lee et al. Architecting phase change memory as a scalable dram alternative. In *ISCA*, 2009.
[11] L. Liu et al. A software memory partition approach for eliminating bank-level interference in multicore systems. In *PACT*, 2012.
[12] M. Mao et al. Exploration of gpgpu register file architecture using domain-wall-shift-write based racetrack memory. In *DAC*, 2014.
[13] J. McCalpin. Stream benchmark. *www.cs.virginia.edu/stream/ref.html*, 1995.
[14] Micron. *8Gb: x4, x8, x16 DDR3L SDRAM Description*. www.micron.com, 2015.
[15] M. Poremba and Y. Xie. Nvmain: An architectural-level main memory simulator for emerging non-volatile memories. In *ISVLSI*, 2012.
[16] M. K. Qureshi et al. Scalable high performance main memory system using phase-change memory technology. *ISCA*, 2009.
[17] A. Ranjan et al. Dyrectape: a dynamically reconfigurable cache using domain wall memory tapes. In *DATE*, 2015.
[18] G. Sun et al. From device to system: cross-layer design exploration of racetrack memory. In *DATE*, 2015.
[19] Z. Sun et al. Design exploration of racetrack lower-level caches. In *ISLPED*, 2014.
[20] Z. Sun and H. Li. Cross-layer racetrack memory design for ultra high density and low power consumption. In *DAC*, 2013.
[21] S. Thoziyoor et al. A comprehensive memory modeling tool and its application to the design and analysis of future memory hierarchies. In *ISCA*, 2008.
[22] R. Venkatesan et al. Stag: Spintronic-tape architecture for gpgpu cache hierarchies. In *ISCA*, 2014.
[23] S. C. Woo et al. The splash-2 programs: Characterization and methodological considerations. In *ACM SIGARCH Computer Architecture News*, volume 23, 1995.
[24] C. Zhang et al. Quantitative modeling of racetrack memory, a tradeoff among area, performance, and power. In *ASP-DAC*, 2015.
[25] P. Zhou et al. A durable and energy efficient main memory using phase change memory technology. In *ISCA*, 2009.