Emerging Non-volatile Memory Technologies Exploration Flow for Processor Architecture

Sophiane Senni, Lionel Torres, Gilles Sassatelli, Abdoulaye Gamatié, Bruno Mussard

Emerging Non-Volatile Memory Technologies Exploration Flow For Processor Architecture

Sophiane Senni[1,2], Lionel Torres[2], Gilles Sassatelli[2] and Abdoulaye Gamatie[2]
LIRMM – UMR CNRS 5506 – University of Montpellier
Montpellier, France
{lastname[2]}@lirmm.fr

Bruno Mussard
Crocus technology
Rousset, France
{ssenni[1], bmussard}@crocus-technology.com

***Abstract*— Most die area of today's systems-on-chips is occupied by memories. Hence, a significant proportion of total power is spent on memory systems. Moreover, since processing elements have to be fed with instructions and data from memories, memory plays a key role for system's performance. As a result, memories are a critical part of future embedded systems. Continuing CMOS scaling leads to manufacturing constraints and power consumption issues for the current three main memory technologies, i.e. SRAM, DRAM and FLASH, which compromises further evolution in upcoming technology node. To face these challenges, new non-volatile memory technologies emerged in recent years. Among these technologies, magnetic RAM (MRAM) is a promising candidate to replace existing memories since it combines non-volatility, high scalability, high density, low latency, and low leakage. This paper describes an evaluation flow to explore next generation of the memory hierarchy of processor-based systems using new non-volatile memory technologies.**

Keywords—MRAM, NVM, Memory hierarhy, VLSI, SoC, Embedded Systems

I. INTRODUCTION

Because of its low access time, SRAM is currently the most suitable memory technology to design the upper level of cache memories to reach the best performance, particularly for multiprocessor architecture. Current issue of SRAM decreasing the technology node is the high leakage current leading to high power dissipation. For decades, DRAM has been used for main memory since it is slower but has higher density than SRAM. This technology is also power consuming due to its mandatory refresh policy. In addition, DRAM faces to manufacturing constraints for the most advanced node. At a lower level of the memory hierarchy, FLASH memory is used for its high storage and non-volatility capabilities. To overcome performance and power challenges of this multi-core era, new non-volatile memory technologies (NVMs) emerged over the past few years. While being non-volatile, MRAM is suitable to become a universal memory as it combines good scalability, low leakage, low access time and high density. However, despite the many attractive features of MRAM, two main issues are still under intensive investigation: write latency and write energy. Compared to SRAM, MRAM write latency is around three to ten times higher, as well as MRAM write energy due to its high current requirement to switch the bit cell.

This paper presents an exploration flow to evaluate integration of MRAM into the memory hierarchy of processor architectures. Both performance and energy are analyzed using both architecture-level and circuit-level tools. Useful information about the memory activity is extracted to better understand the results.

II. MRAM BASICS

MRAM bit is a magnetic tunnel junction (MTJ) which consists of two ferromagnetic layers separated by a thin insulating barrier. The information is stored as the magnetic orientation of one of the two layers, called the free layer (FL). The other layer, called the reference layer (RF), provides the fixed reference magnetic orientation required for reading and writing. Four methods have been proposed to switch the orientation of the FL: toggle [1], thermally assisted switching (TAS) [2], spin transfer torque (STT) [3] and spin orbit torque (SOT) [4].

The toggle write scheme consists of a specific timing sequence of the write-current pulses through the conductive lines to switch the magnetic orientation of the FL to its opposite direction.

TAS-MRAM uses an anti-ferromagnetic layer to block the magnetic orientation of the FL under a threshold temperature. To switch the bit cell, a select transistor provides a flow of current to heat the MTJ above the blocking temperature thereby enabling storage of new information thanks to application of a magnetic field.

STT-MRAM uses the spin transfer torque effect to switch the magnetic orientation of the FL. A highly spin polarized current flowing through the MTJ causes a "torque" applied by the injected electron spins on the magnetization of the FL.

SOT is the most recent MRAM technology. Contrary to STT-MRAM, this new technique uses a three-terminal structure to separate the read and write paths. The physical effect responsible for the reversal of magnetization of the FL is not yet fully understood. According to some authors, the Rashba effect [5] or the spin hall effect [6] could explain the switch in magnetization of the storage layer.

III. NVM EXPLORATION FLOW

To evaluate the impact of integrating MRAM into the memory hierarchy of processor architecture, a framework based on both circuit-level and architecture-level tools is needed (See Fig. 1). A circuit-level tool needs to provide characteristics of a complete memory circuit (i.e. including data array and peripheral circuits). An architecture-level tool simulates a complete processor-based system with its memory hierarchy.

For area, performance, and energy evaluations, the minimum information required is:

- Circuit-level requirements
  - Access latency (read/write)
  - Access energy (read/write)
  - Static power
  - Area
- Architecture-level requirements:
  - Execution time of the simulated applications
  - Amount of memory transactions for each level of the memory hierarchy (reads/writes)

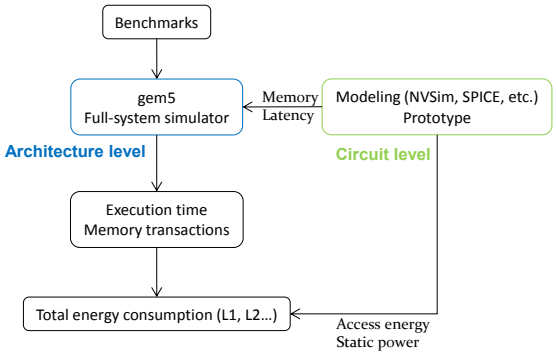

Fig. 1. NVM exploration framework

This section presents a framework based on gem5 [7], a processor architecture simulator widely used by the research community. gem5 currently supports most commercial Instruction Set Architectures (ISA) including ARM, ALPHA, MIPS, Power, SPARC, and x86. gem5 is able to simulate a complete processor-based system with devices and operating system in full system mode (i.e. nothing is emulated). The use of gem5 make it possible to define the total processor system architecture, including memory hierarchy specifications: cache size, cache and main memory latencies, etc. Execution time and memory transactions can be extracted for a given application, i.e. cache read/write accesses including cache hits and misses. In addition, the cache miss rate, the cache miss latency and the memory bandwidth can be monitored over time to better understand the activity of the memory. Hence, a fine-grain analysis of performance and energy results for each simulated workload is possible.

To calibrate the memory hierarchy in terms of access latency, NVSim [9] was used, a circuit-level model for NVM performance, energy, and area estimation, which supports different NVM technologies including STT-MRAM (planar), resistive RAM (RRAM), phase-change RAM (PCRAM). It also models the volatile SRAM memory. NVSim needs two input files to estimate the performance, energy and area of a complete memory circuit:

- An input file specifying the memory cell properties (memory technology, cell area, etc.)
- An input file specifying the memory module parameter (cache or RAM, memory size, etc.)

Using NVSim, a rapid estimation of electrical features of a complete memory chip is possible including read/write access time, read/write access energy and static power. The estimation error is ≤24% [9]. If more precise values are desired, the results of the SPICE simulation of a design or the electrical features of a real prototype can be easily used.

IV. EXPERIMENTAL SETUP

As a case study, some applications of SPLASH-2 benchmark suite [10] were used to explore STT-MRAM and TAS-MRAM based caches for quad-core processor ARM architecture. SPLASH-2 workloads are mostly in the area of high performance computing (HPC). Table I shows the architecture configuration and Table II provides details on the simulated workloads.

TABLE I. ARCHITECTURE CONFIGURATION

Hierarchy Level	Configuration
Processor	4-core, 1 GHz, 32-bit RISC ARMv7 (Linux OS)
L1 I/D cache	Private, 32kB, 4-way associative, 64B cache line
L2 cache	Shared, 512kB, 8-way associative, 64B cache line
Main memory	DRAM, DDR3, 100-cycle latency

TABLE II. SPLASH-2 WORKLOADS

Workloads	Input set
barnes	16K Particles, Timestep = 0.25, Tolerance 1.0
fmm	16K Particles, Timestep = 5
fft	2^{20} total complex data points
lu1	Contiguous blocks, 512x512 Matrix, Block = 16
lu2	Non-contiguous blocks, 512x512 Matrix, Block = 16
ocean1	Contiguous partitions, 514x514 Grid
ocean2	Non-contiguous partitions, 258x258 Grid
radix	4M Keys, Radix = 4K

Note that the cache latency parameters in gem5 were calibrated using simulation results of NVSim for both SRAM and STT-MRAM, while for TAS-MRAM, outcomes from a real prototype were used thanks to support provided by Crocus Technology. To take into account the state of the art of MRAM technology and to evaluate performance and energy fairly, we compare 45 nm STT-MRAM results with a baseline 45 nm SRAM cache, and 130 nm TAS-MRAM results with a baseline 120 nm SRAM cache.

V. PERFORMANCE EVALUATION

Table III shows the latencies of a 512 kB L2 cache for the three memory technologies concerned. As expected, both MRAM technologies have higher write latency than SRAM. Regarding hit latency, STT-MRAM is faster than SRAM (45 nm). This is not surprising since STT-MRAM is denser than SRAM. As a result, for the same capacity, the total L2 cache area for STT-MRAM is smaller than for SRAM (see Table III) resulting in a shorter bit line delay. This difference in hit latency in favor of STT-MRAM is only noticeable in the case of large cache capacity. On the other hand, TAS-MRAM write and hit latencies are respectively 8.5 and 6 times higher than those of SRAM (120 nm).

TABLE III. 512 KB L2 CACHE FEATURES

Technology	Latency		Energy			Cache area	
	Hit (ns)	Write (ns)	Hit (nJ)	Write (nJ)	Leakage (mW)	Total (mm^2)	Cell (F^2)
45 nm SRAM	4.28	2.87	0.27	0.02	320	1.36	146
45 nm STT	2.61	6.25	0.28	0.05	23	0.82	57
120nm SRAM	5.95	4.14	1.05	0.08	82	9.7	146
130 nm TAS	35	35	1.96	4.62	10	11.7	35

TABLE IV. 32 KB L1 CACHE FEATURES

Technology	Latency		Energy			Cache area	
	Hit (ns)	Write (ns)	Hit (nJ)	Write (nJ)	Leakage (mW)	Total (mm^2)	Cell (F^2)
45 nm SRAM	1.25	1.05	0.024	0.006	22	0.091	146
45 nm STT	1.94	5.94	0.095	0.04	3.3	0.117	57

The difference between the latency parameter in SRAM and MRAM will of course depend on the frequency used by the processor. In this study, the frequency used for the processor was 1GHz. Table V shows the access latencies in terms of CPU cycles for L1 and L2. Since TAS-MRAM was evaluated only for L2 cache, L1 latencies for TAS-MRAM and the baseline 120 nm SRAM are not shown.

TABLE V. CACHE LATENCY FOR A 1GHZ PROCESSOR

Technology	Latency (CPU cycle)			
	32kB L1		512kB L2	
	Hit	Write	Hit	Write
45nm SRAM	2	2	5	3
45nm STT-MRAM	2	6	3	7
120nm SRAM	-	-	6	5
130nm TAS-MRAM	-	-	35	35

Fig. 2 shows the execution time of SPLASH-2 workloads for both STT-MRAM and TAS-MRAM based L2 caches. Fig. 2 shows that the performance of STT-MRAM-based L2 scenario is similar and sometimes better (*ocean1*, *ocean2*) than the baseline. This is because STT-MRAM has a smaller hit latency than its SRAM equivalent.

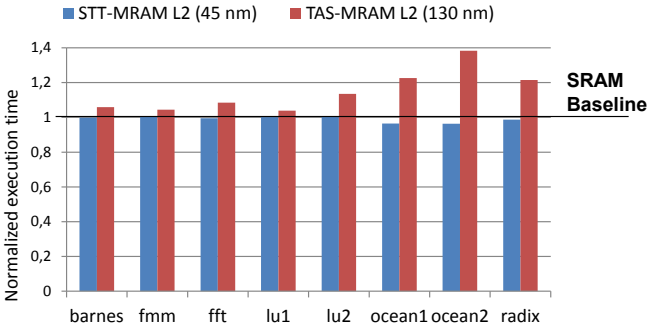

Fig. 2. Execution time with the MRAM-based L2 cache

TAS-MRAM-based L2 performance penalties from 3% (for *lu1*) to 38% (for *ocean2*) were observed. This is understandable since TAS-MRAM has higher access latency than STT-MRAM for both read and write operations. To better understand these results, the L2 cache miss rate was monitored over time (see Fig. 3). For *ocean2*, a high L2 cache miss rate is observed explaining the high penalty on the execution time using TAS-MRAM. For other workloads, such as *barnes*, the small penalty on the execution time is justified by a lower L2 cache miss rate compared to the *ocean2* workload.

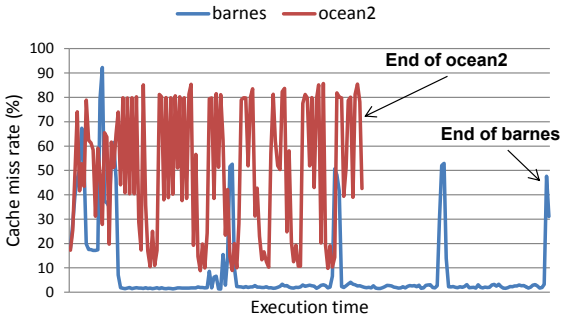

Fig. 3. L2 cache miss rate for barnes and ocean2 workloads

Fig. 4 shows the execution time with different configurations of STT-MRAM-based L1 cache: a first scenario in which both the I-Cache and D-Cache are based on STT-MRAM, a second scenario with STT-MRAM-based I-Cache only and, a third scenario with STT-MRAM-based D-Cache only.

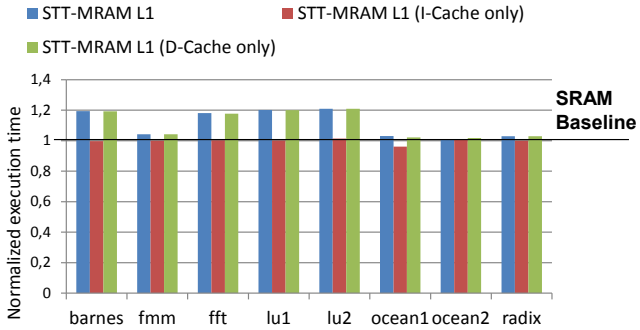

Fig. 4. Execution time of MRAM-based L1 cache

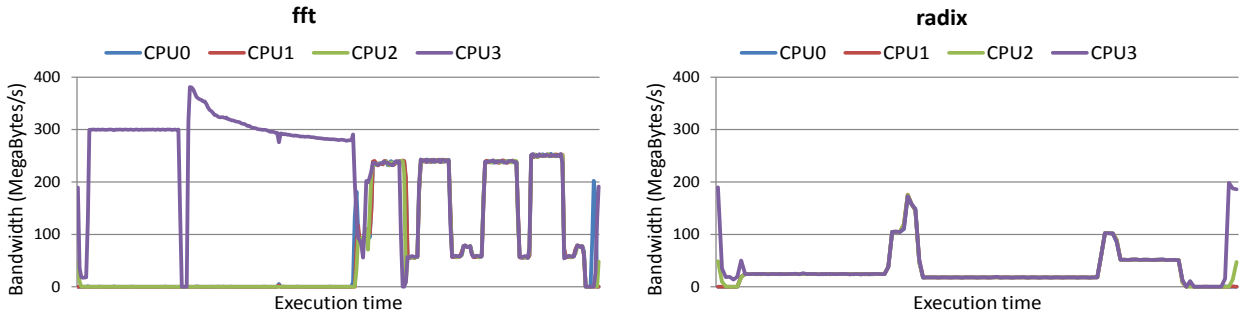

Fig. 5. D-Cache write bandwidth for fft and radix workloads

As already observed in Table V, the L1 hit latency is the same (in terms of CPU cycles) with both technologies. Therefore, STT-MRAM is slower than SRAM only in write operations. Since I-Cache is read only, replacing SRAM by STT-MRAM only in the I-Cache does not affect performance. Penalties are only noticed in the other scenarios in which the D-Cache is based on STT-MRAM. For some workloads (*barnes*, *fft*, *lu1*, *lu2*), using STT-MRAM in the D-cache reduces overall performance by around 20% due to its high write latency. For others, such as *radix*, the execution time penalty is very small, even with a STT-MRAM-based D-Cache. This can be explained by analyzing the cache write bandwidth of the D-Cache (see Fig. 5), which clearly shows that D-Cache is more frequently accessed in write for *fft* than for *radix* (for example). Hence, the impact of the long write latency of STT-MRAM on the execution time is more visible for the *fft* workload. Overall, the simulation results showed that the execution time penalty of STT-MRAM-based D-Cache does not exceed 21%, because the L1 cache is much more accessed in read for the simulated workloads.

VI. ENERGY EVALUATION

Table III provides energy consumption of SRAM, STT-MRAM and TAS-MRAM based L2 caches, while Table IV shows energy consumption of SRAM and STT-MRAM based L1. Regarding L2 energy consumption, using MRAM instead of SRAM results in higher write energy for both STT-MRAM and TAS-MRAM. STT-MRAM hit energy is very similar to that of SRAM, whereas a TAS-MRAM hit consumes around 2 times more energy than SRAM. Concerning L1 energy consumption, STT-MRAM consumes around 4 times and 7 times more energy than SRAM for hit and write operations, respectively.

However, in terms of static power, MRAM has a considerable advantage over SRAM (see Table III and IV): a 45 nm STT-MRAM-based L2 consumes over one order of magnitude less power than a 45 nm SRAM-based L2, while a TAS-MRAM-based L2 consumes around 8 times less power than a 120 nm SRAM-based L2. For L1 cache, a significant gain in static power is also obtained by replacing SRAM with STT-MRAM due to the zero leakage of the MTJ. This is because most of the static power of memories comes from cell arrays. Since MRAM cell has zero standby power and the CMOS access transistor does not require power supply, all the static power in MRAM-based memory is due to peripheral circuitry such as address decoding, drivers and sense amplifiers.

Fig. 6 and Fig. 7 display total L2 energy consumption and total L1 energy consumption, respectively. Concerning L2, simulation results showed using STT-MRAM is 92% more energy efficient, on average, than SRAM, and using TAS-MRAM, 63% energy gain on average were observed. For total L1 energy consumption (i.e. including for the L1 cache of each core), replacing SRAM with STT-MRAM does not gain as much energy as with the L2 cache. The reason is the L1 cache is much more accessed than L2 cache. As a result, the dynamic energy impact of STT-MRAM is more visible. It will be recalled that previous analysis showed that STT-MRAM consumes respectively around 4 times and 7 times more energy than SRAM for hit and write operations in L1 (See Table IV). Fig. 7 shows average energy gains of 38% for STT-MRAM in both I-Cache and D-Cache, of 9% for STT-MRAM in I-Cache only, and of 24% for STT-MRAM in D-Cache only. This notable difference in energy consumption between the two technologies is explained by the low leakage power of MRAM compared to SRAM. Simulations results showed that replacing SRAM with MRAM can dramatically reduce the total energy consumption in the cache. This make MRAM-based cache memory an attractive alternative for energy efficient systems since the performance remains reasonable.

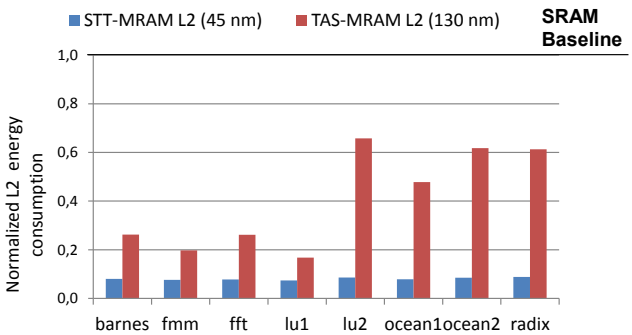

Fig. 6. MRAM-based L2 energy consumption

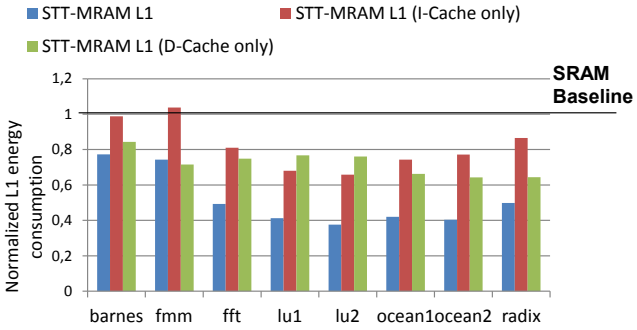

Fig. 7. MRAM-based L1 energy consumption

To better understand the large variations observed in L2 energy consumption between the simulated workloads using TAS-MRAM, the cache bandwidth is monitored over time (see Fig. 8) to see how often the L2 is accessed in read and write (in Bytes per second). Fig. 6 shows for instance a notable difference in energy consumption between *lu1* and *lu2* using TAS-MRAM. This is because the L2 read and write bandwidths for *lu1* are significantly lower than those for *lu2*. As a result, total TAS-MRAM-based L2 energy consumption is lower for *lu1* than for *lu2*. This kind of analysis (i.e. the cache bandwidth analysis) can also be done for the L1 cache to better understand the results on the energy consumption between the simulated workloads. It is not shown in this study for the sake of brevity.

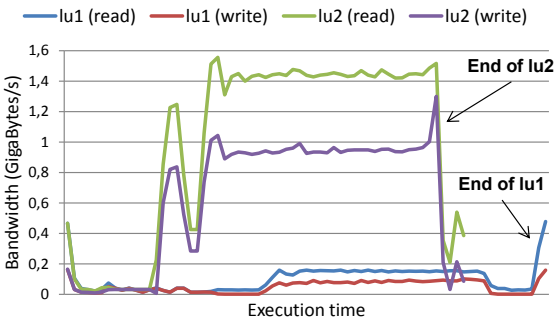

Fig. 8. L2 bandwidth for lu1 and lu2 worloads

VII. EXPLORATION FOR DIFFERENT NUMBER OF CORES

This section aims at analyzing the energy impact of MRAM-based cache when the number of cores is changed: quad-core, dual-core, and single-core. The results shown in this section are the average L1/L2 energy consumption over all the simulated workloads. Performance analysis is not detailed because simulation results reveal that the execution time penalty of MRAM-based cache (compared to the baseline) does not change significantly when increasing the number of cores from one to four. It will be recalled that the L2 is shared for multi-core architectures.

Fig. 9 and Fig. 10 depict respectively total L2 energy consumption and total L1 energy consumption for 4-core, 2-core and 1-core processor architectures. Substantial variations are noticed when the number of cores is changed. Regarding L2 cache in Fig. 9, changing from 4-core to 1-core architecture increases the energy consumption gain by 14% using TAS-MRAM instead of SRAM. On the other hand, no significant change is noticed with STT-MRAM-based L2 when the number of cores is changed.

Fig. 10 shows that when SRAM is replaced by STT-MRAM in L1, changing from 4-core to 1-core architecture increases the energy consumption gain by 18% for STT-MRAM-based I-Cache only, by 4% for STT-MRAM-based D-Cache only, and by 24% for STT-MRAM in both I-Cache and D-Cache.

To better understand this trend (i.e. the energy consumption gain over SRAM-based cache increases when the number of cores is reduced), the cache bandwidth is monitored over time (see Fig. 11) for 4-core, 2-core and 1-core architecture. The analysis is done only for the L2 and only for one workload, since analysis for L1 and for other workloads result in the same conclusions.

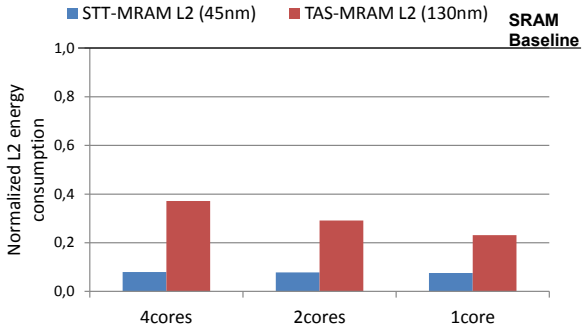

Fig. 9. MRAM-based L2 energy consumption for different number of cores

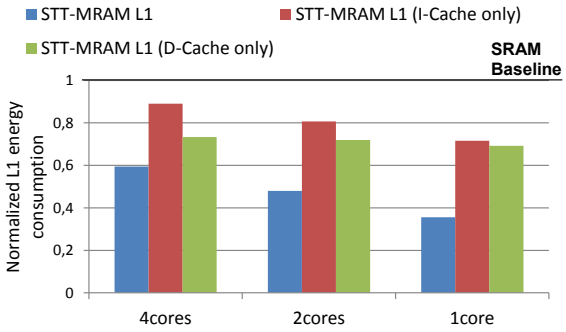

Fig. 10. MRAM-based L1 energy consumption for different number of cores

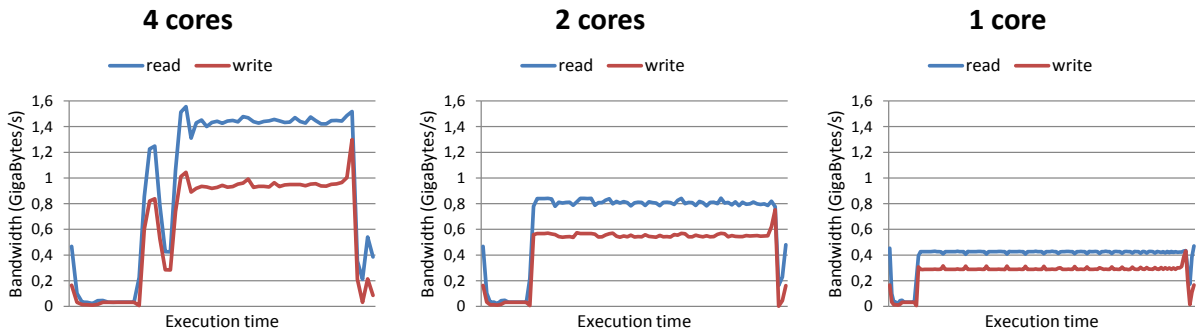

Fig. 11. L2 bandwidth for different number of cores (lu2 workload)

Fig.11 shows that the L2 bandwidth is reduced by around 2 when the number of cores is decreased from 4 to 2, and from 2 to 1. The dynamic part of total L2 energy consumption then is lower for 1-core than for 4-core architecture. Consequently, the loss due to the high dynamic energy of MRAM is reduced. This explains the higher energy gain using MRAM instead of SRAM in L2 for 1-core than for 2-core or 4-core architecture (see Fig. 9). This is particularly visible for TAS-MRAM because it has higher dynamic energy than SRAM for both read and write, whereas for STT-MRAM, this is the case only for writes.

VIII. Related Work

A few studies on integrating NVMs into the memory hierarchy of processor architectures were made in [11], [12], and [13] also using the gem5 simulator. Contrary to these investigations, we do not limit the analysis of the performance and energy of the MRAM-based cache to a direct comparison with that of SRAM-based cache, but we rather observe and analyze memory activity over time to better understand the performance and energy issues.

IX. Conclusions

This paper presented a NVM exploration flow using the gem5 processor architecture simulator to evaluate integration of NVM into a memory hierarchy. Both performance and energy are analyzed using useful information about the memory traffic. Simulations show it is possible to significantly reduce the total energy consumption of caches thanks to the low leakage power of MRAM.

Concerning the future work, evaluation of MRAM at register level is envisaged to not only analyze the performance, energy, and area metrics, but also to explore new possible applications using non-volatile registers inside a processor.

Acknowledgment

The Authors wish to acknowledge all people from ADAC team at LIRMM and people from Crocus technology for their support in this work.

References

[1] B.N. Engel et al., “A 4-Mb Toggle MRAM Based on a Novel Bit and Switching Method,” in IEEE Transactions on Magnetics, vol. 41, no. 1, January 2005.

[2] I.L. Prejbeanu et al., “Thermally assisted MRAM,” in Journal of Physics: Condensed Matter, vol. 19, no. 16, 2007.

[3] A.V. Khvalkovskiy et al., « Basic principles of STT-MRAM cell operation in memory arrays,” in Journal of Physics D: Applied Physics, vol. 46, no. 7, 2013.

[4] D. Gambardella and I. M. Miron, “Current-induced spin-orbit-torques,” in Philosophical transactions A of the Royal Society : Mathematical, Physical and Engineering Sciences, vol. 369, no. 1948, pp. 3175-3197, Aug. 2011.

[5] I. M. Miron et al., “Current-driven spin torque induced by the Rashba effect in aferromagnetic metal layer,” in *Nature Materials*, vol. 9, pp. 230-234, Jan. 2010.

[6] L. Liu et al., “Spin-Torque Switching with the Giant Spin Hall Effect of Tantalum,” in *Science*, vol. 336, no. 6081, May 2012.

[7] N. Binkert, S. Sardashti, R. Sen, K. Sewell, M. Shoaib, N. Vaish,M. D. Hill, D. A. Wood, B. Beckmann, G. Black, S. K. Reinhardt, A. Saidi, A. Basu, J. Hestness, D. R. Hower, and T. Krishna, “The gem5 simulator,” ACM SIGARCH Computer Architecture News, vol. 39, no. 2, pp. 1–7, Aug. 2011.

[8] A. Butko, R. Garibotti, L. Ost, and G. Sassatelli, “Accuracy Evaluation of GEM5 Simulator System,” in the proceedings of the 7th International Workshop on Reconfigurable Communication-Centric Systems-on-Chip (ReCoSoC), pp. 1–7, 2012.

[9] X. Dong, C. Xu, Y. Xie, and N. P. Jouppi, “NVSim: A Circuit-Level Performance, Energy, and Area Model for Emerging Nonvolatile Memory,” IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, vol. 31, no. 7, pp. 994-1007, Jul. 2012

[10] S.C. Woo, M. Ohara, E. Torrie, J.P. Singh and A. Gupta, “The SPLASH-2 Programs: Characterization and Methodological Considerations,” in Proceedings of the 22nd Annual International Symposium on Computer Architecture, pp. 24–36, June 1995.

[11] J. Wang, X. Dong and Y. Xie, “OAP: an obstruction-aware cache management policy for STT-RAM last-level caches,” in Design, Automation & Test in Europe Conference & Exhibition (DATE), pp. 847-852, March 2013.

[12] R. Bishnoi, M. Ebrahimi, F. Oboril and M. Tahoori, “Architectural Aspects in Design and Analysis of SOT-Based Memories,” in the 19th Asia and South Pacific Design Automation Conference (ASP-DAC), pp. 700-707, January 2014.

[13] E. Arima et al., “Fine-Grain Power-Gating on STT-MRAM Peripheral Circuits with Locality-aware Access Control,” in The Memory Forum (in conjounction with the 41st International Symposium on Computer Architecture), June 2014, unpublished.