VAET-STT: A Variation Aware Estimator Tool for STT-MRAM based Memories

Sarath Mohanachandran Nair, Rajendra Bishnoi, Mohammad Saber Golanbari, Fabian Oboril and Mehdi B. Tahoori
Chair of Dependable Nano Computing (CDNC), Karlsruhe Institute of Technology (KIT), Karlsruhe, Germany
Email: {sarath.nair, golanbari, rajendra.bishnoi, fabian.oboril, mehdi.tahoori}@kit.edu

***Abstract*—Spin Transfer Torque Magnetic Random Access Memory (STT-MRAM) is a promising candidate to replace CMOS based on-chip memories due to its advantages such as non-volatility, high density and scalability. However, its stochastic switching and higher sensitivity to process variation compared to CMOS memories can significantly affect its performance, energy and reliability. Although a few works exist which analyze the impact of process variation at the bit-cell level, such analysis at the system level is missing. We have bridged this gap in our work. Specifically, we quantify the effect of stochasticity and process variations from the cell-level to the overall memory system and perform a *variation-aware* memory configuration optimization for energy or performance while meeting reliability constraints. Our system-level variation-aware framework has been built on top of the well-known NVSim engine. The results show that our framework can provide more realistic margins and the optimized variation-aware memory configuration could be significantly different from the conventional framework.**

I. Introduction

Conventional CMOS memories (e.g., SRAM and DRAM) are facing severe challenges in advanced technology nodes due to the increased leakage power [1]. As an emerging technology, *Spin Transfer Torque Magnetic Random Access memory* (STT-MRAM) is well known for its reduced static power because of its non-volatility [2]. It has several distinctive advantages such as scalability, high endurance, long retention and fast accesses, which gives it an edge over other emerging non-volatile memory technologies such as *Phase-change*, *Ferro-electric* and *Resistive* memories. Moreover, STT-MRAM has many favorable qualities of existing memory technologies such as speed of SRAM, density of DRAM and non-volatility of Flash. Therefore, with all its advantages, STT-MRAM has the potential to become a *universal* memory since it can potentially fit into every level of the memory hierarchy [3].

The functionality of STT-MRAM, in addition to variations in CMOS components, is also influenced by variations in magnetic fabrication processes. The manufacturing imperfections in the magnetic devices disturb the magnetic property of the cell such as switching characteristics, resistance differences, etc. Furthermore, the impact of process variation on magnetic devices are exaggerated due to its nature of stochastic switching. As a consequence, the bit-cells have non-normal latency distributions with a long tail [4], and the same is propagated all the way upto the level of memory architecture, severely affecting the required margins. Therefore, the effect of process variation as well as stochastic switching has to be taken into account for a realistic estimation of latency and energy of STT-MRAM based memories.

STT-MRAM is affected by several failure mechanisms while reading from and writing to the bit-cell. A read decision failure can happen due to a wrong decision by the sensing circuitry. A write failure can occur when the write current is lower than the switching threshold current and hence fails to change the value stored in a bit-cell. The read and write failures are typically quantified by the Read Error Rate (RER) and Write Error Rate (WER). The reliability requirements based on target RER/WER should be taken into account while fixing the read/write periods, respectively. This in turn affects the overall read/write latencies and energies of the memory system. Hence it is important to consider the target read/write error rates while finding an optimal memory configuration.

In order to support early-stage memory design space exploration, an estimator tool for STT-MRAM technology is needed for a system-level evaluation before (or without) fabricating the actual chip. NVSim [5] is the most popular among existing estimator tools. However, this tool does not consider variability in its analysis. There is an improved version of NVSim, namely NVSim-VX^S [6] which considers the influence of process variation using statistical compact models. Nevertheless, this tool is limited only to cell-level modeling for write operation. A complete variation-aware estimation and optimization tool for energy, performance and reliability is still not available for system-level evaluation of STT-MRAM based memories.

In this paper, we develop a *Variation Aware Estimator Tool for STT-MRAM based Memories* (VAET-STT), an early stage design exploration tool for STT-MRAM, which considers process variations, stochasticity and reliability requirements in its analysis and memory configuration optimization. In our proposed tool, we employ a hybrid analytical and *Monte-Carlo* simulation based hierarchical approach for the generation of the final memory design parameters such as access latencies, energies and error rates.

Overall, our contributions in this paper are as follows:

- We develop a statistical model for the entire memory architecture including the bit-cell array and the peripheral components.
- The latency and energy values are estimated based on the required margins to satisfy the reliability (overall system-level failure rate) requirements provided by the user.
- We perform a design space exploration that generates an optimal memory configuration satisfying the input reliability requirements.

The results show that for a target error rate of 1×10^{-18}, the read latency is $2.3\times$ larger while the read energy is $2.5\times$ more than the nominal values. For write operation, the latency is $32.5\times$ greater whereas the energy is $31.2\times$ more than the nominal values.

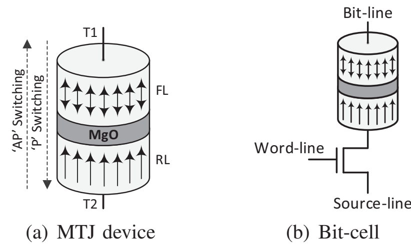
The rest of this paper is organized as follows. In Section II, the basics of STT-MRAM is introduced and related work is discussed. Section III explains the methodology employed to develop the statistical framework, followed by the results which are demonstrated in Section IV. Finally, Section V concludes the paper.

II. Background

A. STT-MRAM technology

In STT-MRAM, the data is stored as resistance states using a *Magnetic Tunneling Junction* (MTJ). An MTJ cell, as shown in Figure 1(a), comprises two ferromagnetic layers separated by a thin oxide layer (e.g. MgO). The magnetic orientation of one of the layers is fixed, known as the *Reference Layer* (RL), whereas for the other one, which is known as the *Free*

978-3-9815370-8-6/17/$31.00 ©2017 IEEE



(a) MTJ device (b) Bit-cell

Fig. 1: STT-MRAM storing device and its bit-cell architecture.

Layer (FL), the magnetization can be freely rotated. When the magnetic orientation of FL is parallel (anti-parallel) to that of RL, the MTJ cell is in low (high) resistance state. The low resistance state is typically denoted by R_P and the high resistance state by R_{AP}. The magnetic orientations can be switched from one state to the other by passing a spin polarized current in the proper direction. On the other hand, to read a value, a low unidirectional current has to flow through the MTJ cell which is sensed using a sense amplifier. The accuracy of sensing depends on the resistance difference between the low (R_P) and high (R_{AP}) resistance quantified by the Tunneling Magnetoresistance (TMR) ratio, given by TMR = (R_{AP} - R_P)/R_P.

A typical STT-MRAM bit-cell architecture is shown in Fig. 1(b). It consists of one MTJ cell and one access transistor. The access transistor is CMOS based and is used to select the MTJ cell for memory operations. A typical STT-MRAM memory system consists of banks, mats and subarrays. A bank consists of several mats, whereas a mat is composed of several subarrays. The subarray is the basic building block of the memory system. It consists of the STT-MRAM bit-cell array and the peripheral components as shown in Fig. 2 [7]. The peripheral components are CMOS based and include row decoders, column decoders, sense amplifiers, column multiplexers, wordline drivers and output drivers.

B. Variation, stochasticity, and failures in STT-MRAM

The fabrication of an STT-MRAM cell requires two different fabrication processes, a *magnetic-process* and a *CMOS-process*. Variations in either process affect the characteristics of the cell. Imperfections in the magnetic manufacturing processes cause variations mainly in tunneling oxide thickness and cross-sectional area. This in turn causes variations in the switching threshold current, resistance values and the TMR ratio [8, 9], significantly affecting the cell-level delay and energy values for both read and write operations. Typically, a large timing margin is used to account for such process variations.

Moreover, the magnetic switching of MTJ is stochastic in nature due to random thermal fluctuations. The write probability of a bit-cell can be modeled by the following equation [10]:

$$WP_{\text{bit}}(t) = \exp\left[\frac{-\pi^2(I-1)\triangle}{4(Ie^{C(I-1)t}-1)}\right],\ I = \frac{I_w}{I_c}$$

where Δ is the thermal stability factor, t is the write period, I_w is the write current, I_c is the critical current and C is a constant determined by the material type and technology parameters. The Write Error rate is given by

$$WER_{bit}(t) = 1 - WP_{bit}(t).$$

For the read operation, a read decision failure, where a wrong decision is made by the sensing circuitry, can extend the timing margin for a given read error rate.

The STT-MRAM bit-cell is also influenced by variations in the CMOS fabrication process which causes variations in the device threshold voltage (V_{th}) primarily due to Random Dopant Fluctuations (RDF) [11, 12]. The standard deviation of threshold voltage (σV_{th}) due to RDF is given by the following equation [13]:

$$\sigma V_{th} \propto \frac{1}{\sqrt{WL}}.$$

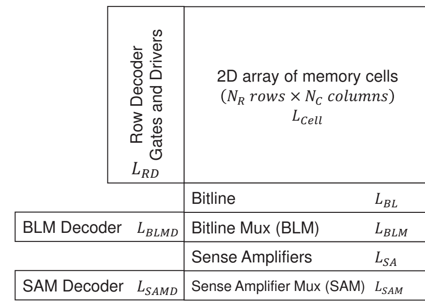


Fig. 2: High-level composition of a subarray with individual components and their latencies [5].

For STT-MRAM, this results in variations in the access transistor causing read/write current fluctuations and disturbance during the cell active durations. The variation in V_{th} also affects the on-current of the peripheral components causing variations in the read/write current to the bit-cell. Hence the combined effect of the MTJ and CMOS variations can significantly influence the overall memory design parameters, such as total access latency and energy.

C. Related work

There are some works considering variability in STT-MRAM [4, 8, 14–18]. The work in [4] proposes an adaptive write current boosting technique to fix the worst-case write latency due to process variations. The work in [8] considers the variations in the bit-cell parameters such as the tunneling oxide thickness and cross-sectional area to come up with an optimal memory configuration, access transistor size and read drive circuitry to maximize the yield. In [14], the authors propose a methodology for an STT-MRAM cell reliability prediction under the joint impact of fabrication and aging-induced process variability, supply voltage and temperature variations. The work described in [15] implements a unified device-circuit-architecture co-design strategy to optimize the yield of STT-MRAM. The memory array is optimized by tuning the bit-cell parameters together with different ECC schemes. The work in [16] considers variations only in the read circuitry and analyzes the impact of variability on sensing schemes. The work in [17] considers the impact of process variations on the stochastic switching behavior of MTJ and proposes probabilistic design techniques to enhance the performance while maintaining a low write failure probability. In [18], the authors have modeled the combined effect of process variation and stochastic write behavior on the write latency of STT-MRAM bit-cell.

The most popular tool for a memory system-level analysis for emerging non-volatile memories is NVSim [5], which is based on the well known CACTI [7]. NVSim finds an optimal memory configuration based on an input design metric and reports out the latencies and energies of the overall memory system. Recently, an improved variation-aware version of NVSim namely NVSim-VX^S was released. However, the variation-aware statistical model in NVSim-VX^S is limited to bit-cell level write operation and does not consider the read operation. In NVSim-VX^S, the bit-cell WER is reported for '0'$\rightarrow$'1' and '1'$\rightarrow$'0' transitions. It also includes a simple extension of the cell-level WER to the block level based on a user input switching pattern. However, the user input switching pattern does not reflect the switching pattern of the optimal

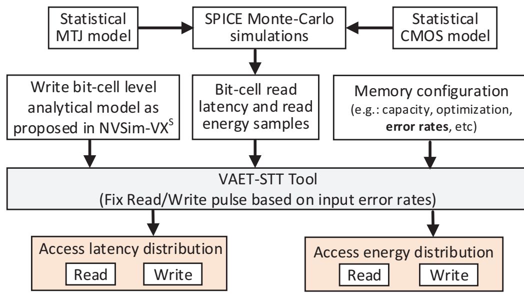


Fig. 3: Proposed VAET-STT tool flow.

memory configuration and hence the block-level extension in NVSim-VX^S does not accurately reflect its energies/error rates.

None of the previous works consider the combined impact of variability and stochasticity at the bit-cell and process variations in the peripherals for a system-level analysis. We have bridged this gap by developing tool to perform system-level variation-aware analysis while meeting reliability constraints.

III. Variation-aware analysis Framework

The VAET-STT tool flow is shown in Fig. 3. The overall memory latencies/energies are composed of latency/energy of the bit-cell and those of the peripheral components. The latency/energy distributions for the peripheral components are obtained from Monte-Carlo simulations. The bit-cell level samples obtained from Monte-Carlo SPICE simulations are used to obtain the bit-cell read latency/energy distributions. The analytical model proposed in NVSim-VX^S is used to obtain the bit-cell write latency/energy distributions. Then, at the system-level, we combine these distributions using a Monte-Carlo approach to obtain the overall latencies and energies.

A. Overview

At the bit-cell and component levels, the delay and energy distributions follow known distributions for reasonable accuracy [6, 19]. Accesses to a memory array does not read/write a single cell but a line which has N bit-cells. The effective cell latency is then the maximum of these N latencies which is shown to follow a Generalized Extreme Value (GEV) distribution [20]. Hence, at the system level, when parameters following diverse and different distributions (e.g., normal, log-normal, GEV) are combined, they cannot simply be fitted to known distributions. For instance, in [19] the cell latency GEV distributions have been approximated with a normal distribution to facilitate an equation based analytical approach. However, such approximations can compromise the accuracy of the results. Hence at the system level, a sampling approach, such as Monte-Carlo can be used to get the overall latency and energy distributions. This hybrid and hierarchical approach allows us to have a good tradeoff of fast run-time, since cell-level samples are taken from their distributions, while achieving high accuracy. Hence by combining the bit-cell and peripheral distributions for different operating temperatures and technology nodes using Monte-Carlo, the system-level results can be obtained to reflect the impact of temperature or technology node on the variation of overall latencies and energies.

B. Component-level Variation Analysis

The first step in latency calculation is to get the latencies at the bit-cell level for which we have used fitted distributions. For the write operation, the distribution fitting is already done in NVSim-VX^S, hence we have used this framework and integrated it into our tool. NVSim-VX^S provides the distribution

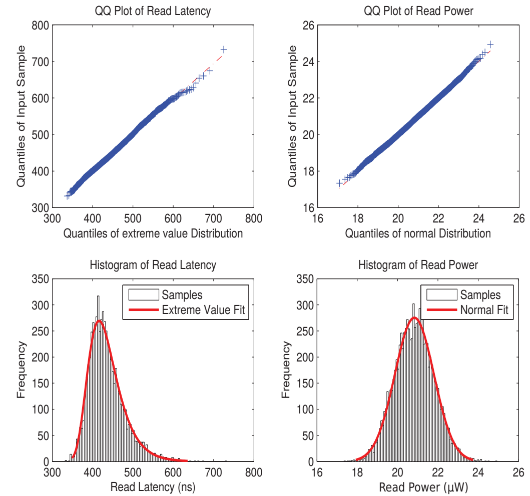


Fig. 4: QQ and Density Plots for bit-cell read latency and read power.

parameters (μ and σ) of the log-normal bit-cell switching time. For the read operation, we have generated samples for the read latency by running Monte-Carlo SPICE simulations on the 1T-1MTJ bit-cell in the presence of variations. We have employed the TSMC general purpose SPICE models for the CMOS access transistor and the model from [21] for the MTJ cell. The results show that the read delay follows an Extreme Value Distribution as shown by the QQ plot and probability density plot in Fig. 4. The parameters of the Extreme Value Distribution (μ and β) are obtained from the bit-cell level samples using the following equations:

$$\mu = m - \gamma\beta, \qquad \beta = \frac{sd \times \sqrt{6}}{\pi},$$

where m is the mean and sd is the standard deviation of the samples respectively and $\gamma = 0.5772$ is the Euler-Mascheroni constant.

Similar to the latency calculation, the normally distributed bit-cell level write dynamic energy parameters (μ and σ) are obtained from NVSim-VX^S. For the read operation, the bit-cell level read power samples are obtained from Monte-Carlo analysis of a 1T-1MTJ bit-cell in the presence of variations. The read power follows a normal distribution as shown by the QQ plot and probability density plot in Fig. 4. These generated samples are used to calculate the parameters (μ and σ) of the normal distribution.

The process variations in the peripheral components, being CMOS based, can be lumped into variations in the threshold voltage (V_{th}). This approach has already been used in process variation aware SRAM tools like VARIUS [19]. The V_{th} variations in turn affect the on-current and mobility which causes variations in the latency and dynamic energy of individual peripheral component. Under a normal V_{th} assumption, the on-current and mobility variations are obtained from SPICE simulations by performing a sensitivity analysis. We then run Monte-Carlo and obtain samples for the peripheral component latencies and energies. The samples are then used to obtain the distribution parameters (μ and σ) of the normally distributed peripheral latencies and energies.

C. System-Level Variation Analysis

We have used a hierarchical and hybrid approach of analytical curve fitting and Monte-Carlo to obtain the overall memory latency and energy distribution samples. This method involves fitting each of the constituent latency samples to known distributions and then combining these distributions using a Monte-Carlo method.

1) Latency Calculation: We follow a two step process to obtain the latencies at the system-level.

- The latency distributions of the individual components in Fig. 2 are extracted by Monte-Carlo and fitted to known distributions such as Normal, Log-normal, and GEV.
- The system-level Monte-Carlo uses the distributions from the previous step to calculate the total read and write latencies.

The statistical latency distribution of the STT memory cells is extracted by running SPICE simulation and is fitted to a GEV distribution as explained in Section III-B. The latency distributions of CMOS based peripheral components shown in Fig. 2 are extracted internally by varying the transistor parameters based on Pelgrom model [13, 22] for local variations in smaller Monte-Carlo loops. It is worth mentioning that the parameters should be assigned randomly to each and every transistor in the CMOS circuits. The peripheral latencies extracted from Monte-Carlo are fitted to known distributions which is used in the system-level Monte-Carlo.

In the system-level Monte-Carlo loop, samples are drawn from the extracted distributions in the previous step, and the total latency is calculated by combining the component latencies (the samples) in an iterative manner. The combining process is either by taking the maximum of the latency samples or adding the latencies. When a number of instances, say M, of an individual component C is accessed in parallel, we take M samples from the corresponding distribution ($\{l_{C,1}, ..., l_{C,M}\}$) and obtain the maximum of these M latencies to generate a single sample of latency. For example, if a memory array has N_C columns (see Fig. 2), a single sample of the memory array latency is obtained by taking the maximum of N_C randomly picked samples from the bit-cell latency distribution. Therefore, the latency of memory array can be explained as:

$$l_{cell,i} \sim Distribution\{Cell\ Latency\},$$
$$L_{cell} = \text{Max}\{l_{cell,1}, ..., l_{cell,N_C}\} \sim GEV,$$

where $l_{cell,i}$ is a random sample taken from the distribution of memory cell latency. The number of columns in a typical memory array is large, hence, L_{cell} can be explained by a GEV distribution, because the maximum of a number of *independent and identically distributed* (i.i.d.) random variables is taken. Similarly, the random variables L_{RD}, L_{BL}, L_{BLM}, L_{BLMD}, L_{SA}, L_{SAM}, and L_{SAMD} follow GEV distributions, because they are the maximums of i.i.d. random variables from the distributions of Row Decoder, Bitline, Bitline Mux, Bitline Mux Decoder, Sense Amplifier, Sense Amplifier Mux, and Sense Amplifier Mux Decoder, respectively. Furthermore, since all the decoders are invoked in parallel, it is possible to simplify the latencies of all decoders into a single *decoder latency* random variable:

$$L_D = \text{Max}\{L_{RD}, L_{BLMD}, L_{SAMD}\}.$$

When the elements are not accessed in parallel, the latency is the summation of the latencies of individual components. Thus the total read latency of the memory subarray is the summation of the decoder latency (L_D), cell read latency (L_{cell}), bitline latency (L_{BL}), and sense-amplifier latency (L_{SA}) as given below:

$$L = L_D + L_{cell} + L_{BL} + L_{SA}.$$

The latency calculation flow described above is applicable to read operation. The write latency can also be calculated in a similar manner. We omit the details due to space constraints.

2) Error Rate Calculation: The STT-MRAM bit-cell is subjected to various failure mechanisms as explained in Section II-B. The bit-cell RER/WER is calculated based on a given read/write pulse. For STT-MRAM, the WER for '0'→'1' switching is different from that of '1'→'0' switching. In our analysis, we have assumed worst case switching of the cells, since the switching pattern of the memory array is not known at design time. Since the bit-cell latencies are fitted to known distributions, the Cumulative Distribution Function (CDF) of the respective distributions is used to calculate the RER/WER. If T is a random variable representing the bit-cell read/write latencies and t_P is the read/write period, then the bit-cell RER/WER e is given by:

$$e = P(T > t_P) = 1 - F_T(t_P), \quad (1)$$

where $F_T(t)$ is the CDF of T.

The bit-cell errors can be assumed to be independent, and hence the block-level error rate E for a memory array with N_C columns can then be calculated by the following equation [8]:

$$E = 1 - (1 - e)^{N_C} \quad (2)$$

3) Energy Calculation: The bit-cell dynamic read/write energy depends on the read/write pulse which is fixed based on a target error rate as explained above. The total dynamic energy of the memory subarray is the summation of the dynamic energies of all the components. Similar to the latency calculation, the overall dynamic read and write energy samples are obtained by combining the bit-cell energy distribution and the peripheral energy distributions using Monte-Carlo method. We follow the same flow as the one performed for total latency calculation with the exception that when M instances are accessed, we take summation of the energies instead of taking the maximum. Using the same notation explaining the latencies, we define E_{cell}, E_{RD}, E_{BL}, E_{BLM}, E_{BLMD}, E_{SA}, E_{SAM}, and E_{SAMD} which are the summation of i.i.d. samples from their corresponding energy distributions. For example, the dynamic energy of the memory array is the summation of N_C instances from cell energy distribution:

$$e_{cell,i} \sim Distribution\{Cell\ Energy\},\ E_{cell} = \sum_{i=1}^{N_C} e_{cell,i}.$$

Therefore, the total dynamic energy is calculated as:

$$E = E_{cell} + E_{RD} + E_{BL} + E_{BLM} + E_{BLMD} + E_{SA} + E_{SAM} + E_{SAMD}.$$

The total leakage power can also be obtained in a similar manner. Since the leakage power of STT-MRAM bit-cell is zero, the total leakage power is the sum of the leakage powers of the peripheral components.

D. Design Space Exploration

NVSim has an optimization engine which performs a design space exploration and finds an optimal memory configuration based on an input design metric. For instance, if the user wants to find a write latency optimized design, NVSim calculates the overall write latency for different memory configurations and finds the configuration which has the minimal write latency. However, this optimization is based on nominal values of bit-cell and peripheral latencies/energies. A more realistic approach would be to find a *variation-aware* optimal memory configuration considering the impact of process variation.

For the variation-aware approach, the read/write period is fixed based on a user input target RER/WER. If the target error rate is E for a memory array with N_C columns, the error rate of a bit-cell *e* can be derived from (2) as $e = 1 - (1 - E)^{(\frac{1}{N_C})}$. Based on e, *quantile* functions of the read/write latency distributions are used to find the read/write period t_P. The

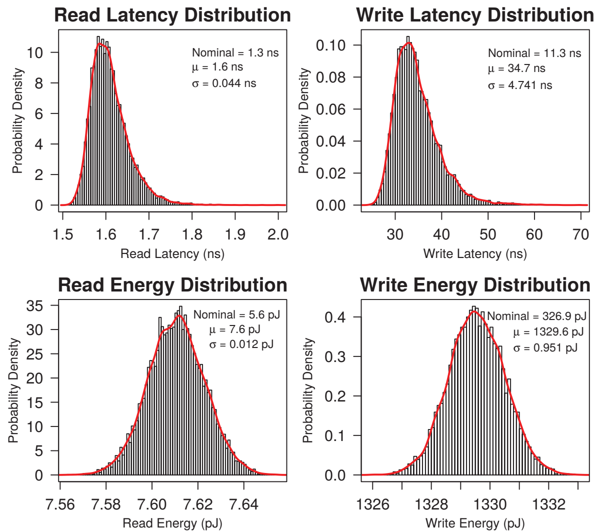


Fig. 5: Overall latency and energy distributions for a subarray size of 1024 × 1024 at a temperature of 300K.

quantile is defined as the inverse of the CDF. From (1), we get $t_P = quantile(1 - e)$.

The bit-cell read/write energy distributions, which depend on the read/write period t_P, is affected by the target error rate. The read/write period, together with the latency distributions of the peripheral components, are then used to calculate the overall read/write latency. Similarly, the overall read/write energies are obtained by combining the read/write energy distributions of the bit-cell as well as the peripheral components.

To perform a variation-aware memory design space exploration, the overall latency/energy distributions should be obtained for each iteration of the memory configuration. This involves running multiple Monte-Carlo simulations for each iteration which require long run-times. Several techniques can be employed to solve this problem. For instance, in a latency optimized design, based on the trends from the previous iterations, certain memory configurations which tend to increase the latency can be excluded from the exploration. Another approach would be to fix the number of banks and/or mats and perform a limited design space exploration. The optimization engine of NVSim is changed to use 0.9987 quantile (equivalent to $\mu+3\sigma$ of a normal distribution) of the overall latency/energy distribution as the optimization metric and perform a variation-aware memory configuration optimization. Our results show that the *variation-aware* optimal memory configuration could be different from that of the nominal case.

IV. Results

A. Experimental Setup

The bit-cell read latency and power samples are obtained from Monte-Carlo SPICE simulations on a 1T-1MTJ bit-cell in the presence of variations. We have employed the TSMC general purpose SPICE models for the CMOS access transistor and the model from [21] for the MTJ cell. The analytical model presented in NVSim-VX^S [6] is used to obtain the bit-cell write latency and energy distributions. For a fair comparison with the variation-aware approach, we have used the mean of the read/write distribution as the read/write period for the nominal case. In the case of read, since we have the bit-cell level samples, the mean of the samples is used as the read pulse. For write, we get the cell level log-normal distribution parameters (μ and σ) from NVSim-VX^S. The mean m of the distribution can be calculated from the following equation:

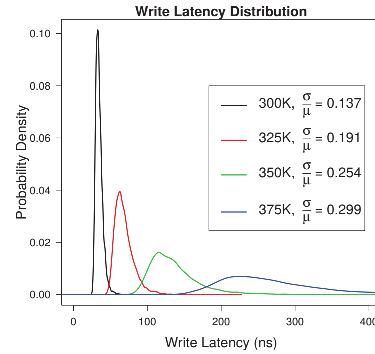


Fig. 6: Variations of overall write latency with temperature for a subarray size of 1024 × 1024.

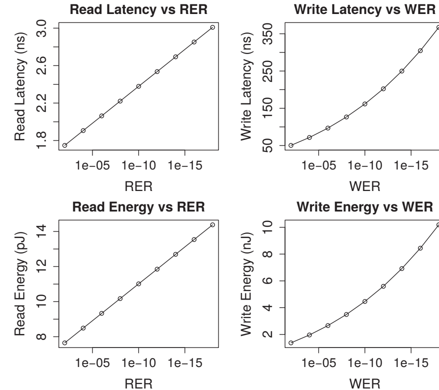


Fig. 7: Overall latency and energy vs Error Rates.

$$m = exp(\mu + \frac{\sigma^2}{2})$$

We have done our analysis on a 128KB RAM with word size of 256 bits. The results have been generated for 45 nm node and a temperature of 300K. We have run Monte-Carlo at the system level to generate 10000 samples for read/write latencies and energies.

B. Results

The results for overall read/write latencies and energies of the memory are given in Fig. 5. The figure also contains the nominal values which are obtained by summing the mean of each of the distributions. The results show that process variations cause a significant increase in the latency and energy values. The variation in energies ($\frac{\sigma}{\mu}$) is less compared to that of latencies. This is because latency calculation involves calculating the maximum of a number of components whereas energy calculation involves calculating the average. The variation of the overall write latency distribution for different temperatures is shown in Fig. 6. It can be seen that as the temperature increases from 300K to 375K, its impact on the write latency also increases as seen by the increase in $\frac{\sigma}{\mu}$ of the distribution.

Fig. 7 presents the overall latencies and energies for different values of input error rates. As the error rates decrease, the read/write latencies increase. This in-turn increases the read/write period which increases the read/write dynamic energies. It can be seen from Fig. 7 that the read latency variation is linear whereas the write latency variation is non-linear. This is because the total latency is dominated by the bit-cell level latency which is log-normal for read and follows an extreme value distribution for write. It can be verified that the quantiles of an extreme value distribution is linear whereas the quantiles of a log-normal distribution is non-linear with respect to the logarithm of the probabilities. The read/write energies depend

TABLE I: Design Space Exploration of a 128KB RAM for an input WER of 1×10^{-18}

Design Metric	NVSim optimization			Variation-aware optimization		
	Rows × Columns	Value		Rows × Columns	Value	
		Nominal	0.9987 quantile		Nominal	0.9987 quantile
Write Latency	1024 × 1024	11.27 ns	367.34 ns	2048 × 512	11.62 ns	357.90 ns
Write Energy	1024 × 1024	326.97 pJ	10.19 nJ	2048 × 512	370.73 pJ	9.91 nJ

TABLE II: Comparison of WER for a Memory Array with 512 Columns

Write Period (ns)	WER		
	NVSim-VX^S [6]	VAET-STT	Monte-Carlo
50	7.882×10^{-03}	4.379×10^{-03}	4.334×10^{-03}
60	1.004×10^{-03}	4.577×10^{-04}	4.463×10^{-04}
70	1.143×10^{-04}	5.708×10^{-05}	5.560×10^{-05}
80	0	8.278×10^{-06}	8.270×10^{-06}
90	0	1.366×10^{-06}	1.160×10^{-06}

TABLE III: Comparison of Run-time of the proposed method with full Monte-Carlo

Subarray Size (Rows × Columns)	Run-time (s)		Speed-up
	Our Method	Monte-Carlo	
256 × 32	7	48	6.8×
1024 × 1024	235	10,112	43×

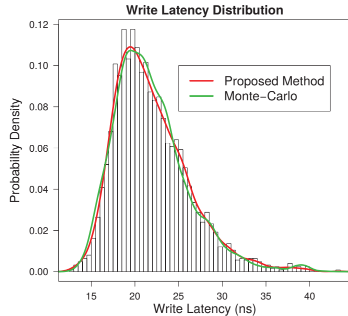


Fig. 8: Validation of our hybrid method with Monte-Carlo for a subarray size of 256 × 32.

on the latencies and hence follow a similar trend.

Table II compares the WER reported by NVSim-VX^S with our VAET-STT tool. It also includes the WER calculated using a detailed Monte-Carlo method. The error rates reported by our tool are more accurate and closer to the detailed Monte-Carlo values. It can also be seen that NVSim-VX^S is not able to report the correct error rates beyond a certain value of write period ($80ns$ as shown in Table II). However, our tool is able to correctly report even low error rates.

Table I shows the results of *variation-aware* design space exploration for the write operation for an input WER of 1×10^{-18}. The table also includes the nominal values where the write period is fixed equal to the mean of the bit-cell write latency distribution. The original NVSim optimization, which is not variation-aware, reports an optimal memory configuration of 1024 × 1024, with a lower nominal latency than that of the 2048 × 512 configuration. However, in the *variation-aware* optimization, the optimal memory configuration is 2048 × 512. This configuration has a lower quantile of the overall latency compared to that of the 1024 × 1024 configuration. A similar trend can be seen in the case of write energy as well. This shows that in some cases, *variation-aware* optimization can lead to a different configuration as compared to the nominal (*variation-unaware*) case.

C. Validation

Fig. 8 shows the comparison of the proposed hybrid method with a full Monte-Carlo analysis for overall write latency. The results show that our proposed method closely approximates a full Monte-Carlo method. However, a full Monte-Carlo requires large number of samples for the individual components, which increases the run-time. Table III shows the run-time comparison for two different subarray sizes. Experiments were run on a 64 bit Linux machine having 16 GB of RAM with 16 Intel Xeon cores clocked at 2.53 GHz. It can be seen that our proposed method is faster than the full Monte-Carlo method. When the number of columns in the subarray increases from 32 to 1024, the run-time for the Monte-Carlo method increases 210×, whereas the run-time for our method increases only 33×. Hence, our method has better scalability compared to the full Monte-Carlo method.

V. Conclusions

Process variations and stochasticity have a significant impact on the performance and energy of STT-MRAM. We have developed a system level *variation-aware* framework to estimate the latencies and energies of STT-MRAM based memories. The tool considers stochastic switching and process variations in the bit-cell as well as process variations in the peripheral components in its system-level analysis. The tool can perform a *variation-ware* memory configuration optimization while meeting reliability constraints. The results show that our framework can provide more realistic margins and the optimized variation-aware memory configuration could be significantly different from the conventional framework.

VI. Acknowledgment

This work was partly supported by the European Commission under the Horizon-2020 Program as part of the GREAT project (http://www.great-research.eu/) and by ANR/DFG as part of the MASTA project.

References

[1] N. S. Kim *et al.*, "Leakage current: Moore's law meets static power," *computer*, vol. 36, no. 12, pp. 68–75, 2003.
[2] M. Hosomi *et al.*, "A novel nonvolatile memory with spin torque transfer magnetization switching: Spin-RAM," *IEDM Tech. Dig*, vol. 459, 2005.
[3] A. Driskill-Smith and Y. Huai, "STTRAM–A new spin on universal memory," *Future Fab Intl. Report*, 2008.
[4] S. Motaman *et al.*, "Impact of process-variations in STTRAM and adaptive boosting for robustness," in *DATE*, pp. 1431–1436, 2015.
[5] X. Dong, C. Xu, N. Jouppi, and Y. Xie, "NVSim: A circuit-level performance, energy, and area model for emerging non-volatile memory," in *Emerging Memory Technologies*, pp. 15–50, Springer, 2014.
[6] E. Eken *et al.*, "NVSim-VX s: an improved NVSim for variation aware STT-RAM simulation," in *DAC*, p. 70, 2016.
[7] S. J. Wilton and N. P. Jouppi, "CACTI: An enhanced cache access and cycle time model," *JSSC*, vol. 31, no. 5, pp. 677–688, 1996.
[8] J. Li *et al.*, "Modeling of failure probability and statistical design of spin-torque transfer magnetic random access memory (STT MRAM) array for yield enhancement," in *DAC*, pp. 278–283, 2008.
[9] W. Zhao *et al.*, "Failure analysis in magnetic tunnel junction nanopillar with interfacial perpendicular magnetic anisotropy," *Materials*, vol. 9, no. 1, p. 41, 2016.
[10] K. Munira, W. H. Butler, and A. W. Ghosh, "A quasi-analytical model for energy-delay-reliability tradeoff studies during write operations in a perpendicular STT-RAM cell," *TED*, vol. 59, no. 8, pp. 2221–2226, 2012.
[11] Y. Ye *et al.*, "Statistical modeling and simulation of threshold variation under random dopant fluctuations and line-edge roughness," *TVLSI*, vol. 19, no. 6, pp. 987–996, 2011.
[12] S. Borkar *et al.*, "Parameter variations and impact on circuits and microarchitecture," in *DAC*, pp. 338–342, 2003.
[13] K. J. Kuhn *et al.*, "Process technology variation," *TED*, vol. 58, no. 8, pp. 2197–2208, 2011.
[14] E. I. Vatajelu *et al.*, "STT-MRAM cell reliability evaluation under process, voltage and temperature (PVT) variations," in *DTIS*, pp. 1–6, IEEE, 2015.
[15] Z. Pajouhi *et al.*, "Device/circuit/architecture co-design of reliable STT-MRAM," in *DATE*, pp. 1437–1442, 2015.
[16] Z. Sun *et al.*, "Variation tolerant sensing scheme of spin-transfer torque memory for yield improvement," in *ICCAD*, pp. 432–437, 2010.
[17] X. Bi *et al.*, "Probabilistic design methodology to improve run-time stability and performance of stt-ram caches," in *ICCAD*, pp. 88–94, 2012.
[18] A. Ahari *et al.*, "Improving reliability, performance, and energy efficiency of STT-MRAM with dynamic write latency," in *ICCD*, pp. 109–116, 2015.
[19] S. R. Sarangi *et al.*, "VARIUS: A model of process variation and resulting timing errors for microarchitects," *TSM*, vol. 21, no. 1, pp. 3–13, 2008.
[20] E. Castillo *et al.*, *Extreme value and related models with applications in engineering and science*. Wiley Hoboken, NJ, 2005.
[21] W. Guo *et al.*, "SPICE modelling of magnetic tunnel junctions written by spin-transfer torque," *Journal of Physics D: Applied Physics*, vol. 43, no. 21, p. 215001, 2010.
[22] M. J. Pelgrom *et al.*, "Matching properties of MOS transistors," *JSSC*, vol. 24, no. 5, pp. 1433–1439, 1989.