UNIVERSITAT POLITÈCNICA
DE CATALUNYA
BARCELONATECH

# *Evaluation of STT-MRAM main memory for HPC and real-time systems*


**by**

**Kazi ASIFUZZAMAN**

UNIVERSITAT POLITÈCNICA DE CATALUNYA

DOCTORAL THESIS

---

# Evaluation of STT-MRAM Main Memory for HPC and Real-time Systems

---

by

Kazi ASIFUZZAMAN

*Supervisor:*
Dr. Petar RADOJKOVIĆ

*Tutor:*
Dr. Eduard AYGUADÉ PARRA

*A thesis submitted in fulfillment of the requirements for the degree of Doctor of Philosophy*

*in the*

Department of Computer Architecture

2019

*"You never fail until you stop trying."*

Albert Einstein

UNIVERSITAT POLITÈCNICA DE CATALUNYA

# *Abstract*

Department of Computer Architecture

Doctor of Philosophy

**Evaluation of STT-MRAM Main Memory for HPC and Real-time Systems**

by Kazi ASIFUZZAMAN

It is questionable whether DRAM 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 an emerging non-volatile memory with a lot of potential that could be exploited for various requirements of different computing systems. Being a novel technology, STT-MRAM devices are already approaching DRAM in terms of capacity, frequency and device size. Special STT-MRAM features such as intrinsic radiation hardness, non-volatility, zero stand-by power and capability to function in extreme temperatures also make it particularly suitable for aerospace, avionics and automotive applications. Intensified efforts in STT-MRAM research by the memory manufacturers may indicate a revolution with STT-MRAM memory technology is imminent, and therefore, it is now the time to explore computing domains that can benefit from this technology. Although STT-MRAM technology got significant attention of various major memory manufacturers, to this day, academic research of STT-MRAM main memory remains marginal. This is mainly due to the unavailability of publicly available detailed timing and current parameters of this novel technology, which are required to perform a reliable main memory simulation on performance and power estimation.

This thesis demonstrates an approach to perform a cycle accurate simulation of STT-MRAM main memory, being the first to release detailed timing and current parameters of this technology from academia — essentially enabling researchers to conduct reliable system level simulation of STT-MRAM using widely accepted existing simulation infrastructure. The results show a fairly narrow overall performance deviation in response to significant variations in key timing parameters. The power consumption experiments also identify the key power component that is mostly affected with STT-MRAM. Furthermore, the thesis analyzes the feasibility of using STT-MRAM in real-time embedded systems by investigating STT-MRAM main memory impact on average system performance and Worst Case Execution Time (WCET). The results show that systems comprising STT-MRAM main memory can be analyzed with the same WCET approaches used in the systems with conventional DRAM. In quantitative terms, the results suggest STT-MRAM main memory in real-time embedded systems provides performance and WCET comparable to conventional DRAM, while opening up opportunities to exploit various advantages.

Overall, this thesis presents the first comprehensive exploration of possibilities to use STT-MRAM in HPC and real-time embedded domain and reveals that STT-MRAM would provide performance and WCET estimates comparable to DRAM while opening up several key advantages that the domains could benefit from.

*I dedicate this thesis*
*to the memory of my dear friend*
*Afnan Hadi Shourov*
*(1984 - 2011)*

# *Acknowledgements*

I start by thanking my supervisors Dr. Petar Radojković and Prof. Eduard Ayguadé Parra for their sincere support and continuous guidance during my thesis. I am truly privileged to conduct my Phd studies under their supervision and leadership.

I can not thank Petar enough for facilitating close supervision and exceptional inspiration, without which it would have been extremely difficult to pass over the challenging path of my PhD studies. He has been an outstanding mentor and leader all the way — assisting me in every obstacle I faced during the course of my PhD. Being a dynamic team leader, he connected me to the people and resources efficiently to solve each and every problem that came on our path. Petar also provided me enough scope to take decisions and grow my skills to be an independent researcher. I genuinely feel honored to pursue my PhD under his supervision.

I would like to thank Barcelona Supercomputing Center (BSC) for accepting me to be a part this prestigious institution. While consistent funding is a common challenge for all PhD students around the world, BSC relieved me from this concern, and allowed me to focus on my work that really matters. I am also grateful to BSC for providing me the office space, equipments and access to its state-of-the-art supercomputing resources. I would like to thank all staff and employees from different groups and departments of BSC with whom I interacted, for their sincere support and assistance. Special thanks to my colleagues Milan Pavlovic, Renan Fischer E Silva, Milan Radulovic and Rommel Sánchez Verdejo for their help and suggestions in professional and practical matters during my stay in Barcelona.

I thank the Department of Computer Architecture (DAC) of Universitat Politècnica de Catalunya (UPC) for carrying out the administrative procedures, providing access to its resources, forming academic committees for evaluation in different stages of the PhD thesis.

I am indebted to my father, Kazi Aktaruzzaman, who encourages me to be ambitious, prepares me to be capable to chase that ambition and supports me unconditionally on all of my endeavors. I would like to express my gratitude to my mother and brothers for always being there and assisting me in every step in my life. I thank my friends back home for their unconditional love

and support all the way. I owe a special thanks to Milladur Rahman (and family) for backing me up during the most difficult time of my life.

It will never be enough to thank my wife Zeba and daughter Yalina for their sacrifices to make this dissertation a reality. Zeba is an extraordinarily kind and cooperative woman with great capability to bring the best out of anyone. She goes an extra mile to accommodate my obligations, making way for me to progress. Without her support and encouragement, it would have been extremely difficult to complete the thesis.

This work is 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). The author wish to thank Terry Hulett, Duncan Bennett and Ben Cooke from Everspin Technologies Inc., for technical support.

# Contents

# List of Figures

# List of Tables

# Chapter 1

# Introduction

Today, we live our life depending on a range of computing systems from smartphones, desktop computers, laptops, embedded systems to high performance computing (HPC) systems. Most people of the society are direct or indirect users of these systems, and such systems provide the crucial support to uphold the modern and efficient lifestyle. All of these computing systems innately incorporate some form of memory systems which are indispensable for their operation and functionality. In modern computing systems, *Memory* usually refers to the *Main Memory* of the system, which holds and provides data to perform computation. Since processor design has undergone an unprecedented development in recent years and subsequently expanded the gap between processor and memory speed, it made the memory system a vital design aspect to improve performance.

For many years, DRAM devices have been the dominant building blocks for main memory systems in most computing systems including desktop, laptop, smartphones and HPC systems. DRAM technology became matured over decades of persistent research and development. For production in mass volume, DRAM has also become a very affordable technology for deployment across all computing domains. However, for a long time researchers have so heavily concentrated on improving processor and cache design that, improving the main memory itself has been far-overlooked. As a result, we are *hitting the memory wall*, which would not be solved unless main memory technology is significantly improved.

Perhaps the greatest challenge that DRAM faces is technology scaling. Extreme scaling is forcing to shrink the size of the DRAM capacitor in an unprecedented ratio making it increasingly vulnerable to errors. Due to several design challenges, this approach of DRAM scaling is not expected to sustain

forever. We need innovations, new methods and perhaps new technologies to move forward to meet next-generation computing needs.

Therefore, significant effort is invested in research and development of novel memory technologies. Among the emerging memory technologies notable candidates are Phase Change Memory (PCM), Resistive RAM (RRAM) and Spin-Transfer Torque Magnetic Random Access Memory (STT-MRAM), along with the 3D-stacked variants of DRAM namely Hybrid Memory Cube (HMC) and High Bandwidth Memory (HBM). Although all of these technologies have their own advantages, two of the major contenders, PCM and RRAM have significantly slower access time in comparison to DRAM, which makes them unsuitable to be considered as a main memory alternative. Since HMC and HBM are practically based on DRAM technology, STT-MRAM stands out to be most promising novel memory technology which has a true potential to be a main memory alternative. 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 require research on the circuit level — involving geometry of the cells and physical properties of their composing materials.

Although STT-MRAM technology was introduced only fourteen years ago [1], STT-MRAM devices are already approaching DRAM in terms of capacity, frequency and device size. Actually, various STT-MRAM commercial products already found their way to some segments of the memory market [2]. Therefore, now is the time to perform system level research to explore use-cases and identify computing domains that could benefit from this technology. System-level research can also detect key STT-MRAM limitations, and estimate their impact on overall system performance and power consumption.

Special STT-MRAM features such as intrinsic radiation hardness, non-volatility, zero stand-by power and capability to function in extreme temperatures also makes it particularly suitable for aerospace, avionics and automotive applications. Such applications often have real-time requirements – that is, certain tasks must complete within a strict deadline. Analyzing whether this *deadline* is met requires Worst Case Execution Time (WCET) Analysis, which is a fundamental part of evaluating any real-time system. Therefore, it is also interesting to see how STT-MRAM would function and comply with the timing

requirements of real-time applications through system level analysis.

STT-MRAM technology has recently got significant attention of various major memory manufacturers; however, academic pursuance to conduct system-level research on this technology is still marginal, mainly due to the lack of publicly available, detailed, and reliable timing parameters of STT-MRAM. These timing parameters are essential to conduct a system level simulation. Some researchers adopt simplistic memory models to simulate main memory, but such models can introduce significant errors in the analysis of the overall system performance [3][4]. Therefore, detailed timing parameters are a *must-have* for any evaluation or architecture exploration study of STT-MRAM main memory. However, these detailed parameters are not publicly available because STT-MRAM manufacturers are reluctant to release any delicate information on the technology. Also, being a rapidly evolving technology, it is difficult even for the manufacturers to predict the exact timing for an upcoming STT-MRAM main memory device.

This thesis attempts to (1) propose a way to reliably model STT-MRAM with existing simulation infrastructures enabling academia to simulate this technology; (2) include STT-MRAM detailed timing parameters to the main repository of two most widely accepted main memory simulators; (3) evaluate system performance impact and power consumption of STT-MRAM in HPC systems; and (4) analyze the performance and WCET implication of STT-MRAM for real-time embedded systems.

## 1.1 Thesis contribution

The main contributions of the thesis are summarized below:

- Demonstrate the approach to perform a cycle accurate simulation of STT-MRAM main memory, being the first to release detailed timing and current parameters of this technology from academia — essentially enabling researchers to conduct reliable system level simulation of STT-MRAM using widely accepted existing simulation infrastructure. The approach that we present converged through research cooperation with Everspin technologies Inc., one of the leading MRAM manufacturers, and it provides reliable STT-MRAM timing parameters while releasing

no confidential information about any commercial products. We seamlessly incorporate our STT-MRAM timing and power analysis into the DRAMSim2 [3] memory simulator and use it as a part of the simulation infrastructure of the high performance computing systems running the SPEC 2006 benchmark suite. Our results show a fairly narrow overall performance deviation in response to significant variations in key timing parameters. The results also identify the key power component that is mostly affected with STT-MRAM.

- Include detailed STT-MRAM main memory timing parameters into the main repositories of DramSim2 [3] and Ramulator [5], two of the most widely used and accepted state-of-the-art main memory simulators. The STT-MRAM timing parameters that have originated as a part of this thesis, are till date the only reliable and publicly available timing information on this memory technology published from academia.

- Evaluate how HPC system performance is affected by STT-MRAM main memory. To that end, we analyze the performance of production HPC applications running on large-scale clusters with STT-MRAM main memory. 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.

- Analyze the feasibility of using STT-MRAM in real-time embedded systems by investigating STT-MRAM main memory impact on average system performance and WCET. We focus on Cobham Gaisler's NGMP architecture [6] as a representative multicore processor. In a validated simulator, we model STT-MRAM main memory with recently-published detailed timing parameters. STT-MRAM's suitability for the real-time embedded systems is validated on benchmarks provided by the European Space Agency (ESA) and EEMBC Autobench suite [7] by analyzing performance and WCET impact. To portray a broader spectrum of scenarios, we further extend the scope of our experiments executing additional benchmarks with high memory utilization from Mediabench [8]. Our results show that systems comprising STT-MRAM main memory can be analyzed with the same WCET approaches used in the

systems with conventional DRAM. This is a compelling finding, as it is fundamental to reduce STT-MRAM adoption costs, without requiring new tools that must undergo a costly qualification process [9]. In quantitative terms, our results show that STT-MRAM main memory in real-time embedded systems provides performance and WCET comparable to conventional DRAM, while opening up opportunities to exploit various advantages.

In this thesis, we target to evaluate different aspects of using STT-MRAM as the main memory across various computing platforms with a range of workloads to understand its feasibility and effectiveness as a potential future memory system. To achieve this, we extend our experiments into different domains, which requires setting-up and using appropriate simulation infrastructures for each platform under observation. It contributes significant overhead to the workload of the thesis, but it also provides a great opportunity to compare and validate that experiments carried out in different simulation infrastructure do not have significant deviation in outcome. All of the results suggest that STT-MRAM is indeed a viable alternative to DRAM from a performance and WCET perspective, while it opens up opportunities to exploit some highly desired properties such as non-volatility, zero stand-by power, and unlimited endurance.

## 1.2 Publications

In this section we list the research articles related to the thesis that have been published, those which are under submission and publications on other topics that are not a part of this thesis.

### 1.2.1 Conferences

- **Kazi Asifuzzaman**, Milan Pavlovic, Milan Radulovic, David Zaragoza, Ohseong Kwon, Kyung-Chang Ryoo, and Petar Radojković. *Performance Impact of a Slower Main Memory: A Case Study of STT-MRAM in HPC*. In the Proceedings of the Second International Symposium on Memory Systems (MEMSYS), Washington DC, USA, 2016.

- **Kazi Asifuzzaman**, Rommel Sánchez Verdejo, and Petar Radojković. *Enabling a reliable STT-MRAM main memory simulation.* In Proceedings of the Third International Symposium on Memory Systems (MEMSYS), Washington DC, USA, 2017.

### 1.2.2 Under submission

- **Kazi Asifuzzaman**, Mikel Fernandez, Petar Radojković, Jaume Abella and Francisco J. Cazorla. *STT-MRAM for Real-Time Embedded Systems: Performance and WCET Implications*. Under submission.

### 1.2.3 Other publications

- Rommel Sánchez Verdejo, **Kazi Asifuzzaman**, Milan Radulovic, Petar Radojković, Eduard Ayguadé, and Bruce Jacob. *Main Memory Latency Simulation: The Missing Link*, in Proceedings of the Fourth International Symposium on Memory Systems (MEMSYS), Washington DC, USA, 2018.

- Milan Radulovic, **Kazi Asifuzzaman**, Darko Zivanovic, Nikola Rajovic, Guillaume Colin de Verdiére, Dirk Pleiter, Manolis Marazakis, Nikolaos Kallimanis, Paul Carpenter, Petar Radojković, and Eduard Ayguadé. *Mainstream vs. Emerging HPC: Metrics, Trade-offs and Lessons Learned*, in 30th International Symposium on Computer Architecture and High Performance Computing (SBAC-PAD), Lyon, France, 2018.

- Milan Radulovic, **Kazi Asifuzzaman**, Paul Carpenter, Petar Radojković, and Eduard Ayguadé. *HPC Benchmarking: Scaling Right and Looking Beyond the Average*, in Euro-Par: Parallel Processing, 2018.

## 1.3 Thesis organization

The thesis is organized as follows:

Chapter 2 provides a brief description of conventional main memory systems based on DRAM, its organization and operation along with the challenges

that this technology faces. In addition, it introduces emerging memory technologies with an in-depth discussion on STT-MRAM's technical overview, development trend, organization, opportunities and challenges.

Chapter 3 investigates the performance impact of a slower main memory (STT-MRAM) in high performance computing (HPC) systems. The chapter describes the simulation methodology, HPC workloads and results obtained from the experiments.

Chapter 4 describes the journey of estimating reliable timing and current parameters of STT-MRAM. It releases detailed main memory parameters explaining the methodologies used to estimate them. We also discuss the experimental setup, applications used and results obtained from the study.

Chapter 5 explores the feasibility of using STT-MRAM in real-time embedded systems by analyzing average system performance impact and WCET implications. This chapter further discusses the timing analysis technique of real-time systems, describes experimental setup, benchmarks used and results obtained from the experiments.

Chapter 6 discusses the related works; Chapter 7 explains possible future avenues of research in continuation to the thesis; and finally, Chapter 8 summarizes the conclusions.

# Chapter 2

# Background

In modern computing systems main memory is closely interfaced with the CPU. There can be multiple memory modules connected to the CPU through different channels. In this chapter, we describe main memory systems, their organization and operation. We discuss the current memory technology (DRAM) that dominates the main memory landscape as well as emerging memory technologies that have strong potential to be a future alternative. Since any new main memory technology is most likely to adopt DRAM standards for easy incorporation, it is very important to understand the organization and operation of DRAM in detail. We further discuss technical details of STT-MRAM, its development trend in recent years, opportunities, advantages as well as the challenges that this technology faces.

## 2.1 DRAM

DRAM is the most widely used main memory technology used till date. This technology developed over several decades through persistent research and development. Since DRAM is being supplied in mass volumes, its production cost reduced to an affordable range making it the most economical option for main memory systems.

### 2.1.1 DRAM organization

DRAM main memory systems usually consist of three fundamental units: Memory controller, memory bus and DRAM devices organized in dual-inline memory modules (DIMMs). Modern CPUs are generally fabricated with

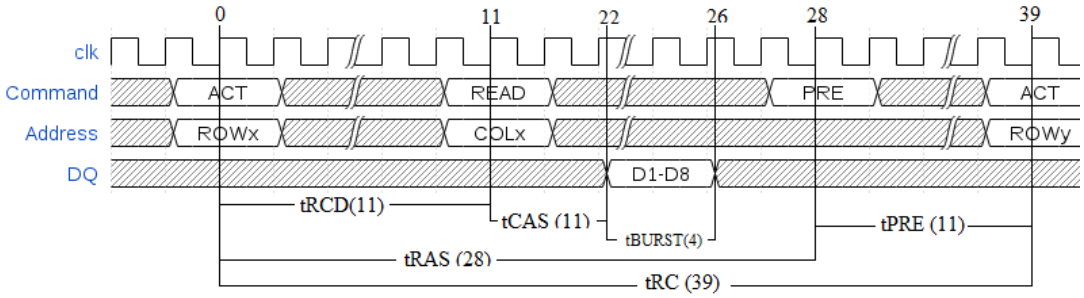


FIGURE 2.1: DRAM Read Cycle (not in scale)

an integrated memory controller. Memory channels connect DIMMs to memory controller in order to transmit data, command and addresses. DIMMs are connected through separate slots on the motherboard to facilitate easy replacement and servicing.

Any memory technology that is built on CMOS stores one bit data in a *cell*, which is considered as the basic building block of that memory. A DRAM cell is constituted of one transistor and one capacitor. In the cell, if a capacitor is fully charged, it represents logical "1" and a discharged capacitor holds the value "0". The transistor of the cell works as a switch to read/write these values. These cells are organized into array structures being connected to a *wordline* and a *bitline*. Specific wordlines are activated with *row address*. Once activated, capacitor charge (stored data) from all the cells connected to that wordline come to the row buffer through bitlines. At this point, the row buffer holds the data of the entire row that has been activated. Then the *column address* further selects a single column out of the entire row buffer and transmits the data to the physical pins of the memory to carry out a read operation. For write operations, row buffer receives data from physical pins of the memory and writes it back to the specific memory location as specified by row address and column address.

DRAM capacitors inherently suffers from *leakage current*. That is, the charge stored in the capacitors dissipates over time. For this reason, DRAM cells are periodically refreshed by the memory controller to preserve the data stored.

### 2.1.2 DRAM operation

In order to access data from DRAM arrays, first a row needs to be *activated* (ACT). Figure 2.1 abstractly illustrates the procedure of a DRAM read cycle [10]. The memory controller transmits the ACT command along with the address of the row that is to be activated. This transfers the data of that particular row to the row buffer. Now, to carry out a *read* operation, the memory controller issues a READ command along with the column address to access the specific column in the row buffer. The selected data of the specific row and column is then placed on the data bus to be transmitted. Since DRAM read is *self-destructive*, meaning when a row is *open* (i.e. having its data in the row-buffer), it needs to be *closed* to have the data written back in the cells. This procedure of closing a row is referred as *precharge* (PRE) operation.

As you can see in Figure 2.1, the delays between issuing commands are denoted by a set of parameters such as Row to column command delay (tRCD), column access strobe latency (tCAS), row precharge (tRP) etc. These are the generic DRAM timing parameters that are used to describe different timing constraints of the device.

### 2.1.3 DRAM challenges

The term *Memory Wall* was introduced in 1995 in a note published in Computer Architecture News [11]. Back in that time, the article implied that researchers so heavily concentrated on improving cache designs and latency-tolerance techniques, that they mostly overlooked enhancing the main memory system itself. The study projected that, no matter how efficient the caches become, increasing main memory latency would always keep the processor waiting, and this situation was termed as *hitting the memory wall* which persists till today.

Another challenge is to achieve higher bandwidth with existing DRAM technology. There are two ways of increasing the bandwidth for off-chip memories: by increasing either memory channel frequencies or their bit width. Both of these approaches have particular challenges. Increasing frequency requires the memory modules to be placed in close proximity to the processor to ensure signal integrity, thus limiting the number of modules that can be used by a processor. Increasing bit width implies to have more physical pins

to the processor chip and contributes to higher cost as well as significantly increases main memory power consumption.

Perhaps the greatest challenge that DRAM faces is technology scaling. Scaling appeared as a promising technique to keep up with the increasing need of capacity, performance and power reduction. To scale down the area occupied by a DRAM cell, it is imperative to take into account the area taken by the capacitor. The capacitor should be able to hold enough capacitance to swiftly set the sense amplifiers above the sensing threshold. It also needs to be strong enough to ignore the bit manipulating disturbances. Extreme scaling requires excessive reduction in the capacitor's size, making it increasingly vulnerable to errors. Smaller capacitors also possess lower retention time, requiring more frequent refresh operations causing additional power consumption, which is further aggregated for the increased capacity of scaled DRAM. Moreover, extremely scaled DRAM devices deployed in special situations such as space bound electronics, suffer from an additional challenge of being erroneously affected by the striking of a charged particle such as ions, photons or alpha particles causing a *Single Event Upset (SEU)*. DRAM devices deployed in such systems are also a cause of continuous power consumption due its refresh requirement.

Due to several such design challenges, this approach of DRAM scaling can not sustain forever. We need innovations, new methods and perhaps new technologies to move forward to meet next-generation computing needs.

## 2.2 Hybrid Memory Cube (HMC)

Hybrid memory cube (HMC) is a DRAM derivative that is constituted with DRAM dies stacked in a 3D fashion. HMC stacks DRAM dies in an innovative approach by interconnecting them with through-silicon vias (TSVs) to achieve higher internal bandwidth, low latency and low energy consumption [13]. Each DRAM die is distributed in *partitions* which are vertically connected with other partitions of adjacent dies forming a *vault*. A *vault* (i.e. a group of *partitions*) is operated collectively through an vault controller residing in the logic base layer (See Figure 2.2). Each vault controller is responsible for managing DRAM banks within the *vault*, thus eliminating the need of an off-chip memory controller in conventional DRAM systems. Thus, a HMC

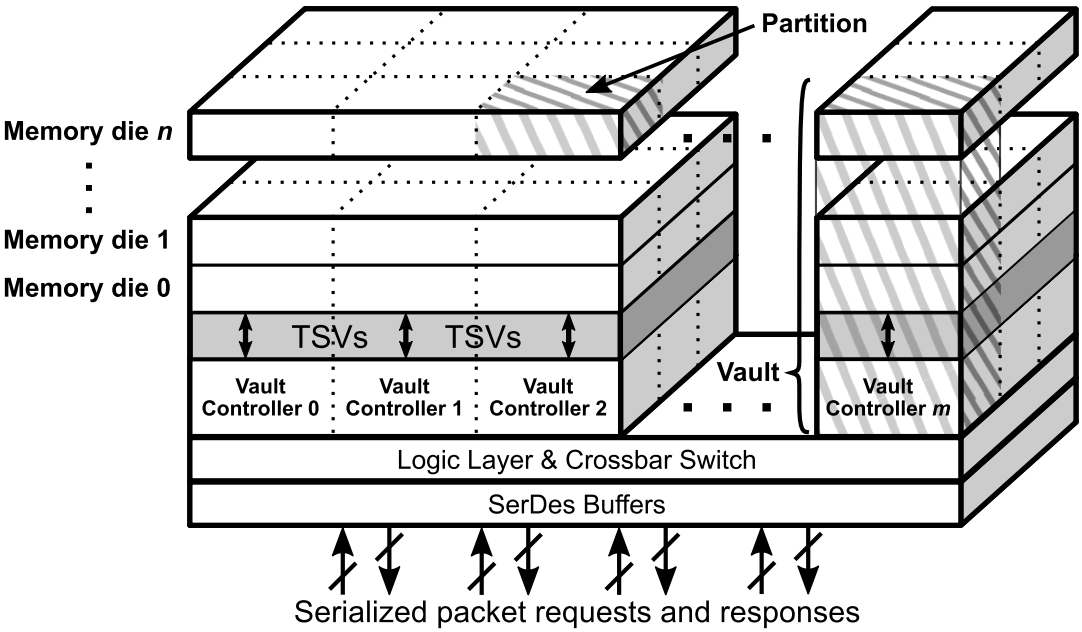


FIGURE 2.2: Internal Structure of Hybrid Memory Cube (HMC) [12]

vault resembles a *channel* in traditional DRAM based memory systems, since it consists of all the units of a DRAM channel — a memory controller, multiple memory ranks and a data bus. This implies, that HMC is essentially a multi-channel DRAM which is capable of handling a large number of concurrent memory requests [14].

Vault controllers are connected with other HMCs or host devices with a packet based communication protocol which is implemented with a high speed serialization/deserialization circuit. This allows to achieve higher link bandwidth in comparison to synchronous bus based interfaces with the standard JEDEC protocol [13]. Packets are grouped into 16-byte elements, called *FLIT* (FLow unIT). A FLIT is the smallest data unit that can be transmitted on the high speed interface. Each transaction supports packet sizes of one FLIT (16 B) to eight FLITs (128 B), depending on the link granularity. In addition to the link granularity, HMC also implements configurable memory address granularity. HMC uses 34 bits for internal memory addressing which consists vault, bank, row, column and byte addresses. Different address mapping schemes can be used by changing the maximum payload size for a packet to take advantage of the configurable memory address granularity. This scheme can be very useful to achieve desired configurations with specific latency and throughput.

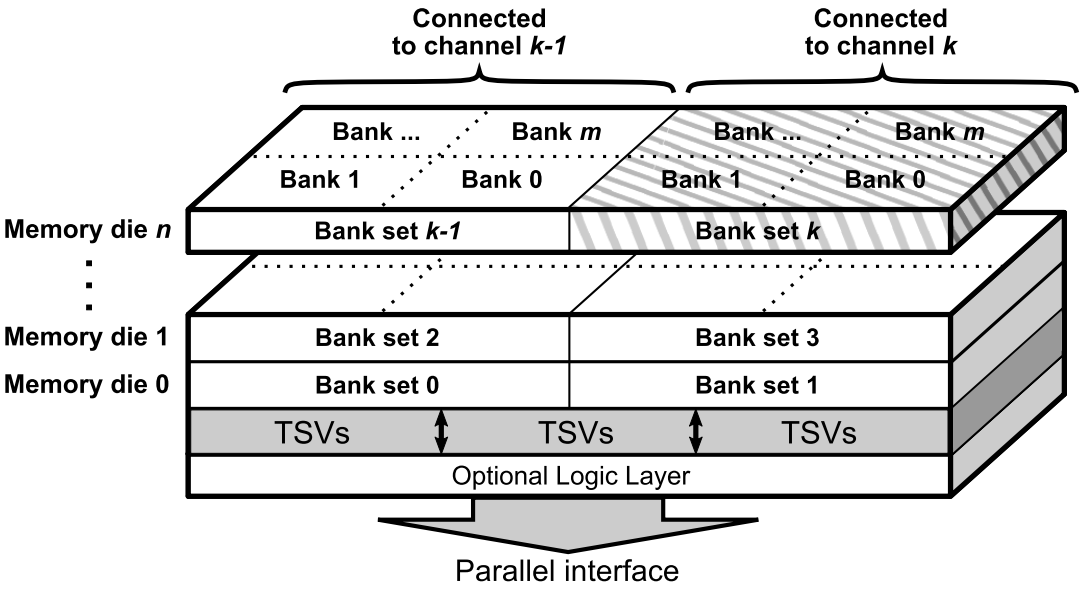


FIGURE 2.3: Internal Structure of High Bandwidth Memory (HBM) [12]

## 2.3 High Bandwidth Memory (HBM)

High bandwidth memory (HBM) is another variant of 3D-stacked DRAM which complies with the JEDEC standard. HBM is introduced in an attempt to reduce the bandwidth gap between processor's memory bandwidth requirements and actual bandwidth performance with conventional DRAM systems. The basic structure of HBM consists of a base logic die and and stacked core DRAM dies, which are interconnected by TSVs as shown in Figure 2.3. A core DRAM die has 2n-prefetch with a minimum access granularity of 16 bytes per channel. To support independent channel operations, each channel of a core DRAM die employs independent address and data TSVs with point-to-point (P2P) connections from the base die. With the semi-independent row and column command interfaces, HBM can allow RAS and CAS commands in parallel [15].

HBM has a DIMM-like system organization: the memory controller is associated with the CPU, and a point-to-point parallel interface links it to the main memory [12]. HBM implements an additional logic layer that is capable of implementing logic functions — opening an opportunity for in-memory computations. Each DRAM die of HBM is grouped into banks and connected to individual channels. HBM is generally targeted to be tightly coupled to the processor die with wide interface in order to facilitate high speed and low-power memory operations.

## 2.4 Phase Change Memory (PCM)

Phase Change Memory (PCM) is one of the emerging non-volatile (NVM) memories which has availed considerable attention in recent years as an alternative memory technology. PCM storage cell consists two electrodes separated by a phase-change substance typically being the chalcogenide material $Ge_2Sb_2Te_5$ (GST) [16]. The physical state of this substance can be changed from crystalline to amorphous, each of which have distinct electrical resistance properties making it an obvious element to consider for data storage.

A PCM memory system has three fundamental operations: *READ*, *SET* and *RESET*. To *READ* data from a PCM cell, small sensing current is applied through it. The current senses the resistance state of the GST to get one bit of data ("0" or "1"). To *RESET* a PCM cell a strong current pulse is applied for a short duration. The strong current pulse quickly increases the temperature of the GST material to the melting point and subsequently programming it in the amorphous state (representing "0"). To *SET* a PCM cell, a programming current pulse is applied for a longer period to program the cell from amorphous state to crystalline state (representing "1"). While the *READ* and *RESET* operations are relatively faster (tens of nanoseconds), the *SET* operation is significantly slower (hundreds of nanoseconds). Therefore, for writing in a PCM based memory system, both *SET* and *RESET* operations will appear but the average write latency will only be limited by the significantly slower *SET* operation [17].

PCM can essentially have a density advantage over DRAM having the possibility of using multiple states of crystallization to hold muti-bit data in one cell [18]. However, despite of having several advantages, PCM suffers from low access latency, which makes it impractical to use as the main memory of high performance computing systems. In addition, having significantly low endurance suggests, PCM may not be a viable option for commercial systems.

PCM has also been reportedly used to construct **3D-XPoint** memory [19]; developed jointly by Intel and Micron technology. As the name suggests, it has a transistor-less 3D structure, where the data resides in the intersection of perpendicular wires. Current is applied to these wires in order to access the data that is stored in the corresponding intersection (i.e. cell). 3D-Xpoint is being projected as superior storage class technology and less likely as a main

memory alternative.

## 2.5 Resistive RAM (RRAM)

Resistive RAM (RRAM) is another potential candidate for future memory systems. An RRAM cell is typically constituted as a two terminal device with a simple metal–insulator–metal structure. RRAM cells are operated by voltage driven resistance switching of the metal oxide insulator between the low resistance state (LRS) and the high resistance state (HRS). This is usually done by creating and dissolving the conductive filament (CF) that has oxygen vacancies [20].

Like PCM, three main operations are usually performed on an RRAM cell: *SET*, *RESET* and *READ*. The *SET* operation is performed by applying a positive voltage between the top and bottom metal layers to transform the cell from the HRS to LRS [21]. In order to limit the current to perform the *SET* operation, a transistor is used. This constitutes or extends a filament of oxygen vacancies from the bottom metal layer to the top metal layer, thus decreasing RRAM cell resistance. Similarly, a negative voltage is applied to perform a *RESET* operation which transforms the cell from LRS to HRS by dissolving the conductive filaments. A *READ* operation is performed to detect the current state (i.e. data) of the cell by applying a sensing voltage which is weak enough not to change the resistance of the cell [22].

RRAM is highly scalable; and having low read/write energy makes it a viable alternative of storage class memories. Recent enhancements improving access latencies could suggest to consider it as a main memory alternative. However, RRAM access time is still significantly slower than other emerging non-volatile memories such as STT-MRAM. It also lacks comparable endurance w.r.t DRAM and STT-MRAM.

## 2.6 STT-MRAM

### 2.6.1 Technology overview

Research exploring the magneto-resistance caused by the spin polarized current can be tracked back to the '90s [23][24][25]. However, significant scientific efforts of optimizing and applying this phenomenon to create a novel non-volatile memory is a relatively new approach. Only around fourteen years ago, in 2005, Hosomi *et al.* [1] 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 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 2.4 depicts, 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 [26].

A more recent variation of MTJ is perpendicular MTJ (pMTJ). In contrast with the conventional MTJ, the poles of pMTJ magnetic layers are perpendicularly aligned with the plane of the wafer; see Figure 2.4 (c) and (d). In 2010, Ikeda et al. presented pMTJ for the first time and demonstrated that it requires much lower write current than the conventional MTJ [27]. Recently, Janusz et al. has reported to achieve good write performance with pMTJ down to 11 nm device size [28].

Other variants of STT-MRAM cell design incorporated advanced 2T-2MTJ, 3T-2MTJ and 4T-2MTJ cells in a pursuit to improve performance and energy efficiency [29][30][31].

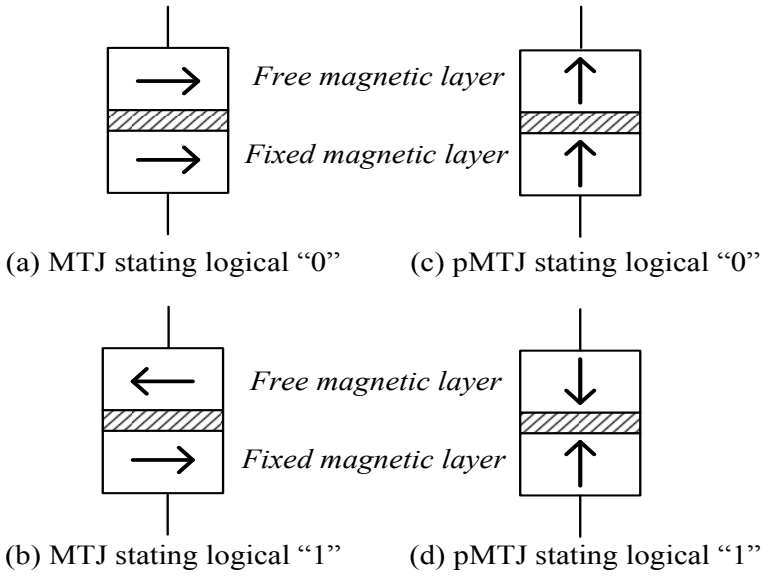


FIGURE 2.4: STT-MRAM cell

### 2.6.2 Development trend

Around fourteen-years-old, STT-MRAM is rapidly catching up with the mature DRAM technology. Figure 2.5 shows an approximate timeline of DRAM and STT-MRAM chip capacity development, and clearly illustrates the diminishing gap between these two technologies.

Development of DRAM devices started back in the ’70s, and by the year 2003, DRAM chip capacity could reach upto 256Mb. Around at the same time, the first reported STT-MRAM chip appeared with the capacity of 128Kb, which is a 2000$\times$ smaller capacity than DRAM (note the logarithmic scale of the vertical axes). DRAM chip capacity gradually increased and reached 16Gb by the year 2016. Following a sharp incline, STT-MRAM chip capacity increased to 4Gb by the same year [32], reducing the capacity gap between these two technologies from 2000$\times$ in 2003 to only 4$\times$ in 2016.

Promising development has also been made improving STT-MRAM’s bus frequency. While the first generation of DDR SDRAM had 133Mhz bus frequency, present day DDR3 and DDR4 compatible STT-MRAM are catching-up with the frequencies of the high-end DRAM devices [33].

The STT-MRAM device improvements come mainly from the enhancements in the MTJ design. With the recent variation of pMTJ, different memory manufacturers have demonstrated a fierce competition to achieve the smallest device size for MTJs. In 2011, Samsung developed pMTJ at 17nm. In 2016, IBM

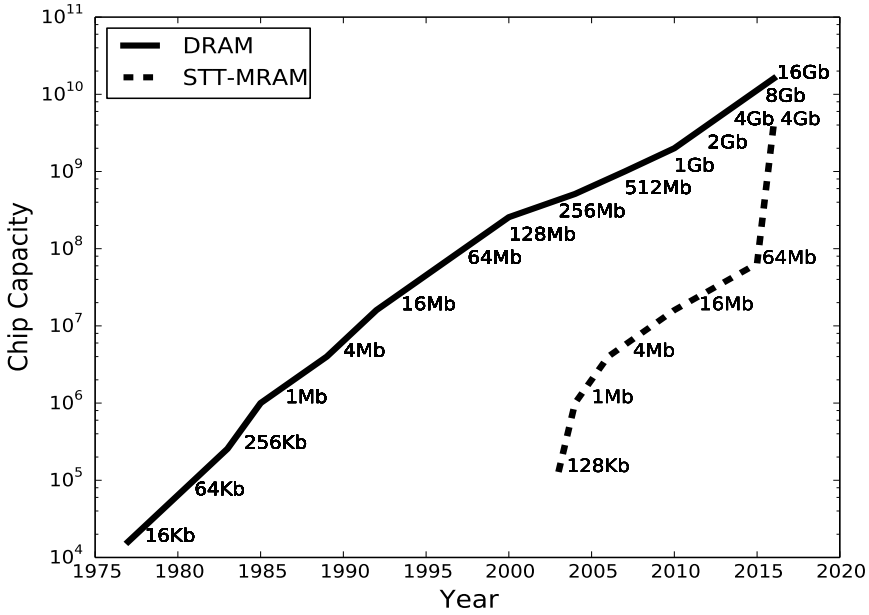


FIGURE 2.5: DRAM and STT-MRAM capacity growth in years

demonstrated an 11nm STT-MRAM junction. By the end of the year 2016, IMEC researchers reported to develop world's smallest pMTJ at 8nm.

An intensified effort in STT-MRAM research by the memory manufacturers may indicate that a revolution with STT-MRAM memory technology is imminent, and we can expect to see a lot of exciting developments with this memory technology in the near future.

### 2.6.3 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 [34] [35] [36]. 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. Performance and energy overheads of DRAM refresh have been deemed insignificant for long, but with the increasing frequency and size of DRAM devices, refresh becomes an important factor of memory performance and energy consumption. In state-of-the-art 32 Gb DRAM devices (specified in the DDR4 standard), refresh contributes to more than 20% of the DRAM energy consumption and degrades the memory bandwidth by more than 30%. In the upcoming 64 Gb devices, it is estimated that refresh will degrade memory throughput and increase the energy consumption by more than 50% [37][38]. STT-MRAM is a non-volatile technology and, therefore it requires no refresh. Thus, a great performance and energy advantage over the 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 [39][40][41][42]. In the 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 [40]. Another source of increasing the memory error rate is the scaling of the DRAM technology [43]. 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.

Clearly, STT-MRAM is not a failure-free technology. However, its retention failures (stochastic in nature), can be efficiently controlled with proper ECC mechanisms [44]. A Recent study of Pajouhi *et al.* [45] analyzes the

interactions between device parameters, bit-cell level parameters and different ECCs to optimize the robustness and energy-efficiency of a STT-MRAM cache. A compelling follow up would be to extend this work on STT-MRAM main memory.

### 2.6.4 STT-MRAM special advantages

In this section, we make a qualitative analysis of STT-MRAM specific features. In particular, intrinsic radiation hardness, non-volatility, zero standby power and the capability to function in extreme temperatures offer a great opportunity to explore its usability in real-time embedded systems in aerospace and automotive domains, where computer systems must operate with guaranteed behavior in harsh environments under stringent constraints. A summary of the main differences between DRAM and STT-MRAM is provided in Table 2.1.

#### Radiation hardness

A *Single Event Upset (SEU)* occurs when the state of a memory cell or transistor is erroneously changes by the striking of a charged particle such as ions, photons or alpha particles [46]. Continuous scaling of CMOS devices has further amplified the chances of being affected, which is a serious concern for DRAM and SRAM main memories and caches, which account for a large fraction of the silicon in computing systems.

Microelectronic devices deployed in space are particularly vulnerable to such events. For instance, it has been reported that soft error rates due to radiation grow by a factor of 650$\times$ when moving from sea level to 12,000m of altitude [47], a usual altitude for commercial planes. Radiation in the space further exacerbates the issue due to the lack of the Earth atmosphere to mitigate radiation. These phenomena lead to increased bit upset rates that require expensive coding and scrubbing techniques to guarantee error correction.

STT-MRAM offers a promising solution to this problem as it replaces charge-based storage with Magnetic Tunnelling Junction (MTJ), which stores data in the form of magnetic resistance that is intrinsically tolerant to radiation.

TABLE 2.1: DRAM vs STT-MRAM for embedded real-time systems

| Feature | DRAM | STT-MRAM |
|---|---|---|
| Radiation-hard | - | +++ |
| Standby power | - | +++ |
| Temperature tolerance | + | +++ |
| Storage capacity | ++ | ++ |
| Access speed | +++ | ++ |
| Endurance | +++ | +++ |

STT-MRAM memory chips have reportedly been deployed on space bound satellites [48].

#### Zero standby power

Electronic devices in the aerospace and automotive domains are usually deployed once to be operated for a long period of time without regular maintenance. Due to its non-volatility, STT-MRAM also ensures that no data is lost if an unexpected power down or voltage drop takes place. Implementing appropriate measures, the operations can resume from the same point as it was interrupted. Also, STT-MRAM having long term data retention with zero standby power is set to offer great advantage from the power consumption perspective. For instance, many instruments in space missions are operated at a given (low) frequency, taking pictures or measurements every second or minute. STT-MRAM allows activating and deactivating systems with negligible power cost, without requiring any form of backup space.

#### Operational temperature

STT-MRAM's another crucial feature is being operational under an extended range of temperatures. One of the STT-MRAM manufacturers states that a Grade 1 qualified MRAM will contain data for 20 years being operational under extreme temperatures ranging from -40C to 125C. [49]. This makes STT-MRAM suitable to be used both in aerospace (extreme cold) and automotive (occasionally extreme hot) parts.

#### Integrated memory

Having comparable speed and density to DRAM, with unlimited endurance and long retention time, STT-MRAM really opens up the opportunity to use it as single memory replacing DRAM and long term storage from the conventional real-time embedded systems.

### 2.6.5 STT-MRAM challenges

Being a promising memory technology with a lot of potential, STT-MRAM also 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 estimations of timing and power consumption parameters. Only three studies simulate and analyze STT-MRAM main memory. Table 2.2 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.* [50] use a cycle-accurate DDR3-DRAM memory simulator and estimate STT-MRAM parameters based on Fujitsu's 16kb test-chip built in 2010 with 0.13$\mu$m technology [53]. 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

TABLE 2.2: 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 [50] | No information | Cycle accurate simulation | $t_{WR}$ and $t_{RCD}$: $0.5 \leqslant \frac{STT}{DRAM} \leqslant 2$ | | - Latency before main memory: No info<br>- Wide range of $t_{WR}$ and $t_{RCD}$ values. |
| Kultursay et. al [51] | Row buffer hit: 30 ns<br>Row buffer miss: 50 ns | | +0 ns | +10 ns | - Main memory latency: No source<br>- Obsolete STT parameters. |
| Suresh et. al [52] | No information | | 35 ns | 35 ns | - Latency before main memory: No info<br>- DRAM latency: No info |

the STT-MRAM $t_{WR}$ and $t_{RCD}$ parameters to the *overall* main memory access latency.

Kultursay *et al.* [51] 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 the 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 [30][31][54].

Suresh *et al.* [52] simulate STT-MRAM read and write operations with a fixed latency of 35 ns, obtaining these estimation from ITRS report, 2013 [55]. 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 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. STT-MRAM main memory evaluation is incomplete without cycle-accurate simulation with reliable timing parameters. The lack of detailed timing parameters is also the main problem for any STT-MRAM microarchitectural exploration, improvement and evaluation.

#### Timing parameters: Dead ends

Our search for reliable STT-MRAM timing parameters was not straightforward; it lasted three years and involved collaboration with two STT-MRAM memory manufacturers.

Initially, we planned to simulate STT-MRAM main memory by using the NVMain simulator [56]. After analyzing NVMain STT-MRAM timings, we noticed that several key parameters had values that differ significantly from

our understanding of STT-MRAM main memory, as well as the timings provided by manufacturers. For example, the NVMain configuration file for a 4GB MRAM[1] listed tRAS (row access strobe) to be 0. Whereas, in DDR*x* standard, tRAS is constituted by tRCD, tCAS, tBURST and the delay for the data restoration which corresponds to 28 cycles in DDR3-1600. In addition, tRCD was set to 14 cycles with an explanation to have it derived from the Everspin MR2A16A product datasheet. We could not verify this derivation to be correct. To clear up the confusion, we contacted the NVMain developers asking for a clarification and source of their STT-MRAM parameters, but got no reply. Since we were unable to verify how these timing parameter values were formulated and we had some serious doubts about their validity, we had to classify the NVMain STT-MRAM main memory parameters as unreliable and discard them from being used in our experiments.

A couple of studies [57][51] simulate STT-MRAM main memory by integrating publicly available STT-MRAM cell parameters into the CACTI [58] cache simulator. Using a *cache simulator* to estimate timing and energy parameters of a *main memory* is not a straightforward approach. Main memory 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 completely different parameter values. We failed to find any information on how CACTI could be adopted for main memory simulation, and the studies that use this approach provide no information on how they bridged the gap between cache and main memory simulation.

#### Our approach

Although STT-MRAM is catching-up rapidly in terms of cell size, capacity and frequency, DRAM still has a great advantage — it is a standardized plug-and-play device. Today, we have various DRAM and CPU manufacturers and OEMs, and we have a full compatibility — we can connect any CPU (Intel, AMD, ARM-based) to any DRAM (Samsung, Micron, Hynix) as long as they follow the same DDR*x* standard. Although we probably take this for granted, it is very important to understand that this standardization requires a tremendous effort and it is done only for mainstream products (technologies) with volumes that justify the investment.

[1] NVMain configuration file describes the parameters of the 4GB MRAM device. This configuration file was released in 2015, even before 64MB devices were manufactured.

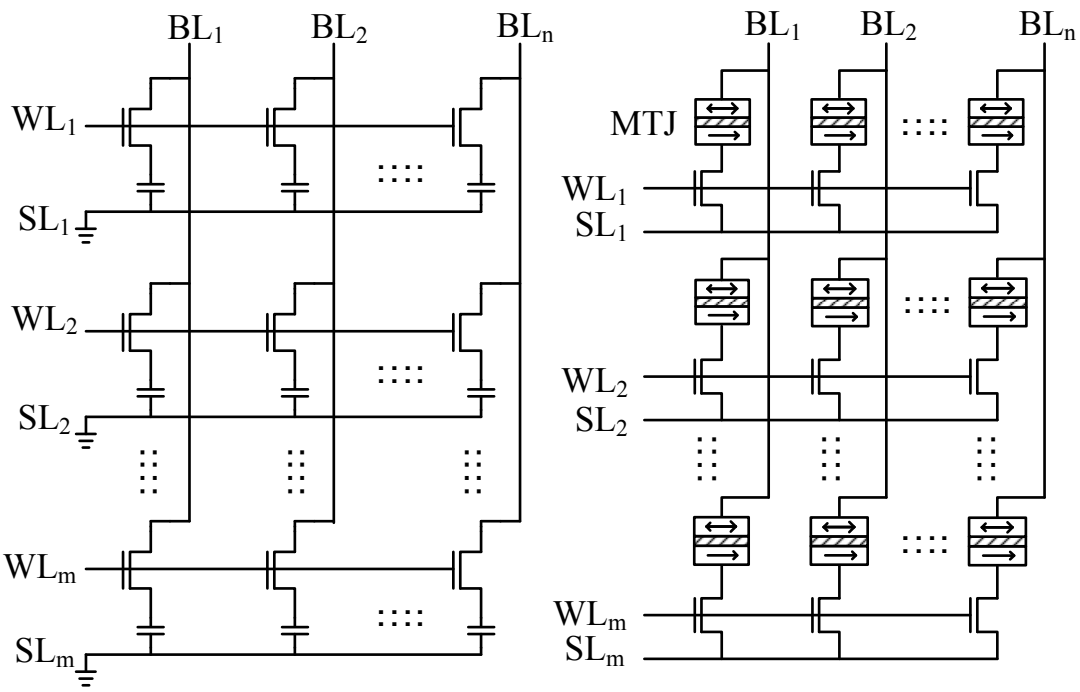


FIGURE 2.6: STT-MRAM and DRAM cell-array

Since STT-MRAM is a new technology with no specific standard, the manufacturers have two options. One would be to make a STT-MRAM main memory system from scratch, leading to fine-tuned microarchitecture, interface and protocols for the STT-MRAM technology. However, this would also require CPU manufacturers and OEMs to adapt their products to STT-MRAM memory, by deploying, e.g., specific STT-MRAM memory controllers. Another option would be to adjust STT-MRAM microarchitecture and interface to the DDR*x* standard. This approach may not lead to an optimal STT-MRAM main memory device, but it probably is the only practical way to make STT-MRAM easily integrated into the existing systems. Publicly available product information and patents [59][60][61] from STT-MRAM manufacturers clearly indicate that they selected the second approach — incorporation of the STT-MRAM technology into DDR*x* interface and protocols enabling a seamless integration into the rest of the system. Therefore, the STT-MRAM data array structure is very similar to that of DRAM (see Figure 2.6). In both designs, DRAM and STT-MRAM, transistors are used to access a selected set of cells, and the only fundamental difference is in the cell type, capacitor in the case of DRAM and MTJ in the case of STT-MRAM. Also, overall STT-MRAM device organization is essentially the same as in DRAM, in terms of number and size of the structures such as ranks, banks,

TABLE 2.3: Comparison of DRAM and STT-MRAM main memory in HPC systems.

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

sub-arrays, row, columns, and row buffers. Finally, STT-MRAM CPU interface is DRAM compatible.

#### Commercial challenges

We summarize our overall comparison between DRAM and STT-MRAM main memory targeting HPC market in Table 2.3. As 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 an 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.

# Chapter 3

# STT-MRAM Performance Impact in HPC Systems

## 3.1 Introduction

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. Being the first set of experiments of the thesis, we conduct a preliminary assessment of HPC system performance impact with STT-MRAM main memory with industry estimations. At this point in time, since reliable timing parameters of STT-MRAM devices were unavailable, we perform a sensitivity analysis that correlates the overall system slowdown trend with respect to average device latency estimated by industry. In this chapter, we discuss the details of the experimental setup, application suite and report the results.

## 3.2 Experimental setup

### 3.2.1 Application suite

We evaluated STT-MRAM main memory on HPC applications included in the Unified European Application Benchmark Suite (UEABS) [62]. UEABS is

TABLE 3.1: 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 |

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 thousand x86 cores.

Table 3.1 summarizes the applications used in the study. The first two columns of the table list the application names and their scientific area. The third column lists the 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.

### 3.2.2 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 tracefile. 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 the TaskSim system simulator [63].

### Target HPC platform

We collected traces of UEABS applications running on the MareNostrum 3 supercomputer [64]. MareNostrum contains 3056 compute nodes (servers) connected with an 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 the 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.

### 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 applications 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 representatives of the overall behavior.

Figure 3.1 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 3.1), 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, intermediate calculations are combined into final results. In production runs of HPC applications, the 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 a few iterations is sufficient to characterize the entire application execution [65]. 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 allow us to simulate the execution of few main-loop iterations of some processes, that are a good representation of the overall behavior [65][66]. That way, we avoid producing traces of unmanageable size (in the order of terabytes) and also bring simulation time to a reasonable level. Therefore, before the detailed instruction tracing, we instrument computation bursts and inter-process communication and analyze the overall application structure. Computation bursts and inter-process communication are instrumented with the Limpio instrumentation framework [67] and the application structure is analyzed with the Paraver visualization tool [68]. Limpio and Paraver are standard tools for this kind of HPC application profiling and analysis.

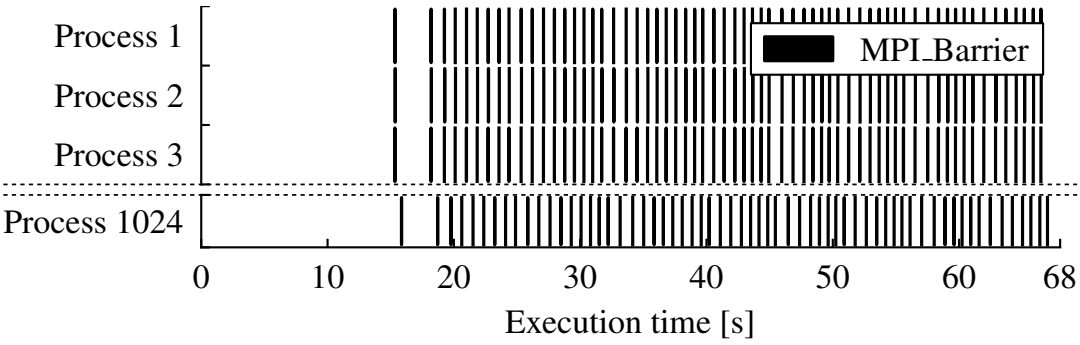


FIGURE 3.1: Repetitive behavior of HPC applications: ALYA, 1024 processes

### Trace Collection

In order to trace instructions in the selected application segments, we developed a tool with the Valgrind instrumentation framework [69]. 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* [70][71]. 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, the filter cache introduces negligible discrepancies in the simulation of the main memory [72]. 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 [63][65][66][73].

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 chapter, we report average slowdown and standard deviation of all the measurements.

### 3.2.3 Simulated CPU

In order to evaluate an STT-MRAM main memory system, we simulated a socket of a MareNostrum-like compute node (see Section 3.2.2), which is the dominant architecture in HPC systems [74]. The simulated hardware platform is comprised of three distinct segments: CPU pipeline, CPU cache hierarchy and main memory.

#### CPU pipeline

Since our study proposes no changes to the CPU microarchitecture, we simulate the CPU pipeline with a simplified model. The model reproduces series of CPU (non-memory) instructions using a constant number of cycles per instruction (CPI) [73]. This approach is used for simulation of changes in memory system because it significantly reduces the simulation time with respect to a detailed pipeline [63]. We repeat our experiments with three values of

TABLE 3.2: 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 |

CPI: 0.5, 1.0 and 2.0. CPI of 0.5 corresponds to a complex core with a strong out-of-order engine that can process two instructions in each cycle. CPI of 1.0 and 2.0 correspond to simpler cores.

#### Cache memory

The simulated hardware platform comprises a detailed model of the Sandy Bridge-EP E5-2670 cache hierarchy [75]. This Sandy Bridge E class processor has eight cores, dedicated L1 instruction and data caches of 32 KB each, a 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 [75], and are summarized in Table 3.2.

### 3.2.4 Simulated main memory

STT-MRAM main memory simulation is a challenging task because its detailed parameters are not yet standardized and released by the industry. 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 the cache memory hierarchy (detailed in Section 3.2.3), but also the latency of the memory controller, the 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 [76], 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) [62]. 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 instructions with the `perf` tool, along with the Precise Event Based Sampling (PEBS) mechanism [75]. 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]

The 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 [30][31][54].[3] In addition to the 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 an average access time of 22.5 ns and 30 ns. The sensitivity analysis is important

[1] The experiments were executed on a stand-alone server (not the 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 [77]. 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.

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 [4]. 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 2.6.5.

Our study focuses on the 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.

## 3.3 Results

The results of our study are organized into two parts. First, we present the performance comparison of an 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.

### 3.3.1 Industry estimate

The performance comparison of HPC systems with conventional DRAM and STT-MRAM main memory being 20% slower than DRAM, is presented in Figure 3.2. For each application, different bars correspond to different simulated CPUs with CPIs 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

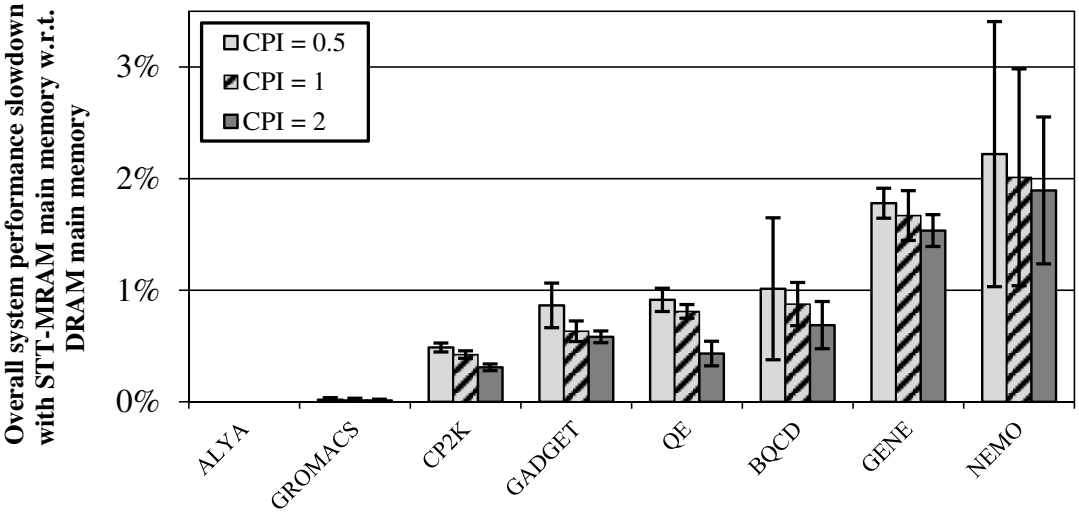


FIGURE 3.2: Performance slowdown with 20% slower STT-MRAM device

Application slowdown ranges from 0% (ALYA) to 2.2% (NEMO), and it is 0.8% on average.

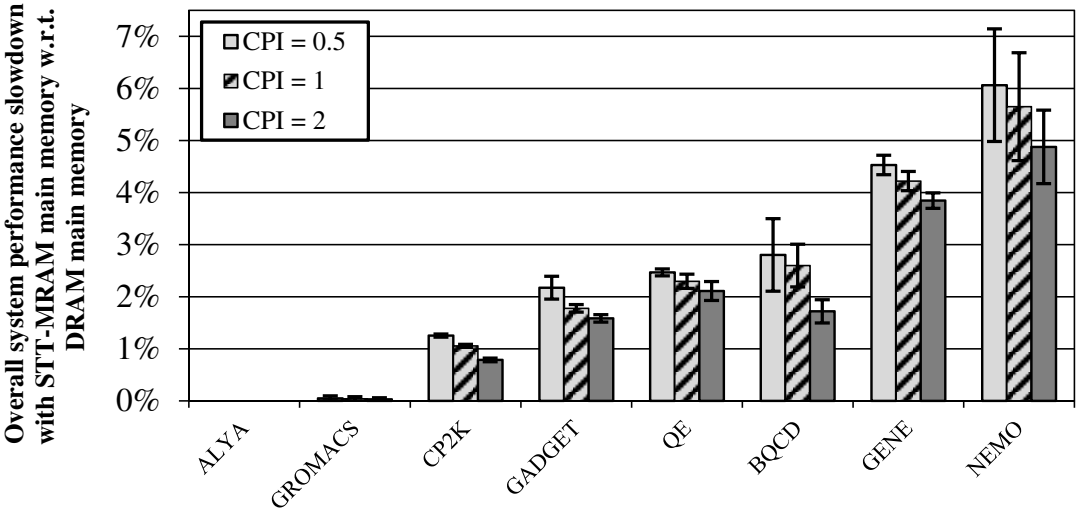


FIGURE 3.3: Performance slowdown with 50% slower STT-MRAM device

Application slowdown ranges from 0% (ALYA) to 6.7% (NEMO), and it is 2.2% on average.

— 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. The processing core with a CPI value of 0.5 refers to an aggressive core which executes two instructions per cycle, while CPIs 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).

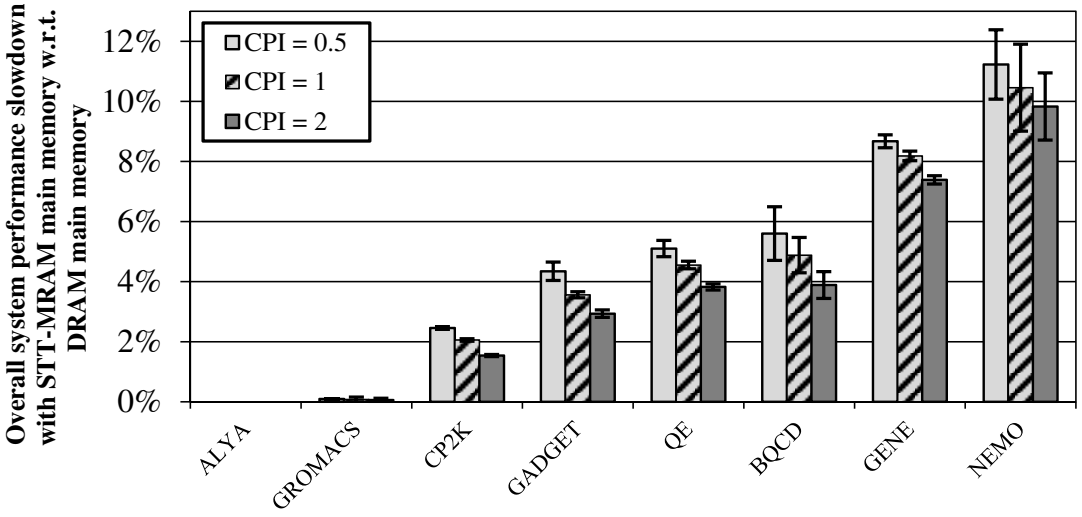


FIGURE 3.4: Performance slowdown with 100% slower STT-MRAM device

Application slowdown ranges from 0% (ALYA) to 11.2% (NEMO), and it is 4.2% on average.

High-CPI cores increase the time spent in the CPU and the execution time of the application. Therefore, a 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 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 an 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.

### 3.3.2 Sensitivity analysis

Figure 3.3 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 3.4. 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).

## 3.4 Summary

We model STT-MRAM main memory with an average latency estimated by industry and incorporate it into the overall simulation of the HPC system executing production applications. Our results suggests that, although STT-MRAM is modelled to be significantly slower than DRAM at the device level, it provides performance comparable to conventional systems, while opening up various opportunities for HPC system improvements. The reasons why a slower main memory does not proportionally degrade performance are because: (1) due to the high efficiency of the caches, only a small portion of the total instructions actually go the main memory, (2) main memory device latency constitutes only a part of the total main memory latency, and (3) the

out-of-order pipeline allows the processor to execute other instructions while waiting for the data from the main memory.

# Chapter 4

# Enabling a Reliable STT-MRAM Main Memory Simulation

## 4.1 Introduction

STT-MRAM is a promising new memory technology with a very desirable set of properties such as non-volatility, byte-addressability and high endurance. It has the potential to become the *universal memory* that could be incorporated to all levels of the memory hierarchy. In this thesis, we perform a detailed analysis of STT-MRAM main memory timing and propose an approach to conduct a reliable system level simulation of the memory technology.

We seamlessly incorporate STT-MRAM timing and current parameters into the DRAMSim2 memory simulator and use it as a part of the simulation infrastructure of the high performance computing (HPC) systems. In this chapter, we discuss the obstacles we face to perform a reliable STT-MRAM simulation, how we approach and analyze the problem, the proposal we develop, the simulation infrastructure used to examine the timing and current parameters we propose, and results of the experiments.

## 4.2 STT-MRAM timing parameters

STT-MRAM technology has recently got significant attention of various major memory manufacturers; however, academic pursuance to conduct system-level research on this technology is still marginal, mainly due to the

lack of publicly available, detailed and reliable timing parameters of STT-MRAM. These timing parameters are essential to conduct a system level simulation. Some researchers adopt simplistic memory models to simulate main memory, but such models can introduce significant errors in the analysis of the overall system performance [3][4]. Therefore, detailed timing parameters are a *must-have* for any evaluation or architecture exploration study of STT-MRAM main memory. However, these detailed parameters are not publicly available because STT-MRAM manufacturers are reluctant to release any delicate information on the technology. Also, being a rapidly evolving technology, it is difficult even for the manufacturers to predict the exact timing for an upcoming STT-MRAM main memory device.

The fact that STT-MRAM memory is DDR$x$ compatible, with the same or very similar organization and CPU interface, provides a lot of information about STT-MRAM timing parameters. Both DRAM and STT-MRAM main memory devices use a row buffer as an interface between the cell-arrays and the memory bus. Since the circuitry beyond the row buffer for DRAM and STT-MRAM would essentially be the same, once the data is in the row buffer, STT-MRAM timing parameters for the consequent operations would be the same as for DRAM. For example, tCWD (Column write delay) corresponds to the delay between issuance of the column write command and placement of the data on the bus. Therefore, the value of this timing parameter does not change for STT-MRAM and DRAM. This apply to all the timing parameters that are not associated with row operations such as tBURST, tCAS, tWTR, etc., as summarized in Table 4.1. The timings are represented in DDR3-1600 cycles, but applies to other DDR$x$ standards as well.

The only fundamental difference in STT-MRAM and DRAM main memory is their storage cell technology, MTJ and capacitor, respectively. Due to the difference in the cell access mechanism of these two memory technologies, the timing parameters associated with the STT-MRAM row operations would deviate from DRAM.[1] DRAM access is a voltage mode operation. To access the cell array, bitlines are precharged to a reference voltage (see Figure 2.6). The timing parameters associated with this operation is tRP (Row Precharge). Then a voltage is applied on the wordline to activate the access transistors allowing the sensing circuit to sense and move the data to the row buffer. The time it takes from a row access to get the data ready at the row buffer is

[1] Rows of the DRAM or STT-MRAM cells constitute the cell arrays, see Figure 2.6. Row operations access directly to the memory cells.

TABLE 4.1: Memory parameters not associated with row operation (DDR3-1600 cycles)

| Timing Parameters | Description | DRAM | ST*x* |
|---|---|---|---|
| tBURST | Burst length | 4 | 4 |
| tAL | Added latency to column access | 0 | 0 |
| tCAS/tCL | Column access strobe latency | 11 | 11 |
| tRTP | Read to precharge delay | 6 | 6 |
| tCCD | Column to column delay | 4 | 4 |
| tWTR | Write to read delay time | 6 | 6 |
| tRTRS | Rank to rank switching time | 1 | 1 |
| tCWD | Column write delay | 10 | 10 |
| tWR | Write recovery time | 12 | 12 |
| tCKE | Next power up for an idle device | 4 | 4 |
| tCMD | Command transport duration | 1 | 1 |
| tXP | Exit power down with DLL on to any valid command | 5 | 5 |

denoted by tRCD (Row to column command delay). On the contrary, STT-MRAM cell array access is a current mode operation and is completely different from the DRAM access mechanism. To read data stored in an MTJ, a wordline is activated and a small amount of current is applied through corresponding bitline to sense the data (in terms of resistance) in a particular MTJ and eventually transferring it to the row buffer.

STT-MRAM specific timing parameters have neither been standardized nor been released by any industry. This is perhaps due to the perpetual evaluation of the STT-MRAM technology that is constantly changing over a short duration of time. Memory manufacturers, who are developing STT-MRAM are judiciously not revealing these parameters ahead of time; so, at this point, we have to accept that there is no reliable information on how these timing parameters will change for the upcoming STT-MRAM devices. Therefore, we strongly argue that the best we can do is a sensitivity analysis on the parameters that will change from DRAM to STT-MRAM. And we would strongly encourage any STT-MRAM related research to validate its analysis and proposals for various potential STT-MRAM parameters — i.e. to consider that uncertainly of the evolution of this technology.

In this study, we selected three sets of timings naming ST-1.2, ST-1.5 and ST-2.0 with deviations of 1.2x, 1.5x and 2x from respective DRAM timing parameters as summarized in Table 5.2. The presented methodology converged through our research cooperation with Everspin Technologies Inc.

TABLE 4.2: Timing parameters associated with row operation (DDR3-1600 cycles)

| Timing Parameters | Description | DRAM | ST-1.2 | ST-1.5 | ST-2.0 |
|---|---|---|---|---|---|
| tRCD | Row to column command delay | 11 | 14 | 17 | 22 |
| tRP | Row precharge | 11 | 14 | 17 | 22 |
| tFAW | Four row activation window | 24 | 29 | 36 | 48 |
| tRRD | Row activation to Row activation delay | 5 | 6 | 8 | 10 |
| tRFC | Refresh cycle time | 208 | 1 | 1 | 1 |

However, the timing parameters used in this study do not specifically correspond to any of their commercial products. We believe simulations performed with these timing parameters gives us a reliable range of possible system performance for upcoming STT-MRAM main memory devices. Although some earlier studies have reported asymmetrical read-write latency for STT-MRAM, we used symmetrical read-write latency in compliance with the latest development and studies of the technology [30][31][54].

In addition to the parameters listed in Table 4.2, there is a change in the STT-MRAM main memory operation sequence as well. In DRAM, when a row is accessed, the storage capacitors are discharged losing the data that they held. This is known as *destructive read*. After the *read* is performed, the data from the row buffer needs to be restored to the data array through a *write-back* before it can issue the next *precharge* command. Whereas, being a non-volatile memory, STT-MRAM read is *non-destructive*; i.e., it does not need to restore the data back to the array. Because of this, STT-MRAM can issue the consequent precharge command sooner [57]. Therefore, in specific cases, STT-MRAM tRC (Row cycle) can be shorter than DRAM even with a longer tRCD and and tRP.

We understand that the ranges of the STT-MRAM timing parameters presented in Table 4.2 may change in the future, with new information publicly released and along with the evolution of the technology, but the overall approach that we propose should persist.

## 4.3 STT-MRAM power estimations

Power estimations are inseparable parts of evaluating any novel technology. Performance estimation accompanied with power consumption help to decide where a particular technology may serve best. A technology that is very good from a performance perspective but has a high power consumption may not be suitable for specific use cases, (i.e. battery supported hand-held devices). Similarly, a technology which is power efficient but severely lacks in performance may also not be an ideal choice for some computing domains. To perform an estimation on STT-MRAM main memory power consumption, it is essential to have detailed configuration parameters for power estimation. Since STT-MRAM is a novel technology which is constantly evolving, it is challenging to have a stable set of power configuration parameters. In addition, since STT-MRAM is in a competitive development stage, the manufacturers are being very careful not to release any key information regarding the power models of their devices. At this point, there are no definitive information released (i.e. detailed data-sheet) from any manufacturer on STT-MRAM current parameters.

As explained in Section 4.2, the only fundamental difference between DRAM and STT-MRAM main memory is their storage cell technology, therefore the only current parameters that would change for STT-MRAM are the ones associated with accessing these cells. We carefully analyze the current parameters that model power consumption of a DRAM device, and based on our understanding of STT-MRAM operation, we assume that among DRAM current parameters, only four will change for STT-MRAM. These parameters correspond to *Active–Precharge Current* (`IDD0`), *Active–Read–Precharge Current* (`IDD1`), *Operating Burst Current* (`IDD4`) and *Bank Interleave Read Current* (`IDD7`). Since, STT-MRAM employs current-mode operation to access its cells, it requires comparatively more current than DRAM's voltage-mode cell operations. Therefore, we perform a sensitivity analysis on the higher side of these parameters using the same methodology that we used to estimate the timing parameters, (see Section 4.2). Considering STT-MRAM does not require refresh, *Refresh Current* (`IDD5`) and *Self Refresh Current* (`IDD6`) are set to 0. The remaining current parameters which generally fall into *Power-Down* and *Standby* category do not change from DRAM to STT-MRAM since they are not associated with any operation accessing the cells. A recent research report from IBM also affirms that these current parameters do not deviate

TABLE 4.3: DRAM and STT-MRAM current parameters used in the study (in mA)

| Current Parameters | Description | DRAM | ST-1.2 | ST-1.5 | ST-2.0 |
|---|---|---|---|---|---|
| **IDD0** | **Active–Precharge Current** | **1305** | **1566** | **1957** | **2610** |
| **IDD1** | **Active–Read–Precharge Current** | **1395** | **1674** | **2092** | **2790** |
| IDD2P | Precharge Power–Down Exit Current | 846 | 846 | 846 | 846 |
| IDD2Q | Precharge Quiet Standby Current | 1030 | 1030 | 1030 | 1030 |
| IDD2N | Precharge Standby Current | 1050 | 1050 | 1050 | 1050 |
| IDD3P | Active Power–Down Current | 990 | 990 | 990 | 990 |
| IDD3N | Active Standby Current | 1310 | 1310 | 1310 | 1310 |
| **IDD4** | **Operating Burst Current** | **1765** | **2118** | **2647** | **3530** |
| IDD5 | Refresh Current | 1940 | 0 | 0 | 0 |
| IDD6 | Self Refresh Current | 246 | 0 | 0 | 0 |
| **IDD7** | **Bank Interleave Read Current** | **2160** | **2592** | **3240** | **4320** |

from DRAM to STT-MRAM [78].

All current parameters used in this study to estimate STT-MRAM power consumption is listed in Table 4.3. In the table, the first column represents the generic representation of the current parameters, the second column lists the description of current parameters, the third column lists DRAM current parameters for a DDR3-1600 device [79]; fourth, fifth and sixth column list the current parameters for ST-1.2, ST-1.5 and ST-2.0, with deviations of 1.2x, 1.5x and 2x from respective DRAM current parameters.

## 4.4 Experimental setup

We analyze the system performance impact with STT-MRAM main memory in comparison to DRAM main memory. In this section we present the application benchmark suite, CPU and main memory simulator used for this study.

### 4.4.1 Benchmark suite

STT-MRAM main memory was evaluated on a set of eleven integer and twelve floating point benchmarks from the SPEC CPU 2006 suite [80]. Table 4.4 lists the benchmarks with their application areas used for the study [2].

TABLE 4.4: SPEC CPU 2006 benchmarks used in the study

| Benchmark | Application Area | Language |
|---|---|---|
| h264ref | Video Compression | C |
| libquantum | Quantum Computing | C |
| perlbench | Programming Language | C |
| gcc | C Compiler | C |
| mcf | Combinatorial Optimization | C |
| gobmk | Artificial Intelligence | C |
| hmmer | Gene Sequence Analysis | C |
| sjeng | Artificial Intelligence | C |
| xalancbmk | XML Processing | C++ |
| aster | Path-finding Algorithm | C++ |
| bzip2 | Compression | C |
| gamess | Quantum Chemistry | Fortran |
| tonto | Quantum Chemistry | Fortran |
| namd | Molecular Dynamics | C++ |
| gromacs | Molecular Dynamics | C,Fortran |
| dealII | Finite Element Analysis | C++ |
| sphinx3 | Speech Recognition | C,Fortran |
| leslie3d | Fluid Dynamics | Fortran |
| cactusADM | General Relativity | C,Fortran |
| GemsFDTD | Computational Electromagnetics | Fortran |
| milc | Quantum Chromodynamics | C |
| bwaves | Fluid Dynamics | Fortran |
| lbm | Fluid Dynamics | C |

### 4.4.2 CPU Simulation

In order to evaluate the STT-MRAM main memory system, we simulated an Intel Sandy Bridge-EP E5-2670 processor, which is a dominant architecture in HPC systems [74]. Intel Sandy Bridge-EP E5-2670 comprises eight cores operating at 3.0 GHz. Although the processors support hyper-threading at core level, this feature is disabled, as in most of the HPC systems. Sandy

[2] The benchmarks `omnetpp`, `zeusmp`, `povray` and `wrf` fail to execute in our simulation infrastructure. We also exclude `calculix` and `soplex` as they produce inconsistent and inconclusive results.

Bridge processors are connected to main memory through four DDR3-1600 channels.

We used the ZSim [81] system simulator for the experiments. Developed by researchers from MIT and Stanford University, ZSim is designed for simulation of large-scale systems. However, ZSim was originally developed to simulate the Intel Westmere architecture which is obsolete at this point. One of the tasks that we had to perform was to upgrade and validate ZSim for Intel Sandy Bridge processors. The ZSim upgrade was done by following the Intel documentation [82], and it comprised several steps. First, we adjusted the latency of numerous instructions, and added support the for the new x86 vector instruction extensions i.e. AVX, SSE3, that are supported by Sandy Bridge and were not supported by Westmere. We also improved the fusion of the instructions into a single micro-op, and we increased the number of entries in the reorder buffer from 128 (Westmere) to 168 (Sandy Bridge). Finally, the simulated hardware platform comprises a detailed model of the Sandy Bridge-EP E5-2670 cache hierarchy [75]. This Sandy Bridge E class processor has eight cores, dedicated L1 instruction and data caches of 32 KB each, a dedicated L2 cache of 256 KB and a shared L3 cache of 20 MB, summarized in Table 3.2. In all three levels of cache memory, we used the Least Recently Used (LRU) cache replacement policy and for the L3 cache level we implemented the slice allocation hash function by Maurice *et al.* [83].

### 4.4.3 Main memory simulation

Both DRAM and STT-MRAM main memory are simulated with DRAMSim2 [3]. DRAMSim2 is a cycle accurate model of a DRAM-based main

TABLE 4.5: Main memory simulator settings

| Parameters | Values |
|---|---|
| NUM_CHANS | 4 |
| JEDEC_DATA_BUS_BITS | 64 |
| TRANS_QUEUE_DEPTH | 32 |
| CMD_QUEUE_DEPTH | 32 |
| EPOCH_LENGTH | 100000 |
| ROW_BUFFER_POLICY | close_page and open_page |
| ADDRESS_MAPPING_SCHEME | scheme2 |
| SCHEDULING_POLICY | rank_then_bank_round_robin |
| QUEUING_STRUCTURE | per_rank |

memory. All major components in a modern memory system are modeled as their own respective objects within the source code, including: ranks, banks, command queue, the memory controller, etc. DRAMSim2 is developed by the University of Maryland and it is validated against manufacturer Verilog models. DRAMSim2 can be integrated with various CPU simulators through a fairly simple interface. Tables 4.1 and 4.2 summarize DRAM and STT-MRAM main memory parameters used in this study, while Table 4.5 lists the simulator settings for the main memory.

Simple integration of ZSim and DRAMSim2 may lead to an underestimation of the main memory access latency. ZSim simulates memory access up to the last level of cache, while DRAMSim2 is focused on the detailed timing simulation of the memory device. This means that a direct merge of ZSim and DRAMSim2 would not consider to the delay contributed by all the circuitry between the last level cache and main memory device, including the memory controller and the memory channel. In order to account for this delay, we introduce an *extra latency* of 70ns between ZSim and DRAMSim2. The estimation of this *extra latency* has been validated on a real machine [84].

### 4.4.4 Validation

We have validated the simulation infrastructure against the actual hardware comprising a Sandy Bridge EP E5-2670 processor connected to four DDR3-1600 channels. The CPU pipeline is validated by using a set of synthetic benchmarks with a main loop comprised of a single instruction type. Different version of the synthetic benchmarks test in-order and out-of-order execution. Our test suite is comprised of 519 synthetic benchmarks, covering almost all instructions included in the instruction set architecture (ISA) of the Sandy Bridge EP E5-2670 processor. Cache hierarchy and main memory latency are validated with *lmbench* [85]. The *lmbench* benchmark essentially measures the access time of random accesses to an array of a given size. In our experiments, we covered the array sizes from 4KB (fitting into the L1 cache), to 4GB (main memory access). Finally, we validated the overall simulation infrastructure using the SPEC CPU 2006 benchmarks, by comparing its execution on the actual hardware with the simulated one.

### 4.4.5 Methodology

In the experiments summarized in this thesis, we simulated eight instances of SPEC CPU 2006 benchmarks running on a single Sandy Bridge socket, i.e. one benchmark instance per core. Each benchmark instance was executed for 50 billion instructions. To compare DRAM and STT-MRAM memory systems, we measured the performance for each process under study in two main memory configurations. We report the average performance difference between the DRAM (baseline) and the STT-MRAM memory system, and standard deviation of all the measurements.

## 4.5 Results

### 4.5.1 Performance analysis

In this section, we present the results of STT-MRAM main memory performance impact in comparison to DRAM. We extend our results for both *open page* and *close page* policies. In *open page* policy, a page is kept open to take advantage of subsequent "page hits" until a "page conflict" appears. In contrary, in *close page* policy, a page is immediately closed after a read or write operation is performed.

For STT-MRAM main memory, we test three sets of timings, namely ST-1.2, ST-1.5, ST-2.0. In the ST-1.2 configuration, the specific STT-MRAM timing parameters: tRCD, tRP, tFAW and tRRD are 1.2× slower w.r.t. the corresponding DRAM timings. Similar applies to ST-1.5 and ST-2.0 configuration, as summarized in Table 4.2.

Figure 4.1 shows the overall system performance impact of the ST-1.2 configuration on SPEC integer benchmarks. The horizontal bars represent system performance deviation for the corresponding benchmarks listed on the *X* axis. This deviation has been measured by the change of Cycles per Instruction (CPI) values between systems with DRAM and STT-MRAM main memory. Actually, for all the integer benchmarks we detect a negative CPI change meaning that the benchmarks experience a speedup with the STT-MRAM main memory.


FIGURE 4.1: ST-1.2 Configuration (integer benchmarks)

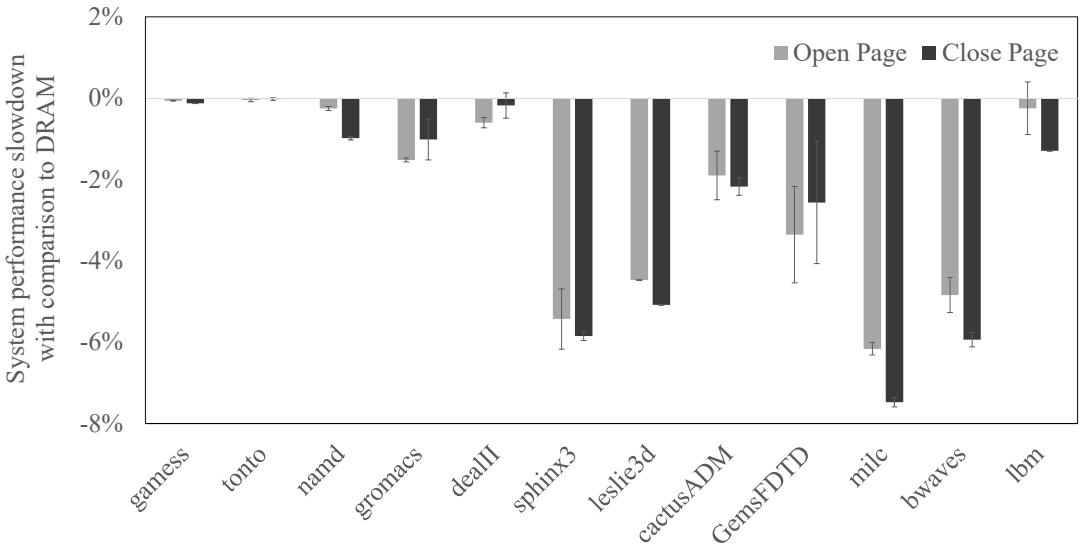


FIGURE 4.2: ST-1.2 Configuration (floating point benchmarks)

With open page policy, the speedup ranges from 0.0% (`h264ref`) to 4.8% (`mcf`); and 2.2% in average. And with close page policy, the results does not change much: speedup ranging from 0.3% (`gobmk`) to 4.8% (`mcf`); and 2.2% in average. Floating point benchmarks with the ST-1.2 configuration follow a similar trend but with a higher amplitude, see Figure 4.2. Four out of twelve benchmarks achieved around 5% system performance improvement in comparison to DRAM for both open and close page policies. Average speedups for all the benchmarks are 2.4% (open page) and 2.7% (close page).

Although ST-1.2 is apparently configured to be comparatively slower w.r.t

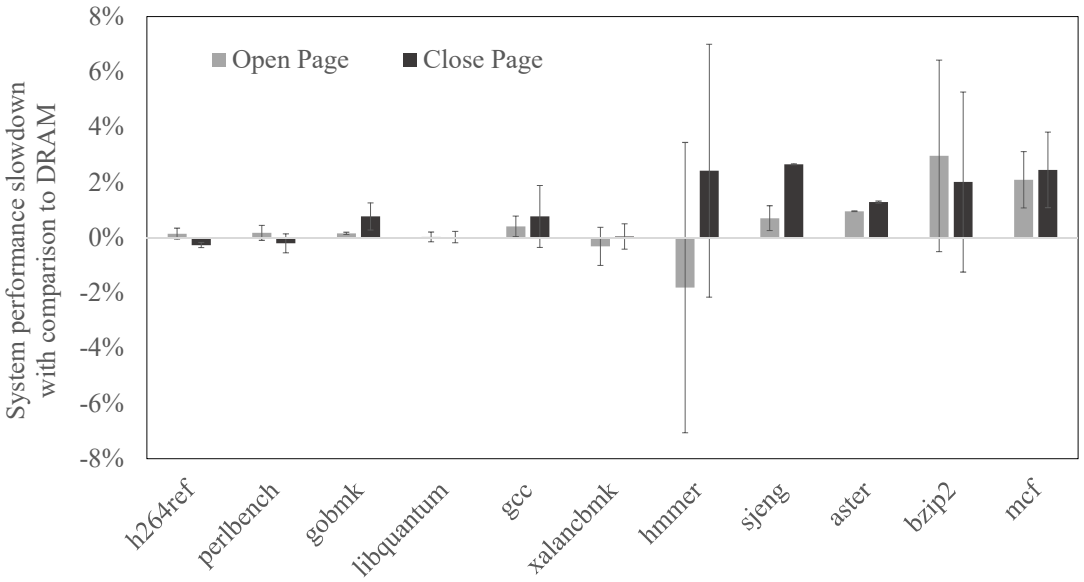


FIGURE 4.3: ST-1.5 Configuration (integer benchmarks)

DRAM, the results with this configuration report performance improvement (speedup) for all benchmarks over DRAM. This is due to the operation sequence of STT-MRAM, which is different from DRAM, as detailed in Section 4.2. Unlike DRAM, STT-MRAM has a non-destructive read which does not have to write-back; meaning it can issue precharge commands sooner [57]. Hence, STT-MRAM tRC (Row cycle) for this configuration can be shorter than DRAM even with a longer tRCD and and tRP.

ST-1.5 results, see Figure 4.3, show performance degradation for most integer benchmarks, 0.5% with open page policy and 1.1% with close page policy on average. However, the benchmarks `h264ref` and `perlbench` with close page policy, as well as `xalancbmk` and `hmmer` with open page policy, still experience a speedup on the STT-MRAM memory systems. Floating point benchmarks, experience higher slowdowns, 2.5% with open page policy and 2.8% with close page policy on average and around 10% in the worst case (`lbm`) for both policies, see Figure 4.4.

For configuration ST-2.0, all benchmarks experience slowdown w.r.t. to the DRAM. With open page policy slowdown of the integer benchmarks ranges between 0.6% (`h264ref`) and 14.1% (`mcf`), and it is 5.4% on average. With close page policy the slowdown of the integer benchmarks ranges between 0.2% (`h264ref`) and 15.8% (`mcf`), and it is 6.5% on average , see Figure 4.5. Floating point benchmarks are even more sensitive to the delays realized in the ST-2.0 configuration, see Figure 4.6. Five of the benchmarks experience a

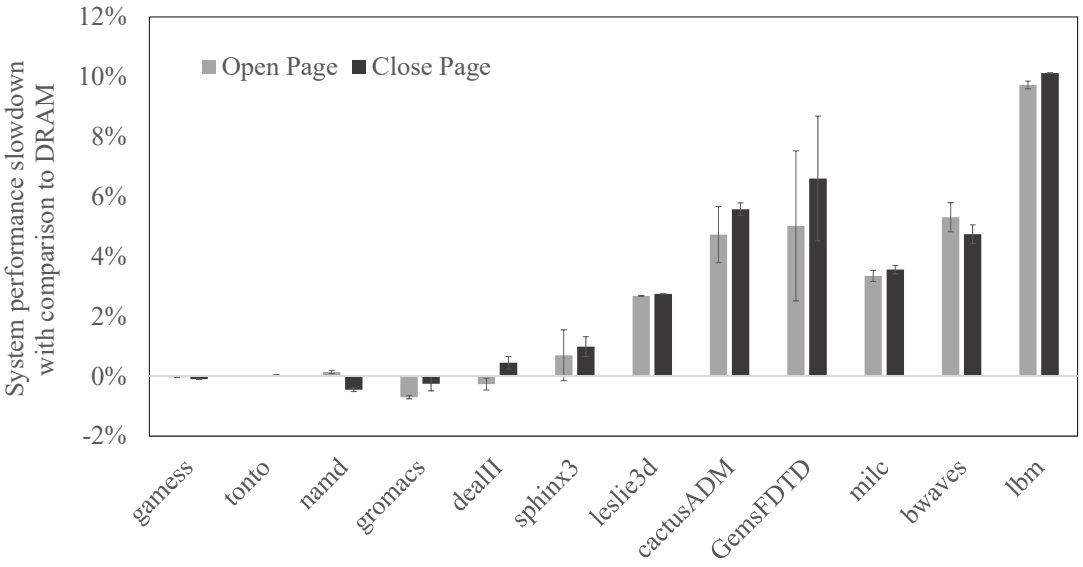


FIGURE 4.4: ST-1.5 Configuration (floating point benchmarks)

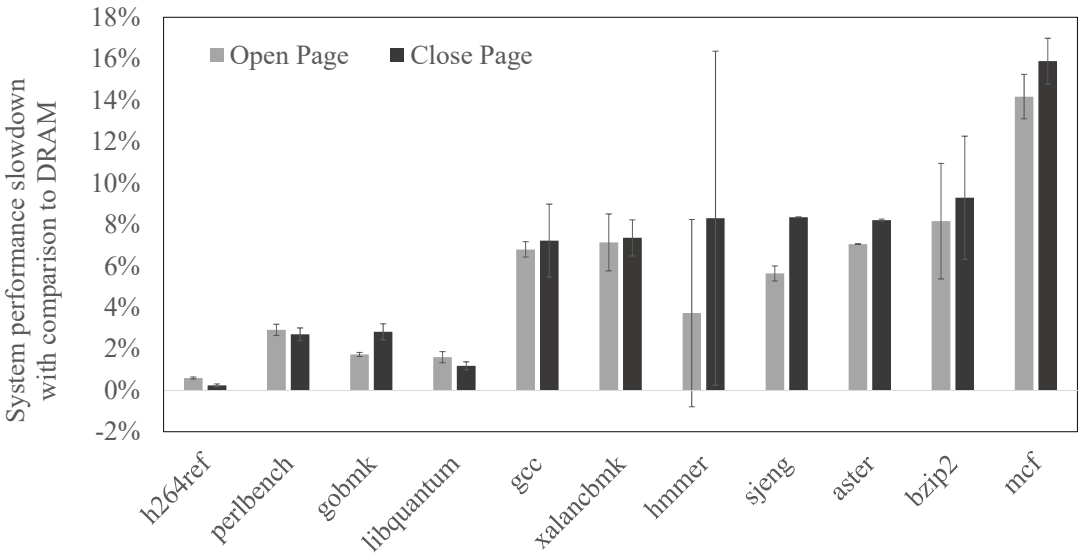


FIGURE 4.5: ST-2.0 Configuration (integer benchmarks)

slowdown of less than 2% with both policies, but for the remaining ones the slowdown ranges between 11.4% (`sphinx3`) and 26.9% (`lbm`) with open page policy; and 12% (`sphinx3`) and 29.6% (`lbm`) with close page policy, leading to the average slowdown of 11.3% and 11.9% for open and close page policy, respectively.

Overall, the results indicate that the system performance experience a minor impact for variation of STT-MRAM timing parameters associated with row operations: tRCD, tRP, tFAW and tRRD. Even when these timing parameters are set to be twice as slow as DRAM, the system performance degrades only by an average of 5.4% and 11.3% for integer and floating point benchmarks

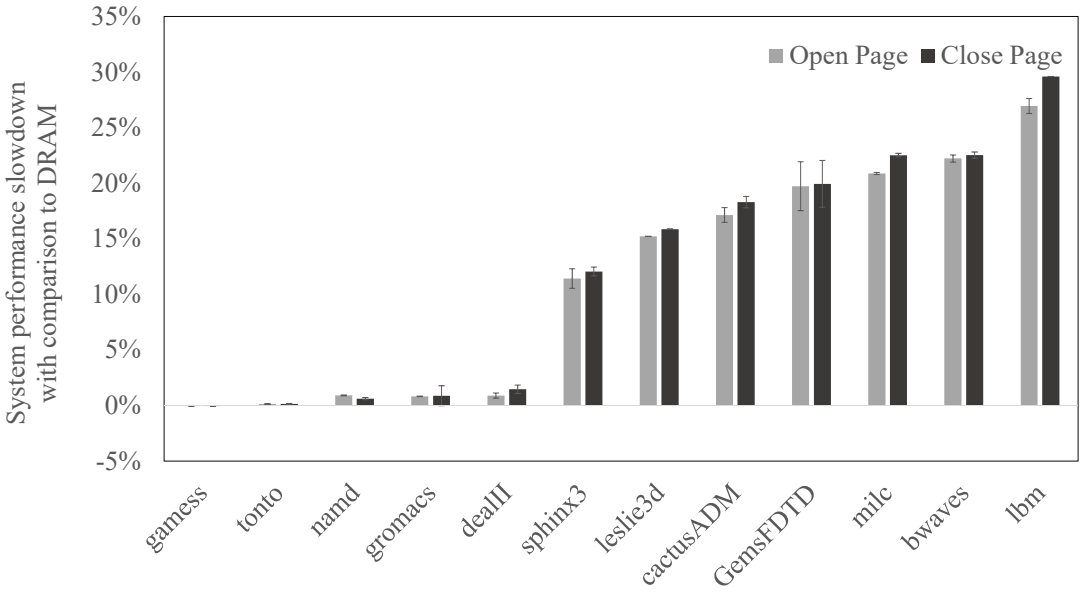


FIGURE 4.6: ST-2.0 Configuration (floating point benchmarks)

(open page policy), respectively. For the ST-1.5 configuration, the STT-MRAM main memory based system experiences an average slowdown of only 0.5% for the integer and 2.5% for the floating point benchmarks (open page policy). For STT-MRAM main memory with the ST-1.2 configuration, we actually measure a speedup w.r.t. to DRAM. The results also imply that page policies (open or close) do not have a significant impact on overall system performance.

### 4.5.2 Power consumption

In this section, we present comparative power measurements of DRAM and STT-MRAM configurations of the main memory. We model DRAM and STT-MRAM power with the current parameters listed in Table 4.3. We incorporate these parameters into DRAMSim2 simulation infrastructure to obtain power measurements in three groups: *Activation and Pre-charge power*, *Burst Power* and *Background Power*. *Activation and Pre-charge power* corresponds to the power consumption to activate/pre-charge rows, *Burst Power* refers to the power consumption for read/write activities, and *Background Power* relates to all other power consumption (excluding *Refresh Power* [3]).

[3]We do not compare *Refresh Power* since STT-MRAM does not require refresh and its corresponding current parameters are set to 0.

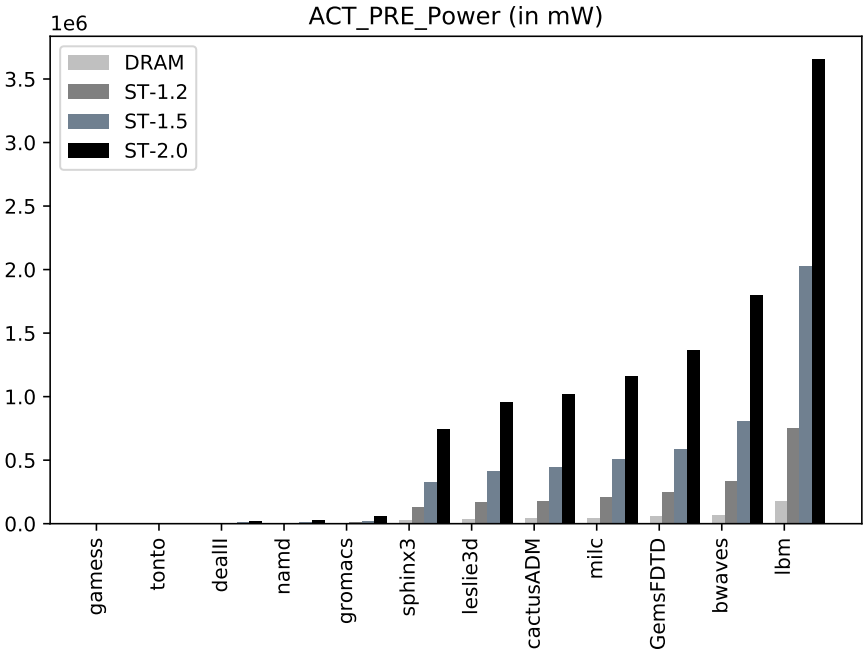


FIGURE 4.7: Activate/Pre-charge power consumption of floating point benchmarks

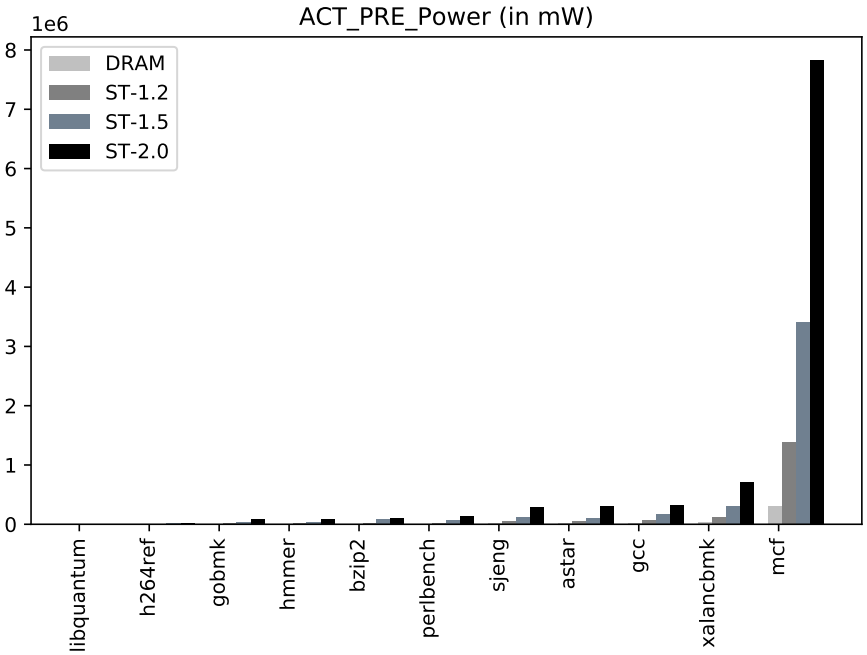


FIGURE 4.8: Activate/Pre-charge power consumption of integer benchmarks

Figure 4.7 to Figure 4.12 present overall power consumption for floating point and integer benchmarks under study. In these figures, X-axis lists the power consumption in mili-Watts (mW) for DRAM and three configurations of STT-MRAM while Y-axis lists the benchmarks.

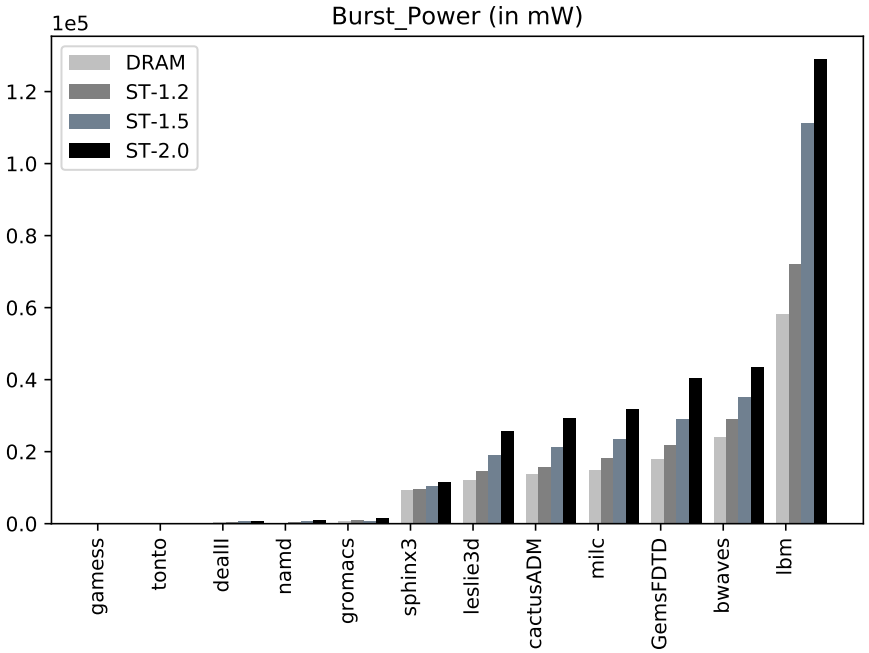


FIGURE 4.9: Burst (Read/Write) power consumption of floating point benchmarks

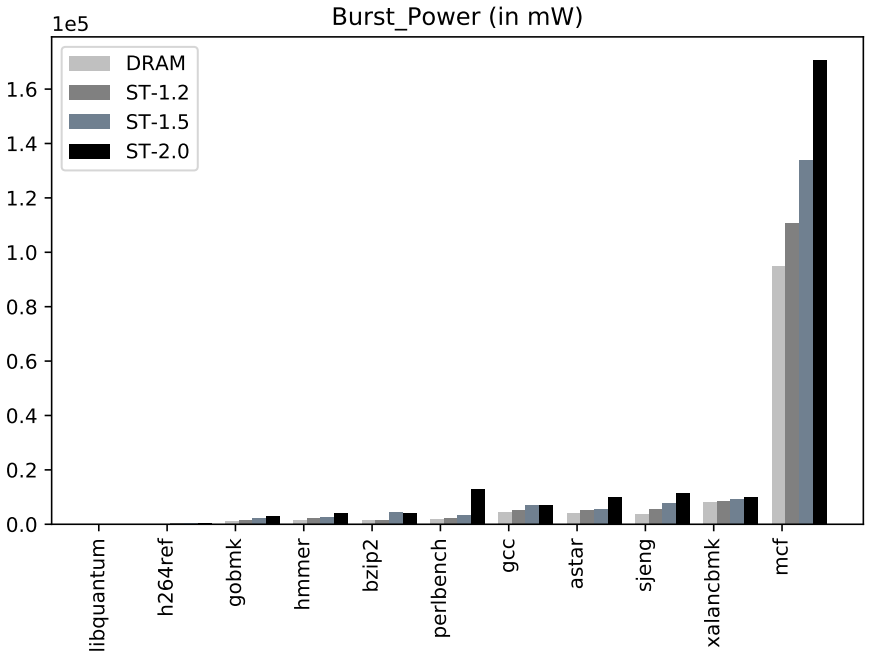


FIGURE 4.10: Burst (Read/Write) power consumption of integer benchmarks

Figure 4.7 and Figure 4.8 shows power consumption due to activation/pre-charge activity for floating point and integer benchmarks, respectively. The results suggest that, *activation and pre-charge power* consumption significantly increases for STT-MRAM configurations w.r.t DRAM. This is expected since, this operation activates rows, and the access mechanism of STT-MRAM

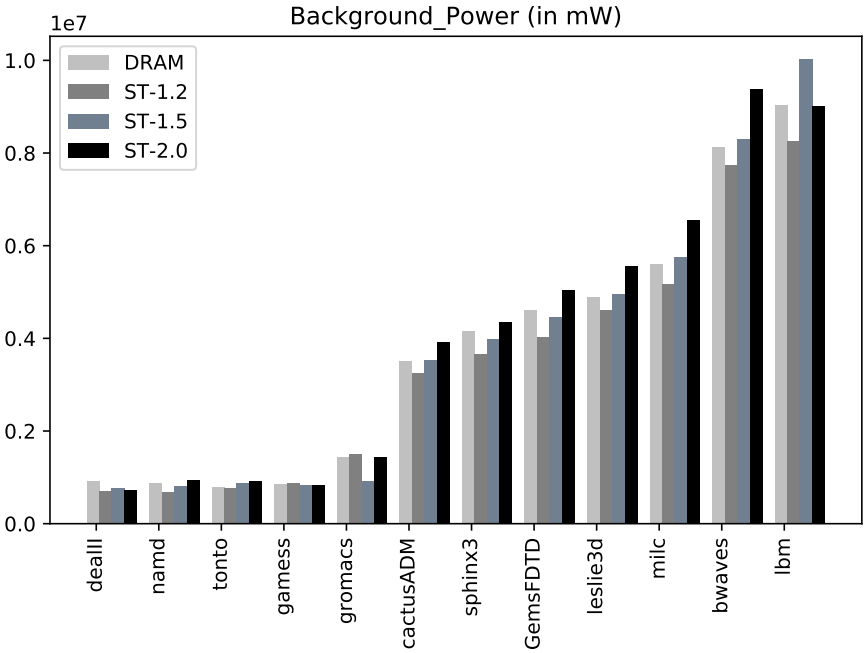


FIGURE 4.11: Background power consumption of floating point benchmarks

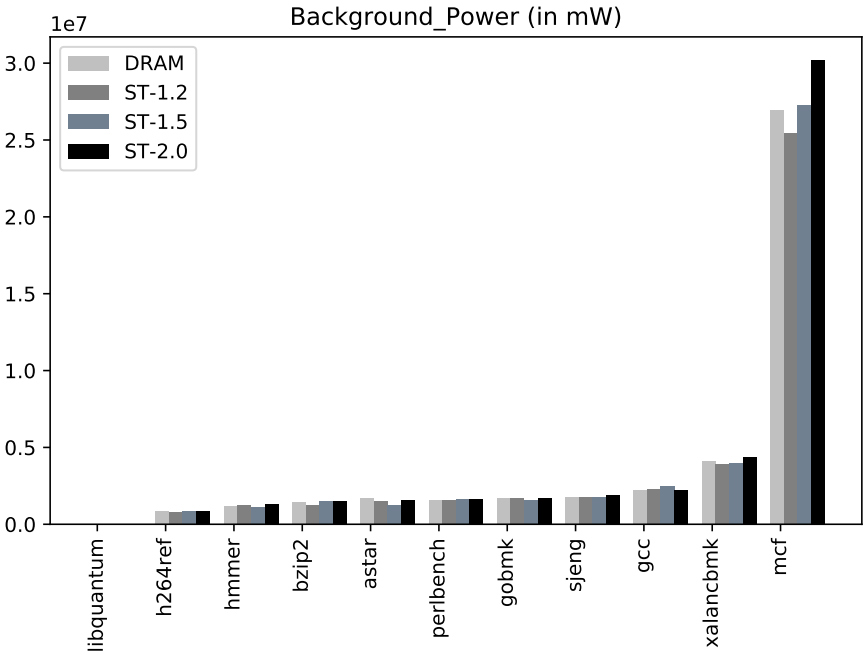


FIGURE 4.12: Background power consumption of integer benchmarks

(current-mode operation) cells are significantly different in comparison to DRAM (voltage-mode operation).

Figure 4.9 and Figure 4.10 report power consumption originating from read/write activities for floating point and integer benchmarks, respectively.

*Burst power* is moderately increased for STT-MRAM configurations in comparison to DRAM.

Figure 4.11 and Figure 4.12 represents *background power* for all benchmarks under study. The results suggest that background power consumption does not deviate much from DRAM to STT-MRAM systems.

## 4.6 Summary

This section of the thesis provides a major breakthrough on STT-MRAM main memory by publishing detailed timing and current parameters for the first time which enables researchers to perform reliable system level simulation of this memory technology. Furthermore, we seamlessly incorporate STT-MRAM timing parameters into the DRAMSim2 memory simulator and use it as a part of the simulation infrastructure for high performance computing systems. The results of our simulations affirm that STT-MRAM main memory would provide performance comparable to DRAM systems and indicates that *Activation and Pre-charge power* escalates the most with STT-MRAM. The detailed timing parameters published with this work are now adopted in the main repositories of leading main memory simulators used today.

# Chapter 5

# Performance and WCET Implications in Real-Time Systems

## 5.1 Introduction

In the final phase of the thesis, we intent to explore a specific domain that has a great potential to exploit the advantages of STT-MRAM. Special STT-MRAM features such as intrinsic radiation hardness, non-volatility, zero stand-by power and capability to function in extreme temperatures makes it particularly suitable for aerospace, avionics and automotive applications. Such applications often have real-time requirements — that is, certain tasks must complete within a strict deadline. Analyzing whether this *deadline* is met requires Worst Case Execution Time (WCET) analysis, which is a fundamental part of evaluating any real-time system.

In this section, we investigate the feasibility of using STT-MRAM in real-time embedded systems by analyzing average system performance impact and WCET implications.

## 5.2 Experimental setup

### 5.2.1 Processor platform

For our experiments, we selected the Cobham Gaisler LEON4 Next Generation Microprocessor (NGMP) [6]. The NGMP is targeted to be used for future

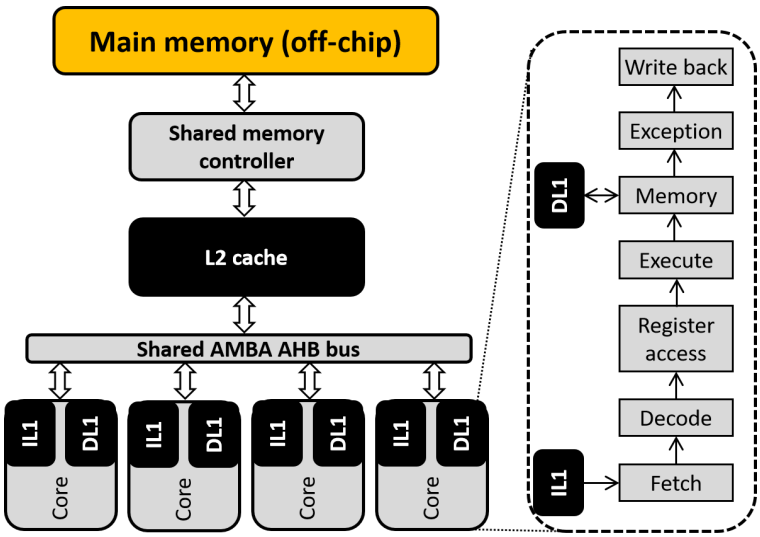


FIGURE 5.1: Schematic view of the Next Generation Microprocessor (NGMP)

space missions by the European Space Agency (ESA). It is a good representative of advanced real-time embedded processors, which start introducing multiple cores per processor. The most important features of NGMP CPU are summarized in Figure 5.1 and Table 5.1. Each core has a private 16 KB L1 instruction and data cache, while a 256KiB L2 cache is shared among all four cores. The cores are connected through a 128-bit AHB AMBA bus arbitrated by a round-robin policy.

### 5.2.2 Main Memory platform

We model the STT-MRAM timings based on the parameters that are derived in section 4.2 in collaboration with Everspin Technologies Inc., one of the leading STT-MRAM manufacturers [86]. The published timing parameters enable a reliable methodology to simulate STT-MRAM without releasing confidential information about any product. We briefly summarize the procedure and reasoning of estimating the timing parameters.

STT-MRAM memory devices are DDR$x$ compatible, with the same or very similar organization and CPU interface, as conventional DRAM. Also, both DRAM and STT-MRAM main memory devices use a row buffer as the interface between the cell-arrays and the memory bus. Since the circuitry beyond the row buffer for DRAM and STT-MRAM is essentially the same — once the data is in the row buffer, STT-MRAM timing parameters for the consequent

TABLE 5.1: Next Generation Microprocessor (NGMP): Main features

| Feature | Description |
|---|---|
| Cores | 4 |
| ISA | SPARC v8 |
| Pipeline stages | Fetch, decode, register, execute, memory, exceptions, commit |
| Core Frequency | 150 MHz |
| Superscalar | No |
| Out-of-Order | No |
| L1 D-cache | Private (per-core)<br>16KB, 32 byte/line, 4-way<br>Write-through, Write no-allocate |
| L1 I-cache | Private (per-core)<br>16KB, 32 byte/line, 4-way |
| L2 cache | Shared: 4 cores<br>Unified: Data and Instructions<br>256KB, 32 byte/line, 4-way<br>Copy-back, Write-allocate |
| FPU | Double precission IEEE-754 |

operations are the same as for DRAM. Therefore, the values of all the timing parameters that are not associated with row operations do not change from DRAM to STT-MRAM (e.g. tBURST, tCAS, tRTP, tWTR etc.).

The only fundamental difference in STT-MRAM and DRAM main memory is their storage cell technology — MTJ and capacitor, respectively. Due to the difference in the cell access mechanism of these two memory technologies, the timing parameters associated with STT-MRAM row operations would deviate from DRAM (as detailed in Section 4.2). In this study, we select three sets of STT-MRAM timings, with 1.2×, 1.5× and 2× slower row-related operations w.r.t. DRAM, as summarized in Table 5.2. Simulations performed with these timing parameters give us a reliable range of possible system performance impact for upcoming STT-MRAM main memory devices [86].

### 5.2.3 Simulation infrastructure: SoCLib & DRAMSim2

The NGMP processor is simulated with a SoCLib-based simulator [87]. The simulator is designed to conceptually separate the functional emulation

TABLE 5.2: DRAM and STT-MRAM timing parameters associated with row operation (DDR2-667 cycles)

| Timing Parameters | Description | DRAM | ST-1.2 | ST-1.5 | ST-2.0 |
|---|---|---|---|---|---|
| tRCD | Row to column command delay | 5 | 6 | 8 | 10 |
| tRP | Row precharge | 5 | 6 | 8 | 10 |
| tFAW | Four row activation window | 13 | 16 | 20 | 26 |
| tRRD | Row to Row activation delay | 3 | 4 | 5 | 6 |
| tRFC | Refresh cycle time | 43 | 0 | 0 | 0 |

from the timing behavior. Functional emulation executes the instructions according to a particular Instruction Set Architecture (ISA) and provides all the relevant information about the instruction execution, such as the instruction address, register use, instruction type, result, etc. The timing simulator analyzes the timing behavior of instructions for a given hardware implementation, e.g., it determines the latency of load instructions. It is built in a modular way so that each hardware component maps to a component of the timing simulator. This allows for extensions such as the addition of more accurate memory models. Simulator parameters used in the study have been previously verified [88] to accurately model the behavior of the GR-CPCI-LEON4-N2X [89], a board implementing the NGMP processor.

In this study we consider a system in which the NGMP CPU is connected to a DDR2-667 memory device. Both DRAM and STT-MRAM main memories are simulated with the DRAMSim2 simulator [90], that is integrated with the SoCLib simulator through a fairly simple interface. DRAMSim2 is a cycle accurate model of a DRAM main memory validated against manufacturer Verilog models. For the simulation of the DRAM, we use the timing parameters from the automotive DDR2 SDRAM data-sheet provided by Micron Technology, Inc [91]. The estimation of the timing parameters for the STT-MRAM main memory is described in Section 5.2.2.

### 5.2.4 Timing analysis of real-time systems

Real-time embedded systems are subject to strict timing constraints as defined by applicable safety standards. Failing to meet specific deadlines for

those systems may lead to fatal consequences, specially for the most (safety or mission) critical applications.

Since timing is a critical concern in real-time embedded systems, validation and certification of these systems requires sufficient evidence that tasks will complete within assigned time budgets, i.e. before specific deadlines. This evidence is typically provided using timing analysis techniques that estimate the Worst-Case Execution Time (WCET) of tasks running in the target system.

There are several approaches to perform the timing analysis from static timing analysis (STA) to measurement-based timing analysis (MBTA), each one with its own pros and cons [92]. STA is performed statically without executing the code [93]. It has been proven the most convenient solution for very simple microcontrollers, where accurate and reliable timing models of the processor can be built. However, STA faces severe limitations when considering complex hardware and software, which challenge the reliability of timing models and the tightness of timing bounds.

In this paper we focus on measurement-based timing analysis (MBTA) [94][95]. MBTA builds on the collection and operation of execution time measurements of the application running on the target platform. It is the most widely adopted solution by industry due to its relatively low cost of applicability. MBTA has been shown to provide trustworthy estimates for highest-criticality software in Avionics [96] when it runs on simple processors.

Increased hardware and software complexity, however, reduces the confidence that can be placed on WCET estimates derived with MBTA [93]. Statistical techniques have been studied for MBTA to derive bounds to execution time distributions. In particular, measurement-based probabilistic timing analysis (MBPTA) [97] has matured in recent years. MBPTA delivers a probabilistic WCET (pWCET) function that upper-bounds the (probabilistic) execution time distribution of the program (pET) at any exceedance probability, see Figure 5.2. MBPTA has been successfully applied to industrial case studies [98][99] and its impact on certification has been addressed [100].

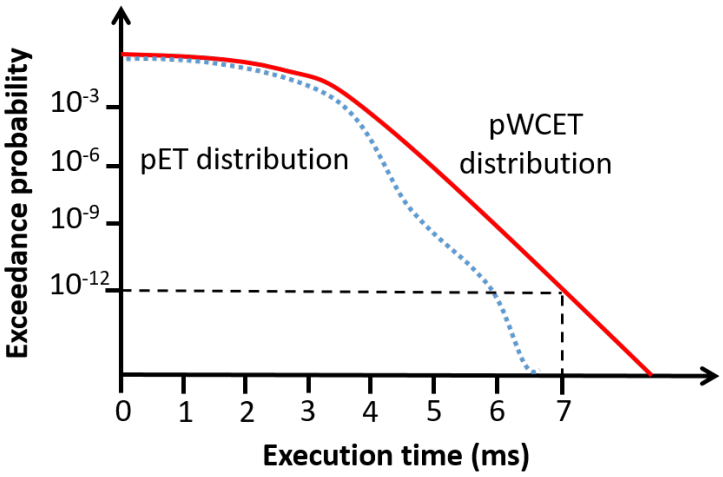


FIGURE 5.2: pWCET distribution

Probability (Y-axis) that the application execution time (in any given run) exceeds the corresponding time on the X-axis. In this example, the pWCET estimate (7ms) is exceeded with a probability of $10^{-12}$.

### 5.2.5 Measurement-based probabilistic timing analysis

MBPTA has been complemented with solutions that inject randomization in a program's timing behavior to relieve the user from controlling those *jittery resources* affecting the execution time variability of a program. Randomization makes sure that the potential behavior that a given jittery resource (e.g. caches) can exhibit, is naturally (and randomly) explored in every new test, enabling the derivation of probabilistic guarantees. This is contrary to *deterministic* platforms where randomization is not enabled and execution time is not expected to deviate. For the probabilistic platform, randomization has been implemented at hardware level (e.g. random arbitration policies and random placement/re-placement techniques) that are now part of a commercial product for the space domain [101]; and with software techniques that work at the compiler/linker level [102] or source code level [103]. In order to enforce probabilistic guarantees to hold during operation, randomizations must be kept enabled, so that the execution time distribution analyzed matches (or upper-bounds) that during operation. Then, by using MBPTA on the collected execution measurements, reliable pWCET estimates are obtained. Figure 5.3 illustrates the process of obtaining pWCET estimates. In this work we build upon MBPTA-CV [97], a MBPTA technique whose implementation has been recently made publicly available [104]. In the following section we summarize the steps of the MBPTA process used in this study.

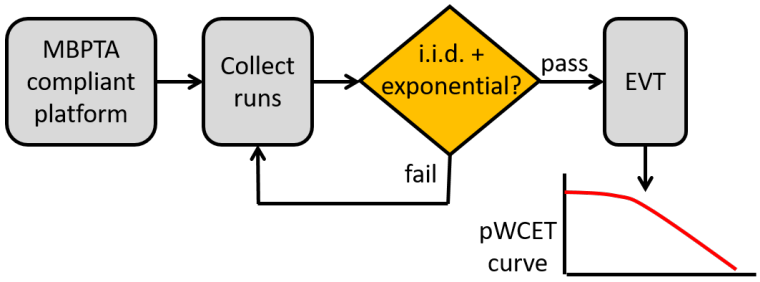


FIGURE 5.3: Schematic of the MBPTA application process

**MBPTA compliant platform.**

MBPTA relies on the use of platforms with specific timing properties [105] – either provided by hardware or software means – that allow obtaining measurements at analysis that *represent* the behavior during operation. In particular, those platforms build upon time upperbounding and time randomization so that measurements at analysis correspond to a distribution (random variable) that upperbounds probabilistically the behavior during operation. By introducing those properties in the hardware/software platform, which has been proved to cause marginal performance degradation [105], collecting representative measurements has been shown to be independent of the use of complex hardware features such as cache hierarchies with intricate behavior (unified data/instruction caches, inclusive caches, etc) and multicores, among others. In fact, MBPTA does not pose any explicit constraint on the use of *any* hardware feature as long as specific properties are met in the hardware/software platform and measurement collection process [106].

**Collecting Runs**

Once measurements are guaranteed to match or upperbound operation time behavior, MBPTA requires a sufficiently large execution time sample. In this study, each benchmark is executed for a thousand times with each memory configuration (*DRAM*, *ST-1.2*, *ST-1.5* and *ST-2.0*). It takes a few minutes to several hours (depending on the benchmark) to perform one execution.

### Independent and identically distributed – i.i.d. test

After we accumulate sufficient execution time measurements from a probabilistic platform for a benchmark with a specific memory configuration, MBPTA-CV assesses whether the execution time measurement are statistically independent and identically distributed (i.i.d.) with appropriate tests. While the process measured is probabilistically i.i.d. by construction (each benchmark is executed independently with an identical configuration), random samples might sporadically fail to achieve those properties statistically. For instance, we may roll a dice 6 times and obtain statistically non-i.i.d. samples (e.g. six times the same value). However, since the variable observed (execution time) is probabilistically i.i.d., whenever tests are failed, a larger sample needs to be collected since the sample will converge to the variable studied eventually.

### Extreme value theory

Once the sample is accepted as i.i.d., MBPTA-CV builds upon Extreme Value Theory (EVT) [107] to deliver a probabilistic WCET (pWCET) estimate of the program. A pWCET estimate (an example is shown in Figure 5.2 for illustrative purposes) is a continuous function upperbounding the exceedance probability for any high execution time. pWCET functions are typically plotted in the form of a Complementary Cumulative Distribution Function (CCDF), also known as tail distributions, with logarithmic y-axis scale. Their intrepretation is such that the particular probability on the y-axis is an upperbound to the true exceedance probability of the execution time on the x-axis. The exceedance probability can be set arbitrarily low so that it can be deemed irrelevant w.r.t. the requirements of the function (e.g. below $10^{-8}$ failures per hour). Note that, contrarily to some people's believings, WCET estimates can potentially be exceeded since even the most stringent timing analysis processes have some form of residual risk associated to the modelling of the hardware (for STA) or the measurement collection process (MBTA/MBPTA). Hence, upperbounds to the exceedance rates are compatible with vefirication and validation processes even for the most critical functions. In general, appropriate safety measures are designed along those critical functions to either set the system to a safe state on a failure (e.g. stopping the car) or to keep it operational by means of diverse redundancy so that a single fault cannot lead

to a full-system failure. We refer the interested reader to the corresponding functional safety standards in each domain, such as ISO26262 in automotive [9] and DO178B in avionics [108]. In order to set an appropriate EVT distribution to the sample, MBPTA-CV builds upon the fact that execution times of real-time programs are finite. This guarantees that execution times can be upperbounded with exponential tails [97]. Hence, MBPTA-CV selects automatically those measurements that belong to the tail of the distribution from the sample, and tests their exponentiality. If the best fit can be rejected to be exponential or a light tail[1], then the sample does not have enough tail measurements and MBPTA-CV instructs the user to collect further measurements. Eventually, since the random variable (execution time) observed has a maximum value, this process converges and sufficient values of the tail will be collected, so they will be properly upperbounded with an exponential tail. Note that MBPTA-CV imposes, by construction of the method, that no less than 50 tail measurements can be accepted to fit the appropriate exponential distribution to the tail. Hence, not only a reliable tail model is obtained, but the confidence interval is necessarily narrow, thus preserving the tightness of the pWCET estimate.

#### pWCET estimate with confidence interval

The confidence interval, set to usual values in statistics (e.g. 90%, 95% or 99%) illustrates that tail fitting introduces low variability. In general, despite confidence intervals could be used to select the pWCET estimate, we stick to point estimation since the process already inherits some sources of pessimism that guarantee the need for upperbounding, such as the pessimism incurred by enforcing worst-case operation conditions for pWCET estimation, and using exponential tails instead of light tails[2].

[1] Light tails fall at a higher rate than exponential tails and approach a maximum value asymptotically, so they are naturally upperbounded by exponential tails.

[2] Note that the true bound must be a light tail, but due to the difficulties of selecting the right one reliably, we stick to exponential tails, which upperbound all light tails by construction [97]

TABLE 5.3: Benchmarks used in the study

| Suite | Benchmarks | Domain |
|---|---|---|
| ESA Applications | obdp, debie | Space |
| EEMBC Autobench | a2time01, aifftr01, aifirf01, aiifft01, basefp01, bitmnp01, cacheb01, canrdr01, idctrn01, iirflt01, matrix01, pntrch01, puwmod01, rspeed01, tblook01, ttsprk01 | Automotive |
| Mediabench | mesa.texgen, mesa.mipmap, epic.decode, mesa.osdemo | Media |

### 5.2.6 Benchmarks

STT-MRAM's suitability for real-time embedded systems is validated on benchmarks provided by the European Space Agency (ESA), EEMBC Autobench suite [7] and Mediabench [8]. Table 5.3 lists the benchmarks used in the study.

The European Space Agency provided two applications, On-board Data Processing (`obdp`) and `debie`. `obdp` contains the algorithms used to process raw frames coming from the state-of-the-art near infrared (NIR) HAWAII-2RG detector, already used in production systems, such as the Hubble Space Telescope. `debie` is the software that controls an instrument that observes micrometeoroids and small space debris. It has been already used in the PROBA-1 satellite.

EEMBC Autobench includes 16 benchmark kernels that mimic functionalities of production automotive, industrial, and general-purpose applications. General purpose kernels include bit-manipulation, multiplication, floating-point, matrix, cache and pointer chasing benchmarks, as well as the pulse-width modulation and shift operations typical of encryption algorithms.

In order to further investigate STT-MRAM's performance and WCET traits, we execute four more applications with high memory utilization [109] from Mediabench benchmark suite.

## 5.3 Results

### 5.3.1 Performance estimation

We execute each application under study with DRAM and three sets of STT-MRAM timing configurations, namely *ST-1.2*, *ST-1.5* and *ST-2.0*. From their individual *execution time* and *number of executed instructions* we calculate Cycles Per Instruction (CPI) values for each application with DRAM and each of the STT-MRAM configurations. We measure the overall performance slowdown by the change of CPI values between systems with DRAM and STT-MRAM. Figure 5.4 shows overall system performance impact of different STT-MRAM configurations for the benchmarks under study. The bars represent the system performance degradation (from DRAM) for the corresponding benchmark listed at the X-axis. The different bars represent different STT-MRAM configurations.

The results show STT-MRAM produces a negligible performance impact for all the benchmarks. For the *ST-1.2* configuration, slowdown ranges from 0% (`matrix01`) to 0.04% (`obdp`). *ST-1.5* introduces slowdown ranging from 0% (`basefp01`) to 0.22% (`obdp`). *ST-2.0* shows a similar trend of low impact to the overall system performance. In the worst case, the system performance degradation is 0.37% (`obdp`).

To investigate more in this regard, we execute four applications with high memory utilization [109] from the Mediabench benchmark suite. Figure 5.5 shows benchmarks with high memory utilization pay a higher performance penalty, although not very significant. For the *ST-1.2* configuration, slowdown ranges from 0.44% (`mesa.texgen`) to 1.03% (`mesa.osdemo`). *ST-1.5* introduces slowdown ranging from 1.57% (`mesa.mipmap`) to 3.39% (`mesa.osdemo`). In the worst case, the performance degradation is 5.62% (`mesa.osdemo`) for the *ST-2.0* configuration.

Within the scope of the performance analysis, we go a step further and analyze the performance degradation due to the modifications required to introduce *Randomization* to the target platform (See Section 5.2.4). The results show that *Randomization* does not introduce any significant performance overhead to the system (0.061% deviation is the worst case — `mesa.osdemo` with *ST-2.0* configuration).

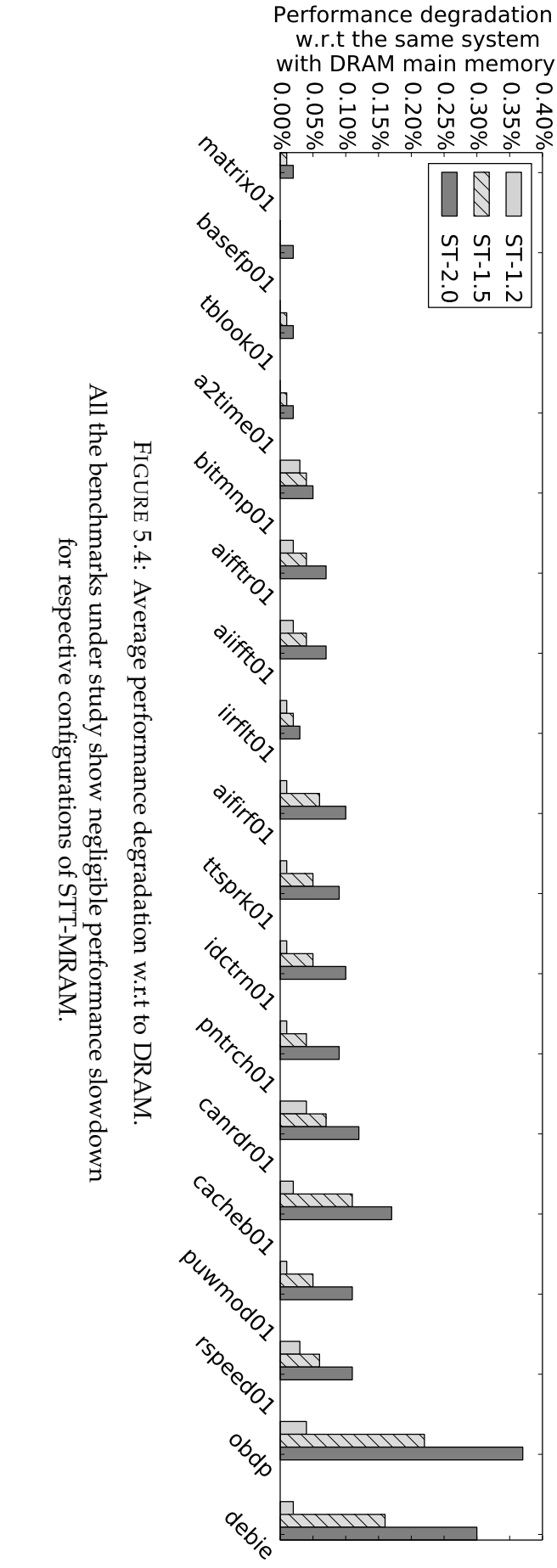


FIGURE 5.4: Average performance degradation w.r.t to DRAM.
All the benchmarks under study show negligible performance slowdown
for respective configurations of STT-MRAM.

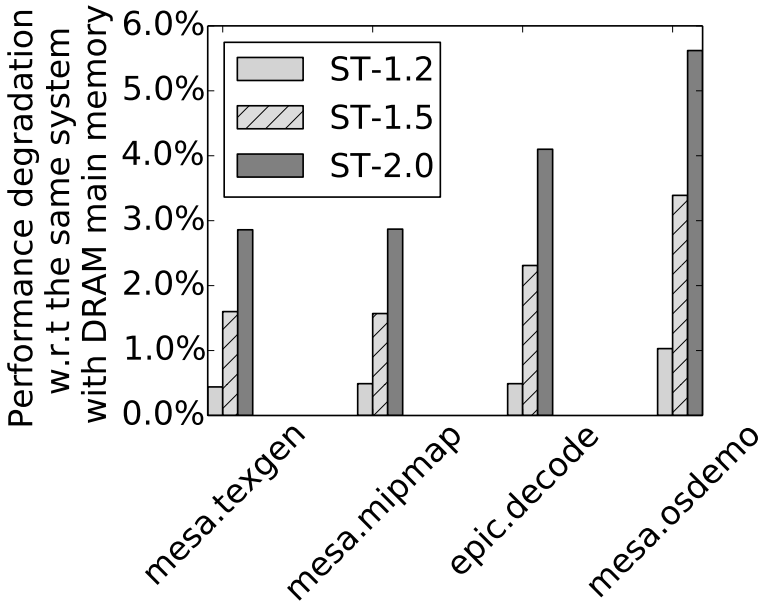


FIGURE 5.5: Performance degradation for applications with high memory utilization

The results suggest that, in the aspect of performance, STT-MRAM can be a good contender for aerospace and automotive applications.

### 5.3.2 Worst case execution time (WCET) analysis

In this section we present and compare WCET estimates between the conventional DRAM and STT-MRAM memory systems. In particular, we use the measurement-based probabilistic timing analysis (MBPTA) described in Section 5.2.4 to compute probabilistic WCET (pWCET) estimates for each benchmark and system configuration.

Figure 5.6 illustrates the pWCET distribution for the `cacheb01` benchmark executed in the system with the DRAM memory. The figure shows the probability (Y-axis) that the execution time in any `cacheb01` run exceeds the corresponding pWCET on the X-axis. The solid line corresponds to the pWCET point estimate, while the thin (blue) lines correspond to the 95% confidence interval. For example, the chart shows that, for the exceedance probability of $10^{-12}$, the pWCET ranges between $19.3e^6$ and $20.5e^6$ cycles (95% confidence interval) while the point estimate is $19.8e^6$ cycles. The dotted (red) line represents actually measured execution times.

We estimate the pWCET distribution for 80 scenarios: two ESA, fourteen

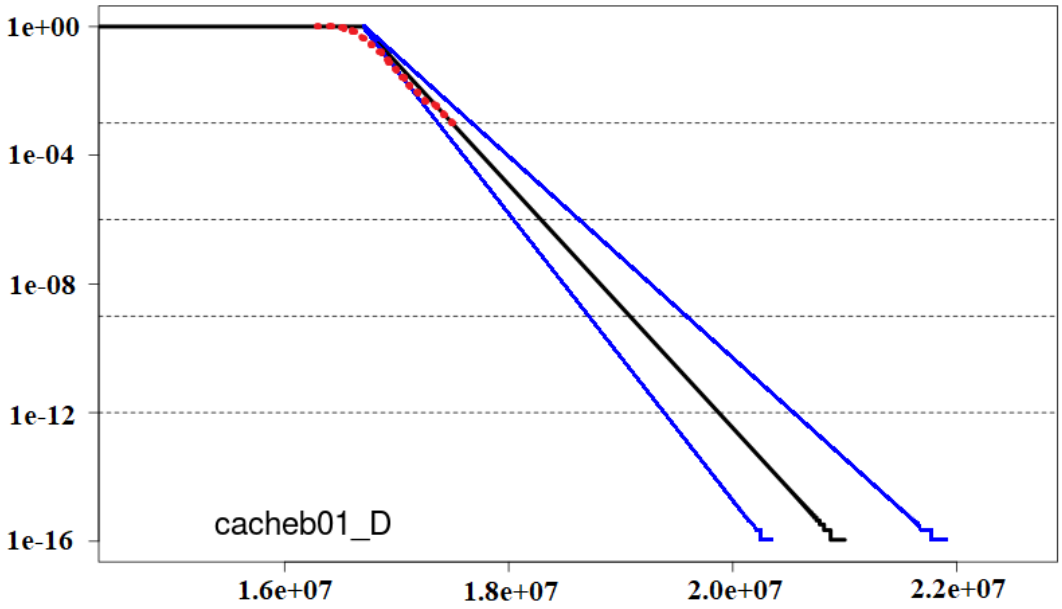


FIGURE 5.6: pWCET distribution for `cacheb01` with DRAM.

One of the scenarios under experiment: EEMBC benchmark `cacheb01` executed with DRAM main memory. Exceedance probability (Y-axis) of the corresponding execution time (X-axis), at any given run.

EEMBC [3], and four Mediabench benchmarks with four memory timing parameters: *DRAM*, *ST-1.2*, *ST-1.5*, and *ST-2.0*.

Since it would be a tedious task to plot and compare individual pWCET distribution charts for the 80 scenarios, we represent the results with a summarized appearance. In Figures 5.7 and 5.8 we plot the pWCET (Y-axis) for five different exceedance probabilities, $10^{-3}$, $10^{-6}$, $10^{-9}$, $10^{-12}$, $10^{-15}$ (Y-axis). The solid bars show the pWCET point estimate, while the error bars denote 95% confidence interval, as explained in Figure 5.6 . In order to increase the visibility of the results, the benchmarks are distributed among the figures based on their pWCET.

From the charts in Figures 5.7 and 5.8, we can see that for all benchmarks under study, pWCET slightly increases for decreasing exceedance probability, as expected. As shown in Figure 5.8, `mesa.texgen` has the widest confidence interval in relative terms across benchmarks. In this case, the confidence interval for DRAM at an exceedance probability of $10^{-15}$ is 98.6%-101.7%, normalized w.r.t. the point estimation for DRAM. Using the same

[3] We exclude `bitmnp01` and `matrix01` EEMBC benchmarks for WCET estimation because they did not fulfil the requirements of MBPTA statistical analysis (See Section 5.2.5).

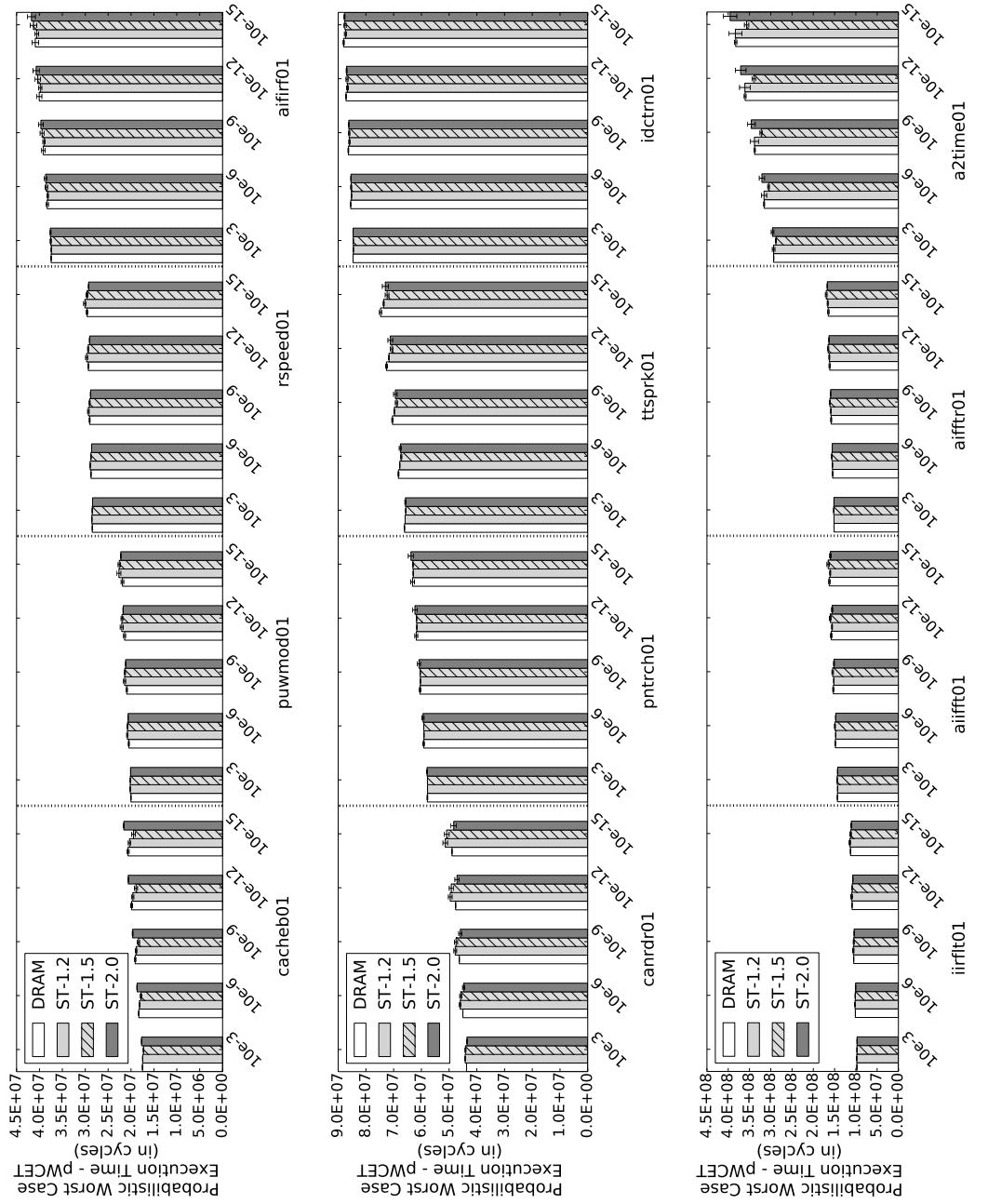


FIGURE 5.7: Probabilistic Worst Case Execution Time (pWCET)

For benchmarks `cacheb01`, `puwmod01`, `rspeed01`, `aifirf01` (Top); `canrdr01`, `pntrch01`, `ttsprk01`, `idctrn01` (Middle); and `iirflt01`, `aiifft01`, `aifftr01`, `a2time01` (Bottom). X-axis lists Benchmarks & Exceedance probability.

reference, ST-2.0, the most pessimistic scenario, has a confidence interval of 94.9%-101.1%, thus overlapping with the confidence interval for DRAM, which allows claiming that no configuration can be proven superior to the other. In general, we observe the very same trend for all benchmarks, with overlapping confidence intervals between DRAM and all STT-MRAM configurations.

Most of the benchmarks show insignificant deviation in pWCET estimates for DRAM and STT-MRAM configurations. Some benchmarks (e.g. `cacheb01`, `canrdr01`, `a2time01`, `basefp01` etc.) show minor but visible fluctuations in pWCET estimations for different memory configurations. For example, in a few cases STT-MRAM configurations offer better results than expected (i.e. faster than DRAM and/or faster STT-MRAM configurations). This relates to the intrinsic pessimism of EVT to fit pWCET curves. Eventually, high values in the random samples may fit in a slightly narrower value range due to pure random reasons, which allows EVT to find slightly tighter pWCET estimates. As we decrease the exceedance probability (e.g. down to $10^{-15}$), discrepancies naturally amplify, but they are still within few percent points w.r.t. the reference DRAM setup.

There are two main observations from the WCET estimation results. First, The results confirm that the measurement-based probabilistic timing analysis (MBPTA) described in Section 5.2.4 can be applied for the system comprising STT-MRAM main memory. The effort for the pWCET analysis, including the benchmark runs and the statistical analysis, does not change from the DRAM to the STT-MRAM main memory. Each benchmark converged to produce a valid WCET estimate approximately with a thousand runs. This is fundamental to reduce STT-MRAM adoption costs, without requiring new tools that must undergo a costly qualification process [9]. Second, the results show that the pWCET estimates have a very narrow confidence interval, and that there is a negligible difference between WCET estimations with DRAM and STT-MRAM systems.

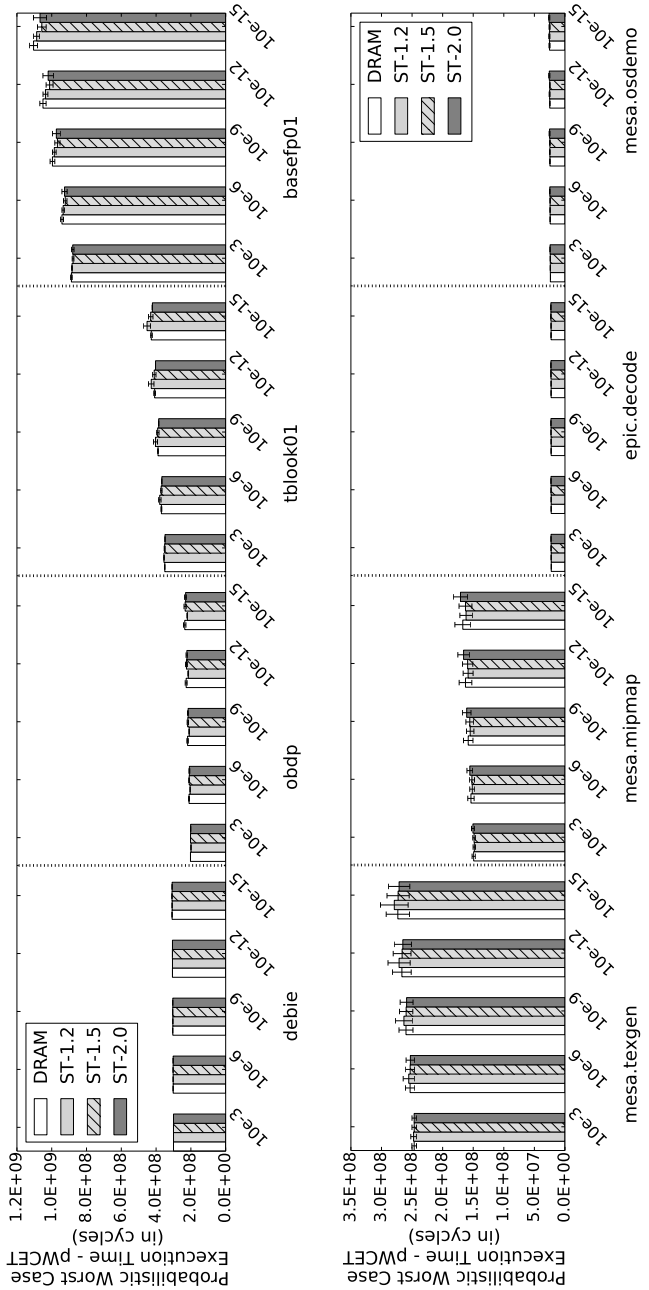


FIGURE 5.8: Probabilistic Worst Case Execution Time (pWCET)

For benchmarks `debie`, `obdp`, `tblook01`, `basefp01`. (Top); and `mesa.texgen`, `mesa.mipmap`, `epic.decode`, `mesa.osdemo`. (Bottom). X-axis lists Benchmarks & Exceedance probability.

## 5.4 Summary

In the final part of the thesis, we extend our experiments into real-time embedded systems, a domain that has great potential for the STT-MRAM due to its intrinsic radiation hardness, non-volatility, zero stand-by power and extended temperature tolerance. We investigate the feasibility of using STT-MRAM in real-time systems by analyzing system performance impact and worst case execution time (WCET) implications. The results suggest that, in the aspect of performance, STT-MRAM can be a good contender for aerospace and automotive applications. For WCET analysis, the results confirm that MBPTA can be applied for the system comprising STT-MRAM main memory. The effort for the WCET analysis, including the benchmark runs and the statistical analysis, does not change from the DRAM to the STT-MRAM main memory. This is fundamental to reduce STT-MRAM adoption costs, without requiring new tools that must undergo a costly qualification process [9]. The results also show that the WCET estimates have a very narrow confidence interval, and that there is negligible difference between WCET estimates with DRAM and STT-MRAM systems.

# Chapter 6

# Related Works

In this chapter we discuss the related studies that investigated STT-MRAM as main memory or cache memory alternative.

## 6.1 STT-MRAM as main memory

Most of the STT-MRAM system-level research so far focused on the suitability of this technology for on-chip cache memories. Only a few studies analyze STT-MRAM as the main memory.

Meza *et al.* [50] 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 the cost of some performance degradation. Kultursay *et al.* [51] evaluate STT-MRAM as a main memory for SPEC CPU2006 workloads and show that, without any optimizations, early-design STT-MRAM [110] 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.* [52] analyze the 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.

In all studies that target the HPC and server domains, DRAM and various STT-MRAM main memory designs are evaluated by using *average* read and write latencies. This approach fails to account for the highly complex behavior of modern memory systems and may under-report their effect on the overall system performance [90][4].

Jiang *et al.* [111] propose using STT-MRAM main memory in mobile devices. The main objective of their study is to save the energy of the DRAM refresh, by using the non-volatile memory technology. The authors also propose two STT-MRAM microarchitectual enhancements that would improve the STT-MRAM performance in the presence of the read disturbance errors. The proposal is evaluated based on the STT-MRAM parameters targeting LPDDR devices estimated by Wang *et al.* [57] using CACTI [58] cache simulator and NVSim [112].

Simone *et al.* [113] advocate to exploit the unlimited endurance of STT-MRAM as small-capacity rad-hard memories due to their inherent resistance to radiation.

Manu *et al.* [114] use realistic, calibrated STT-MRAM models with a general purpose multicore architecture. In their experiments, the most energy-efficient STT-MRAM main memory saves an average of 27% energy at the cost of $2x$ of area, and the least energy-efficient STT-MRAM main memory saves an average of 8% energy using comparable area with DRAM.

Guo *et al.* [115] propose a new approach to protect STT-MRAM main memory against retention errors. The authors introduce an architecture named *Sanitizer*, which mitigates the performance and energy overheads of ECC and scrubbing in future STT-MRAM based main memories.

Armin *et al.* [116] propose a hybrid row-buffer management scheme that monitors row buffer hit rates at run-time and optimize the policy selection for STT-MRAM main memory. The authors claim to achieve significant improvement in system performance and energy efficiency.

Lei *et al.* [117] present architectural techniques to improve read performance for STT-MRAM main memories. The authors propose *Smash Read* to shorten

read latency when the destructive high current restore required (HCRR) is adopted and *Flexible Read* to dynamically switch between *Smash Reads* and non-destructive low current long latency (LCLL) read.

## 6.2 STT-MRAM as on-chip caches

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

Li *et al.* [118] 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 a 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.* [119] observe that many bits in the STT-MRAM cache are rewritten 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.* [120] compare 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 that, 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 [121][122][123][124]. Li *et al.* [122] indicate that the majority of cache data stay active for a 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.* [123] 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.* [121] propose a ultra-low retention time STT-MRAM caches supported by a DRAM-like refresh policy. Sun *et al.* [124] further exploit the scenario by deploying STT-MRAM with multiple retention levels. Smullen *et al.* [121] and Sun *et al.* [124] 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 [58]. 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.

# Chapter 7

# Future work

This thesis provides a major breakthrough in STT-MRAM main memory research by publishing the detailed timing parameters for the first time and evaluating its performance aspects across various computing platforms with a range of workloads to understand its feasibility and effectiveness as a potential future memory system. However, there is still a lot more to explore in this research direction. In this section, we discuss potential research paths that can be explored in the future in continuation to this thesis.

## 7.1 Exploiting non-volatility

In this thesis, we evaluated STT-MRAM as a clean replacement to conventional DRAM assuming that any new main memory technology has to adopt into the DDR$x$ protocol to be easily incorporated into the existing systems. Meaning STT-MRAM is operated using DRAM protocols and therefore its non-volatility is not being used as a feature. Since, DRAM has been dominating the main memory landscape for a long time, all applications, operating system and other modules are adopted to the idea that main memory is volatile. Investigating what advantages can be drawn by exploiting the non-volatility of STT-MRAM main memory would be a compelling area of research.

## 7.2 Check-pointing

HPC applications consist of thousands of tightly-coupled processes, so failure of a single process results in a global failure that terminates the whole application. Therefore, system reliability is a first-class requirement, mainly because of a costly re-execution of long-running HPC jobs. In large HPC systems, fault tolerance is usually provided by the *checkpoint-restart* approach. Checkpointing periodically saves the system recovery information into a persistent storage. In case that a failure occurs, instead of repeating the whole experiment, the recovery information is used to *restart* the application from the last checkpoint [125]. Checkpoint-restart comes at an additional cost because it interrupts the normal execution of the applications, and puts significant stress to the interconnect network and the IO storage. In current HPC systems, between 15% and 45% of the time is spent on checkpointing, restarting and partial re-computation of the work lost since the last checkpoint [126][127]. This overhead will increase with the size of HPC systems, and it is estimated that on a 100,000-node cluster, checkpoint-restart activities will require 65% of the overall machine time [128]. One of the main sources of checkpoint-restart overheads is moving large amounts of data through the network to the remote storage [129]. In HPC systems with the STT-MRAM main memory, the memory system itself can be used to permanently store system recovery information. In order to prevent the system crashes in case that the whole server fails (e.g. due to power supply failure), system recovery information could be stored in main memory of the near-by servers. This would remove the pressure from the interconnect network and storage system and therefore would significantly reduce the checkpointing overhead. Low-overhead checkpoint-restart would also enable advanced techniques in batch scheduling, because it would relax the requirement that the jobs are executed continuously and without interruption, from the beginning until the end [130]. In order to improve the system utilization, the scheduler could perform a checkpoint of a given job, put it on hold and then restart it at an appropriate moment. This could, for example, allow system shutdown without queue draining period, or running large jobs in off-peak hours without the need to limit their execution time.

An extensive research in this field could hugely benefit high performance computing domain.

# Chapter 8

# Conclusion

STT-MRAM is an emerging non-volatile memory with a lot of potential that could be exploited for various requirements of different computing systems. Being a novel technology, STT-MRAM devices are already approaching DRAM in terms of capacity, frequency and device size. Intensified efforts in STT-MRAM research by the memory manufacturers may indicate that a revolution with STT-MRAM memory technology is imminent, and therefore, it is now the time to explore computing domains that can benefit from this technology.

In this thesis, we evaluate different aspects of of using STT-MRAM as the main memory across various computing platforms with a range of workloads to understand its feasibility and effectiveness as a potential future memory system.

In the first set of experiments, we conduct a preliminary analysis on whether STT-MRAM is a candidate for future HPC memory systems. We model STT-MRAM main memory latency using industry estimation and incorporate it into the overall simulation of the HPC system executing production applications. Results suggest, being 20% slower at the device level, STT-MRAM yields an average of only 0.8% system performance slowdown for production HPC workloads. Therefore, STT-MRAM provides performance comparable to conventional systems, while opening up various opportunities for HPC system improvements.

STT-MRAM main memory got significant attention of various major memory manufacturers, and is expected to bring a revolution in the memory market. However, academic research on this technology is still marginal, and

academia is struggling to conduct a reliable STT-MRAM main memory simulation. In order to overcome this problem, in the second phase of the thesis, we thoroughly analyze and publish detailed STT-MRAM main memory timing parameters enabling a reliable system level simulation of this technology. The study is based on the fact that STT-MRAM main memory devices will be incorporated into the DDR*x* interface and protocol, indicating that most of the timings will not change from DRAM to STT-MRAM main memory. For the parameters that will change due to differences in DRAM and STT-MRAM storage cells, we have to accept that there is no reliable information on how these timing parameters will change for the upcoming STT-MRAM devices. Therefore, we strongly argue that the best we can do at this point is a sensitivity analysis on these parameters. The approach that we present converged through research cooperation with Everspin technologies Inc., and it provides reliable STT-MRAM timing parameters while releasing no confidential information about any commercial products.

We apply similar methodology to estimate current parameters of STT-MRAM. We identify four current parameters that would change for STT-MRAM w.r.t DRAM and we perform a sensitivity analysis on these parameters to achieve an estimation of STT-MRAM power consumption.

We also seamlessly incorporate STT-MRAM timing and current parameters into the DRAMSim2 memory simulator and use it as a part of the simulation infrastructure of the high performance computing systems. The results of our simulations show that, for the most realistic (considering the ongoing development) configuration *ST-1.2*, the SPEC 2006 benchmarks suffers an average system slowdown of less than 3%. Results from the power estimation indicates that STT-MRAM power consumption increases significantly for *Activation and Pre-charge power* while *Burst Power* increases moderately and *Background Power* does not deviate much from DRAM.

An intensified effort of memory manufacturers in STT-MRAM research promises exciting developments on this technology in the near future. Now, with the reliable detailed timing parameters that we publish, we would strongly encourage academia to also explore the opportunities that this technology has to offer.

Special STT-MRAM features such as intrinsic radiation hardness, non-volatility, zero stand-by power and capability to function in extreme temperatures offer a great opportunity to explore its usability in real-time embedded

systems, particularly in the space, avionics and automotive domains.

In the final part of the thesis, we investigate the feasibility of using STT-MRAM in these domains by analyzing the system performance impact and worst case execution time (WCET) implications. In our opinion, it is of vital importance to perform a head-to-head comparison of a new technology to the conventional one on the same platform without proposing ambitious optimizations, because such proposals may actually obscure the critical information where the technology stands as-is, or how far it is from being used as a standard replacement of the conventional one.

The results suggest that, in the aspect of performance, STT-MRAM can be a good contender for aerospace and automotive applications. There is almost no system performance slowdown for the *ST-1.2* configuration (below 0.04%). Even for the *ST-2.0* configuration, the worst slowdown is reported to be below 0.4%.

For WCET analysis, the results confirm that MBPTA can be applied for systems comprising STT-MRAM main memory. The effort for the WCET analysis, including the benchmark runs and the statistical analysis, does not change from the DRAM to the STT-MRAM main memory. This is fundamental to reduce STT-MRAM adoption costs, without requiring new tools that must undergo a costly qualification process [9]. The results also show that the WCET estimates have a very narrow confidence interval, and that there is negligible difference between WCET estimates for DRAM and STT-MRAM systems.

Overall, this study presents the first comprehensive exploration of possibilities to use STT-MRAM in the real-time embedded domain and reveals that STT-MRAM would provide performance and WCET estimates comparable to DRAM while opening up several key advantages that the domain could benefit from.

Finally, STT-MRAM's adoption as an 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.

# Bibliography


[1] M. Hosomi et al. “A Novel Nonvolatile Memory with Spin Torque Transfer Magnetization Switching: Spin-RAM”. In: *IEEE International Electron Devices Meeting*. 2005.

[2] Everspin Technologies, Inc. *STT-MRAM Products*. https://www.everspin.com/stt-mram-products. 2018.

[3] P. Rosenfeld, E. Cooper-Balis, and B. Jacob. “DRAMSim2: A Cycle Accurate Memory System Simulator”. In: *IEEE Computer Architecture Letters* (2011).

[4] David Wang et al. “DRAMsim: A Memory System Simulator”. In: *SIGARCH Comput. Archit. News* 33.4 (2005).

[5] Yoongu Kim, Weikun Yang, and Onur Mutlu. “Ramulator: A Fast and Extensible DRAM Simulator”. In: *IEEE Comput. Archit. Lett.* (2016).

[6] European Space Agency. *GR740: The ESA Next Generation Microprocessor (NGMP)*. http://microelectronics.esa.int/gr740/index.html.

[7] J. A. Poovey et al. “A Benchmark Characterization of the EEMBC Benchmark Suite”. In: *IEEE Micro* (2009).

[8] Chunho Lee, Miodrag Potkonjak, and William H. Mangione-Smith. “MediaBench: A Tool for Evaluating and Synthesizing Multimedia and Communicatons Systems”. In: *Proceedings of the 30th Annual ACM/IEEE International Symposium on Microarchitecture*. MICRO 30. 1997.

[9] International Organization for Standardization. *ISO/DIS 26262. Road Vehicles – Functional Safety*. 2009.

[10] Rajinder Gill. *Everything You Always Wanted to Know About SDRAM (Memory): But Were Afraid to Ask*. https://www.anandtech.com/show/ 3851/everything-you-always-wanted-to-know-about-sdram-memory -but-were-afraid-to-ask. 2015.

[11] Wm Wulf and Sally A. McKee. *Hitting the Memory Wall: Implications of the Obvious*. Tech. rep. Charlottesville, VA, USA, 1994.

[12] Milan Radulovic et al. "Another Trip to the Wall: How Much Will Stacked DRAM Benefit HPC?" In: *Proceedings of the 2015 International Symposium on Memory Systems*. 2015.

[13] R. Hadidi et al. "Demystifying the characteristics of 3D-stacked memories: A case study for Hybrid Memory Cube". In: *2017 IEEE International Symposium on Workload Characterization (IISWC)*. 2017.

[14] Jialiang Zhang, Soroosh Khoram, and Jing Li. "Boosting the Performance of FPGA-based Graph Processor Using Hybrid Memory Cube: A Case for Breadth First Search". In: *Proceedings of the 2017 ACM/SIGDA International Symposium on Field-Programmable Gate Arrays*. FPGA '17. 2017. ISBN: 978-1-4503-4354-1.

[15] H. Jun et al. "HBM (High Bandwidth Memory) DRAM Technology and Architecture". In: *2017 IEEE International Memory Workshop (IMW)*. 2017.

[16] H. Horii et al. "A novel cell technology using N-doped GeSbTe films for phase change RAM". In: *2003 Symposium on VLSI Technology. Digest of Technical Papers (IEEE Cat. No.03CH37407)*. 2003.

[17] I. G. Thakkar and S. Pasricha. "DyPhase: A Dynamic Phase Change Memory Architecture With Symmetric Write Latency and Restorable Endurance". In: *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* (2018).

[18] F. Bedeschi et al. "A Multi-Level-Cell Bipolar-Selected Phase-Change Memory". In: *2008 IEEE International Solid-State Circuits Conference - Digest of Technical Papers*. 2008.

[19] K. Son et al. "Modeling and Signal Integrity Analysis of 3D XPoint Memory Cells and Interconnections with Memory Size Variations During Read Operation". In: *2018 IEEE Symposium on Electromagnetic Compatibility, Signal Integrity and Power Integrity (EMC, SI PI)*. 2018.

[20] P. Chen and S. Yu. "Compact Modeling of RRAM Devices and Its Applications in 1T1R and 1S1R Array Design". In: *IEEE Transactions on Electron Devices* (2015).

[21] H. . P. Wong et al. "Metal–Oxide RRAM". In: *Proceedings of the IEEE* (2012).

[22] T. F. Wu et al. "Hyperdimensional Computing Exploiting Carbon Nanotube FETs, Resistive RAM, and Their Monolithic 3D Integration". In: *IEEE Journal of Solid-State Circuits* (2018).

[23] B. Dieny et al. "Giant magnetoresistive in soft ferromagnetic multilayers". In: *Phys. Rev. B* (1991).

[24] J.K. Spong et al. "Giant Magnetoresistive Spin Valve Bridge Sensor". In: *IEEE Transactions on Magnetics* (1996).

[25] J. A. Katine et al. "Current-Driven Magnetization Reversal and Spin-Wave Excitations in Co /Cu /Co Pillars". In: *Phys. Rev. Lett.* (2000).

[26] Yuan Xie. "Modeling, Architecture, and Applications for Emerging Memory Technologies". In: *IEEE Design Test of Computers* (2011).

[27] S. Ikeda et al. "A perpendicular-anisotropy CoFeB–MgO magnetic tunnel junction". In: *Nature Materials*. Vol. 9. 2010, pp. 721–724.

[28] J. J. Nowak et al. "Dependence of Voltage and Size on Write Error Rates in Spin-Transfer Torque Magnetic Random-Access Memory". In: *IEEE Magnetics Letters* 7 (2016), pp. 1–4.

[29] K. Abe et al. "Novel Hybrid DRAM/MRAM Design for Reducing Power of High Performance Mobile CPU". In: *IEEE International Electron Devices Meeting (IEDM)*. 2012.

[30] H. Noguchi et al. "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.

[31] R. Nebashi et al. "A 90nm 12ns 32Mb 2T1MTJ MRAM". In: *IEEE International Solid-State Circuits Conference*. 2009.

[32] K. Rho et al. "23.5 A 4Gb LPDDR2 STT-MRAM with compact 9F2 1T1MTJ cell and hierarchical bitline architecture". In: *2017 IEEE International Solid-State Circuits Conference (ISSCC)*. 2017.

[33] Everspin Technologies, Inc. *Everspin displays both the 1Gb DDR4 Perpendicular ST-MRAM device and a 1GByte DDR3 Memory Module (DIMM) at Stand A3-545*. https://www.everspin.com/news/everspin-previews-upcoming-products-electronica. 2016.

[34] Dong Li et al. “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.

[35] A.M. Caulfield et al. “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.

[36] Jeffrey S. Vetter and Sparsh Mittal. “Opportunities for Nonvolatile Memory Systems in Extreme-Scale High Performance Computing”. In: *Computing in Science and Engineering special issue* (2015). ISSN: 1521-9615.

[37] Ishwar Bhati et al. “DRAM Refresh Mechanisms, Penalties, and Trade-Offs”. In: *IEEE Transactions on Computers*. 2015.

[38] Jamie Liu et al. “RAIDR: Retention-Aware Intelligent DRAM Refresh”. In: *39th Annual International Symposium on Computer Architecture*. 2012.

[39] 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.

[40] 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*. 2012.

[41] 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.

[42] Vilas Sridharan et al. “Feng Shui of Supercomputer Memory: Positional Effects in DRAM and SRAM Faults”. In: *International Conference on High Performance Computing, Networking, Storage and Analysis*. 2013.

[43] Yoongu Kim et al. “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.

[44] C. Augustine et al. "Numerical Analysis of Typical STT-MTJ Stacks for 1T-1R Memory Arrays". In: *IEEE International Electron Devices Meeting (IEDM)*. 2010.

[45] Zoha Pajouhi, Xuanyao Fong, and Kaushik Roy. "Device/Circuit/Architecture Co-design of Reliable STT-MRAM". In: *Proceedings of the Design, Automation & Test in Europe Conference & Exhibition*. 2015.

[46] D. Chabi et al. "Design and Analysis of Radiation Hardened Sensing Circuits for Spin Transfer Torque Magnetic Memory and Logic". In: *IEEE Transactions on Nuclear Science* (2014). DOI: 10.1109/TNS.2014.2370735.

[47] M. Riera et al. "A detailed methodology to compute Soft Error Rates in advanced technologies". In: *2016 Design, Automation Test in Europe Conference Exhibition (DATE)*. 2016, pp. 217–222.

[48] Everspin Technologies, Inc. *Case Study: SpriteSat (Rising) Satellite*. https: //www.everspin.com/aerospace. 2018.

[49] Everspin Technologies, Inc. *Automotive*.

[50] Jing Li Justin Meza and Onur Mutlu. "Evaluating Row Buffer Locality in Future Non-Volatile Main Memories". In: *Safari Technical Report No. 2012-002* (2012).

[51] E. Kultursay et al. "Evaluating STT-RAM as an Energy-Efficient Main Memory Alternative". In: *IEEE International Symposium on Performance Analysis of Systems and Software*. 2013.

[52] 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.

[53] D. Halupka et al. "Negative-Resistance Read and Write Schemes for STT-MRAM in 0.13um CMOS". In: *IEEE International Solid State Circuits Conference*. 2010.

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

[55] Iternational Technology Roadmap for Semiconductors. *ITRS 2013 Edition*. http://www.itrs.net/reports.html.

[56] M. Poremba and Y. Xie. "NVMain: An Architectural-Level Main Memory Simulator for Emerging Non-volatile Memories". In: *2012 IEEE Computer Society Annual Symposium on VLSI*. 2012.

[57] Jue Wang, Xiangyu Dong, and Yuan Xie. "Enabling High-performance LPDDRx-compatible MRAM". In: ISLPED. 2014.

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

[59] H. Kim et al. *Magneto-resistive memory device including source line voltage generator*. 2013.

[60] H.R. Oh. *Resistive Memory Device, System Including the Same and Method of Reading Data in the Same*. 2014.

[61] C. Kim et al. *Magnetic Random Access Memory*. 2013.

[62] *Unified European Applications Benchmark Suite*. Partnership for Advanced Computing in Europe (PRACE). 2013.

[63] Alejandro Rico et al. "On the Simulation of Large-scale Architectures Using Multiple Application Abstraction Levels". In: *ACM Trans. Archit. Code Optim.* (2012).

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

[65] R. Preissl et al. "Detecting Patterns in MPI Communication Traces". In: *37th International Conference on Parallel Processing*. 2008.

[66] 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.

[67] Milan Pavlovic et al. "Limpio — LIghtweight MPI instrumentatiOn". In: *IEEE 23rd International Conference on Program Comprehension*. 2015. URL: http://www.bsc.es/computer-sciences/computer-architecture/memory-systems/limpio.

[68] Barcelona Supercomputing Center. *Paraver*. http://www.bsc.es/computer-sciences/performance-tools/paraver.

[69] *Valgrind*. http://valgrind.org/.

[70] Thomas Roberts Puzak. "Analysis of Cache Replacement Algorithms". PhD thesis. 1985.

[71] Wen-Hann Wang and Jean-Loup Baer. "Efficient Trace-driven Simulation Methods for Cache Performance Analysis". In: *ACM Trans. Comput. Syst.* (1991).

[72] Richard A. Uhlig and Trevor N. Mudge. "Trace-driven Memory Simulation: A Survey". In: *ACM Comput. Surv.* (1997).

[73] 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*. 2013.

[74] Top500. *Top500 Supercomuter Sites*. http://www.top500.org/.

[75] 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.

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

[77] B. Jacob. *The Memory System: You Can't Avoid It; You Can't Ignore It; You Can't Fake It. M*. Reading, Massachusetts: Morgan & Claypool Publishers, 2009. ISBN: 978-1598295870.

[78] Janani Mukundan, Alexandre Ferreira, Karthick Rajamani, Jente Kuang, Kyu-Hyoun Kim, Hillery Hunter, Luis Lastras, Xiaochen Guo. *Design of High-performance, Resilient, STT-MRAM-based Main Memory*. https:// domino.research.ibm.com/library/cyberdig.nsf/papers/83C96B8F98D488A9852582B20043C181. 2018.

[79] Samsung Electronics Co., Ltd. *240pin Registered DIMM based on 2Gb D-die*. https://www.samsung.com/semiconductor/global.semi/file/resource/2017/11/ ds_ddr3_2gb_d-die_based_1_35v _rdimm_rev12-2.pdf. 2011.

[80] John L. Henning. "SPEC CPU2006 Benchmark Descriptions". In: *SIGARCH Comput. Archit. News* (2006).

[81] Daniel Sanchez and Christos Kozyrakis. "ZSim: Fast and Accurate Microarchitectural Simulation of Thousand-core Systems". In: *Proceedings of the 40th Annual International Symposium on Computer Architecture*. ISCA. 2013.

[82] Intel Corporation. *Intel 64 and IA-32 Architectures Software Developer Manuals*. 2017.

[83] Clémentine Maurice et al. "Reverse Engineering Intel Last-Level Cache Complex Addressing Using Performance Counters". In: *Research in Attacks, Intrusions, and Defenses: 18th International Symposium, RAID, Kyoto, Japan. Proceedings*. Ed. by Herbert Bos, Fabian Monrose, and Gregory Blanc. 2015.

[84] Kazi Asifuzzaman et al. "Performance Impact of a Slower Main Memory: A Case Study of STT-MRAM in HPC". In: *Proceedings of the Second International Symposium on Memory Systems*. (MEMSYS). 2016.

[85] Larry McVoy and Carl Staelin. "Lmbench: Portable Tools for Performance Analysis". In: *Proceedings of the Annual Conference on USENIX Annual Technical Conference*. (ATEC). 1996.

[86] Kazi Asifuzzaman, Rommel Sánchez Verdejo, and Petar Radojković. "Enabling a Reliable STT-MRAM Main Memory Simulation". In: *Proceedings of the International Symposium on Memory Systems*. MEMSYS '17. Alexandria, Virginia, 2017, pp. 283–292. ISBN: 978-1-4503-5335-9. URL: http://doi.acm.org/10.1145/3132402.3132416.

[87] SoCLib. -. http://www.soclib.fr/trac/dev. 2003-2012.

[88] L. Fossati M. Zulianello F. J. Cazorla J. Jalle J. Abella. "Validating a Timing Simulator for the NGMP Multicore Processor". In: *2016 DASIA*. ISBN: 789292213015.

[89] Cobham Gaisler. *GR-CPCI-LEON4-N2X Quad-Core LEON4 Next Generation Microprocessor Evaluation Board*. http://www.gaisler.com/index.php/products/boards/gr-cpci-leon4-n2x.

[90] P. Rosenfeld, E. Cooper-Balis, and B. Jacob. "DRAMSim2: A Cycle Accurate Memory System Simulator". In: *IEEE Computer Architecture Letters* (2011).

[91] Micron Technology, Inc. *Automotive DDR2 SDRAM*. 2011.

[92] J. Abella et al. "WCET analysis methods: Pitfalls and challenges on their trustworthiness". In: *10th IEEE International Symposium on Industrial Embedded Systems (SIES)*. 2015, pp. 1–10. DOI: 10.1109/SIES.2015.7185039.

[93] Reinhard Wilhelm. et al. "The worst-case execution time problem: overview of methods and survey of tools". In: *ACM TECS* 7.3 (2008), pp. 1–53.

[94] I. Wenzel et al. "Measurement-Based Timing Analysis". In: *ISOLA*. 2008.

[95] I. Wenzel et al. "Measurement-Based Worst-Case Execution Time Analysis". In: *SEUS Workshop*. 2005.

[96] S. Law and I. Bate. "Achieving Appropriate Test Coverage for Reliable Measurement-Based Timing Analysis". In: *ECRTS*. 2016.

[97] Jaume Abella et al. "Measurement-Based Worst-Case Execution Time Estimation Using the Coefficient of Variation". In: *ACM Trans. Des. Autom. Electron. Syst.* 22.4 (June 2017), 72:1–72:29. ISSN: 1084-4309. DOI: 10.1145/3065924. URL: http://doi.acm.org/10.1145/3065924.

[98] F. Wartel et al. "Timing Analysis of an Avionics Case Study on Complex Hardware/Software Platforms". In: *DATE*. 2015.

[99] M. Fernandez et al. "Probabilistic Timing Analysis on Time-randomized Platforms for the Space Domain". In: *DATE*. 2017.

[100] Z. Stephenson et al. "Supporting Industrial Use of Probabilistic Timing Analysis with Explicit Argumentation". In: *INDIN*. 2013.

[101] Cobham Gaisler. *LEON3 Processor (Probabilistic platform)*. http://www.gaisler.com/index.php/products/processors/leon3.

[102] L. Kosmidis et al. "Probabilistic Timing Analysis on Conventional Cache Designs". In: *DATE*. 2013.

[103] L. Kosmidis et al. "TASA: Toolchain-agnostic Static Software Randomisation for Critical Real-time Systems". In: *ICCAD*. 2016.

[104] Jaume Abella. *MBPTA-CV*. Nov. 2017. DOI: 10.5281/zenodo.1065776. URL: https://doi.org/10.5281/zenodo.1065776.

[105] Leonidas Kosmidis et al. "Fitting processor architectures for measurement-based probabilistic timing analysis". In: *Microprocessors and Microsystems* 47 (2016), pp. 287 –302. ISSN: 0141-9331. DOI: https://doi.org/10.1016/j.micpro.2016.07.014. URL: http://www.sciencedirect.com/science/article/pii/S0141933116300977.

[106] Francisco J. Cazorla et al. "Upper-bounding Program Execution Time with Extreme Value Theory". In: *13th International Workshop on Worst-Case Execution Time Analysis*. Ed. by Claire Maiza. Vol. 30. OpenAccess Series in Informatics (OASIcs). Dagstuhl, Germany: Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, 2013, pp. 64–76. ISBN: 978-3-939897-54-5. DOI: 10.4230/OASIcs.WCET.2013.64. URL: http://drops.dagstuhl.de/opus/volltexte/2013/4123.

[107] S. Kotz et al. *Extreme value distributions: theory and applications*. World Scientific, 2000, p. 185. ISBN: 1860942245, 9781860942242.

[108] RTCA and EUROCAE. *DO-178B / ED-12B, Software Considerations in Airborne Systems and Equipment Certification*. 1992.

[109] B. Bishop, T. P. Kelliher, and M. J. Irwin. "A detailed analysis of MediaBench". In: *1999 IEEE Workshop on Signal Processing Systems. SiPS 99. Design and Implementation (Cat. No.99TH8461)*. 1999.

[110] Guangyu Sun et al. "A Novel Architecture of the 3D Stacked MRAM L2 Cache for CMPs". In: *IEEE 15th International Symposium on High Performance Computer Architecture*. 2009.

[111] Lei Jiang et al. "Improving read performance of STT-MRAM based main memories through Smash Read and Flexible Read". In: *21st Asia and South Pacific Design Automation Conference (ASP-DAC)*. 2016.

[112] X. Dong et al. "NVSim: A Circuit-Level Performance, Energy, and Area Model for Emerging Nonvolatile Memory". In: *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* 31.7 (2012), pp. 994–1007.

[113] S. Gerardin and A. Paccagnella. "Present and Future Non-Volatile Memories for Space". In: *IEEE Transactions on Nuclear Science* (2010).

[114] M. Komalan et al. "Main memory organization trade-offs with DRAM and STT-MRAM options based on gem5-NVMain simulation frameworks". In: *2018 Design, Automation Test in Europe Conference Exhibition (DATE)*. 2018.

[115] X. Guo et al. "Sanitizer: Mitigating the Impact of Expensive ECC Checks on STT-MRAM Based Main Memories". In: *IEEE Transactions on Computers* (2018).

[116] A. H. Aboutalebi and L. Duan. "Work-in-progress: enabling reliable main memory using STT-MRAM via restore-aware memory management". In: *2017 International Conference on Compilers, Architectures and Synthesis For Embedded Systems (CASES)*. 2017.

[117] Lei Jiang et al. "Improving read performance of STT-MRAM based main memories through Smash Read and Flexible Read". In: *2016 21st Asia and South Pacific Design Automation Conference (ASP-DAC)*. 2016.

[118] 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.

[119] Ping Zhou et al. "Energy Reduction for STT-RAM Using Early Write Termination". In: *IEEE/ACM International Conference on Computer-Aided Design - Digest of Technical Papers*. 2009.

[120] 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*. 2013.

[121] C.W. Smullen et al. "Relaxing non-volatility for fast and energy-efficient STT-RAM caches". In: *IEEE 17th International Symposium on High Performance Computer Architecture (HPCA)*. 2011.

[122] Hai Li et al. "Performance, Power, and Reliability Tradeoffs of STT-RAM Cell Subject to Architecture-Level Requirement". In: *IEEE Transactions on Magnetics* (2011).

[123] A. Jog et al. "Cache Revive: Architecting Volatile STT-RAM Caches for Enhanced Performance in CMPs". In: *49th ACM/EDAC/IEEE Design Automation Conference (DAC)*. 2012.

[124] Zhenyu Sun et al. "Multi Retention Level STT-RAM Cache Designs with a Dynamic Refresh Scheme". In: *44th Annual IEEE/ACM International Symposium on Microarchitecture*. 2011.

[125] E. N. (Mootaz) Elnozahy et al. "A Survey of Rollback-recovery Protocols in Message-passing Systems". In: *ACM Comput. Surv.* (2002).

[126] John T. Daly. "ADTSC Nuclear Weapons Highlights: Facilitating High Throughput ASC Calculations". In: *Technical Report LALP-07-041, Los Alamos National Laboratory, Los Alamos, NM, USA* (2007).

[127] John T. Daly, Lori A. Pritchett-Sheats, and Sarah E. Michalak. "Application MTTFE vs. Platform MTTF: A Fresh Perspective on System Reliability and Application Throughput for Computations at Scale". In: *Workshop on Resiliency in High Performance Computing*. 2008.

[128] Kurt Ferreira et al. "Increasing Fault Resiliency in a Message-Passing Environment". In: *Sandia National Laboratories, Tech. Rep.* (2009).

[129] R.A. Oldfield et al. "Modeling the Impact of Checkpoints on Next-Generation Systems". In: *24th IEEE Conference on Mass Storage Systems and Technologies*. 2007.

[130] Paul H Hargrove and Jason C Duell. "Berkeley Lab Checkpoint/Restart (BLCR) for Linux Clusters". In: *Journal of Physics* (2006).