Enabling High-Performance LPDDR3-Compatible MRAM

Jue Wang
Pennsylvania State University
jzw175@cse.psu.edu

Xiangyu Dong
Qualcomm Technologies, Inc.
xydong@acm.org

Yuan Xie
Pennsylvania State University
yuanxie@cse.psu.edu

ABSTRACT

DRAM consumes a significant amount of energy in mobile computing devices today. Emerging non-volatile memory such as magnetoresistive memory (MRAM) offers a DRAM alternative and can potentially lead to a more energy-efficient memory system. The MRAM technology is already mature, but considering the memory industry is highly standardized, we are still unable to see any MRAM used in mainstream products. To tackle this problem, we design an LPDDR3-compatible MRAM interface by considering both MRAM pros and cons. Our design solves the pin-compatibility and the performance issues caused by MRAM small page size, and it optimizes the interface protocol by leveraging the MRAM unique feature of non-destructive reads. Combining our techniques, we boost the MRAM performance by 14% and provide a DRAM-swappable MRAM solution with 20% less energy.

Categories and Subject Descriptors

B.7.1 [Integrated Circuits]: Types and Design Styles—*Memory Technologies*

Keywords

MRAM, spin-transfer torque, LPDDR

1. INTRODUCTION

Battery-backed mobile devices require low energy consumption. The memory subsystem in mobile devices is unfortunately not energy-efficient, e.g. the DRAM in a smartphone today can consume 34.5% of the total energy [8]. It is because DRAM by nature is volatile: DRAM needs periodic refreshes, which can cause a 20% energy waste [17]. Ever worse, the DRAM refresh issue will soon become a system performance bottleneck [19]. Therefore, it is necessary to explore alternative memory technologies.

Magnetoresistive memory (MRAM), or known as spin-transfer torque memory (STT-RAM), is an emerging non-volatile memory (NVM), and it has potentials to provide an energy-efficient memory subsystem [15, 18]. However, enjoying the MRAM energy-saving benefit is not free. It is a consensus that MRAM

Figure 1: The conceptual view of an MRAM cell.

cannot compete with DRAM in terms of performance, and more importantly, MRAM chip internal structure is incompatible to today's memory interfaces. This compatibility is not only highly required for a successful MRAM early adoption, but also critical for enabling a tiered memory system using refresh-free MRAM and high-performance DRAM [10, 20]. Everspin's effort [22] to produce a DDR3-compatible MRAM is another example of this.

The goal of this paper is to introduce an optimized MRAM interface but totally compatible to the state-of-the-art LPDDR3 specification originally designed for DRAM. All the optimizations are done by tweaking the existing timing parameters. We start with a background introduction in Section 2, followed a discussion in Section 3 on two unique MRAM properties: MRAM small page size and MRAM non-destructive reads. These two properties bring us both challenges and opportunities to reach our goal. Then, we detail three optimization techniques in Section 4: *ComboAS* and *DynLat* to solve the compatibility and performance issues caused by MRAM small page size; *EarlyPA* to leverage the non-destructive reads. Combined all these together, we show in Section 5 that the MRAM performance is improved by 14% (up to 36%), and this DRAM-swappable MRAM solution saves 20% energy consumption.

2. BACKGROUND

We briefly explain DRAM and MRAM technology background first in this section.

2.1 MRAM Technology

Compared to DRAM, MRAM is non-volatile and consumes zero standby power [6, 13, 22, 27]. Figure 1 illustrates the basic concept of MRAM. Instead of using electrical charges, MRAM uses magnetic tunnel junctions (MTJs) to store its binary data. Each MTJ consists of two ferromagnetic layers: a pinned layer with a fixed magnetization direction and a free layer with a switchable direction. The relative direction of these two layers determines the stored data. Previous work [6] has shown that the unit cell dimension of MRAM below 30 nm can be smaller than $8F^2$, which is comparable to DRAM's $6F^2$ size.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from Permissions@acm.org
ISLPED'14, August 11–13, 2014, La Jolla, CA, USA.
Copyright 2014 ACM 978-1-4503-2975-0/14/08 ...\$15.00.
<http://dx.doi.org/10.1145/2627369.2627610>.

Figure 2: The memory organization for LPDDRx chips.

2.2 DRAM and LPDDRx Interface

LPDDRx memory interface is dominant in modern mobile devices. JEDEC released the first LPDDR specification in 2009. Today, almost all the mobile SoC use LPDDR2 or LPDDR3 [12]. Figure 2 shows an exemplary LPDDRx configuration, which is a 1-channel, 2-rank memory subsystem with four x32 DRAM chips.

LPDDRx uses a multiplexed command/address (CA) bus to reduce the pin count. The 10-bit CA bus contains command, address, and bank information. Each LPDDRx DRAM internally has 8 banks, and each bank can independent process a different memory request. Same as DDRx, LPDDRx accesses begin with an activation command (ACT), which includes a row access signal, a bank address, and a row address. Memory controllers send ACT commands to DRAMs, and a corresponding DRAM row is then activated (opened). After that, memory controllers issue column read or write commands with a column access signal and the starting column address for burst accesses.

2.3 MRAM Device Projection

Everspin announced the world's first MRAM (based on STT-RAM) DDR3 device in 2013 [22]. This product is DDR3-compatible, but its capacity is only 64Mb, far away from modern gigabit-scale DRAMs. Before detailing how to build DRAM-comparable gigabit-scale MRAM, we first project how the future MRAM will look like.

We use our modified CACTI [26] and NVSim [7] to simulate a 4Gb LPDDR3 DRAM and a 4Gb LPDDR3 MRAM on a 28nm process, respectively. Table 1 lists the timing and power parameters. We can verify that the estimated numbers match to the actual LPDDR3 DRAM and MRAM prototypes. The major differences between DRAM and MRAM LPDDR3 devices are:

- *Small page size*: MRAM page (row buffer) size is 16 folds smaller than DRAM. This results in an unbalanced MRAM row/column address bit ratio – 18:6 versus DRAM's 14:10. The details of this constraint is discussed later in Section 3.
- *Non-volatility*: MRAM is non-volatile and needs no refresh. Hence, MRAM has zero tREF, tRFC, auto-refresh current (IDD5¹) and self-refresh current (IDD6).
- *Non-destructive read*: MRAM has smaller tRTP and can issue precharge command sooner because MRAM reads are non-destructive and do not need write-back.
- *Fast page close*: DRAM precharge needs to balance bitlines (BL and $\overline{BL}$) to VDD/2, but MRAM precharge can skip this step. Therefore, MRAM precharge (tRP) is faster.
- *Slow page open*: MRAM MTJ has small on/off resistance ratio (e.g. 200%), and it is hard to sense the data. Therefore, the MRAM row activation (tRCD) is slower.

¹In auto-refresh mode, MRAM peripheral circuitry still consumes power so that IDD5 is essentially IDD2P.

Table 1: Comparison of DRAM and MRAM LPDDR3 devices

Parameters	LPDDR3 DRAM	LPDDR3 MRAM		
Clock	533 MHz	533 MHz		
Page size	4 KByte	256 Byte		
Bank bit	BA2-BA0	BA2-BA0		
Row bit	R13-R0	R17-R0		
Column bit	C9-C0	C5-C0		
tREF	3900 ns	N/A		
tRCD	10 cycle	13 cycle		
tRL	8 cycle	6 cycle		
tWL	4 cycle	4 cycle		
tRP	10 cycle	7 cycle		
tRC	32 cycle	18 cycle		
tRTP	4 cycle	2 cycle		
tRRD	6 cycle	6 cycle		
tCCD	4 cycle	4 cycle		
tWTR	4 cycle	4 cycle		
tWR	8 cycle	14 cycle		
tFAW	27 cycle	27 cycle		
tRFC	70 cycle	N/A		
Voltage	VDD1 (1.8 V)	VDD2 (1.2 V)	VDD1 (1.8 V)	VDD2 (1.2 V)
IDD0	7.8 mA	28.3 mA	4.8 mA	41.2 mA
IDD2N	1.8 mA	3.7 mA	1.8 mA	3.7 mA
IDD2P	1.8 mA	1.7 mA	1.8 mA	1.7 mA
IDD3N	3.5 mA	6.9 mA	3.5 mA	6.9 mA
IDD3P	3.5 mA	6.9 mA	3.5 mA	6.9 mA
IDD4R	3.5 mA	140.4 mA	0 mA	170.9 mA
IDD4W	3.5 mA	145.4 mA	0 mA	267.2 mA
IDD5	23.8 mA	78.2 mA	1.8 mA	1.7 mA
IDD6	1.0 mA	3.8 mA	0.0 mA	0.0 mA

- *Slow write*: MRAM has longer write latency and higher write energy. Thus, MRAM has larger tWR and IDD4W.

3. CHALLENGE AND OPPORTUNITY

The underlying technology difference poses both challenges and opportunities in designing an LPDDRx-compatible MRAM.

3.1 Challenge: Small Page Size

Although most MRAM parameters have their DRAM counterparts as we list in Table 1, a key difference is MRAM's page size.

The fundamental constraint of MRAM small page size is that MRAM reads require current sensing. Current-mode S/A is much more complicated than voltage-mode S/A used in DRAM, and it is also significantly larger. Our circuit simulation shows an MRAM S/A is 16 times larger than a DRAM S/A. To maintain chip area utilization, MRAM has to use less S/A causing a smaller page size (e.g. 16X smaller). MRAM industry also has consensus on this. For example, a 2012 EverSpin patent [2] discloses that their MRAM page is only 512-bit large, 32X smaller than a DRAM page. Unfortunately, such a small difference makes MRAM incompatible with existing memory interfaces, and worse degrades the system performance.

Unbalanced address bits: A 16x smaller page size means that a same-sized MRAM requires 4 more row address bits but 4 less column address bits². However, LPDDRx uses a multiplexed CA bus for both command and address, and it can only carry 20 bits per cycle. For a row activation command, we need 2 bits for command decoding, 3 bits of bank addresses, and only have up to 15 bits for row addresses. An MRAM with 256K rows is obviously unsupported. Even though we could possibly add more CA pins to future memory interface, MRAM's unbalanced row/column address bit ratio still makes the row/column multiplexing idea highly inefficient.

²The "row" and "column" are both logical concept here. MRAM can still maintain a reasonable geometric aspect ratio, but partial logical row address bits are used for physical column muxing.

Figure 3: The IPC and page hit ratio comparison between a DRAM system and an MRAM system.

Figure 4: The timing diagrams of Unlimited-pin and ComboAS.

Performance degradation: Although a smaller page size is preferred to avoid over-activation in a memory system that uses close-page policy [28], modern mobile devices heavily adopt open-page policy, and smaller page size means high page miss rate and low performance. Figure 3 shows the page hit rate and the performance impact due to MRAM’s 16x smaller page size (see Section 5 for simulation methodology details). On average, the page hit ratio is decreased by 66% and the resulting performance degradation is around 10%³.

3.2 Opportunity: Non-destructive Read

DRAM reads are destructive and need data restoration, thus the DRAM S/A in an active bank always connect to bitlines and serves as a row buffer. On the contrary, MRAM reads are non-destructive. In another word, we can treat each row buffer as a copy of the original MRAM row data. This extra redundancy allows us to utilize the MRAM row buffer in a more aggressive way, and this is a unique opportunity for MRAM performance optimization.

4. SOLUTIONS

To overcome the drawback and leverage the advantage of MRAM, we gradually propose three optimization techniques as our solution.

4.1 ComboAS: Balance Row/Column Address

As explained in Section 3.1, the LPDDR_x interface only carries 15 row address bits, but our targeted MRAM has a highly skewed row/column bit ratio (e.g 18:6). Adding two more CA pins can temporarily solve this problem, but it requires an industry-wide PHY redesign. Worse, it implies that such MRAM is not DRAM-swappable and prohibits any mixture uses of DRAM and MRAM.

Therefore, the first technique we propose is *Combinational Row/Column Address Strobe (ComboAS)*, and its goal is to re-balance the address bits carried by RAS (row access strobe, e.g. ACT command) and CAS (column access strobe, e.g. READ and WRITE commands). The basic concept is straightforward: offloading the overflowed row address from RAS to CAS.

Since we split the row address into RAS and CAS, ComboAS needs both commands before activating a new row. Consequently, instead of waiting for tRCD, we should issue a CAS command immediately after every RAS command. Figure 4 compares the

³Note that for the workloads whose page hit ratio is less sensitive to the page size, MRAM outperforms DRAM (e.g. *e.nat*) because MRAM has faster precharge speed and needs no refresh.

Table 2: Timing parameters for Unlimited-pin and ComboAS

	Unlimited-pin system	ComboAS
tRCD	tRCD ₀	1 tCK
tCCD	tCCD ₀	tCCD ₀
tRL	tRL ₀	tRCD ₀ + tRL ₀
tWL	tWL ₀	tRCD ₀ + tWL ₀
tRTP	tRTP ₀	tRCD ₀ + tRTP ₀

Figure 5: The timing diagrams of ComboAS and DynLat.

timing diagram of ComboAS against the incompatible solution of adding 2 more CA pins. In ComboAS, the actual row activation is delayed by 1 cycle to wait the remaining row address bits from CAS; read or write accesses are delayed by tRCD₀ (the original row activation delay). Table 2 lists the detailed adjustments.

To implement ComboAS, the modifications include:

MRAM device: We need three minor changes. (1) a new register to hold the partial row address bits carried by ACT and then later combined with the remaining row bits from READ or WRITE commands. (2) a signal generator to latch the remaining row address from the first-arrived CAS command. (3) a small register sets to temporarily hold column addresses. This is because the latency to activate a row (tRCD₀) is larger than the minimum delay between two column commands (tCCD₀), and multiple CAS commands might arrive during a new row activation. In this work, the size of this register set is 3 (i.e. $\lceil tRCD_0/tCCD_0 \rceil$).

Memory Controller: Only a small latch (4-bit in this work) after the PHY interface is needed to temporarily hold the extra row address bits from RAS command and later deposit them into the next CAS command. Second or latter CAS commands do not need to carry any row address bits. In addition, the memory timing parameters are adjusted according to Table 2.

4.2 DynLat: Remove Unnecessary Latencies

Figure 4 shows that ideally ComboAS only causes 1-cycle delay. However, that is not always true. ComboAS unconditionally adds tRCD₀ on top of every tRL, tWL, and tRTP to avoid internal bus conflicts, which is unnecessary for non-back-to-back accesses. For those non-ideal cases, we define a metric, *bubble*, to indicate the difference between the actual interval and the minimum interval⁴. ComboAS can be further improved if bubbles exist.

Taking a deep look into this issue, while the conventional tRL₀, tWL₀, and tRTP₀ are all static values and determined by the memory hardware limitation, the new tRL, tWL, and tRTP parameters in ComboAS become variable as they include the row activation latency (tRCD₀) which we only need to pay once for one opened row. We can deduct this tRCD₀ overhead from tRL/tWL/tRTP if we find bubbles on the command bus. This observation leads us to a dynamic timing parameter settings where tRL, tWL, and tRTP are adjustable on-the-fly. We call this technique *Dynamic Latency (DynLat)*.

⁴The minimum interval between two column accesses is $\max(tCCD, BL/2)$ where BL is the burst length. LPDDR3 has tCCD=4 and BL=8.

Table 3: Simulation settings

Core	2GHz, out-of-order ARM core
SRAM I-L1/D-L1 caches	private, 32KB/32KB, 8-way, LRU, 64B cache line, write-back, write allocate
SRAM L2 cache	shared, 1MB, 16-way, LRU, 64B cache line, write-back, write allocate
Memory controller	open page policy, 32-entry read/write request queues per controller, FR-FCFS scheduler
Main Memory	1 channel, 2 ranks-per-channel, 8 banks-per-rank. Timing is configured as Table 1

Figure 8: Normalized IPC of main memory system with each technique: ComboAS as the baseline, Unlimited-pin, DynLat, EarlyPA, and DRAM systems.

tRCD₀ after every ACT command and reset to tWL+BL/2+tWR or 3+tWL+BL/2+tWR after every WRITE command depending on whether the counter reaches zero or not at the WRITE command arrival. Furthermore, the memory device skips precharge-related timing rule (e.g. tRTP checking) except the tRAS checking as S/As and row buffers are decoupled in the EarlyPA mode.

Memory controller: Symmetrically, the memory controller manipulates the timing parameters in the same way as memory devices do. An additional modification to the memory controller is the write-to-precharge latency control: after issuing a WRITE command, the minimum required delay for the next PRE command is tWL+BL/2+tWR+tRP₀ instead of tWL+BL/2+tWR.

5. EXPERIMENTS

To quantify the performance and energy improvement achieved by our techniques, we detail our simulation methodology and experiment results in this section.

5.1 Simulation Methodology

We model a 2GHz out-of-order ARMv7 microprocessor using our modified version of gem5 [3]. DRAMSim2 [23] is integrated and modified to model the main memory system. Open-page policy with FR-FCFS [21] scheduling is accurately modeled.

We use the parameters in Table 1 to simulate DRAM and MRAM and use open-page policy with row-interleaving to maximize the memory-level parallelism. More details for the simulation setting are provided in Table 3. The memory-intensive benchmarks are selected from SPEC 2006 [24], EEMBC 2.0 [9], and HPEC [11]. We fast-forward each simulation to the pre-defined breakpoint at the code region of interest, warm-up 10 million instructions, and simulate for at least 1 billion instructions.

5.2 Performance speedup

Figure 8 shows the performance speedup of the MRAM system with each proposed technique. The DRAM system performance is also provided for comparison. We use ComboAS as the baseline which has the worst performance. The second bar is the performance of an impractical implementation where 2 more pins are added (referred to as Unlimited-pin in the chart). Compared to Unlimited-pin, Figure 8 shows that the performance of ComboAS is degraded by 5% on average. However, after adopting DynLat, the system performance is bounced back by 3% on average (up to 14%) and is comparable to the Unlimited-pin in most cases. In addition, by leveraging the MRAM non-destructive read with EarlyPA, we improve the performance further. The total

Figure 9: Normalized energy consumption of DRAM system and MRAM system adopted different techniques.

performance speedup is 14% (up to 36%). Generally, memory-intensive workloads benefit more from our proposed techniques because more efficient memory accesses provide larger system performance improvement.

After adopting all the proposed techniques, the overall MRAM performance is competitive to the DRAM counterpart (about 98%).

5.3 Energy consumption analysis

While the performance of our optimized MRAM system is similar to the conventional DRAM system, the real deal breaker is the energy consumption saving. Figure 9 shows the comparison of energy consumption between DRAM and MRAM systems, in which each value is divided to refresh energy, burst energy, activation/precharge energy and background peripheral circuit energy⁶. The energy overhead of each proposed technique is also included.

Compared to DRAM, MRAM-based system consumes zero refresh energy because of its non-volatility, and this is the major source of the MRAM energy saving. However, as shown in Table 1, the read/write energy of MRAM is larger than DRAM because MRAM has smaller sense margin and the memory cell is difficult to write. Thus, the MRAM burst energy is usually larger than the DRAM one. The energy consumed by peripheral circuits is similar between DRAM and MRAM because we do not apply any circuit optimization in this work. But we should note that the peripheral energy of MRAM can be further reduced if MRAM is allowed to go into power-collapse mode frequently during the idle state.

Figure 9 shows the ComboAS MRAM system reduces the total energy consumption by 17% on average compared to DRAM system. After adopting DynLat and EarlyPA techniques, the energy

⁶In this work, we do not model the self-refresh mode for DRAM systems and the power-collapse mode for MRAM systems. The realistic background energy can be smaller than our simulated numbers.

consumption of MRAM system can be further reduced by 4% on average since the performance is increased and the total execution time is reduced. Considering the comparable performance and smaller energy consumption, MRAM is an attractive candidate to build the main memory system.

6. RELATED WORK

Many previous works are focused on increasing DRAM energy efficiency by re-designing DRAM organization architecture [1, 28, 29]. Others are focused on improving DRAM performance [14, 25]. The posted-RAS scheme [28] is the most similar work to our ComboAS technique as we both issue RAS and CAS commands back-to-back. However, posted-RAS only works for close-page policy where there is only one CAS command after opening one row, and it does not optimize the timing parameters to mitigate the performance overhead.

Other work aimed at NVM optimizations. PCM is studied as a main memory candidate and some techniques are proposed to reduce its energy overhead and improve its performance [4, 5, 30]. Lee *et al.* [16] proposed the technique to separate sense amplifiers and row buffers in PCM-based main memory, but unlike *EarlyPA*, their technique did not issue precharge commands in advance to hide the latency. Meza *et al.* [18] and Emre *et al.* [15] also evaluated MRAM as a main memory alternative, but they did not discuss the issue of how to build a compatible interface. Everspin [22] demonstrates a DDR3-compatible MRAM, but their MRAM capacity is so small (i.e. 64Mb) that the pin-compatibility problem is naturally hidden. Also, Everspin does not optimize for MRAM performance.

7. CONCLUSION

The shift from PCs to mobile devices is requesting low-power memory solutions, and non-volatile memories such as MRAM are promising candidates. Compared to DRAM, MRAM has many unique features such as *small page size* and *non-destructive read*. The smaller page size brings challenges in designing commodity MRAM that can be deployed on the same LPDDR interface for DRAM memory, and can cause performance degradation to mobile systems where the page hit ratio is important. In this work, we propose three techniques: *ComboAS* and *DynLat* to solve the DRAM-compatibility issue; *EarlyPA* to further improve the performance. Combined together, our solution enables a commodity MRAM on LPDDR3 interface with a much optimized performance (14% on average and up to 36%). It makes LPDDR3 MRAM have competitive performance but save 20% energy compared to LPDDR3 DRAM does. The proposed architecture is a step forward to the future energy-efficient memory design.

7.1 Acknowledgments

This research was funded by NSF grants 1218867, 1313052, 1409798, and Department of Energy under Award Number DE-SC0005026.

8. REFERENCES

- [1] J. H. Ahn *et al.*, "Multicore DIMM: an energy efficient memory module with independently controlled DRAMs," *Computer Architecture Letters*, 2009.
- [2] S. M. Alam *et al.*, "Memory controller and method for interleaving DRAM and MRAM accesses," US Patent 2012/0 155 160 A1, 6 21, 2012.
- [3] N. Binkert *et al.*, "The gem5 simulator," *SIGARCH Computer Architecture News*, 2011.
- [4] Y. Byung-Do *et al.*, "A low power phase-change random access memory using a data-comparison write scheme," in *ISCAS*, 2007.
- [5] S. Cho *et al.*, "Flip-N-Write: A simple deterministic technique to improve PRAM write performance, energy and endurance," in *MICRO*, 2009.
- [6] S. Chung *et al.*, "Fully integrated 54nm STT-RAM with the smallest bit cell dimension for high density memory application," in *IEDM*, 2010.
- [7] X. Dong *et al.*, "NVSim: A circuit-level performance, energy, and area model for emerging non-volatile memory," *TCAD*, 2012.
- [8] R. Duan *et al.*, "Exploring memory energy optimizations in smartphones," in *IGCC*, 2011.
- [9] EEMBC, "EEMBC benchmark," <http://www.eembc.org/>.
- [10] S. Hellmold, "Delivering nanosecond-class persistent memory," in *FMS*, 2013.
- [11] HPEC, "HPEC benchmark," <http://www.ll.mit.edu/HPECchallenge/>.
- [12] JEDEC Solid State Technology Association, "JESD209-3 LPDDR3 low power memory device standard."
- [13] W. Kim *et al.*, "Extended scalability of perpendicular STT-MRAM towards sub-20nm MTJ node," in *IEDM*, 2011.
- [14] Y. Kim *et al.*, "A case for exploiting subarray-level parallelism (SALP) in DRAM," in *ISCA*, 2012.
- [15] E. Kultursay *et al.*, "Evaluating STT-RAM as an energy-efficient main memory alternative," in *ISPASS*, 2013.
- [16] B. C. Lee *et al.*, "Architecting phase change memory as a scalable DRAM alternative," in *ISCA*, 2009.
- [17] J. Liu *et al.*, "RAIDR: Retention-aware intelligent DRAM refresh," in *ISCA*, 2012.
- [18] J. Meza, J. Li, and O. Mutlu, "A case for small row buffers in non-volatile main memories," in *ICCD*, 2012, pp. 484–485.
- [19] J. Mukundan *et al.*, "Understanding and mitigating refresh overheads in high-density DDR4 DRAM systems," in *ISCA*, 2013.
- [20] M. K. Qureshi *et al.*, "Scalable high performance main memory system using phase-change memory technology," in *ISCA*, 2009.
- [21] S. Rixner *et al.*, "Memory access scheduling," in *ISCA*, 2000.
- [22] N. Rizzo *et al.*, "A fully functional 64Mb DDR3 ST-MRAM built on 90nm CMOS technology," *Magnetics, IEEE Transactions on*, vol. 49, no. 7, 2013.
- [23] P. Rosenfeld *et al.*, "DRAMSim2: A cycle accurate memory system simulator," *Computer Architecture Letters*, 2011.
- [24] SPEC CPU, "SPEC CPU2006," <http://www.spec.org/cpu2006/>.
- [25] K. Sudan *et al.*, "Micro-pages: increasing DRAM efficiency with locality-aware data placement," in *ASPLOS*, 2010.
- [26] S. Thoziyoor *et al.*, "A comprehensive memory modeling tool and its application to the design and analysis of future memory hierarchies," in *ISSCC*, 2008.
- [27] K. Tsuchida *et al.*, "A 64Mb MRAM with clamped-reference and adequate-reference schemes," in *ISSCC*, 2010.
- [28] A. N. Udipi *et al.*, "Rethinking DRAM design and organization for energy-constrained multi-cores," in *ISCA*, 2010.
- [29] H. Zheng *et al.*, "Mini-rank: Adaptive DRAM architecture for improving memory power efficiency," in *MICRO*, 2008.
- [30] P. Zhou *et al.*, "A durable and energy efficient main memory using phase change memory technology," in *ISCA*, 2009.