

Exploiting Memory Resilience for Emerging Technologies: An Energy-Aware Resilience Exemplar for STT-RAM Memories



Amir Mahdi Hosseini Monazzah, Amir M. Rahmani, Antonio Miele, and Nikil Dutt

1 Introduction

In the recent years, the aggressive progress in technology scaling has allowed to integrate a larger number of processing cores in the same chip thus leading to the fabrication of multicore and manycore devices. To efficiently exploit such processing power, it is imperative to proportionally increase the performance and bandwidth of the on-chip cache memory sub-system.

Unfortunately, the commonly-used static RAM (SRAM) technology imposes fundamental limitations for the quest of high memory performance in the next generation computing systems. Indeed, the low density of SRAM cells forces to dedicate approximately 60% of the area of today's chips to the cache memories [7, 19]. Moreover, SRAM memories present a considerably high leakage power consumption becoming a considerable issue with the continuous technology scaling

A. M. H. Monazzah (✉)

Iran University of Science and Technology (IUST), Tehran, Iran

Institute for Research in Fundamental Sciences (IPM), Tehran, Iran

e-mail: monazzah@iust.ac.ir

A. M. Rahmani

University of California, Irvine (UCI), Irvine, CA, USA

e-mail: a.rahmani@uci.edu

Institute of Computer Technology, TU Wien, Austria

e-mail: antonio.miele@polimi.it

A. Miele

Politecnico di Milano, Milano, Italy

e-mail: dutt@uci.edu

N. Dutt

University of California, Irvine (UCI), Irvine, CA, USA

© The Author(s) 2021

J. Henkel, N. Dutt (eds.), *Dependable Embedded Systems*, Embedded Systems,

https://doi.org/10.1007/978-3-030-52017-5_21

beyond 40 nm leading to the leakage power to contribute up to 80% of the overall energy consumption of the cache memories [7].

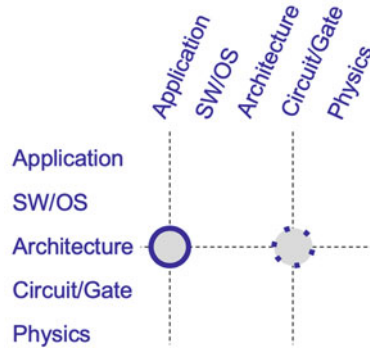
For these reasons, a recent trend is to consider Non-Volatile Memory (NVM) technologies, such as Phase Change Memory (PCM), Resistive RAM (ReRAM), Ferroelectric RAM (FeRAM), and Spin Transfer Torque Magnetic RAM (STT-MRAM or STT-RAM) as alternative solutions to SRAMs in multicore and many-core chips [23]. Among them, various studies [2, 3, 11, 17, 20] have observed that STT-RAM represents the most promising technology for on-chip cache memories. In particular, STT-RAM allows to increase the memory size, thanks to its higher density, and outperforms the SRAM counterpart in terms of energy consumption. In fact, since STT-RAM is a non-volatile technology, its leakage power consumption is negligible.

Unfortunately, STT-RAM technology suffers from a different set of reliability issues that has to be carefully addressed in order to realize its deployment in commercial products. STT-RAMs present a high susceptibility to failure both during write/read operations and in idle status [2, 23]. More precisely, the memory cell may suffer from *retention errors*, which are caused by thermal noises that lead to an unintentional bit flip of the value stored in an idle cell. Moreover, during read operations, it may occur that the cell content incorrectly flips leading to the so-called *read disturbance*, or the returned value has an undistinguished state, dubbed as *false read*. Finally, also the write operation may suffer from *write errors*, which are caused by thermal fluctuations in the magnetization process that lead to storing a wrong value in the cell w.r.t. the one in the input. Among these reliability threats, write errors impose the most challenging issue [2, 7].

A circuit-level strategy to eliminate write errors is to increase the current (voltage) applied during the write operation [23]. However, it mainly leads to higher energy consumption, and secondarily to a higher probability of permanent failures of the device. Indeed, higher current implies an increase in the temperature and in turn to an increase in the probability of junction barrier breakdown [7]. An alternative design strategy is the use of Error Correction Codes (ECCs) to harden the architecture of the cache memory [1]. However, if it is applied in a naive way, it may result in a significant area overhead. For instance, when Bose–Chaudhuri–Hocquenghem (BCH) 7 ECC is employed, such overhead can be as high as 15% of the data block area [2]. Such additional area causes a significant energy overhead. Given such rationale, several studies [2, 3, 6, 7, 17, 18, 20, 21, 24] have individually investigated these techniques and proposed strategies to improve their effectiveness. Given this background, we believe that there is an opportunity for a larger improvement of these hardening schemes by holistically amalgamating these two techniques. Moreover, an opportunistic integration and tuning of these two techniques can also lead to a considerable improvement in energy consumption of the cache memory architecture while at the same time guaranteeing the error rate threshold.

This chapter proposes FlexRel, a reliability improvement technique which utilizes the STT-RAM write current as an actuation knob and multi-level ECC

Fig. 1 Positioning of the proposed approach in the overall cross-layer vision of the book



protection scheme to conduct an optimal trade-off between reliability and energy consumption in STT-RAM cache memories. Targeting an overall block write error rate threshold that should be guaranteed in applications running on a platform, FlexRel proposes a cache way partitioning scheme that utilizes different combinations of write currents and ECC protection codes in each partition to satisfy that threshold. Then, during the run-time of applications, the FlexRel controller redirects the more vulnerable blocks to more robust partitions to keep the write error rate below the write error threshold. Within the overall cross-layer vision of the book, the main contribution of this chapter can be primarily classified as an architecture-to-application cross-layer approach, also using gate/circuit-level actuation, as illustrated in Fig. 1. In fact, this approach exploits application-level profiled information as an input to an architecture-level memory hardening technique based on ECC while using current tuning at circuit-level, with the final goal of optimizing energy consumption and application reliability.

We evaluate FlexRel using gem5 simulator [4] running SPEC CPU2006 [9] workloads. We compare the efficiency of FlexRel against an optimized uniform protection (OUP) scheme from reliability, energy, area, and performance perspectives. The simulation results show that, while FlexRel meets the write error rate threshold, it outperforms OUP scheme in terms of energy and area by up to 13.2 and 7.9%, respectively. Furthermore, The restriction of write traffics to specific partitions in FlexRel incurs only a 1.7% performance overhead to the system, on average.

The rest of the chapter is organized as follows. Section 2 introduces the necessary background presenting the basic architecture of a STT-RAM cell and the energy/error rate issues of this technology. Section 3 briefly surveys the previous approaches for hardening STT-RAMs highlighting the adopted strategies and differentiating them from the proposed approach. Section 4 is the core of this chapter and presents our proposed FlexRel approach, consisting of an enhanced memory architecture capable of trading off reliability and energy consumption. The proposed solution has been experimentally validated and results are discussed in Sect. 5. Finally, Sect. 6 draws conclusions.

2 STT-RAMs and Their Energy-Reliability Challenges

This section introduces the basics of STT-RAM technology, its architecture and how read/write operations are performed. The second part of the section focuses on energy vs. write error issues in this type of memory. This discussion represents the preliminaries for the proposed energy-aware error-tolerant scheme for STT-RAM.

2.1 Basic Architecture of STT-RAM

Figure 2 shows the basic cell structure of a STT-RAM, called 1 Transistor 1 Magnitude Tunnel Junction (MTJ), shortly 1T-1J. The cell is constructed from an MTJ element and an access NMOS transistor. MTJ itself includes three layers which are a MgO-based barrier (called *tunneling oxide barrier*), a ferromagnetic layer with fixed magnetic field direction (called *reference layer*), and a ferromagnetic layer with free magnetic field direction (called *free layer*). The MgO-based barrier layer is sandwiched between two ferromagnetic layers. STT-RAM works based on the relative magnetic field direction of the free layer and the reference layer (parallel or anti-parallel states). As shown in the figure, the parallel state will represent a logic value "0" while the anti-parallel one a logic value "1." The different relative ferromagnetic field directions lead to different resistances in MTJ, i.e. R_{High} and R_{Low} (R_H and R_L) [5]. In the following, we explore the read and write operation mechanisms in a STT-RAM cell.

The read operation in a STT-RAM cell is initiated by setting the word line (WL in the figure) to turn on the access NMOS transistor. Then, a small read current I_R (or read voltage, V_R) is applied to the MTJ from the source line (SL in the figure) through an access transistor [30]. By applying I_R to MTJ, a current (or voltage) is

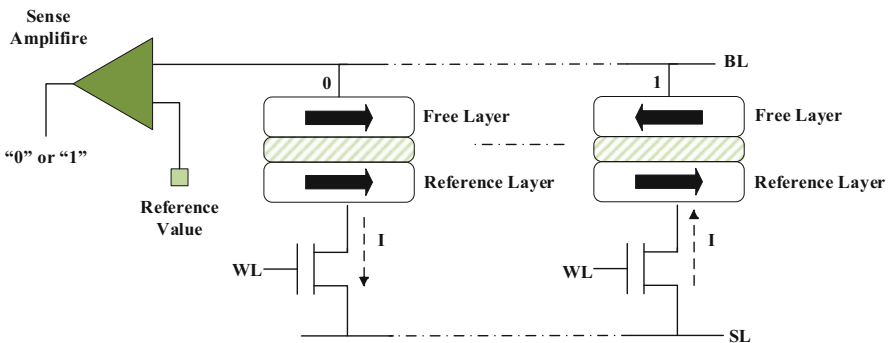


Fig. 2 A typical STT-RAM cell structure: on the left side, a parallel magnetic field direction represents a logic value 0, while on the right side, an anti-parallel magnetic field direction a logic value "1"

sensed on the bit line (BL in the figure). Based on the sensed values of current or voltage in the BL during the read operations, the resistance of MTJ is determined as high (R_H) or low (R_L). The STT-RAM cell value is determined by calculating the Tunneling Magneto Resistance (TMR), a ratio parameter defined as

$$\text{TMR} = \frac{(R_H - R_L)}{R_L} \quad (1)$$

The parallel (anti-parallel) state of MTJ leads to low (high) resistance of MTJ, calculated by means of the following equations:

$$V_{BL_L} = I_R \times (R_L + R_{NMOS}) \quad (2)$$

$$V_{BL_H} = I_R \times (R_H + R_{NMOS}) \quad (3)$$

If the sense amplifier which calculates the voltage (current) of bit line decides the sensed voltage is the same as V_{BL_L} , the MTJ value is logically “0,” otherwise, if the sensed voltage is the same as V_{BL_H} , MTJ value is logically “1.”

To perform a write operation and modify the value stored in MTJ, we need to change the magnetic field direction of MTJ free layer. By changing the magnetic field direction of the free layer, the resistance of MTJ will change [5, 10]. To this end, again the access transistor should be turned on by setting the word line. Then, a write current is applied from the source line to the baseline or vice versa. The direction of applied write current determines the magnetic field direction of the free layer. By applying the write pulse to the MTJ, when the amount of spin polarized current exceeds a threshold value, the magnetic field direction of the free layer flips.

2.2 Error Rate vs. Energy Consumption Trade-Off

One of the main issues in STT-RAM is its stochastic switching nature caused by the effects of thermal fluctuations. Among the side-effects of stochastic switching, write failure is the most important reliability challenge [2, 7]. More precisely, write failure occurs during the write operation and its effect is that the value stored in the MTJ will be different from the one provided as data input. From a physical point of view, it happens according to the stochastic behavior of STT-RAM cell when the magnetic field direction of the free layer could not change during the pre-determined write pulse width [12, 27]. There are many parameters that contribute to switching the MTJ state during the write operations, e.g., MTJ switching current, process variations, thermal fluctuations, and switching pulse width. According to [16], the write failure probability can be calculated using the following equation:

$$P_{wf}(t_w) = \exp\left(-t_w \cdot \frac{2\mu_B p (I_w - I_{C_0})}{\left(c + \ln\left(\frac{\pi^2 \Delta}{4}\right)\right) \cdot (em(1 + p^2))}\right) \quad (4)$$

where Δ is the thermal stability factor, I_{C_0} is the critical MTJ switching current at 0°K, c is the Euler constant, e is the magnitude of electron charge, m denotes the magnetic momentum of the free layer, p is the tunneling spin polarization, μ_B is the Bohr magneton, I_w is the write current, and t_w is the write pulse width.

While STT-RAM technology is a promising candidate to resolve the static energy challenge of SRAM technology in on-chip memories, from the dynamic energy consumption perspective, it imposes considerable energy consumption for a reliable write operation due to its stochastic switching feature; the higher the I_w , the lower the write error rate of STT-RAM will be. For example, we used NVSim [8] to experimentally compare two alternative implementations, in SRAM and STT-RAM technologies, with the same 32 KB cache architecture with 64 Byte word lines implemented in 45 nm. Our results show that from the leakage power point of view, the SRAM cache imposes 41.896 mW power consumption, while STT-RAM cache only charges 9.066 mW power consumption to the design. On the other hand, each read operation in SRAM and STT-RAM implementations uses 11.421 and 82.493 pJ dynamic energy consumption, respectively. Finally, each write operation in SRAM and STT-RAM technologies enforces 5.712 and 534.375 pJ dynamic energy consumption to the design, responsively. As a conclusion, I_w in Eq. 4 is the main contributor for dynamic energy consumption in STT-RAM. Accordingly, I_w is one of the effective circuit-level knobs available to control the reliability-energy trade-off during a write operation. Generally, a lower I_w decreases the write energy, but it also amplifies the probability of write failure.

To systematically analyze this aspect, we performed a quantitative evaluation of the write error rate of STT-RAM at different write current amplitudes by using the STT-RAM SPICE model introduced in [14]. In particular, we characterized a STT-RAM cell by using parameters reported in Table 1 and ran several Monte Carlo simulations. Figure 3 depicts the write error rate of the STT-RAM cell when the write current is varied and the cell is flipped from 0 \rightarrow 1 (in red) or vice versa (in blue). As shown in Fig. 3, we retrieved the trend lines of error rate patterns in both directions to generate the STT-RAM write error rate formulas. These formulas are useful to estimate the write error rates of STT-RAM at any write current. We therefore conclude that I_w is an effective circuit-level knob available to control the

Table 1 STT-RAM HSPICE model configurations

Parameter	Value ($\mu \pm 3\sigma$)
MTJ length	32 nm
MTJ width	96 nm
MTJ thickness	2.44 nm
Relative initial angle	$0 \pm 35^\circ / 180 \pm 35^\circ$
Transistors technology size	32 ± 1 nm

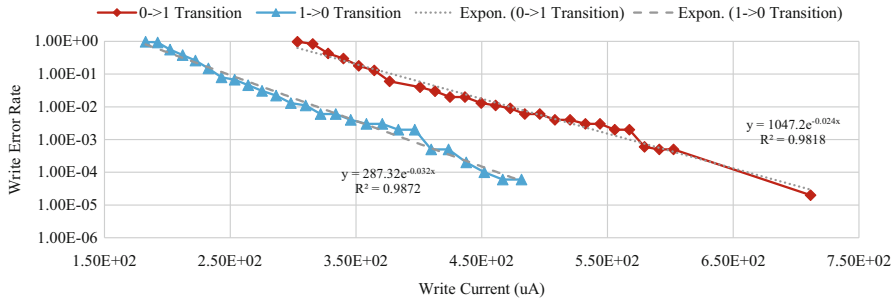


Fig. 3 STT-RAM cell write error rate vs. write current in different MTJ state transitions (write pulse width fixed at 10 ns)

quality-energy trade-off during a write operation. Moreover, it can be noted that the write error rate is asymmetric and $0 \rightarrow 1$ is the most critical transition. This is due to the fact that the initial MTJ state affects the total energy required to change its resistance [29]. For $0 \rightarrow 1$ bit transition, to change the MTJ state from parallel (low resistance) to anti-parallel (high resistance), more energy (power \times time) needs to be spent compared to the amount of energy needed for the transition in the opposite direction ($1 \rightarrow 0$) [28]. As a consequence, the MTJ state transition is asymmetric from the error rate vs. energy consumption perspective.

Finally, it is worth mentioning that unlike the provided knob in SRAM technology (memory bank voltage scaling) that is coarse-grained and affects a large portion of data in memory, STT-RAM exploits I_w which is fine-grained and can be tuned for granularity of a data block in memory. As we show in the following, this feature offered by STT-RAM provides a unique opportunity for flexible adjustment of the energy-reliability knob.

3 Related Work on STT-RAM Reliability

In recent years, several studies have addressed reliability issues of STT-RAM. In [23], the authors present a survey of the preliminary approaches addressing faults in various non-volatile memory technologies with the focus on both permanent and transient faults. Regarding permanent faults, the basic strategies are to (1) increase the current during write operations and (2) augment the architecture by using Error Correction Codes (ECCs). Indeed these strategies are the ones later used by many subsequent approaches.

In [22], the authors propose a strategy, Verify and Correct (VnC), which consists of reading each value immediately after the write operation in the STT-RAM cache to assess its correctness. Since read delay is negligible, such an approach may lead to performance degradation in case of high failure rate. The approach is later enhanced by combining VnC with a limited ECC to reduce the need of rewriting upon

failures. Ahn et al. [1] propose a scheme where ECC is shared among several cache blocks thus reducing its hardware cost. Cheshmikhani et al. [6] and Azad et al. [3] present ECC schemes with an optimized interleaved bit selection mechanism to minimize codewords' vulnerability variations. Finally, the authors of [25] introduce adaptiveness in the hardening scheme; where two different levels of ECC can be selectively chosen for each incoming block in the STT-RAM cache. The level of protection is selected based on the vulnerability of the incoming block which is calculated by enumerating the number of $0 \rightarrow 1$ bit transitions. The idea of using several ECC scheme in an adaptive way is further explored in [2] by defining an STT-RAM cache architecture, called A²PT using several ECC levels and integrating a specific hardware module which selects the replacement candidate in order to minimize the Hamming distance between the stored block the newly incoming one.

A different strategy is proposed in [11] where the classical Least-Recently-Used (LRU) cache replacement policy is substituted with a new algorithm which performs a Least-Error-Rate (LER) replacement. To reduce the probability of write errors in STT-RAM, this algorithm tries to write the incoming block in a location which imposes the least number of $0 \rightarrow 1$ bit transitions among the victim block candidates. In [7], the authors observe that the stochastic switching in write operations is mainly caused by the device heating. Therefore, to reduce the write errors, they propose to replace the LRU policy with a thermal-aware counterpart, which tries to write the incoming block in a location which imposes the least temperature increase among the victim block candidates.

As discussed in Sect. 2, acting on the current applied during the write operation sensibly affects the correctness of the stored value; moreover, $0 \rightarrow 1$ is the most susceptible bit transition. For these reasons, Kim et al. [13] propose two different circuit design techniques applied at each single bit-line to balance out the asymmetric write current and optimize the memory design in terms of write-power and reliability. In a similar manner, Monazzah et al. [17] exploit the tuning of the write current to explicitly trade memory reliability for energy saving in the context of approximate computing. The approach considers software applications capable of tolerating a certain degree of errors in the results, such as image processing applications. Therefore, for each write operation, the current to be applied is dynamically selected based on the reliability requirement annotated in the application source code as well as the Hamming distance between the block to be written and the candidate to be replaced. In such a way, the energy consumption is minimized under a predefined number of errors that can be ignored in the application output.

The main contribution of our approach presented in this chapter is to holistically integrate ECC deployment and write current tuning based on our prior works presented in [2] and [17]. The main property of our approach is its self-adaptiveness to dynamically tune the system operating point to the characteristics of the running applications to optimize the energy consumption of the system while keeping the observed error rate under control.

4 FlexRel: An Energy-Aware Reliability Improvement Approach for STT-RAM Caches

In this section, we present our proposed approach called FlexRel. As a preliminary discussion, first we explore the conventional ECC deficiency in tolerating the write failures of STT-RAM caches. Then, we observe how different data patterns lead to different write error rates in cache blocks. At the end, we discuss in detail the FlexRel approach for the STT-RAM caches which utilizes the STT-RAM write current actuation knob and multi-level ECC protection scheme to conduct an optimum trade-off between reliability and energy consumption.

4.1 The Effects of Write Patterns on ECC Protection Level

As mentioned in previous sections, the write error rate in the STT-RAM cache depends on the bit differences between the contents of data that was previously stored in the cache block and the contents of the new incoming block. Indeed, during a write operation while failure may occur in the bit locations that should be toggled, for the other bit locations that will not experience any toggle, we do not observe any write error. In Sect. 2.2, we observed that in STT-RAM the write error rates of $0 \rightarrow 1$ bit transitions is higher than $1 \rightarrow 0$ bit transitions by about two orders of magnitude for the same current amplitude in both directions. Accordingly, FlexRel will mainly focus on $0 \rightarrow 1$ bit transitions since they represent the main contributor to write error rate in STT-RAM.

Generally, the total number of $0 \rightarrow 1$ bit transitions in a STT-RAM cache block is proportional to the Hamming Weight (HW) of the new incoming block, that is the total number of 1 in the bit representation of each block [2, 26]. On the other hand, the maximum number of $0 \rightarrow 1$ bit transitions in a cache block write operation happens when all of the bit locations storing value “1” in the new incoming block should store on bit locations that previously contained “0.” Considering this fact, for a STT-RAM cache that is protected with an ECC code with t -bit error correction capability, we can estimate the Block Error Rate (BER) of a STT-RAM cache write operation according to Eq. 5 [26]:

$$\text{BER}(w, t) \approx 1 - \sum_{i=0}^t C_w^i P_{ER0 \rightarrow 1}^i (1 - P_{ER0 \rightarrow 1})^{w-i} \quad (5)$$

where, $P_{ER0 \rightarrow 1}$ is the bit failure rate in $0 \rightarrow 1$ switching, t the error correction capability of ECC, w the HW of the incoming data, and C_w^i the combination of HW taken i at a time.

We conducted an experimental evaluation of the BER of STT-RAM cache write operations for the incoming blocks when varying HW. In particular, we configured

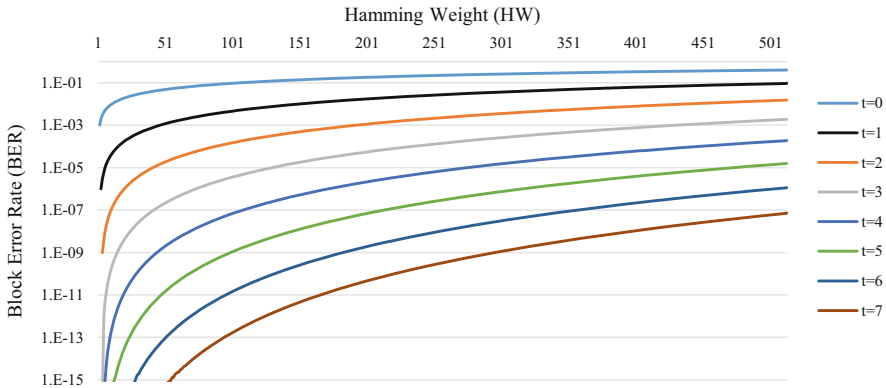


Fig. 4 Block Error Rate (BER) in different ECC schemes for various Hamming Weight (HW) data

a 512-bit STT-RAM cache block, with $0 \rightarrow 1$ write error rate, i.e., $P_{ER0 \rightarrow 1}$ of 10^{-3} as reported in [1, 22, 26]. Figure 4 depicts the results of the experiments which verify that the BER of the cache blocks with different ECC protection scheme are considerably affected by the HW of the incoming data. For example, if we consider ECC with protection capability of t bit errors in a cache block, for HW of 25 we observed the BER of 10^{-15} , while for HW of 50 and 100 this protection scheme delivers BER of 10^{-13} and 10^{-11} , respectively, which is considerably different.

The results shown in Fig. 4 are calculated based on Bose, Chaudhuri, and Hocquenghem (BCH) coding scheme [15] which is a well-known scheme in protecting memory architectures. In BCH code, ECC converts to k -bit data and $(n - k)$ ECC check bits. The complexity of the peripherals that is required to protect the k -bit data is also increased with the increase in the ECC protection capability. Generally, for protecting a $k = 512$ bit cache line using BCH code with $(t + 1)$ bit error correction capability, we require $(10t + 1)$ check bits [2]. The results demonstrated in Fig. 4 is calculated for various coding schemes: SECDED (Single Error Correction-Double Error Detection), DEC-TED (Double Error Correction-Triple Error Detection), 3EC4ED, 4EC5ED, 5EC6ED, 6EC7ED, and 7ED8EC codes with $t = 1$, $t = 2$, $t = 3$, $t = 4$, $t = 5$, $t = 6$, and $t = 7$ error(s) correction capabilities, respectively.

With the emergence of ECC protection scheme in the cache memories to protect the data, conventionally, all the cache blocks are protected with the same ECC protection capability (t). This conventional ECC protection scheme is called Uniform, i.e., all of the blocks in the cache utilize the same ECC protection level. However, as we can see in Fig. 4, the different HWs in write requests lead to various BER for each cache block during the execution time. Accordingly, in the Uniform protection scheme the highest HW needs to be considered to select an ECC protection level that satisfies the write error rate threshold. As an example, considering Fig. 4, the ECC protection level $t = 6$ should be selected to satisfy the

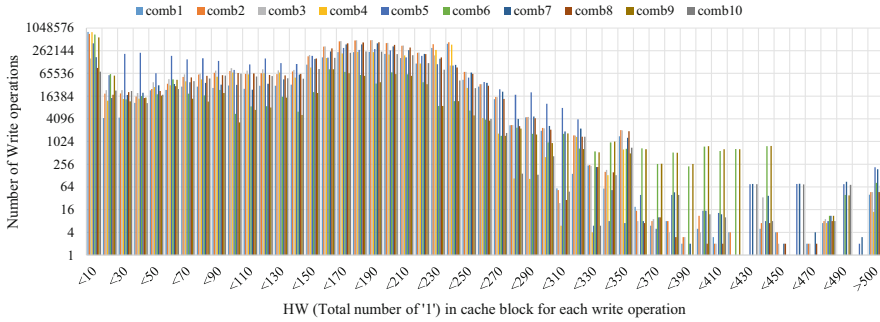


Fig. 5 The distribution of write operations based on Hamming Weight (HW) across a 4-MB 16-way shared L2 cache

write error rate threshold of 10^{-6} in the worst case where all of the bit locations in the target cache block experience toggle for the new incoming write request. However, satisfying a write request that all of its bit locations experience $0 \rightarrow 1$ bit transition is a very rare event. Therefore, using ECC with $t = 6$ correction capability is usually more than enough.

Accordingly, we investigated the distribution of HW across a 4-MByte 16-way set associative shared L2 cache during the execution of workloads. To this end, we ran combinations of benchmarks that are selected from SPEC CPU2006 [9] benchmark suite.¹ Figure 5 depicts the distribution of write requests' HWs across the shared L2 cache. Figure 5 illustrates that most of the write requests had less than 350 HW, while for the uniform full-protection ECC schemes we need to consider the worst-case HW (512 in 512 bit cache line size) for reliable write operations leading to significant under-utilization of resources.

One of the main concerns in utilizing uniform full-protection ECC configuration for the caches is the amount of energy consumption that is imposed to the system. Indeed, as illustrated in Figs. 4 and 5, while utilizing the full-protection configuration guarantees the reliable write operations, it imposes high energy consumption for most of the time that the write operations contain lower number of value 1 than the considered ECC-related threshold. For this reason, FlexRel exploits a non-uniform multi-protection level ECCs scheme to save energy. In addition, it improves the hardening scheme by deploying different write current levels introducing different STT-RAM write error rates.

¹The details of simulator configurations and workload combinations will be mentioned later in Sect. 5, in particular in Tables 3 and 4.

4.2 *FlexRel Organization*

In FlexRel, we divide a cache memory into several protection level zones. Unlike the previous studies that benefit only from different ECC codes to conduct multi-level protection scheme (e.g. [2, 3, 25]), in FlexRel we consider a combination of write current level and ECC protection level to satisfy a pre-determined write error rate threshold at each zone. The main strategy that is considered in FlexRel for cache partitioning is to assign the lowest possible write current for the zones that face high amounts of write operations to alleviate STT-RAM high write energy consumption and instead apply stronger ECC protection codes in these zones to satisfy the write error threshold. On the other hand, for the zones that experience low amounts of write operations we consider high write current with weaker ECC protection codes to alleviate the static energy consumption of ECC parts of the cache ways. For the sake of better intuition, in the following we explain the FlexRel approach considering a STT-RAM L2 cache memory architecture being 16-way set-associative and having a 64 Byte (512 bit) cache line as our case study example, while in general, FlexRel approach is applicable to all associative STT-RAM caches with any configuration and at any memory abstraction level.

The first step in designing a FlexRel-equipped cache is to classify the write requests of the cache based on the HW (which shows the vulnerability of write requests) and the portion of write requests. The partitioning in FlexRel applies at way granularity. Here, as an example, we consider four protection levels in our case study FlexRel-equipped cache. The straightforward approach to assign the cache ways to one of the four protection levels is the uniform assignment (in case of 16-way set associative cache it implies to assign four ways to each protection level). Considering the efficiency challenge that was mentioned for the uniform ECC protection technique, this straightforward assignment may face a considerable waste of resources. Therefore, our approach to enhance FlexRel has been to consider once again the write request patterns depicted in Fig. 5 to configure the portion of cache ways in each protection level. For this decision, we need to consider the cumulative amount of write requests in each protection level to provide enough space for them and keep the system performance as high as possible.

Thus, we partition the 16-way FlexRel-equipped cache to four protection levels as depicted in Fig. 6. According to the figure, we assign half of the cache ways to protection level 2 ($101 \leq HW \leq 250$) which should serve the most amount of write requests. Furthermore, protection level 1 ($HW \leq 100$) which should serve the second most amount of write requests benefits from a quarter of cache ways, while each of protection levels 3 and 4 only utilizes two ways to serve their low-intensive write requests. Figure 7 depicts the proposed FlexRel scheme for our case study example which includes four zones regarding protection levels.

After the way partitioning of FlexRel-equipped cache is completed, we should determine combinations of STT-RAM write current and ECC code to deliver a reliable write operation in each zone. To select these combinations, first we should consider a write error rate threshold to meet during the write operations in FlexRel-

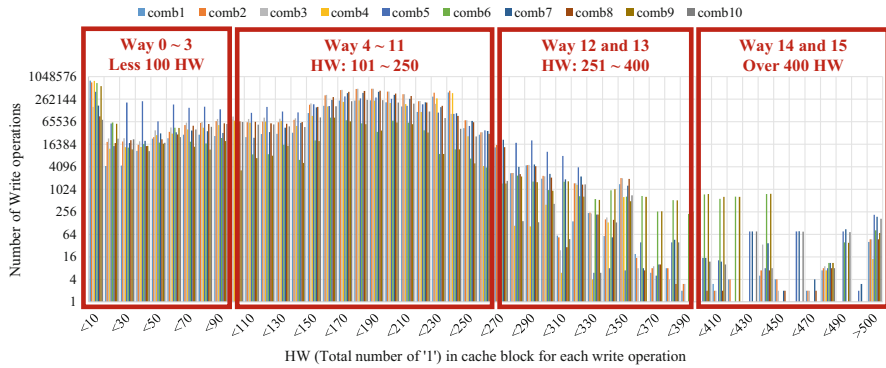


Fig. 6 Way partitioning across a 4-MB 16-way FlexRel-equipped shared L2 cache

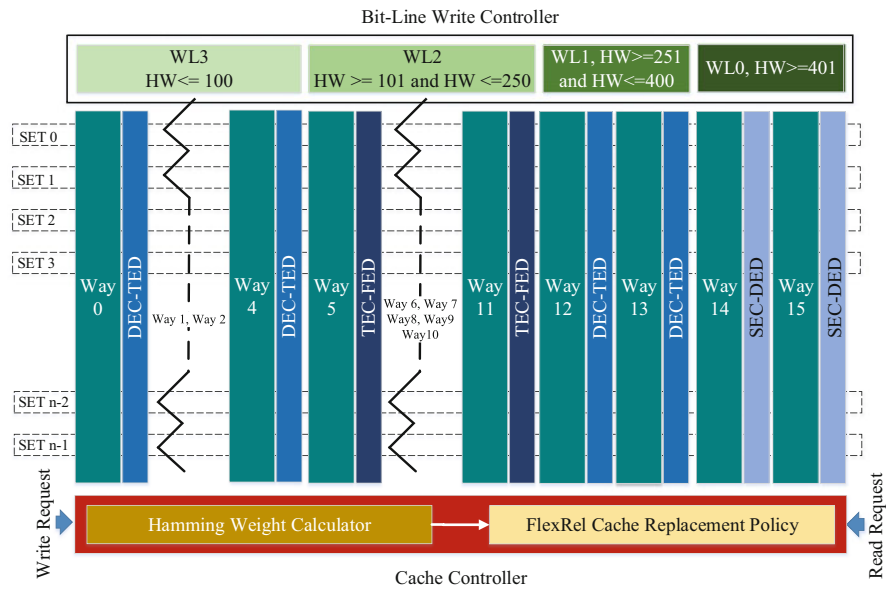


Fig. 7 A 16-way set associative FlexRel-equipped cache

equipped cache. To guarantee this threshold, FlexRel can either increase the write voltage and decrease the protection level of ECCs or vice versa. As an example, we consider 10^{-8} as write error rate threshold that should be met in all protection levels considering the write requests' HWs.

Consequently, Table 2 depicts a transducer map considered for FlexRel. As shown in the table, based on the configuration of write currents and ECC protection capabilities different dynamic energies and static powers are consumed in FlexRel-equipped cache ways. The last row of the table shows the amount of dynamic energy and static power of a uniform full-protection scheme with the same write error rate

Table 2 Transducer map for a 256 KB way embedded in a 4 MB 16-way set-associative FlexRel-equipped cache

Protection level	Write current	0 → 1 Error rate	ECC protection level (t)	Cache 0 → 1 block error rate		Way dynamic energy	Way static energy	
				Lowest HW	Highest HW		Read	Write
1	706.2 μ A	3×10^{-5}	2	None	4.4×10^{-9}	67.3 pJ	4.79 nJ	50.6 mW
2	666.9 μ A	8×10^{-5}	3	1.5×10^{-10}	6.4×10^{-9}	68.5 pJ	4.61 nJ	53.1 mW
3	778.2 μ A	5×10^{-6}	2	3.2×10^{-10}	1.3×10^{-9}	67.5 pJ	5.27 nJ	50.6 mW
4	935.0 μ A	1×10^{-7}	1	7.9×10^{-10}	1.3×10^{-9}	66.6 pJ	6.19 nJ	48.5 mW
OUP	586.0 μ A	6×10^{-4}	7	None	1.42×10^{-9}	73.3 pJ	4.37 nJ	60.6 mW

The energy consumption and Error rates depicted in this table are retrieved with the aid of NVSim [8] and HSPICE Monte Carlo[®] simulations

threshold mentioned for FlexRel-equipped cache. It is worthy of mentioning that in each protection level, FlexRel should provide the required facilities so that the highest HW write request in that level meets write error rate thresholds. The write currents at different protection levels in Table 2 reveal the mentioned strategy in FlexRel way partitioning. For example, comparing protection level 2 (with high amounts of write operations) with protection level 4 (with low amounts of write operations), for level 2 we considered lower write current ($666.9 \mu\text{A}$) with stronger ECC code ($t = 3$) to alleviate the high write energy consumption of STT-RAMs. On the other hand, for level 4, we consider higher write current ($935.0 \mu\text{A}$) with weaker ECC code ($t = 1$) to alleviate the ECC static energy consumption.

Now that the data storage architecture of FlexRel is explored, the final element that we should consider in the architecture of FlexRel-enabled cache is to design a mechanism to redirect the write request to their corresponding ways based on their HW during the execution. As depicted in Fig. 7, we developed a new replacement policy to perform this redirection.

Considering our case study example, Algorithm 1 depicts the traffic controller and replacement policy defined for FlexRel-equipped cache. This algorithm can be easily modified to apply to any other FlexRel-equipped cache with different configurations than the case study example. The FlexRel controller is responsible for calculating the HW of the incoming write request (Line 1). Then, if the write request is hit in the cache (Lines 3–10), FlexRel controller will verify the possibility of writing the new request to the hit block. To this end, FlexRel controller checks the HW of the new incoming request with the HW boundaries of the hit block's way protection level (Line 4). If the new incoming block satisfies the HW boundaries of the hit block's way protection level, FlexRel controller will satisfy the write request (Lines 5–7). Otherwise, the hit block will become invalid, and a cache miss signal will be triggered for the new incoming write request (Lines 7–10).

On the other hand, if the new incoming write request is missed in the cache (Line 11–24), based on the HW of this request that was calculated previously (Line 1), the FlexRel replacement policy should select the appropriate protection level for this request and evict a block from the protection level's assigned ways. It should be noted that FlexRel replacement policy uses LRU replacement policy in each protection level to evict the blocks (Lines 13, 16, 19, and 22).

5 Experimental Results

To explore the effectiveness of FlexRel in saving the energy consumption while meeting the reliability constraints we conducted a set of simulations. To this end, we used gem5 [4] simulating a quad-core ARM processor. The frequency of this processor is set to 1 GHz. The details of simulation configurations are summarized in Table 3. We extracted the dynamic and leakage power of STT-RAM cache ways from NVSim [8] with the aid of HSPICE. SPEC CPU2006 benchmark suites [9]

Algorithm 1: FlexRel controller and replacement policy for 16-way set associative cache

```

input : New incoming write request (WR)
output : The target block in cache for satisfying request
1 HW = calculate_HW(WR);
2 blk = Hit(WR);
  /* Check the availability of the requested block.          */
3 if blk then
  /* The requested blk is found in the cache (hit).          */
4   satisfy_request = check_way_boundary(blk→way,HW);
5   if satisfy_request then
6     | return(blk);
7   else
8     | Invalidate(blk);
9     | blk = null;
      /* After generating a miss signal for this request,
      FlexRel replacement policy will decide about the new
      location of this block.                                */
10  end
11 else
  /* The requested blk is not found in the cache (miss).    */
12  if HW ≤ 100 then
13    | blk = LRU(way0 ~ way3);
14    | return(blk);
15  else if 101 ≤ HW ≤ 250 then
16    | blk = LRU(way4 ~ way11);
17    | return(blk);
18  else if 251 ≤ HW ≤ 400 then
19    | blk = LRU(way12 and way13);
20    | return(blk);
21  else
22    | blk = LRU(way14 and way15);
23    | return(blk);
24  end
25 end

```

were used as the workloads in this study. Table 4 depicts the combination of benchmarks in each workload.

It should be noted that, for the sake of improving the accuracy of the experiments, all of the simulation results were retrieved after skipping the L2 cache warm-up phase. During the experiments, we implemented and compared the following schemes:

- **Optimized Uniform Protection**—In this scheme, L2 cache ways were protected with uniform 7EC8ED BCH code with low write current mentioned in the last row of Table 2 (OUP). This uniform protection satisfied the considered block write error rate considered in this study (i.e., 10^{-8}).

Table 3 Experimental setup for gem5 simulations

Parameter	Value
ISA	ARMv7-A
No. of cores	4
L1 \$ size, assoc.	32 KB, 4
L2 \$ size, assoc.	4 MB, 16
Cache configuration	L1 (Private) L2 (Shared, FlexRel-enabled)
Cache block size	64B
Cache warm-up instructions	100 million
No. simulated instructions	100 million

Table 4 Workload combinations

Combination	Core 0	Core 1	Core 2	Core 3
Comb1	perlbench	bzip2	mcf	soplex
Comb2	perlbench	bzip2	omnetpp	xalancbmk
Comb3	perlbench	mcf	omnetpp	xalancbmk
Comb4	bzip2	mcf	soplex	xalancbmk
Comb5	gcc	bwaves	mcf	cactusADM
Comb6	namd	dealII	soplex	calculix
Comb7	perlbench	gcc	mcf	namd
Comb8	perlbench	namd	soplex	xalancbmk
Comb9	bwaves	dealII	namd	calculix
Comb10	gcc	bwaves	soplex	xalancbmk

- **FlexRel**—In this scheme, L2 cache ways were protected with variable strength ECCs and write currents mentioned in Table 2, according to the discussed cache structure in FlexRel.

Figure 8 depicts the normalized energy consumption of FlexRel-equipped shared L2 cache compared with the optimized uniform scheme. We evaluated the efficiency of FlexRel in terms of energy consumption from three perspectives, i.e., dynamic energy consumption, static energy consumption, and overall energy consumption. According to Fig. 8, FlexRel increased the dynamic energy consumption of ways by up to 19% in *comb1* which intensively used the protection levels that consume high write energy consumption. It is worth noting that since the optimized uniform scheme utilizes the least write current, the FlexRel scheme will impose more dynamic energy. On the other hand, since FlexRel utilizes low protection ECCs in comparison with optimized uniform ECC scheme, it significantly improves the static energy consumption in almost all of the combinations. Indeed, while the leakage power of the ways in the FlexRel-equipped cache was significantly lower than the optimized uniform scheme, the high-performance overhead experienced in *comb2* (see Fig. 9) led to the same static energy consumption in both schemes, and further increased the overall energy consumption by 3%.

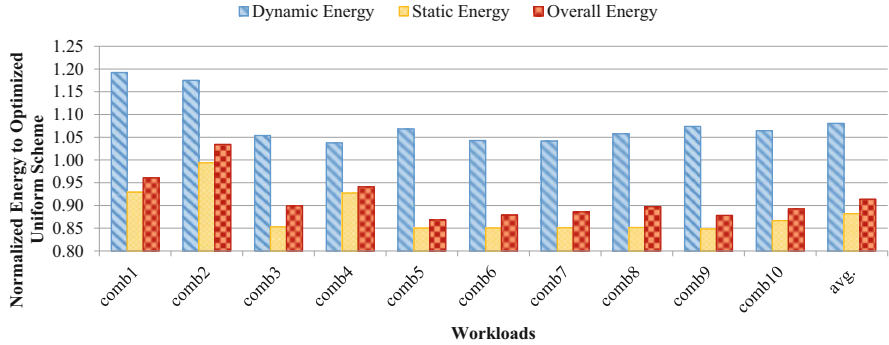


Fig. 8 Energy consumption of the ways of 4 MB shared FlexRel-equipped L2 cache normalized to optimized uniform scheme

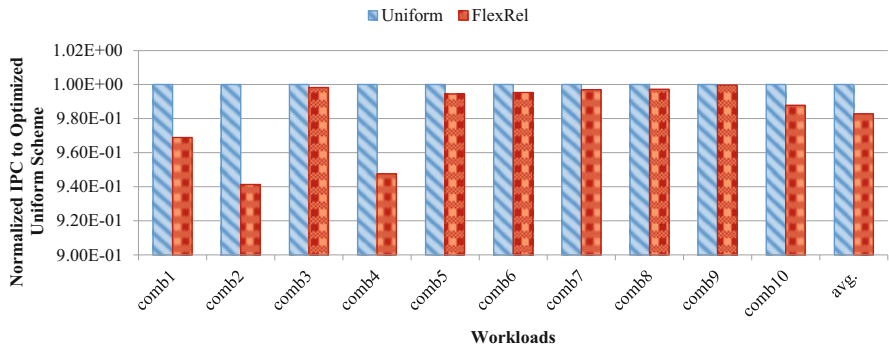


Fig. 9 The Instruction per Clock (IPC) of a system includes a 4 MB shared FlexRel-equipped L2 cache normalized to the IPC of a system that includes optimized uniform scheme in its L2 cache

In general, since static energy consumption is the main contributor in the energy consumption of the cache ways, FlexRel achieves a considerable improvement in the overall energy consumption (static + dynamic) of ways in the L2 cache. On average, while FlexRel increases the dynamic energy consumption of the ways by 8%, it saves the static energy consumption and overall energy consumption of L2 cache ways by 12% and 9%, respectively. The calculated amount of saved static power in each FlexRel-equipped cache’s way is about 70.6 mW. With this amount of power, we will be able to supply the static power of more than seven 32 KB L1 caches (like the ones considered in this study) each consuming 8.9 mW static power.

Figure 9 shows the Instruction per Clock (IPC) of a system using a 4 MB shared FlexRel-equipped L2 cache normalized to the IPC of a system that incorporates the optimized uniform scheme in its L2 cache. Since FlexRel modifies the replacement policy of the L2 cache to redirect the write requests to their corresponding protection level’s ways it may impose some performance penalty to the system in the situations when (1) the target protection level’s ways face intensive write requests from

Table 5 The amount of way area saving in FlexRel at different protection levels compared with optimized uniform scheme

Protection Level	ECC protection level (t)	ECC check bits	% of area improvement
1	2	21	8.5
2	3	31	6.6
3	2	21	8.5
4	1	11	10.2
% of average area improvement	7.9		

different addresses or (2) we should evict a hit block because the total number of “1”s in this block forces FlexRel to change its location.

According to Fig. 9, in the worst case scenario, i.e., *comb2* that the miss penalty in FlexRel approach is considerable, the IPC of the system is degraded by 6%. This result illustrates the importance of cache partitioning in the performance of the system. In other words, while the considered partitioning provides reasonable performance for most of the workloads, for *comb1*, *comb2*, and *comb4*, the considered partitioning utilized by Algorithm 1 led to high-performance overheads for these workloads. Previously we mentioned that these performance overheads even affect the energy efficiency of FlexRel for *comb1*, *comb2*, and *comb4* workloads. On average, FlexRel decreases the IPC of the system by a negligible 1.7%.

Finally, w.r.t. area, ECC check bits assigned at each protection level are the main contributor. Accordingly, Table 5 reports ECC check-bits, and the area saving at different protection levels of FlexRel compared with the optimized uniform protection scheme. In general, considering a 16-way set associative L2 cache, the flexible scheme provided by FlexRel could save the occupied ways’ area by about 7.9%, on average.

6 Conclusions and Future Work

In this chapter, we proposed FlexRel, an energy-aware reliability improvement architectural scheme for STT-RAM cache memories. FlexRel is an architecture-to-application cross-layer approach that considers a memory architecture provided with Error Correction Codes (ECCs) and a custom current regulator for the various cache ways and conducts a trade-off between reliability and energy consumption. The FlexRel cache controller dynamically profiles the number of $0 \rightarrow 1$ bit transitions of each write operation and, based on this critical parameter it selects the most-suitable cache way and current level to deliver the necessary reliability level (in terms of occurred write errors) while minimizing the energy consumption.

The results of evaluating FlexRel show that, while the scheme satisfies the reliability requirements, it delivers up to 13.2% energy saving and up to 10.2% cache ways’ area saving, compared with the most efficient uniform protection scheme. The

performance overhead imposed by FlexRel to the system due to the modifications of cache ways' access traffics is 1.7%, on average.

As future work, we will further improve and refine the FlexRel in two aspects. First, we will focus on the proposed replacement policy to improve the performance of the system for workloads that face significant block evictions due to HW boundary violations. To minimize the performance overhead, we will attempt to devise a dynamic cache partitioning scheme capable of changing the configuration of the FlexRel-equipped cache when considerable performance degradation is observed.

References

1. Ahn, J., Choi, K.: Selectively protecting error-correcting code for area-efficient and reliable STT-RAM caches. In: Proceedings of the Asia and South Pacific Design Automation Conference (ASP-DAC), pp. 285–290 (2013)
2. Azad, Z., Farbeh, H., Monazzah, A.M.H., Miremadi, S.G.: An efficient protection technique for last level STT-RAM caches in multi-core processors. *IEEE Trans. Parallel Distr. Syst.* **28**(6), 1564–1577 (2017)
3. Azad, Z., Farbeh, H., Monazzah, A.M.H.: ORIENT: Organized interleaved ECCs for new STT-MRAM caches. In: Proceedings of the Design, Automation Test in Europe Conference and Exhibition (DATE), pp. 1187–1190 (2018)
4. Binkert, N., Beckmann, B., Black, G., Reinhardt, S.K., Saidi, A., Basu, A., Hestness, J., Hower, D.R., Krishna, T., Sardashti, S., et al.: The gem5 simulator. *ACM SIGARCH Comput. Archit. News* **39**(2), 1–7 (2011)
5. Chen, Y., Li, H., Wang, X., Zhu, W., Xu, W., Zhang, T.: A nondestructive self-reference scheme for spin-transfer torque random access memory (STT-RAM). In: Proceedings of the Design, Automation Test in Europe Conference and Exhibition (DATE), pages 148–153, 2010.
6. Cheshmikhani, E., Farbeh, H., Asadi, H.: ROBIN: Incremental oblique interleaved ECC for reliability improvement in STT-MRAM caches. In: Proceedings of the Asia and South Pacific Design Automation Conference (ASP-DAC), pp. 173–178 (2019)
7. Cheshmikhani, E., Farbeh, H., Miremadi, S.G., Asadi, H.: TA-LRW: A replacement policy for error rate reduction in STT-MRAM caches. *IEEE Trans. Comput.* **68**(3), 455–470 (2019)
8. Dong, X., Xu, C., Xie, Y., Jouppi, N.: NVSim: A circuit-level performance, energy, and area model for emerging nonvolatile memory. *IEEE Trans. Comput. Aided Design Integr. Circuits Syst.* **31**(7), 994–1007 (2012)
9. Henning, J.L.: SPEC CPU2006 benchmark descriptions. *ACM SIGARCH Comput. Archit. News* **34**(4), 1–17 (2006)
10. Hosomi, M., Yamagishi, H., Yamamoto, T., Bessho, K., Higo, Y., Yamane, K., Yamada, H., Shoji, M., Hachino, H., Fukumoto, C., Nagao, H., Kano, H.: A novel nonvolatile memory with spin torque transfer magnetization switching: Spin-ram. In: Proceedings of the IEEE International Electron Devices Meeting (IEDM), pp. 459–462 (2005)
11. Monazzah, A.M.H., Farbeh, H., Miremadi, S.G.: LER: Least-error-rate replacement algorithm for emerging STT-RAM caches. *IEEE Trans. Device Mat. Rel.* **16**(2), 220–226 (2016)
12. Jin, Y., Shihab, M., Jung, M.: Area power and latency considerations of STT-MRAM to substitute for main memory. In: Proceedings of the Symposium on Computer Architecture (ISCA) (2014)

13. Kim, Y., Gupta, S., Park, S., Panagopoulos, G., Roy, K.: Write-optimized reliable design of STT MRAM. In: Proceedings of the International Symposium on Low Power Electronics and Design (ISLPED), ISLPED '12, pp. 3–8 (2012)
14. Kim, J., Chen, A., Behin-Aein, B., Kumar, S., Wang, J., Kim, C.: A technology-agnostic MTJ SPICE model with user-defined dimensions for STT-MRAM scalability studies. In: Proceedings of the Custom Integrated Circuits Conference (CICC), pp. 1–4 (2015)
15. Lin, S., Costello, D.: Error Control Coding: Fundamentals and Applications. Prentice Hall, Upper Saddle River (1983)
16. Marins de Castro, M., Sousa, R.C., Bandiera, S., Ducruet, C., Chavent, A., Auffret, S., Papisoi, C., Prejbeanu, I.L., Portemont, C., Vila, L., et al.: Precessional spin-transfer switching in a magnetic tunnel junction with a synthetic antiferromagnetic perpendicular polarizer. *J. Appl. Phys.* **111**(7), 07C912 (2012)
17. Monazzah, A.M.H., Shoushtari, M., Miremadi, S.G., Rahmani, A.M., Dutt, N.: QuARK: Quality-configurable approximate STT-MRAM cache by fine-grained tuning of reliability-energy knobs. In: Proceedings of the International Symposium on Low Power Electronics and Design (ISLPED), pp. 1–6 (2017)
18. Oboril, F., Shirvanian, A., Tahoori, M.: Fault tolerant approximate computing using emerging non-volatile spintronic memories. In: Proceedings of the VLSI Test Symposium (VTS), pp. 1–1 (2016)
19. Rahmani, A.M., Liljeberg, P., Hemani, A., Jantsch, A., Tenhunen, H.: *The Dark Side of Silicon*, 1st edn. Springer, Berlin (2016)
20. Ranjan, A., Venkataramani, S., Fong, X., Roy, K., Raghunathan, A.: Approximate storage for energy efficient spintronic memories. In: Proceedings of the Design Automation Conference (DAC), pp. 1–6 (2015)
21. Shoushtari, M., Rahmani, A.M., Dutt, N.: Quality-configurable memory hierarchy through approximation: Special session. In: Proceedings of the International Conference on Compilers, Architectures and Synthesis for Embedded Systems (CASES), pp. 1–2 (2017)
22. Sun, H., Liu, C., Zheng, N., Min, T., Zhang, T.: Design techniques to improve the device write margin for MRAM-based cache memory. In: Proceedings of the Great Lakes Symposium on VLSI (GLSVLSI), pp. 97–102 (2011)
23. Swami, S., Mohanram, K.: Reliable nonvolatile memories: techniques and measures. *IEEE Design Test* **34**(3), 31–41 (2017)
24. Teimoori, M.T., Hanif, M.A., Ejlali, A., Shafique, M.: Adam: adaptive approximation management for the non-volatile memory hierarchies. In: Proceedings of the Design, Automation Test in Europe Conference and Exhibition (DATE), pp. 785–790 (2018)
25. Wang, X., Mao, M., Eken, E., Wen, W., Li, H., Chen, Y.: Sliding basket: An adaptive ECC scheme for runtime write failure suppression of STT-RAM cache. In: Proceedings of the Design, Automation Test in Europe Conference and Exhibition (DATE), pp. 762–767 (2016)
26. Wen, W., Mao, M., Zhu, X., Kang, S., Wang, D., Chen, Y.: CD-ECC: Content-dependent error correction codes for combating asymmetric nonvolatile memory operation errors. In: Proceedings of the International Conference on Computer-Aided Design (ICCAD), pp. 1–8. IEEE, New York (2013)
27. Xu, W., Sun, H., Wang, X., Chen, Y., Zhang, T.: Design of Last-level on-chip cache using spin-torque transfer RAM (STT RAM). *IEEE Trans. Very Large Scale Integr. Syst.* **19**(3), 483–493 (2011)
28. Zhang, Y., Wang, X., Li, Y., Jones, A.K., Chen, Y.: Asymmetry of MTJ switching and its implication to STT-RAM designs. In: Proceedings of the Design, Automation Test in Europe Conference and Exhibition (DATE), pp. 1313–1318 (2012)
29. Zhang, Y., Zhang, L., Chen, Y.: MLC STT-RAM design considering probabilistic and asymmetric MTJ switching. In: Proceedings on International Symposium on Circuits and Systems (ISCAS), pp. 113–116 (2013)
30. Zhang, Y., Li, Y., Sun, Z., Li, H., Chen, Y., Jones, A.K.: Read performance: The newest barrier in scaled STT-RAM. *IEEE Trans. Very Large Scale Integr. Syst.* **23**(6), 1170–1174 (2015)

Open Access This chapter is licensed under the terms of the Creative Commons Attribution 4.0 International License (<http://creativecommons.org/licenses/by/4.0/>), which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons license and indicate if changes were made.

The images or other third party material in this chapter are included in the chapter's Creative Commons license, unless indicated otherwise in a credit line to the material. If material is not included in the chapter's Creative Commons license and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder.

