Performance Impact of a Slower Main Memory: A case study of STT-MRAM in HPC

Kazi Asifuzzaman
Barcelona Supercomputing Center (BSC)
Universitat Politècnica de Catalunya (UPC)
Barcelona, Spain

Milan Pavlovic
BSC & UPC, Barcelona, Spain

Milan Radulovic
BSC & UPC, Barcelona, Spain

David Zaragoza
BSC & UPC, Barcelona, Spain

Ohseong Kwon
Memory Planning Group
Samsung Electronics Co., Ltd,
Seoul, Korea

Kyung-Chang Ryoo
Memory Planning Group
Samsung Electronics Co., Ltd,
Seoul, Korea

Petar Radojković
BSC, Barcelona, Spain

ABSTRACT

In high-performance computing (HPC), significant effort is invested in research and development of novel memory technologies. One of them is Spin Transfer Torque Magnetic Random Access Memory (STT-MRAM) — byte-addressable, high-endurance non-volatile memory with slightly higher access time than DRAM. In this study, we conduct a preliminary assessment of HPC system performance impact with STT-MRAM main memory with recent industry estimations. Reliable timing parameters of STT-MRAM devices are unavailable, so we also perform a sensitivity analysis that correlates overall system slowdown trend with respect to average device latency. Our results demonstrate that the overall system performance of large HPC clusters is not particularly sensitive to main-memory latency. Therefore, STT-MRAM, as well as any other emerging non-volatile memories with comparable density and access time, can be a viable option for future HPC memory system design.

CCS Concepts

•**Computer systems organization** → *Processors and memory architectures;* •**Hardware** → Non-volatile memory; •**Computing methodologies** → Massively parallel and high-performance simulations;

Keywords

STT-MRAM, Main memory, High-performance computing.

MEMSYS 2016 October 3–6, 2016, Washington, DC, USA

DOI: http://dx.doi.org/10.1145/2989081.2989082

1. INTRODUCTION

Memory systems are major contributors to the deployment and operational costs of large-scale high-performance computing (HPC) clusters [1][2][3], as well as one of the most important design parameters that significantly affect system performance [4][5]. For decades, DRAM devices have been dominant building blocks for main memory systems in server and HPC domains. However, it is questionable whether this technology will continue to scale and will meet the needs of next-generation systems. Therefore, significant effort is invested in research and development of novel memory technologies. One of the candidates for next-generation memory is Spin-Transfer Torque Magnetic Random Access Memory (STT-MRAM). STT-MRAM is a byte-addressable, high-endurance non-volatile memory, with access time comparable to DRAM. STT-MRAM is still a novel technology with a lot of scope to be improved in terms of cell size, read/write latency and energy. These improvements requires research in low level — involving geometry of the cells and physical properties of their composing materials. On the other hand, it is important to conduct system-level research on STT-MRAM in order to determine potential application domains which would be benefited by incorporation of this technology. System-level research also detects key STT-MRAM limitations, and estimates their impact on overall system performance and energy consumption.

In this work, we explore how HPC system performance is affected with STT-MRAM main memory. To that end, we analyze performance of production HPC applications running on large-scale clusters with STT-MRAM main memory. We simulate STT-MRAM main memory module with recent industry estimations. Our results show that, 20% slower STT-MRAM main memory device (w.r.t. DRAM) introduces only around 1% overall performance loss for most of the applications under experiment. We also perform a sensitivity analysis, and repeat the simulation with pessimistic 50% and 100% slower STT-MRAM devices w.r.t. DRAM. Again, the results show a small overall performance difference between HPC systems with STT-MRAM and DRAM

main memory.

The rest of the article is organized as follows. Section 2 introduces the STT-MRAM technology. Section 3 describes the experimental environment used in our study, while Section 4 presents and analyzes the results. Section 5 discusses opportunities and challenges of STT-MRAM memory systems for HPC. Finally, Section 6 discusses the related work, and Section 7 presents the conclusions of the study.

2. STT-MRAM

2.1 Technology overview

Research exploring the magneto-resistance caused by the spin polarized current can be tracked back in the '90s [6][7][8]. Although, significant scientific efforts of optimizing and applying this phenomenon to create a novel non-volatile memory is a relatively new approach. Only around ten years ago, in 2005, Hosomi *et al.* [9] presented a non-volatile memory utilizing spin transfer torque magnetization switching for the first time. In the following years, there has been a notable dedication of academic scientists and memory manufacturers researching this novel non-volatile memory technology.

The storage and programmability of STT-MRAM revolve around a Magnetic Tunneling Junction (MTJ). An MTJ is constituted by a thin tunneling dielectric being sandwiched between two ferro-magnetic layers. One of the layers has a fixed magnetization while the other layer's magnetization can be flipped. As Figure 1(a) and (b) depict, if both of the magnetic layers have the same polarity, the MTJ exerts low resistance therefore representing a logical "0"; in case of opposite polarity of the magnetic layers, the MTJ has a high resistance and represents a logical "1". In order to read a value stored in an MTJ, a low current is applied to it. The current senses the MTJ's resistance state in order to determine the data stored in it. Likewise, a new value can be written to the MTJ through flipping the polarity of its free magnetic layer by passing a large amount of current through it [10].

Figure 1(c) illustrates a simplified STT-MRAM array based on 1T-1MTJ cell [11][12][13]. The cells are organized into rows and columns, similar to the conventional DRAM modules. The main difference is that, instead of the capacitor used in DRAM, one bit of data is stored in the MTJ. In this design, the word lines $WL_{1..m}$ activate particular rows of the cell array, while the bit lines $BL_{1..n}$ are used to perform read or write operation to the corresponding MTJs. The current required for these operations is driven by the source lines $SL_{1..m}$.

Further research on the STT-MRAM cell design revealed advanced 2T-2MTJ, 3T-2MTJ and 4T-2MTJ cells in a pursuit to improve performance and energy efficiency [14][15][16].

STT-MRAM can be used to build byte-addressable memory devices with pin structure compatible to the conventional DRAM chips [11]. Therefore, existing DRAM modules can be seamlessly replaced with STT-MRAM modules, without requiring any modification in the rest of the system architecture. This may suggest an easier incorporation of STT-MRAM in the existing systems.

2.2 Ongoing research and development

STT-MRAM is often referred to as a *universal memory*

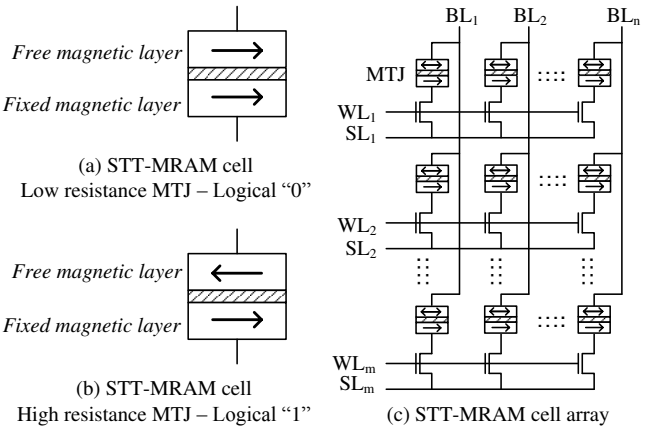

Figure 1: STT-MRAM cell and cell-array

which can be incorporated in all levels of memory hierarchy. Nevertheless, most of the system-level research so far, focused on suitability of STT-MRAM for on-chip cache memories (see Section 6). In general, these studies propose to exploit STT-MRAM's non-volatility, zero stand-by power, and higher density with respect to SRAM to design next-generation caches. Another potential market for STT-MRAM could be the main memory, which is currently dominated by mature DRAM technology. Small-capacity STT-MRAM devices that mainly target embedded systems have already hit the market [17], and large-capacity (high density) STT-MRAM based main memory systems are expected to be on their way. Still, very limited information on STT-MRAM main memory parameters are publicly available till date, and only a few studies analyze STT-MRAM as main memories [18][19][20].

3. EXPERIMENTAL ENVIRONMENT

We analyze performance of large-scale HPC applications running on clusters with STT-MRAM memory. This section presents the application suite, simulation methodology and infrastructure used for this study.

3.1 Application suite

We evaluated STT-MRAM main memory on HPC applications included in the Unified European Application Benchmark Suite (UEABS) [21]. UEABS is the latest benchmark suite distributed by Partnership for Advanced Computing in Europe (PRACE) and it represents a good coverage of production HPC applications running on European Tier-0 and Tier-1 HPC systems. All UEABS applications are parallelized using Message Passing Interface (MPI) and they are regularly executed on hundreds or thousands of processing cores. UEABS also includes input data-sets that characterize production use of the applications. In our experiments, we executed UEABS applications with Test Case A, input data-set that is designed to run on Tier-1 sized systems, up to around thousand x86 cores.

Table 1 summarizes applications used in the study. First two columns of the table list application names and their scientific area. The third column lists number of application processes used in the experiments. All the applications were executed on 1024 cores, except Quantum Espresso, which does not scale on more than 256 cores.

Table 1: UEABS applications used in the study

Application	Scientific area	Cores
ALYA	Computational mechanics	1024
BQCD	Particle physics	1024
CP2K	Computational chemistry	1024
GADGET	Astronomy and cosmology	1024
GENE	Plasma physics	1024
GROMACS	Computational chemistry	1024
NEMO	Ocean modeling	1024
Quantum Espresso	Computational chemistry	256

3.2 HPC system simulation

Simulation of HPC applications that comprise thousands of processes is a challenging task. One of the approaches is a trace-driven simulation which includes two steps. First, the application is executed on a real HPC cluster with an instrumentation tool that records the executed instructions into a trace-file. In the second step, the instruction trace is reproduced on a simulator that can mimic various CPU or memory architectures. In our study, HPC servers with DRAM and STT-MRAM main memory were simulated with TaskSim system simulator [22].

3.2.1 Target HPC platform

We collected traces of UEABS applications running on MareNostrum supercomputer [23]. MareNostrum contains 3056 compute nodes (servers) connected with the Infiniband network. Each node contains two Intel Sandy Bridge-EP E5-2670 sockets that comprise eight cores operating at 2.6 GHz. Although Sandy Bridge processors support hyper-threading at core level, this feature is disabled, as in most of the HPC systems. Sandy Bridge processors are connected to main memory through four channels and each channel is connected to a single 4GB DDR3-1600 DIMM.

3.2.2 HPC application behavior

In order to perform complex numerical computations in a reasonable time, HPC applications use numerous simultaneous processes. Trace collection and simulation of entire HPC application that comprises thousands of processes is infeasible. Therefore, first we had to analyze the application structure to detect relatively smaller application segments that are good representative of its overall behavior.

Figure 2 illustrates a visual representation of an HPC application's execution (ALYA). For different application processes (Process 1–1024), the figure shows repetitive appearance of *MPI_Barrier* — the iterating function of the application. At the beginning of the execution (up to approximately 17s in Figure 2), in the *pre-processing* phase, HPC applications divide and distribute input data over a large number of processes. Then, in the application *main loop*, through a series of computation bursts and inter-process communication steps, the intermediate calculations are combined into final results. In production runs of HPC applications, duration of the pre-processing phase is negligible, so the analysis of HPC applications is primarily focused on the main loop. Since the main loop naturally follows repetitive patterns, characterizing of its few iterations is sufficient to characterize the entire application execution [24]. Similarly, most of the processes execute the same algorithm on different data, so, in general, the behavior of a few processes represents the behavior of the entire application.

These properties of scientific HPC applications allowed us to simulate execution of few main-loop iterations of some processes that are a good representative of its overall behavior [24][25]. That way, we avoided producing traces of unmanageable size (in the order of terabytes) and also brought simulation time to a reasonable level. Therefore, before the detailed instruction tracing, we instrumented computation bursts and inter-process communication and analyzed the overall application structure. Computation bursts and inter-process communication were instrumented with Limpio instrumentation framework [26] and the application structure was analyzed with the Paraver visualization tool [27]. Limpio and Paraver are standard tools for this kind of HPC application profiling and analysis.

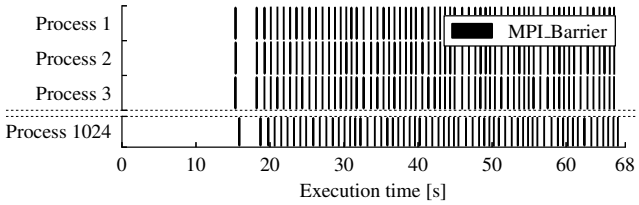

Figure 2: Repetitive behavior of HPC applications: ALYA, 1024 processes

3.2.3 Trace Collection

In order to trace instructions in the selected application segments, we developed a tool with Valgrind instrumentation framework [28]. This tool instruments all the instructions that are executed while extracting only the information required for detailed memory-system simulation.

To simulate non-memory instructions, the tool records the number of instructions that are executed between two consecutive memory operations. To further reduce the trace size of memory instructions, the tool simulates a small 16 KB direct-mapped cache which is referred to as *filter cache* [29][30]. The tool records the number of the filter cache hits and logs detailed information (instruction type, address and data size in bytes) only for instructions that miss the cache. Since dedicated per-core L1 cache of the Sandy Bridge is larger than the trace filter cache, the memory instructions that hit in the filter cache will also hit the L1 cache on the target processor. Therefore, filter cache introduces negligible discrepancies in the simulation of the main memory [31]. On the other hand, as most of the memory instructions hit in the filter cache (more than 90% in our experiments), the resulting trace file is significantly reduced.

All aforementioned approaches for HPC application tracing and trace filtering are validated and regularly used by researchers pursuing similar studies on memory systems [22][24][25][32].

We simulated eight application processes that were executed on a single Sandy Bridge socket. For each process, we traced several main-loop iterations that corresponded to 10–15 seconds of the native execution. To compare DRAM and STT-MRAM memory systems, we measured their performance difference in each main-loop iteration of each process under study. In this paper, we report average slowdown and standard deviation of all the measurements.

3.3 Simulated CPU

In order to evaluate STT-MRAM main memory system, we simulated a socket of MareNostrum-like compute node (see Section 3.2), which is the dominant architecture in HPC systems [33]. The simulated hardware platform is comprised of three distinct segments: CPU pipeline, CPU cache hierarchy and main memory.

3.3.1 CPU pipeline

Since our study proposes no changes in the CPU microarchitecture, we simulate CPU pipeline with a simplified model. The model reproduces the series of CPU (non-memory) instructions using a constant number of cycles per instruction (CPI) [32]. This approach is used for simulation of changes in memory system because it significantly reduces the simulation time with respect to the detailed model of CPU pipeline [22]. We repeat our experiments with three values of CPI: 0.5, 1.0 and 2.0. CPI of 0.5 corresponds to a complex core with strong out-of-order engine that can process two instructions in each cycle. CPI of 1.0 and 2.0 correspond to simpler cores.

3.3.2 Cache memory

The simulated hardware platform comprises a detailed model of Sandy Bridge-EP E5-2670 cache hierarchy [34]. This Sandy Bridge E class processor has eight cores, dedicated L1 instruction and data cache of 32 KB each, dedicated L2 cache of 256 KB and a shared L3 cache of 20 MB. In all three levels of cache memory, we implemented the Least Recently Used (LRU) cache replacement policy. The on-chip cache latencies are detailed in the Sandy Bridge E specification [34], and are summarized in Table 2.

Table 2: Cache parameters of Sandy Bridge E class processor used in the study

	L1-Data	L2	L3
Size	32 KB	256 KB	20 MB
Latency (in CPU cycles)	4	12	31
Cache line size	64 Byte	64 Byte	64 Byte
Set associativity	8 way	8 way	12 way

3.4 Simulated main memory

STT-MRAM main memory simulation is a challenging task because its detailed parameters are not yet standardized and released by industries or academia. In addition to this, to conduct such a simulation correctly, it is essential to estimate also the latency components *before main memory device*. These latency components include not only cache memory hierarchy (detailed in Section 3.3.2), but also the latency of memory controller, memory channel and all the circuitry between the last-level cache and the main memory device itself.

In this study, all memory access latencies were estimated by memory planning group of Samsung Electronics Co., Ltd. The main memory access time in DRAM systems was simulated with 85 ns, from which 15 ns correspond to the DRAM device latency, and the remaining 70 ns account for all the latencies before the memory device — mainly CPU pipeline, cache hierarchy, memory controller and interconnect circuitry. To validate these estimations we measured main memory access time for HPC applications running on a real system — Dual-socket Sandy Bridge E5-2620 server [35], with each socket containing 6 cores and 64 GB of DDR3-1333 main memory.[1] We executed ALYA, GROMACS and NAMD production HPC applications from the Unified European Application Benchmark Suite (UEABS) [21]. The remaining UEABS benchmarks could not be executed because their input datasets exceed the available main memory of the server.

We measured the latency of load instruction with the `perf` tool, along with Precise Event Based Sampling (PEBS) mechanism [34]. The PEBS mechanism samples load instructions and records the number of cycles between the execution of the instruction and actual delivery of the data. The average main memory latency measured in these experiments is 83.6 ns, which closely corresponds to 85 ns used in this study.[2]

Memory planning group of Samsung Electronics Co., Ltd also estimates that the high-density STT-MRAM main memory devices will be approximately 20% slower than conventionally used DRAM. Therefore, the average STT-MRAM access time was simulated with 18 ns (1.2×15 ns DRAM latency), featuring a symmetrical read and write scheme which is in compliance with several scientific studies and products released recently [15][16][36].[3] In addition to 20% slower STT-MRAM device, we performed a sensitivity analysis over this estimate, simulating a pessimistic 50% and 100% device level slowdown, i.e. STT-MRAM devices with average access time of 22.5 ns and 30 ns. The sensitivity analysis is important because it correlates the overall system slowdown trend with respect to device level slowdown, which has not been performed by any previous STT-MRAM main memory studies. We acknowledge the importance of cycle-accurate main memory simulation [37]. However, at this point, this level of details in the STT-MRAM simulation is infeasible due to the lack of *reliable* timing parameters, as we discuss in Section 5.2.

Our study focuses on performance impact of HPC systems with slower STT-MRAM main memory. For the primary assessment, we take DRAM average access latency as the baseline and investigate how the system performance deviates for a specific STT-MRAM device level slowdown. Although we understand the importance of evaluating energy consumption, at this point, such evaluation on energy components of high-density STT-MRAM main memory that would be used in HPC domain is infeasible due to the lack of publicly available up-to-date resources. Simulation of STT-MRAM and DRAM main memory with detailed timing parameters

[1]The experiments were executed on a stand-alone server (not MareNostrum supercomputer) because the software tool for measuring memory access latency requires *root* privileges that we could not obtain on a production HPC cluster.

[2]The average memory latency is application dependent, and it is a subject to the stress that application puts to the memory system — the higher is number of concurrent memory requests (memory bandwidth), the higher the stress to the memory system and the longer the main memory access time [4]. In our experiments the average memory latency ranges from 81 ns (GROMACS) to 87 ns (ALYA).

[3]Memory planning group of Samsung Electronics Co., Ltd also estimates that capacity of high-density STT-MRAM devices will be comparable with DRAM modules. Microarchitecture and detailed timings of Samsung high-density STT-MRAM main memory devices can not be disclosed due to confidentiality issues.

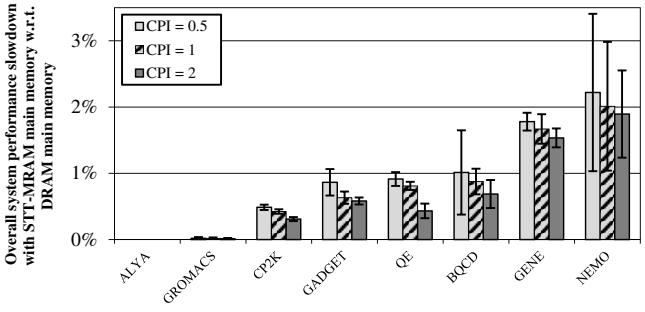

Figure 3: 20% slower STT-MRAM device (industry estimated): Application slowdown ranges from 0% (ALYA) to 2.2% (NEMO), and it is 0.8% on average.

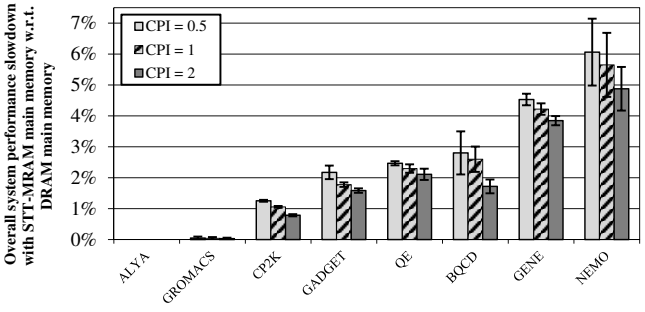

Figure 4: 50% slower STT-MRAM device (pessimistic estimate): Application slowdown ranges from 0% (ALYA) to 6.7% (NEMO), and it is 2.2% on average.

along with estimation of STT-MRAM energy components are parts of our ongoing work.

4. RESULTS

The results of our study are organized into two parts. First, we present the performance comparison of STT-MRAM main memory device being 20% slower than DRAM, corresponding to the recent industry estimation. Then, we present the results of our sensitivity analysis on STT-MRAM main memory performance with a 50% and 100% slower STT-MRAM device.

4.1 Industry estimate: 20% slower STT-MRAM device

The performance comparison of HPC systems with conventional DRAM and STT-MRAM main memory being 20% slower than DRAM, is presented in Figure 3. For each application, different bars correspond to different simulated CPU with CPI of 0.5, 1 and 2. The solid bars represent the average STT-MRAM slowdown, and the error bars show the standard deviation for various application processes and main-loop iterations.

For ALYA and GROMACS, we detect almost no performance difference between STT-MRAM and DRAM main memory systems. Four out of the remaining six applications, CP2K, GADGET, QE and BQCD, experience less than 1% slowdown. Finally, GENE slowdown ranges between 1.5% and 1.8%, while the slowdown of NEMO is around 2%. Overall, the impact of higher STT-MRAM latency on the HPC application performance is very low — for six out of eight applications the slowdown is below 1% and it is only 2.2% in the worst case.

We also analyze the impact of CPU complexity on the performance of STT-MRAM main memory. Processing core with CPI value 0.5 refers to an aggressive core which executes two instructions per cycle, while CPI of 1 and 2 model simpler cores. With an increasing CPI value, we detect slight STT-MRAM performance improvement (lower slowdown w.r.t. DRAM). High-CPI cores increase the time spent in the CPU and the execution time of the application. Therefore, smaller portion of the overall time is spent in the memory and higher STT latency has less impact on the overall performance. However, it is also important to notice that the impact of CPI values on the results is very low — it ranges from 0% for ALYA to only 0.5% for BQCD.

Our analysis identifies three key reasons why yet being

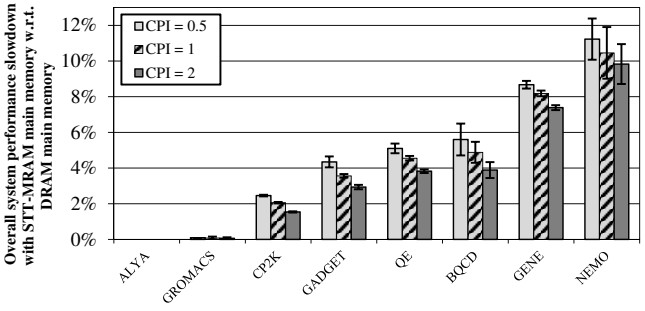

Figure 5: 100% slower STT-MRAM device (extremely pessimistic estimate): Application slowdown ranges from 0% (ALYA) to 11.2% (NEMO), and it is 4.2% on average.

20% slower than DRAM, STT-MRAM main memory yields a negligible impact on overall performance. Firstly, 20% slower STT-MRAM main memory affects only the instructions that access the main memory, which is a fairly small portion of the total instructions. CPU instructions and memory instructions that hit the cache memory are not affected with the slower main memory device. Secondly, main memory device latency constitutes only a portion of the overall main memory access time. The time spent in CPU caches, memory controller, memory channel and the corresponding circuitry does not change when moving from DRAM to STT-MRAM main memory system. And finally, with a out-of-order pipeline, the slowdown of the instructions that access the main memory can be reduced as the processor can execute independent instructions while waiting for data from the main memory.

4.2 Sensitivity analysis: 50% and 100% slower STT-MRAM device

Figure 4 shows the HPC system performance degradation for a STT-MRAM main memory device which is assumed to be 50% slower with respect to DRAM. The results indicate that, ALYA and GROMACS still yields almost no performance penalty, CP2K, GADGET, QE and BQCD introduces less than 3% systems performance slowdown while GENE and NEMO perform around 4% and 5.5% slower, respectively. On average, for a 50% slower STT-MRAM device, overall system performance penalty is 2.2%.

An extremely pessimistic estimation assuming a 100% slower STT-MRAM main memory device also generates a similar chart for HPC performance slowdown, see Figure 5. Even with a 100% slower STT-MRAM device, we observe a negligible system performance impact for ALYA and GROMACS. CP2K, GADGET, QE and BQCD slowdown ranges between 2% to 5%. GENE and NEMO performs around 8% and 10% slower, respectively. The average slowdown of applications is 4.2%.

We analyze the impact of CPU complexity (CPI value of 0.5, 1 and 2) for the 50% and 100% slower STT-MRAM device as well. The results show slight performance improvement for increasing CPI value for both 50% and 100% slower STT-MRAM device. The performance impact for CPI values ranges from 0% for ALYA to 1.2% for NEMO (50% slower STT-MRAM device) and 1.7% for BQCD (100% slower STT-MRAM device).

5. STT-MRAM OPPORTUNITIES AND CHALLENGES

In this section we discuss STT-MRAM's possible advantages in HPC domain and the challenges it faces to be adopted as an alternative main memory technology.

5.1 STT-MRAM opportunities

Some of the STT-MRAM main memory advantages were already analyzed in the context of other non-volatile memory technologies and other application domains [38] [39] [40]. Here, we briefly summarize the ones that are of main interest in the HPC domain.

DRAM refresh: In a DRAM cell, the information is stored as a charge in small capacitors that have to be refreshed periodically in order to preserve the content. DRAM refresh degrades system performance because it interferes with application memory accesses. Also, refresh increases energy consumption, directly, because refresh operations consume energy, and indirectly, because degradation of system performance increases execution time, and therefore overall energy consumption. STT-MRAM is a non-volatile technology and therefore, it requires no refresh. Thus, a performance and energy advantage over DRAM technology can come from resolving the memory refresh problem.

Memory errors: One of the leading causes of hardware failures in modern HPC clusters are main memory DRAM errors [41][42][43][44]. In future, DRAM errors will pose an even larger threat to the reliability of HPC systems. First, the number of memory errors will increase because the amount of DRAM in HPC systems keeps growing at a consistent rate [42]. Another source of increasing the memory error rate is the scaling of the DRAM technology [45]. DRAM cells are getting smaller and they hold a decreasing amount of charge, which makes them more vulnerable to any disturbance and data corruption. Also, the distance between DRAM elements is already so small that electromagnetic coupling causes undesired interactions between the adjacent cells. STT-MRAM is a non-volatile technology that mitigates the transient faults (caused by magnetic or electrical interference), that account for a significant portion of the overall memory faults. Since STT-MRAM technology would improve the reliability of the memory systems, the complexity and overheads of the contemporary error correction approaches can be reduced.

5.2 STT-MRAM challenges

As a novel technology, STT-MRAM faces specific challenges on its way to be a future memory alternative. There are simulation challenges which correspond to the struggle of performing a reliable simulation of the technology, and there are commercial challenges, which refer to the obstacles that is preventing STT-MRAM to appear in the market as a competing main memory technology.

Simulation challenges: To find suitable use cases for STT-MRAM main memory, it is essential to conduct reliable simulation of STT-MRAM. However, simulation of STT-MRAM main memory with detailed timing parameters has been a challenging task due to the unavailability of reliable estimation of timing parameters. Only three studies simulate and analyze STT-MRAM main memory. Table 3 summarizes timing parameters used in these studies — parameters of main memory devices (DRAM and STT-MRAM) along with *before main memory device* latency.

Meza *et al.* [18] use a cycle-accurate DDR3-DRAM memory simulator and estimate STT-MRAM parameters based on Fujitsu's 16kb test-chip built in 2010 with 0.13μm technology [46]. The authors assume, that t_{WR} and t_{RCD} parameters for STT-MRAM main memory would change on a range of twice as slow to twice as fast with respect to DRAM. In our opinion, it is difficult to estimate a reasonable assessment of STT-MRAM main memory using such a wide range of values for key latency parameters. The study, also does not provide any information about latency components before main memory device, making it it infeasible to repeat the study or to quantify the impact of the STT-MRAM t_{WR} and t_{RCD} parameters to the *overall* main memory access latency.

Kultursay *et al.* [19] compare DRAM and STT-MRAM performance by simulating fixed latencies for row buffer hit (30 ns) and conflict (50 ns) without specifying the breakdown or inclusion of latency components for this delay. The source of these estimations are not revealed in the paper. The authors also state that they modified CACTI to model STT-MRAM, however there is no information how this modification was formulated, taking into account the fact that CACTI is widely used as a cache memory simulator, but least likely to be used to simulate main memories. The study also proposes an additional 10 ns penalty for STT-MRAM write, which as an obsolete parameter used in early STT-MRAM designs. Practically all recent studies and commercial products suggest STT-MRAM cells with symmetrical (same latency) read and write operations [15][16][36].

Suresh *et al.* [20] simulate STT-MRAM read and write operations with fixed latency of 35 ns, obtaining these estimation from ITRS report, 2013 [47]. The study provides no information about the latency components before main memory device, or DRAM device latencies.

To summarize, previous studies use obsolete STT-MRAM timing parameters or the parameters with no reliable source. In addition to this, the before main memory device latency is not validated versus real systems, or it is directly omitted from the simulation infrastructure analysis. Our work tries to advance previous studies by performing the sensitivity analysis that correlates overall system slowdown trend with respect to device-level slowdown. However, STT-MRAM main memory evaluation is incomplete without cycle-accurate simulation with reliable timing parameters. The lack of detailed timing parameters is also

Table 3: STT-MRAM simulation parameters. Previous studies use obsolete STT-MRAM parameters or the parameters with no available source. The *before main memory device* latency is not validated versus real systems, or it is directly omitted from the simulation infrastructure analysis.

Study	Memory access latency				Observations
	Before memory device	Main memory device			
		DRAM	STT Read	STT Write	
Meza et. al [18]	No information	Cycle accurate simulation	t_{WR} and t_{RCD}: $0.5 \leqslant \frac{STT}{DRAM} \leqslant 2$		- Latency before main memory: No info - Wide range of t_{WR} and t_{RCD} values.
Kultursay et. al [19]	Row buffer hit: 30 ns Row buffer miss: 50 ns		+0 ns	+10 ns	- Main memory latency: No source - Obsolete STT parameters.
Suresh et. al [20]	No information		35 ns	35 ns	- Latency before main memory: No info - DRAM latency: No info
This study	70 ns	15 ns	18 ns	18 ns	- Latency before main memory: Estimated by industry, validated in real HPC system. - STT latency estimated by industry - STT latency: Sensitivity analysis

the main problem for any STT-MRAM microarchitectural exploration, improvement and evaluation.

Commercial challenges: We summarize overall comparison between DRAM and STT-MRAM main memory targeting HPC market in Table 4. As it can be seen from the table, STT-MRAM main memory would provide performance and capacity comparable to DRAM systems, while opening up various opportunities for HPC system improvements. However, its adoption as alternative main memory technology is limited due its high production cost as compared to DRAM — a mature technology with huge production volumes. Therefore, if we really want to make STT-MRAM an alternative to DRAM in main memory systems, we have to find domains and use cases so that STT-MRAM primary development cost can be justified with *significant* improvements in features of interest.

Table 4: Overall comparison of DRAM and STT-MRAM main memory in HPC systems.

Attribute	DRAM	STT-MRAM
Performance	Comparable	
Capacity	Comparable	
Refresh-less	No	Yes
Permanent Memory	No	Yes
Resiliency	Low	High
Maturity of technology	Mature	Novel
Production volume	Very high	Very low
Production cost	Very low	High

6. RELATED WORK

6.1 STT-MRAM main memory

To the best of our knowledge, only three studies analyze suitability of STT-MRAM for main memory.

Meza *et al.* [18] analyze architectural changes to enable small row buffers in non-volatile memories, PCM, STT-MRAM, and RRAM. The study concludes that NVM main memories with reduced row buffer size can achieve up to 67% energy gain over DRAM at a cost of some performance degradation.

Kultursay *et al.* [19] evaluate STT-MRAM as a main memory for SPEC CPU2006 workloads and show that, without any optimizations, early-design STT-MRAM [48] is not competitive with DRAM. The authors also propose *partial write* and *write bypass* optimizations that address time and energy-consuming STT-MRAM write operation. Optimized STT-MRAM main memory achieves performance comparable to DRAM while reducing memory energy consumption by 60%.

Suresh *et al.* [20] analyze design of memory systems that match the requirements of data intensive HPC applications with large memory footprints. The authors propose a complex 5-level memory hierarchy with SRAM caches, EDRAM or HMC last level cache, and non-volatile PCM, STT-MRAM, or FeRAM main memory. The study also analyzes using a small DRAM off-chip cache that filters most of the accesses to the non-volatile main memory and therefore reduces a negative impact on performance and dynamic energy consumption of NVM technologies.

Our study evaluates STT-MRAM main memory for high-performance computing (HPC) and analyzes the performance impact when DRAM is simply replaced with STT-MRAM. The presented results suggests that 20% slower STT-MRAM main memory induces negligible system performance impact, while opening up opportunities to provide some highly desired properties such as non-volatility, zero stand-by power and high endurance.

6.2 STT-MRAM on-chip caches

Advantages of STT-MRAM over SRAM motivated numerous studies to analyze STT-MRAM as cache memory.

Li *et al.* [49] propose to integrate STT-MRAM with SRAM to construct a hybrid adaptive on-chip cache architecture that offers low power consumption, low access latency and high capacity. The authors evaluate hybrid SRAM / STT-MRAM cache on a set of PARSEC and SPLASH-2 workloads, and report a 37% reduction of power consumption along with 23% performance improvement compared to SRAM based design.

Zhou *et al.* [50] observe that many bits in the STT-MRAM

cache are re-written with the same value. As, early STT-MRAM cell design write operation requires significant energy, such unnecessary writes can be avoided to reduce power consumption. They introduce *early write termination*, a scheme which terminates redundant bit writes for STT-MRAM caches and achieves upto 80% of write energy reduction for SPEC 2000, SPEC 2006 and SPLASH-2 benchmarks.

Chang *et al.* [51] compares STT-MRAM and eDRAM as a replacement of SRAM for last level caches. The study identifies specific weaknesses of each technology and analyzes the trade-offs associated with each of these technologies for implementing last level caches. The study concludes, if refresh is effectively controlled, eDRAM based last level cache becomes a viable, energy-efficient alternative for multi-core processors.

Various studies propose to trade-off STT-MRAM's non-volatility to improve write latency and energy consumption [52][53][54][55]. Li *et al.* [53] indicate that majority of cache data stay active for much shorter time duration than the data retention time assumed in the STT-MRAM designs. The authors suggest that, the retention time can be aggressively reduced to achieve significant switching performance and power improvements. Jog *et al.* [54] formulate the relation between retention time and write latency in order to find optimal retention time for an efficient STT-MRAM cache hierarchy. Smullen *et al.* [52] propose a ultra-low retention time STT-MRAM caches supported by a DRAM-like refresh policy. Sun *et al.* [55] further exploit the scenario by deploying STT-MRAM with multiple retention levels. Smullen *et al.* [52] and Sun *et al.* [55] propose architectures with SRAM L1 cache along with relaxed-retention STT-MRAM L2 and L3 cache. The hybrid cache architectures are evaluated on SPEC 2006 and PARSEC benchmarks and they show significant performance improvement over conventional SRAM-based designs while reducing energy consumption.

The studies perform analysis of STT-MRAM cache latencies, area, leakage and dynamic power based on publicly available STT-MRAM cell parameters and CACTI [56]. Unfortunately, these STT-MRAM timing and energy parameters could not be used to simulate main memory because such devices have higher capacity (by several orders of magnitude), different organization (DIMMs, ranks, banks, chips, rows, columns) and interface (e.g. row buffer), which would yield a completely different set of values for STT-MRAM main memory.

7. CONCLUSIONS

In this study, we conduct a preliminary analysis whether STT-MRAM is a candidate for future HPC memory systems. We model STT-MRAM main memory latency using recent industry estimation and incorporate it into the overall simulation of the HPC system executing production applications. Although STT-MRAM is significantly slower than DRAM at the device level, it provides performance comparable to conventional systems, while opening up various opportunities for HPC system improvements. Exploration of these opportunities as well as any further research on STT-MRAM main memory, however, is conditioned by a release of reliable timing parameters. Finally, although STT-MRAM has a potential as alternative main memory technology, the extent of its adoption will likely be limited by its high production cost. Therefore, future of STT-MRAM main memory depends mainly on whether we find domains and use cases in which its cost can be justified with significant improvements in other features of interest.

8. ACKNOWLEDGMENTS

This work was supported by the Collaboration Agreement between Samsung Electronics Co., Ltd. and BSC, Spanish Government through Programa Severo Ochoa (SEV-2015-0493), by the Spanish Ministry of Science and Technology through TIN2015-65316-P project and by the Generalitat de Catalunya (contracts 2014-SGR-1051 and 2014-SGR-1272). This work has also received funding from the European Union's Horizon 2020 research and innovation programme under ExaNoDe project (grant agreement No 671578).

9. REFERENCES

[1] Peter Kogge, Keren Bergman, Shekhar Borkar, Dan Campbell, William Carlson, William Dally, Monty Denneau, Paul Franzon, William Harrod, Kerry Hill, Jon Hiller, Sherman Karp, Stephen Keckler, Dean Klein, Robert Lucas, Mark Richards, Al Scarpelli, Steven Scott, Allan Snavely, Thomas Sterling, R. Stanley Williams, and Katherine Yelick. ExaScale Computing Study: Technology Challenges in Achieving Exascale Systems, September 2008.

[2] Avinash Sodani. Race to Exascale: Opportunities and Challenges. Keynote Presentation at the 44th Annual IEEE/ACM International Symposium on Microarchitecture (MICRO), 2011.

[3] Rick Stevens, Andy White, Pete Beckman, Ray Bair-ANL, Jim Hack, Jeff Nichols, Al GeistORNL, Horst Simon, Kathy Yelick, John Shalf-LBNL, Steve Ashby, Moe Khaleel-PNNL, Michel McCoy, Mark Seager, Brent Gorda-LLNL, John Morrison, Cheryl Wampler-LANL, James Peery, Sudip Dosanjh, Jim Ang-SNL, Jim Davenport, Tom Schlagel, BNL, Fred Johnson, and Paul Messina. A Decadal DOE Plan for Providing Exascale Applications and Technologies for DOE Mission Needs. Presentation at Advanced Simulation and Computing Principal Investigators Meeting, March 2010.

[4] B. Jacob. *The Memory System: You Can't Avoid It; You Can't Ignore It; You Can't Fake It. M.* Morgan & Claypool Publishers, Reading, Massachusetts, 2009.

[5] Wm. A. Wulf and Sally A. McKee. Hitting the memory wall: Implications of the obvious. *SIGARCH Comput. Archit. News*, 1995.

[6] B. Dieny, V. S. Speriosu, S. S. P. Parkin, B. A. Gurney, D. R. Wilhoit, and D. Mauri. Giant magnetoresistive in soft ferromagnetic multilayers. *Phys. Rev. B*, 1991.

[7] J.K. Spong, Speriosu, Robert E. Fontana, Moris M. Dovek, and T.L. Hylton. Giant Magnetoresistive Spin Valve Bridge Sensor. *IEEE Transactions on Magnetics*, 1996.

[8] J. A. Katine, F. J. Albert, R. A. Buhrman, E. B. Myers, and D. C. Ralph. Current-Driven Magnetization Reversal and Spin-Wave Excitations in Co /Cu /Co Pillars. *Phys. Rev. Lett.*, 2000.

[9] M. Hosomi, H. Yamagishi, T. Yamamoto, K. Bessho, Y. Higo, K. Yamane, H. Yamada, M. Shoji,

H. Hachino, C. Fukumoto, H. Nagao, and H. Kano. A Novel Nonvolatile Memory with Spin Torque Transfer Magnetization Switching: Spin-RAM. In *IEEE International Electron Devices Meeting*, 2005.

[10] Yuan Xie. Modeling, Architecture, and Applications for Emerging Memory Technologies. *IEEE Design Test of Computers*, 2011.

[11] C. Kim, D. Kang, H. Kim, C.W. Park, D.H. SOHN, Y.S. Lee, S. Kang, H.R. Oh, and S. Cha. Magnetic random access memory, 2013. US Patent App. 13/768,858.

[12] H. Kim, S.K. Kang, D.H. SOHN, D.M. Kim, and K.C. Lee. Magneto-resistive memory device including source line voltage generator, 2013. US Patent App. 13/832,101.

[13] H.R. Oh. Resistive Memory Device, System Including the Same and Method of Reading Data in the Same, 2014. US Patent App. 14/094,021.

[14] K. Abe, H. Noguchi, E. Kitagawa, N. Shimomura, J. Ito, and S. Fujita. Novel Hybrid DRAM/MRAM Design for Reducing Power of High Performance Mobile CPU. In *IEEE International Electron Devices Meeting (IEDM)*, 2012.

[15] H. Noguchi, K. Kushida, K. Ikegami, K. Abe, E. Kitagawa, S. Kashiwada, C. Kamata, A. Kawasumi, H. Hara, and S. Fujita. A 250-MHz 256b-I/O 1-Mb STT-MRAM with Advanced Perpendicular MTJ Based Dual cell for Nonvolatile Magnetic Caches to Reduce Active Power of Processors. In *Symposium on VLSI Technology (VLSIT)*, 2013.

[16] R. Nebashi, N. Sakimura, H. Honjo, S. Saito, Y. Ito, S. Miura, Y. Kato, K. Mori, Y. Ozaki, Y. Kobayashi, N. Ohshima, K. Kinoshita, T. Suzuki, K. Nagahara, N. Ishiwata, K. Suemitsu, S. Fukami, H. Hada, T. Sugibayashi, and N. Kasai. A 90nm 12ns 32Mb 2T1MTJ MRAM. In *IEEE International Solid-State Circuits Conference*, 2009.

[17] Everspin Technologies, Inc. Everspin Embedded MRAM. http://www.everspin.com/everspin-embedded-mram, 2015.

[18] Jing Li Justin Meza and Onur Mutlu. Evaluating Row Buffer Locality in Future Non-Volatile Main Memories. *Safari Technical Report No. 2012-002*, 2012.

[19] E. Kultursay, M. Kandemir, A. Sivasubramaniam, and O. Mutlu. Evaluating STT-RAM as an Energy-Efficient Main Memory Alternative. In *IEEE International Symposium on Performance Analysis of Systems and Software (ISPASS)*, 2013.

[20] A. Suresh, P. Cicotti, and L. Carrington. Evaluation of Emerging Memory Technologies for HPC, Data Intensive Applications. In *IEEE International Conference on Cluster Computing (CLUSTER)*, 2014.

[21] PRACE. *Unified European Applications Benchmark Suite*, 2013.

[22] Alejandro Rico, Felipe Cabarcas, Carlos Villavieja, Milan Pavlovic, Augusto Vega, Yoav Etsion, Alex Ramirez, and Mateo Valero. On the Simulation of Large-scale Architectures Using Multiple Application Abstraction Levels. *ACM Trans. Archit. Code Optim. (TACO)*, 2012.

[23] Barcelona Supercomputing Center. MareNostrum III System Architecture. http://www.bsc.es/marenostrum-support-services/mn3, 2013.

[24] R. Preissl, T. Kockerbauer, M. Schulz, D. Kranzlmuller, B. Supinski, and D.J. Quinlan. Detecting Patterns in MPI Communication Traces. In *37th International Conference on Parallel Processing*, 2008.

[25] L. Alawneh and A. Hamou-Lhadj. Identifying Computational Phases from Inter-Process Communication Traces of HPC Applications. In *IEEE 20th International Conference on Program Comprehension*, 2012.

[26] Milan Pavlovic, Milan Radulovic, Alex Ramirez, and Petar Radojkovic. Limpio — LIghtweight MPI instrumentatiOn. In *IEEE 23rd International Conference on Program Comprehension (ICPC)*, 2015. http://www.bsc.es/computer-sciences/computer-architecture/memory-systems/limpio.

[27] Barcelona Supercomputing Center. Paraver. http://www.bsc.es/computer-sciences/performance-tools/paraver, 2015.

[28] Valgrind. http://valgrind.org/, 2015.

[29] Thomas Roberts Puzak. *Analysis of Cache Replacement Algorithms*. PhD thesis, 1985.

[30] Wen-Hann Wang and Jean-Loup Baer. Efficient Trace-driven Simulation Methods for Cache Performance Analysis. *ACM Trans. Comput. Syst.*, 1991.

[31] Richard A. Uhlig and Trevor N. Mudge. Trace-driven Memory Simulation: A Survey. *ACM Comput. Surv.*, 1997.

[32] M. Pavlovic, N. Puzovic, and A. Ramirez. Data Placement in HPC Architectures with Heterogeneous Off-Chip Memory. In *IEEE 31st International Conference on Computer Design (ICCD)*, 2013.

[33] Top500. Top500 Supercomuter Sites. http://www.top500.org/, 2015.

[34] Intel. Intel® 64 and IA-32 Architectures Optimization Reference Manual. http://www.intel.com/content/www/us/en/architecture-and-technology/64-ia-32-architectures-optimization-manual.html, 2015.

[35] Supermicro. SuperServer 6017R-WRF. www.supermicro.com/products/system/1U/6017/SYS-6017R-WRF.cfm, 2015.

[36] Everspin Technologies, Inc. Everspin Enhances RIM Smart Meters with Instantly Non-Volatile, Low-Energy MRAM Memory. http://www.everspin.com/everspin-embedded-mram, 2015.

[37] David Wang, Brinda Ganesh, Nuengwong Tuaycharoen, Kathleen Baynes, Aamer Jaleel, and Bruce Jacob. Dramsim: A memory system simulator. *SIGARCH Comput. Archit. News*, 33(4), 2005.

[38] Dong Li, Jeffrey S. Vetter, Gabriel Marin, Collin McCurdy, Cristian Cira, Zhuo Liu, and Weikuan Yu. Identifying Opportunities for Byte-Addressable

Non-Volatile Memory in Extreme-Scale Scientific Applications. In *Proceedings of the 26th IEEE International Parallel and Distributed Processing Symposium (IPDPS)*, 2012.

[39] A.M. Caulfield, J. Coburn, T. Mollov, A. De, A. Akel, Jiahua He, A. Jagatheesan, R.K. Gupta, A. Snavely, and S. Swanson. Understanding the Impact of Emerging Non-Volatile Memories on High-Performance, IO-Intensive Computing. In *International Conference for High Performance Computing, Networking, Storage and Analysis (SC)*, 2010.

[40] Jeffrey S. Vetter and Sparsh Mittal. Opportunities for nonvolatile memory systems in extreme-scale high performance computing. *Computing in Science and Engineering special issue*, 2015.

[41] Bianca Schroeder, Eduardo Pinheiro, and Wolf-Dietrich Weber. DRAM Errors in the Wild: A Large-scale Field Study. In *11th International Joint Conference on Measurement and Modeling of Computer Systems*, 2009.

[42] Andy A. Hwang, Ioan Stefanovici, and Bianca Schroeder. Cosmic Rays Don't Strike Twice: Understanding the Nature of DRAM Errors and the Implications for System Design. In *17th International Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS)*, 2012.

[43] Vilas Sridharan and Dean Liberty. A Study of DRAM Failures in the Field. In *International Conference on High Performance Computing, Networking, Storage and Analysis*, 2012.

[44] Vilas Sridharan, Jon Stearley, Nathan DeBardeleben, Sean Blanchard, and Sudhanva Gurumurthi. Feng Shui of Supercomputer Memory: Positional Effects in DRAM and SRAM Faults. In *International Conference on High Performance Computing, Networking, Storage and Analysis*, 2013.

[45] Yoongu Kim, R. Daly, J. Kim, C. Fallin, Ji Hye Lee, Donghyuk Lee, C. Wilkerson, K. Lai, and O. Mutlu. Flipping Bits in Memory without Accessing them: An Experimental Study of DRAM Disturbance Errors. In *ACM/IEEE 41st International Symposium on Computer Architecture (ISCA)*, 2014.

[46] D. Halupka, S. Huda, W. Song, A. Sheikholeslami, K. Tsunoda, C. Yoshida, and M. Aoki. Negative-Resistance Read and Write Schemes for STT-MRAM in 0.13um CMOS. In *IEEE International Solid State Circuits Conference*, 2010.

[47] ITRS. 2013 International Technology Roadmap for Semiconductors. http://www.itrs.net/Links/2013ITRS/Home2013.htm, 2015.

[48] Guangyu Sun, Xiangyu Dong, Yuan Xie, Jian Li, and Yiran Chen. A Novel Architecture of the 3D Stacked MRAM L2 Cache for CMPs. In *IEEE 15th International Symposium on High Performance Computer Architecture (HPCA)*, 2009.

[49] Jianhua Li, C.J. Xue, and Yinlong Xu. STT-RAM Based Energy-Efficiency Hybrid Cache for CMPs. In *IEEE/IFIP 19th International Conference on VLSI and System-on-Chip (VLSI-SoC)*, 2011.

[50] Ping Zhou, Bo Zhao, Jun Yang, and Youtao Zhang. Energy Reduction for STT-RAM Using Early Write Termination. In *IEEE/ACM International Conference on Computer-Aided Design - Digest of Technical Papers*, 2009.

[51] M. T. Chang and P. Rosenfeld and S. L. Lu and B. Jacob. Technology comparison for large last-level caches (L3Cs): Low-leakage SRAM, low write-energy STT-RAM, and refresh-optimized eDRAM. In *IEEE 19th International Symposium on High Performance Computer Architecture (HPCA2013), 2013*, 2013.

[52] C.W. Smullen, V. Mohan, A. Nigam, S. Gurumurthi, and M.R. Stan. Relaxing non-volatility for fast and energy-efficient STT-RAM caches. In *IEEE 17th International Symposium on High Performance Computer Architecture (HPCA)*, 2011.

[53] Hai Li, Xiaobin Wang, Zhong-Liang Ong, Weng-Fai Wong, Yaojun Zhang, Peiyuan Wang, and Yiran Chen. Performance, Power, and Reliability Tradeoffs of STT-RAM Cell Subject to Architecture-Level Requirement. *IEEE Transactions on Magnetics*, 2011.

[54] A. Jog, A.K. Mishra, Cong Xu, Yuan Xie, V. Narayanan, R. Iyer, and C.R. Das. Cache Revive: Architecting Volatile STT-RAM Caches for Enhanced Performance in CMPs. In *49th ACM/EDAC/IEEE Design Automation Conference (DAC)*, 2012.

[55] Zhenyu Sun, Xiuyuan Bi, Hai (Helen) Li, Weng-Fai Wong, Zhong-Liang Ong, Xiaochun Zhu, and Wenqing Wu. Multi Retention Level STT-RAM Cache Designs with a Dynamic Refresh Scheme. In *44th Annual IEEE/ACM International Symposium on Microarchitecture (MICRO)*, 2011.

[56] Naveen Muralimanohar, Rajeev Balasubramonian, and Norman P. Jouppi. CACTI 6.0: A Tool to Understand Large Caches. *HP Technical Report HPL-2009-85*, 2009.