Normally-Off Computing and Checkpoint/Rollback for Fast, Low-Power, and Reliable Devices

Sophiane Senni, Lionel Torres, Pascal Benoit, Abdoulaye Gamatié, Gilles Sassatelli

Normally-Off Computing and Checkpoint/Rollback for Fast, Low-Power and Reliable Devices

Sophiane Senni, Lionel Torres, Pascal Benoit, Abdoulaye Gamatie, Gilles Sassatelli
University of Montpellier, LIRMM UMR CNRS 5506
Email: senni@lirmm.fr

Since the advent of complementary metal oxide semiconductors (CMOS), the number of transistor per die never stops increasing to reach today several billion of transistors. As a result, design and fabrication of smart devices able to run at high speed has been possible. However, the power consumption of systems-on-chip has significantly increased due to the high density integration and the high leakage power of current CMOS transistors. As a result, a heat dissipation wall makes difficult further improvement in performance. Having high autonomy for battery-powered devices becomes a real challenge. To deal with these issues, STT-MRAM technology is seen as a promising solution. In addition to its attractive performance features, STT-MRAM can bring non-volatility inside a system to allow full data retention after a complete shutdown while keeping a fast wake-up time. Considering two 32-bit embedded processors, this paper shows how STT-MRAM can improve energy efficiency and reliability of future embedded systems thanks to normally-off computing and checkpointing/rollback techniques. Finally, a detailed analysis is performed to evaluate the cost related to the backup/recovery of the system.

***Index Terms*—Spintronic memory and logic, embedded processor.**

I. Introduction

THE scaling limits of complementary metal oxide semiconductors (CMOS) are mainly due to the high heat dissipation observed in current systems-on-chip. As a consequence, speed and density are limited and the thermal constraints oblige the system to be partially turned off by using power gating techniques. However, this solution is clearly constrained by the inherent volatility of CMOS devices, since turning off the memory part also means losing the execution state. For beyond CMOS systems, the spin-transfer-torque magnetic random access memory (STT-MRAM) is a promising solution by combining non-volatility, high density, low leakage and competitive access time compared to CMOS-based memories such as static random access memory (SRAM) and flash memory. Recently, new technologies known as spin-hall-effect MRAM ([1], [2], [3]) and voltage-controlled MRAM ([4], [5], [6]) appear to be promising for fast and ultra-low power applications. Since these technologies are quite young and require further development, they are not considered in this work. On the contrary, several prototypes demonstrated the maturity of the STT-MRAM technology ([7], [8], [9], [10]).

Considering two 32-bit embedded processors (The MIPS-like SecretBlaze [11] and the ARM-like Amber [12]), this paper shows how perpendicular STT-MRAM can help to design fast, low-power and reliable devices. The rest of the paper is organized as follows: Section II gives the basics of STT-MRAM technology. Section III presents the experimental setup to evaluate a non-volatile processor based on STT-MRAM. Section IV describes two features introduced by the non-volatility of STT-MRAM for future computing: *normally-off computing* and *rollback*. Section V shows a detailed performance/energy analysis of the considered embedded processors based on STT-MRAM considering various architecture scenarios. Section VI concludes this paper.

Compared to the state of the art, this work validated the *normally-off computing* and *rollback* techniques on two different 32-bit embedded processors, which clearly strengthen our approach. Moreover, this work has also validated a recovery of the system at run time (e.g. in the case of a soft error), contrary to other related works which only considered the recovery process after a shutdown of the system.

Manuscript received 24 January 2017, revised 3 May 2017, accepted 16 May 2017. Corresponding author: M. Senni (email: senni@lirmm.fr).

II. Spin Transfer Torque MRAM: Basics

A bit of information is stored as the resistance of a magnetic tunnel junction (MTJ) which consists of two ferromagnetic layers separated by a thin insulating barrier (Figure 1). The parallel (antiparallel) state causes a low (high) resistance value and can be characterized as a logic zero (one). A read operation consists in measuring the resistance thanks to a sensing current flowing through the MTJ. For the write operation, a spin-polarized current flips the magnetization of the storage layer by direct transfer of the spin angular momentum from spin-polarized electrons. The direction of the current flow through the MTJ determines the final state of the bit cell.

Regarding the last advances of the STT-MRAM technology, IBM and SAMSUNG [13] demonstrated that a specific MTJ stack with perpendicular magnetic anisotropy is capable of delivering good STT performance down to 10^{-6} write error rate (WER) in a broad range of device sizes from $50nm$ to $11nm$, on a statistically relevant sample of several hundred of devices. They demonstrated an individual 11-nm device switching down to WER = 7×10^{-10} using only $7.5\mu A$.

III. Experimental setup

The first step to evaluate the integration of STT-MRAM into the considered processors is to validate two capabilities brought by the non-volatility: the *normally-off computing* for

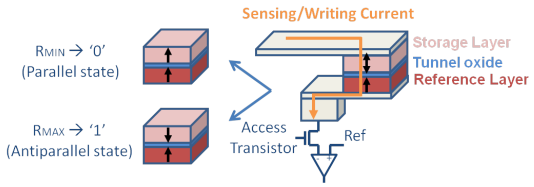

Fig. 1: STT-MRAM bit cell structure

TABLE I: STT-MRAM based flip-flop, cache and main memory performances

Memory element ($45nm$)	Latency (ns)		Energy (pJ)	
	Read	Write	Read	Write
Flip-Flop [14]	0.2	4	0.012	0.5
Cache ($8kB$) [15]	1.5	10.5	53	268
Main memory ($1MB$) [15]	5	14	36	52
Checkpoint memory ($4kB$) [15]	1.06	10.3	8.7	34.5

TABLE II: Architecture scenarios

Scenario	Architecture
s1	Processor core (STT-MRAM based flip-flops) No Cache Memory Main Memory ($1MB$, STT-MRAM)
s2	Processor core (STT-MRAM based flip-flops) Instruction/Data Cache ($8kB$ each, write-trough, STT-MRAM) Main Memory ($1MB$, STT-MRAM)
s3	Processor core (STT-MRAM based flip-flops) Instruction/Data Cache ($8kB$ each, write-trough, SRAM) Main Memory ($1MB$, STT-MRAM)

near-zero leakage power during sleep mode, and the *rollback* to be tolerant against soft errors and power failures. The synthesizable hardware description language (HDL) codes of the processors were modified to implement the aforementioned techniques. The registers retaining the state of the processor were duplicated to emulate the non-volatile STT-MRAM registers, and control logic were added to enable the backup/recovery of the system state.

The second step is to quantify the cost in terms of speed, energy and area of implementing the two prior capabilities. Data from the current state-of-the-art of STT-MRAM based FFs [14] are used to evaluate the cost at register level. On the other hand, the cost at cache and memory levels are quantified thanks to NVSim [15]. Tables I and II respectively detail the performance data of each memory element based on STT-MRAM and the three architecture scenarios considered in this work. The performance data of the SRAM cache used in the scenario 3 are $0.43ns$, $0.4ns$, $8pJ$, and $7pJ$ respectively for read latency, write latency, read energy, and write energy.

IV. Non-volatile computing

A. Normally-off computing

The *normally-off computing* consists in saving the state of the processor before a complete shutdown, then restoring the state after a new power-up. To make this possible, it is required to insert STT-MRAM at both register level and main memory level. Among all of the flip-flops (FFs) of the SecretBlaze (Amber), 1,986 (1,644) FFs contain the state of the processor. A typical FF based on STT-MRAM is designed to have a dual-storage facility (Figure 2a) [16]. The CMOS stage of the FF uses cross-coupled inverters (latch) to store one data bit in its electrical (volatile) form. On the other hand, the magnetic stage uses a MTJ to store one non-volatile data bit. Such a hybrid CMOS/MTJ FF allows fast processing during active mode, while the leakage power is significantly reduced during sleep mode, especially thanks to the non-volatility of the MTJ. Figure 2b shows a snapshot of a silicon prototype with two FFs based on STT-MRAM (200 nm) and the 28nm fully depleted silicon on insulator (FDSOI) CMOS technology. In brief, assuming the FFs and the main memory of the processor are based on STT-MRAM, then a backup/recovery of the processor is as described in Algorithm 1.

Algorithm 1: Normally-off computing: backup/recovery

(1) *For each FF, save the current state by writing the value from the CMOS volatile stage into the non-volatile magnetic stage*;
(2) *Power down the processor. As the main memory is non-volatile, data are preserved*;
(3) *Power up the processor. As the main memory is non-volatile, data are available*;
(4) *For each FF, restore the state by reading the value from the non-volatile magnetic stage to the volatile CMOS stage.*

B. Checkpointing/Rollback mechanism

The *rollback* technique is the ability to restore a safe state of the processor in the case of a system failure (Figure 3). This work assumes that an error detection mechanism is available into the processor architecture to identify errors during the execution, for instance as proposed in [17]. To avoid resetting the application, *checkpoints* can be created at runtime by saving the state of the processor either periodically or at strategic instant during the execution of the application. Then, if a system failure occurs, the last *checkpoint* is recovered. To keep a backup of the system state, a *checkpoint* has to retain the state of the registers and the main memory. Thanks to the dual-storage structure of the STT-MRAM FFs, a checkpoint of the registers is performed by copying data from the volatile CMOS stage to the non-volatile magnetic stage of each FF.

After a *checkpoint*, the main memory contents will most probably be modified. For the *rollback* procedure, a dual-bank memory architecture can be considered. One bank (main memory) is dedicated to the execution of the application whereas the other bank (*checkpoint memory*) is used for

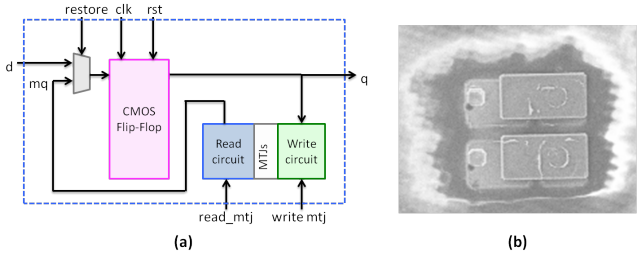

Fig. 2: MRAM-based non-volatile flip-flop: (a) architecture (b) silicon prototype (28nm FDSOI CMOS and 200nm STT-MRAM)

Fig. 3: Rollback principle

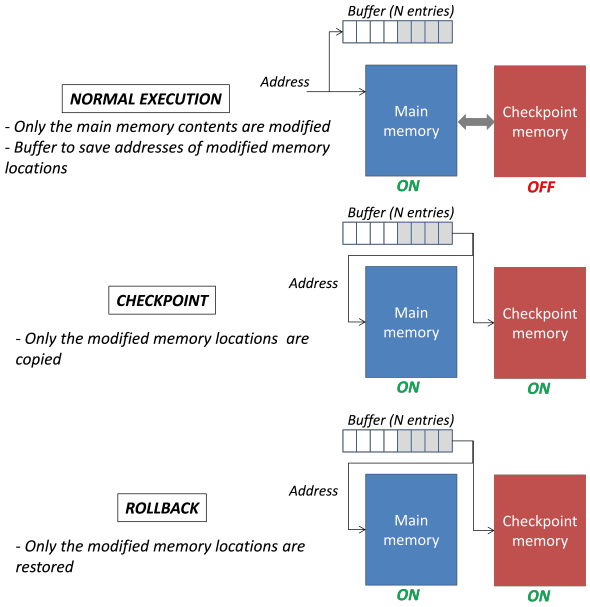

Fig. 4: Checkpointing/Rollback mechanism for main memory

the backup. It is worth noting that the *checkpoint memory* would be smaller than the main memory in a real application. Indeed, only a few memory locations are modified between two *checkpoints*. The size of the *checkpoint memory* depends on the application and the *checkpointing* period. Figure 4 describes how this work implements the backup of the main memory. A buffer is used to save the addresses of the modified memory locations during the execution of the application. If a *checkpoint* is desired, only the modified memory locations (from the last *checkpoint*) are backed up. If the address buffer is full, a creation of a *checkpoint* is forced. In a similar way, if a *rollback* is needed before the next *checkpoint* (e.g. because of an execution error), only the modified memory locations are restored. An alternative solution to perform a *checkpoint* at memory level could be the use of a double context non-volatile SRAM cell as proposed in [16]. Considering such a non-volatile memory based on this cell, it is possible to optimize the silicon area overhead and to greatly simplify the backup of the main memory. In brief, assuming the aforementioned considerations, then the *checkpointing/rollback* mechanism is as described in Algorithm 2.

C. Validation

A complete backup/recovery of the system state has been validated through register-transfer level simulations for the Secretblaze and the Amber processors. A *checkpoint/rollback* procedure is demonstrated in Figure 5 which shows the output terminals of both processors running the Data Encryption

Algorithm 2: Checkpointing/Rollback mechanism

(1) *Create checkpoints during the execution of the application*;
(2) *A system failure is detected*;
(3) *Stall the processor*;
(4) *Restore the last checkpoint consisting in*;
- Restoring the state of the FFs by reading the value from the non-volatile magnetic stage into the volatile CMOS stage;
- Restoring the main memory contents by copying data from the checkpoint memory to the main memory;
(5) *Resume the execution of the application.*

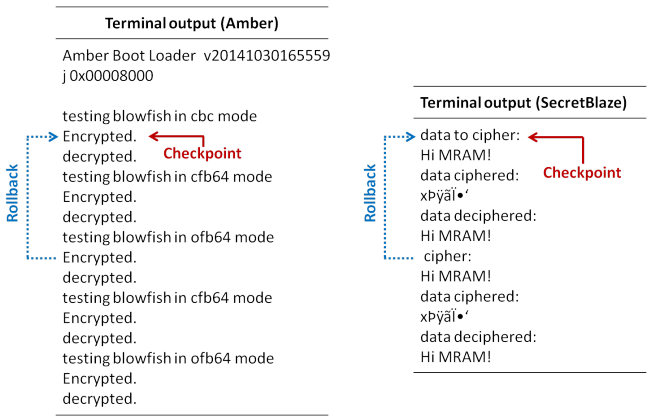

Fig. 5: Checkpoint/rollback validation

Standard (DES) and the blowfish cipher algorithms. In both cases, a rollback is carried out at runtime, then the applications are properly re-executed from the checkpoint.

V. Analysis of STT-MRAM based embedded processor

This section analyzes the cost of integrating the *normally-off computing* and *rollback* features into the considered processors. All the results are summarized in Tables III and IV. The core frequency is set to $50MHz$, and this work assumes a *checkpoint memory* size of $4kB$. The cost has been evaluated separately for each memory element (i.e. FFs, cache, and main memory), then the total cost for the complete system has been reported. This work assumes the backup/recovery processes of the FFs, cache, and main memory are carried out in parallel. Therefore, the latency cost at system level represents the highest latency between the three memory elements. For the energy cost at system level, a simple addition is performed.

A. Register level

Considering data from Table I, each FF consumes $500fJ$ ($12fJ$) to save (restore) the state of the CMOS stage into (from) the non-volatile magnetic stage. As a result, the energy cost to backup the system at register level comes to about $1nJ$ for both processors, and the recovery energy comes to $24pJ$ and $19.7pJ$ respectively for the Secretblaze and the Amber.

Regarding the backup latency, it is worth noting that backing up all the FFs at the same time can lead to a high peak current, which could cause electrical integrity issues. Therefore, a progressive backup is considered in this work. A maximum of 500 FFs are backed up (in parallel) at a time. As $4ns$ is

TABLE III: Cost of the normally-off computing: backup/recovery

Memory element		Backup						Restore					
		Latency			Energy			Latency			Energy		
		s1	s2	s3	s1	s2	s3	s1	s2	s3	s1	s2	s3
Flip-Flops	(Secretblaze)	$16ns$			$993pJ$			$0.8ns$			$24pJ$		
	(Amber)				$822pJ$						$19.7pJ$		
Cache ($8kB$)		$0s$			$0J$			$0s$	warmup		$0J$	warmup	
Main memory ($1MB$)		$0s$			$0J$			$0s$			$0J$		
Total	**(Secretblaze)**	**16ns**			**993pJ**			**0.8ns**	**0.8ns + warmup**		**24pJ**	**24pJ + warmup**	
	(Amber)				**822pJ**						**19.7pJ**	**19.7pJ + warmup**	

TABLE IV: Cost of the checkpointing/rollback

Memory element		Backup (Checkpoint)						Restore (Rollback)					
		Latency			Energy			Latency			Energy		
		s1	s2	s3	s1	s2	s3	s1	s2	s3	s1	s2	s3
Flip-Flops	(Secretblaze)	$16ns$			$993pJ$			$0.8ns$			$24pJ$		
	(Amber)				$822pJ$						$19.7pJ$		
Cache ($8kB$)		$0s$			$0J$			$0s$	$5.12\mu s$ + warmup		$0J$	$19.6nJ$ + warmup	$0.376nJ$ + warmup
Main memory ($1MB$) Checkpoint memory ($4kB$)		$15.7\mu s$			$72.2nJ$			$15.4\mu s$			$62.2nJ$		
Total	**(Secretblaze)**	**15.7 μs**			**73.2nJ**			**15.4 μs**	**15.4 μs + warmup**		**62.2nJ**	**81.8nJ + warmup**	**62.6nJ + warmup**
	(Amber)				**73nJ**								

required to back up one FF, it will take $16ns$ to save all the FFs for both processors. This corresponds to one clock cycle latency if the system frequency is lower or equal to 62.5 MHz. On the other hand, it only takes $0.8ns$ to restore all the FFs (500 FFs restored at a time, $0.2ns$ restore time per FF).

B. Cache level

During a backup, there is no cost associated to the cache memory, even for the scenario 3 which considers a volatile SRAM cache since a write-through policy is used. When restoring the system state, a first cost is related to the warmup. The second cost is related to the flush process to avoid memory inconsistency after a *rollback* at run time. The considered cache architecture has 256 lines. This work implemented the cache flush process by invalidating one cache line per clock cycle. As a result, the latency cost of the *rollback* at cache level comes to $5.12\mu s$. In terms of energy, the *rollback* costs $19.6nJ$ and $0.376nJ$ respectively for the second and the third scenarios. It is worth noting that to invalidate a cache line, only a write into the tag array is required. Therefore, the energy costs reported in Table IV for the cache correspond to the energy of 256 writes into the tag array.

C. Main memory level

For *normally-off computing*, there is no cost related to the main memory. Since the latter is based on STT-MRAM, data are preserved before a complete shutdown of the system. For the *checkpointing/rollback* implementation, the cost to perform a *checkpoint* will depends on the number of bytes it is required to save into the *checkpoint memory*. In this work, the worst case is considered (i.e. the size of the *checkpoint memory* which is $4kB$). The memory architecture is implemented with a 32-bit word width. Therefore, the energy cost to backup $4kB$ (i.e. 1024 words) is represented by the equation 1, where N_{words}, E_{read}, and E_{write} are respectively the number of words to backup, the read energy per access of the main memory and the write energy per access of the *checkpoint memory*. As a result, considering the data in Table I, the backup energy cost at main memory level comes to $72.2nJ$ for both processors. In a similar way, the restore energy comes to $62.2nJ$. Regarding the latency, creating/restoring a *checkpoint* takes about $16\mu s$. It is worth noting that this work does not consider the cost related to error correction code (ECC) to minimize errors. However, it has been demonstrated that every cell out of a $8Mb$ STT-MRAM chip could be written without the use of ECC down to a pulse length of $4.5ns$ [18].

$$E_{Backup} = N_{words} \times (E_{read} + E_{write}) \quad (1)$$

D. Area analysis

Although MRAM is denser than SRAM thanks to its small bit cell structure, the drawback of this technology is the high peripheral circuitry area due to the large CMOS transistors required to generate sufficient write current. As a result, a ratio of 1.5 to 3 is noticed between hybrid CMOS/MTJ FFs and standard CMOS FFs [14]. The STT-MRAM cache area of scenario 2 ($0.048mm^2$) is also bigger than the SRAM cache area of scenario 3 ($0.026mm^2$) by a ratio of 2. On the other hand, the STT-MRAM based $1MB$ main memory is clearly denser ($0.98mm^2$) than its SRAM equivalent ($2.5mm^2$) by a ratio of 2.5. This is because the area of the cell array occupies a large proportion of the total memory area compared to the area of the peripheral circuitry [19].

VI. Conclusion

This paper showed how STT-MRAM could help designing energy-efficient and reliable devices thanks to the *normally-off computing* and the *checkpointing/rollback* mechanisms. Perspectives of this work is to strengthen the results by building a real silicon prototype of a non-volatile processor based on STT-MRAM.

ACKNOWLEDGMENT

This work has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 687973 - (GREAT project), and the French National Research Agency under grant ANR-15-CE24-0033-01 (MASTA project).

REFERENCES

[1] W. Kang, W. Lv, Y. Zhang, and W. Zhao, "Low store power high-speed high-density nonvolatile sram design with spin hall effect-driven magnetic tunnel junctions," *IEEE Transactions on Nanotechnology*, vol. 16, no. 1, pp. 148–154, 2017.

[2] S. Aradhya, G. Rowlands, S. Shi, J. Oh, D. Ralph, and R. Buhrman, "Few-nanosecond pulse switching with low write error for in-plane nanomagnets using the spin-hall effect," in *APS Meeting Abstracts*, 2016.

[3] J. Kim, B. Tuohy, C. Ma, W. H. Choi, I. Ahmed, D. Lilja, and C. H. Kim, "Spin-hall effect mram based cache memory: A feasibility study," in *Device Research Conference (DRC), 2015 73rd Annual*. IEEE, 2015, pp. 117–118.

[4] W. Kang, Y. Ran, W. Lv, Y. Zhang, and W. Zhao, "High-speed, low-power, magnetic non-volatile flip-flop with voltage-controlled, magnetic anisotropy assistance," *IEEE Magnetics Letters*, vol. 7, pp. 1–5, 2016.

[5] H. Noguchi, K. Ikegami, K. Abe, S. Fujita, Y. Shiota, T. Nozaki, S. Yuasa, and Y. Suzuki, "Novel voltage controlled mram (vcm) with fast read/write circuits for ultra large last level cache," in *Electron Devices Meeting (IEDM), 2016 IEEE International*. IEEE, 2016, pp. 27–5.

[6] C. Grezes, H. Lee, A. Lee, S. Wang, F. Ebrahimi, X. Li, K. Wong, J. A. Katine, B. Ocker, J. Langer *et al.*, "Write error rate and read disturbance in electric-field-controlled magnetic random-access memory," *IEEE Magnetics Letters*, vol. 8, pp. 1–5, 2017.

[7] K. Rho, K. Tsuchida, D. Kim, Y. Shirai, J. Bae, T. Inaba, H. Noro, H. Moon, S. Chung, K. Sunouchi *et al.*, "23.5 a 4gb lpddr2 stt-mram with compact 9f2 1t1mtj cell and hierarchical bitline architecture," in *Solid-State Circuits Conference (ISSCC), 2017 IEEE International*. IEEE, 2017, pp. 396–397.

[8] S.-W. Chung, T. Kishi, J. Park, M. Yoshikawa, K. Park, T. Nagase, K. Sunouchi, H. Kanaya, G. Kim, K. Noma *et al.*, "4gbit density stt-mram using perpendicular mtj realized with compact cell structure," in *Electron Devices Meeting (IEDM), 2016 IEEE International*. IEEE, 2016, pp. 27–1.

[9] Y. Song, J. Lee, H. Shin, K. Lee, K. Suh, J. Kang, S. Pyo, H. Jung, S. Hwang, G. Koh *et al.*, "Highly functional and reliable 8mb stt-mram embedded in 28nm logic," in *Electron Devices Meeting (IEDM), 2016 IEEE International*. IEEE, 2016, pp. 27–2.

[10] Y. Lu, T. Zhong, W. Hsu, S. Kim, X. Lu, J. Kan, C. Park, W. Chen, X. Li, X. Zhu *et al.*, "Fully functional perpendicular stt-mram macro embedded in 40 nm logic for energy-efficient iot applications," in *Electron Devices Meeting (IEDM), 2015 IEEE International*. IEEE, 2015, pp. 26–1.

[11] L. Barthe, L. V. Cargnini, P. Benoit, and L. Torres, "The secretblaze: A configurable and cost-effective open-source soft-core processor," in *Parallel and Distributed Processing Workshops and Phd Forum (IPDPSW), 2011 IEEE International Symposium on*. IEEE, 2011, pp. 310–313.

[12] C. Santifort, "Amber arm-compatible core," *OpenCores. org*, 2010.

[13] J. J. Nowak, R. P. Robertazzi, J. Z. Sun, G. Hu, J.-H. Park, J. Lee, A. J. Annunziata, G. P. Lauer, R. Kothandaraman, E. J. OSullivan *et al.*, "Dependence of voltage and size on write error rates in spin-transfer torque magnetic random-access memory," *IEEE MAGNETICS LETTERS*, vol. 7, 2016.

[14] D. Chabi, W. Zhao, E. Deng, Y. Zhang, N. B. Romdhane, J.-O. Klein, and C. Chappert, "Ultra low power magnetic flip-flop based on checkpointing/power gating and self-enable mechanisms," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 61, no. 6, pp. 1755–1765, 2014.

[15] X. Dong, C. Xu, Y. Xie, and N. P. Jouppi, "Nvsim: A circuit-level performance, energy, and area model for emerging nonvolatile memory," *Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on*, vol. 31, no. 7, pp. 994–1007, 2012.

[16] B. Jovanovic, R. M. Brum, and L. Torres, "Comparative analysis of mtj/cmos hybrid cells based on tas and in-plane stt magnetic tunnel junctions," *Magnetics, IEEE Transactions on*, vol. 51, no. 2, pp. 1–11, 2015.

[17] I. Wali, A. Virazel, A. Bosio, P. Girard, S. Pravossoudovitch, and M. S. Reorda, "A hybrid fault-tolerant architecture for highly reliable processing cores," *Journal of Electronic Testing*, pp. 1–15, 2016.

[18] G. Jan, L. Thomas, S. Le, Y.-J. Lee, H. Liu, J. Zhu, R.-Y. Tong, K. Pi, Y.-J. Wang, D. Shen *et al.*, "Demonstration of fully functional 8mb perpendicular stt-mram chips with sub-5ns writing for non-volatile embedded memories," in *VLSI Technology (VLSI-Technology): Digest of Technical Papers, 2014 Symposium on*. IEEE, 2014, pp. 1–2.

[19] S. Senni, L. Torres, G. Sassatelli, A. Gamatie, and B. Mussard, "Exploring mram technologies for energy efficient systems-on-chip," *IEEE Journal on Emerging and Selected Topics in Circuits and Systems (JETCAS)*, 2016.

[20] Z. Or-Bach, "Dark silicon – are the dark days coming?" *News & Analysis. EE Times*, 2015.

[21] E. Kültürsay, M. Kandemir, A. Sivasubramaniam, and O. Mutlu, "Evaluating stt-ram as an energy-efficient main memory alternative," in *Performance Analysis of Systems and Software (ISPASS), 2013 IEEE International Symposium on*. IEEE, 2013, pp. 256–267.

[22] L. Jiang, W. Wen, D. Wang, and L. Duan, "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)*. IEEE, 2016, pp. 31–36.

[23] K. Ikegami, H. Noguchi, C. Kamata, M. Amano, K. Abe, K. Kushida, E. Kitagawa, T. Ochiai, N. Shimomura, S. Itai *et al.*, "Low power and high density stt-mram for embedded cache memory using advanced perpendicular mtj integrations and asymmetric compensation techniques," in *2014 IEEE International Electron Devices Meeting*. IEEE, 2014, pp. 28–1.

[24] A. Khvalkovskiy, D. Apalkov, S. Watts, R. Chepulskii, R. Beach, A. Ong, X. Tang, A. Driskill-Smith, W. Butler, P. Visscher *et al.*, "Basic principles of stt-mram cell operation in memory arrays," *Journal of Physics D: Applied Physics*, vol. 46, no. 7, p. 074001, 2013.

[25] J. S. Meena, S. M. Sze, U. Chand, and T.-Y. Tseng, "Overview of emerging nonvolatile memory technologies," *Nanoscale research letters*, vol. 9, no. 1, pp. 1–33, 2014.

[26] C. Layer, L. Becker, K. Jabeur, S. Claireux, B. Dieny, G. Prenat, G. D. Pendina, S. Gros, P. Paoli, V. Javerliac *et al.*, "Reducing system power consumption using check-pointing on nonvolatile embedded magnetic random access memories," *ACM Journal on Emerging Technologies in Computing Systems (JETC)*, vol. 12, no. 4, p. 32, 2016.

[27] H. Koike, T. Ohsawa, S. Ikeda, T. Hanyu, H. Ohno, T. Endoh, N. Sakimura, R. Nebashi, Y. Tsuji, A. Morioka *et al.*, "A power-gated mpu with 3-microsecond entry/exit delay using mtj-based nonvolatile flip-flop," in *Solid-State Circuits Conference (A-SSCC), 2013 IEEE Asian*. IEEE, 2013, pp. 317–320.

[28] B. Lacoste, M. M. de Castro, R. Sousa, I. Prejbeanu, L. Buda-Prejbeanu, S. Auffret, U. Ebels, B. Rodmacq, and B. Dieny, "Control of sub-nanosecond precessional magnetic switching in stt-mram cells for sram applications," in *Memory Workshop (IMW), 2016 IEEE 8th International*. IEEE, 2016, pp. 1–4.