RHS-TRNG: A Resilient High-Speed True Random Number Generator Based on STT-MTJ Device

Siqing Fu[1], Tiejun Li[1], Chunyuan Zhang, Hanqing Li, Sheng Ma, Jianmin Zhang, Ruiyi Zhang and Lizhou Wu*

***Abstract*—High-quality random numbers are very critical to many fields such as cryptography, finance, and scientific simulation, which calls for the design of reliable true random number generators (TRNGs). Limited by entropy source, throughput, reliability, and system integration, existing TRNG designs are difficult to be deployed in real computing systems to greatly accelerate target applications. This study proposes a TRNG circuit named RHS-TRNG based on spin-transfer torque magnetic tunnel junction (STT-MTJ). RHS-TRNG generates resilient and high-speed random bit sequences exploiting the stochastic switching characteristics of STT-MTJ. By circuit/system co-design, we integrate RHS-TRNG into a RISC-V processor as an acceleration component, which is driven by customized random number generation instructions. Our experimental results show that a single cell of RHS-TRNG has a random bit generation speed of up to 303 Mb/s, which is the highest among existing MTJ-based TRNGs. Higher throughput can be achieved by exploiting cell-level parallelism. RHS-TRNG also shows strong resilience against PVT variations thanks to our designs using bidirectional switching currents and dual generator units. In addition, our system evaluation results using gem5 simulator suggest that the system equipped with RHS-TRNG can achieve 3.4–12x higher performance in speeding up option pricing programs than software implementations of random number generation.**

***Index Terms*—TRNG, MTJ, Monte Carlo, Circuit/System Codesign.**

I. Introduction

With the proliferation of semiconductor products, random numbers play an increasingly vital role in many fields such as cryptography, computational finance, scientific simulation, artificial intelligence(AI), and stochastic computing [1]–[3]. Existing computing systems mainly rely on pseudo-random number generators (PRNGs) to generate random numbers. However, this software-based method compromises generation speed and quality because of its predictable and periodic characteristics; it may even open doors to potential attacks that compromise keys, intercept data, and ultimately hack devices and communication channels. As an alternative, true random number generators (TRNGs) are able to produce random number sequences that are truly uniformly distributed and unpredictable by sampling physically random processes [4]. In addition, TRNGs are typically implemented in hardware, thus making them much faster than PRNGs in generating random bit stream. Therefore, it is crucial to design a quality high-speed TRNG that can be integrated into computing systems to accelerate and secure critical applications.

Hardware-based TRNG designs rely on random physical phenomena as entropy sources, which are typically obtained from existing commodity circuits such as DRAM or from dedicated CMOS designs. In the former, random phenomena that can be utilized include DRAM retention failures [5], [6], start-up value variations [7], or random reads caused by illegal DRAM commands [8], [9]. However, these entropy sources are mostly slow physical processes, making them difficult for TRNG designs to meet the throughput and latency requirements of some applications. The latter, such as differential ring oscillators [10], [11] and metastability [12], [13], rely on CMOS circuit thermal noise or oscillator jitter. These CMOS-based TRNGs also have some drawbacks such as low generation speed [10], [14]. Additionally, the area and power consumption of CMOS-based TRNG designs are also unsatisfactory.

As semiconductor technologies evolve, some emerging post-CMOS devices with lower power consumption and smaller areas than transistors can provide new solutions for TRNG designs. For example, magnetic tunnel junction (MTJ) as a promising Spintronic device has been widely studied over the past decades, and its random switching behavior driven by spin-transfer torque (STT) provide opportunities for designing high-quality TRNGs. More specifically, the switching process of STT-MTJ between its two magnetic states is intrinsically stochastic under the influence of thermal fluctuation, thus making it a perfect entropy source [15]. There exist some works on MTJ-based TRNG, but the performance of MTJ has not been fully exploited [16]–[22]. Examining the prior art reveals the following four drawbacks: 1) The random bit generation latency is high, reaching for example 80 ns in [18]; 2) lifetime is limited by the high reset current; 3) random number quality is susceptible to process/voltage/temperature (PVT) variations; 4) system integration and acceleration effects on applications are unclear.

In this paper, we design a high-speed STT-MTJ-based TRNG which offers high resilience to PVT variations. It can be integrated into computing systems as an acceleration

[1]These authors contributed equally to this work.

Manuscript received 12 February 2023; revised 17 May 2023 and 6 July 2023; accepted 19 July 2023. This work was supported by the State Key Laboratory of High Performance Computing Foundation (202201-13), Science and Technology on Parallel and Distributed Processing Laboratory Foundation (WDZC20205500115), National Key R&D Project 2021YFB0300300, the NSFC 62172430, the NSF of Hunan Province 2021JJ10052, and the STIP of Hunan Province 2022RC3065.(*Corresponding author: Lizhou Wu.)

Siqing Fu, Tiejun Li, Chunyuan Zhang, Hanqing Li, Sheng Ma, Jianmin Zhang, and Lizhou Wu are with the College of Computer Science and Technology, National University of Defense Technology, Changsha 410073, China (e-mail: {fusiqingnudt, tjli, cyzhang, lihanqing23013, masheng, jmzhang, lizhou.wu}@nudt.edu.cn).

Ruiyi Zhang is with the Division of Science and Technology, Beijing Normal University - Hong Kong Baptist University United International College, Zhuhai 519087, China (e-mail: q030018105@mail.uic.edu.cn).

component with customized instructions to produce high-quality random bit stream for performance/security-critical applications. In our RHS-TRNG design, the number of write operations in a random bit generation cycle is reduced from two to one by feedback control, which reduces the generation latency. Eliminating the reset operation also greatly boosts the TRNG lifetime since the high reset current no longer flows through the underlying STT-MTJ device. In addition, we leverage two mechanisms: 1) bidirectional switching currents and 2) dual generator units to equip RHS-TRNG with resilience capability to PVT variations. Finally, we integrate RHS-TRNG into the instruction pipeline of RISC-V cores and design three custom instructions to drive the TRNG acceleration component. To evaluate the performance, power and area of RHS-TRNG, we have implemented its circuit and layout in the Cadence Virtuoso tool. Simulation results shows that the latency is $3.3\,\mathrm{ns/bit}$, power is 2.65-5.3 $\mathrm{pJ/bit}$, and area is 14.5-24.29 $\mu\mathrm{m}^2/\mathrm{bit}$. By integrating multiple STT-MTJ cells, the circuit can achieve better performance in power consumption and area per bit. To evaluate the hardware acceleration effect on target applications, we have simulated our system design in gem5 and used Monte Carlo option pricing program as a test benchmark. The experimental results show that by integrating RHS-TRNG, we can obtain more than 3.4x speedup when running Monte Carlo option pricing program when compared to generating random numbers using software methods.

In summary, the main contributions of this paper are listed as follows:

- We design a high-speed two-phase TRNG circuit based on STT-MTJ. With feedback control, we avoid the reset phase in each random bit generation cycle to achieve the generation speed of $303\,\mathrm{Mb/s}$ for a single TRNG cell.
- We enhance the resilience of RHS-TRNG to tolerate PVT variations by utilizing bidirectional write current and dual-generator XOR design.
- We propose circuit-system co-design for RHS-TRNG by customizing a RISC-V processor and instructions. Gem5 simulation results show 3.4-12x performance acceleration for Monte Carlo option pricing applications in comparison to software implementations.

The rest of the paper is organized as follows: Section. II introduces the fundamentals of TRNG, MTJ device and its switching behavior. Section. III elaborates the motivation of this paper. Section. IV presents the circuit and system co-design of RHS-TRNG. In Section. V, we present the experimental setups and results at both circuit and system levels. Section. VI compares this work with the prior art, while Section. VII discusses some valuable future research topics. Finally, Section. VIII concludes the paper.

II. Background

A. True Random Number Generator

Random number generators are typically classified into two categories: pseudo-random number generators (PRNGs) and true random number generators (TRNGs). PRNGs generate random sequences by algorithms that transform an internal state and calculates an output value upon request. Once the initial seed is set, the next state only depends on the previous state. As a result, pseudo-random sequences can be predictable and controllable; they are only mathematically consistent with a random distribution [4]. In contrast, TRNGs generate random numbers by sampling random physical phenomena such as thermal noise, electromagnetic behavior, and quantum behavior. Since these entropy sources are intrinsically non-deterministic, the state of each cycle in a TRNG cannot be predicted even if all states are known when it runs. Therefore, it can produce a truly random bit stream to applications which have stringent requirements on the quality of random numbers.

Conventional TRNGs are typically implemented by leveraging key features of CMOS circuits, e.g., ring oscillation (RO) [10], [11] and metastability [12], [13]. However, these CMOS-based TRNG designs do not provide adequate performance. For example, the TRNG in [14] using time-dependent dielectric breakdown provides only $0.011\,\mathrm{Mb/s}$ random sequences, and the speed of the differential OR-based TRNG in [10] is $8.28\,\mathrm{Mb/s}$. To design faster TRNG hardware, researchers are looking for new technologies. Among the emerging devices, STT-MTJ has a smaller area and lower power consumption than CMOS transistors [23]. Moreover, its operating principle involves inherent physical random processes, making it a promising option for designing TRNGs [16]–[22].

B. Magnetic Tunnel Junction

Magnetic tunnel junction (MTJ) is a widely used Spintronic device [24]–[26]. The core of an MTJ is a three-layer structure consisting of two ferromagnetic layers and one dielectric tunnel barrier (TB) layer sandwiched between them, as shown in Fig. 1.a. The bottom ferromagnetic layer is called pinned layer (PL), whose magnetization is fixed along the MTJ's easy axis [27]. The top ferromagnetic layer is called free layer (FL); its magnetization is either in parallel (P) or anti-parallel (AP) to that of the PL [28]. Due to the tunneling magneto-resistance (TMR) effect [29], the AP and P magnetic states have different resistance values: the high-resistance value observed under the AP state (R_{AP}) represents logic 1, while the low-resistance state under the P state (R_{P}) represents logic 0. The difference between R_{AP} and R_{P} is commonly quantified using the TMR ratio, given by: $(R_{\mathrm{AP}} - R_{\mathrm{P}})/R_{\mathrm{P}} \times 100\%$.

The MTJ used in this paper is perpendicular MTJ composed of ferromagnetic layers with perpendicular magnetic anisotropy (PMA) [30]. This type of MTJ has become the

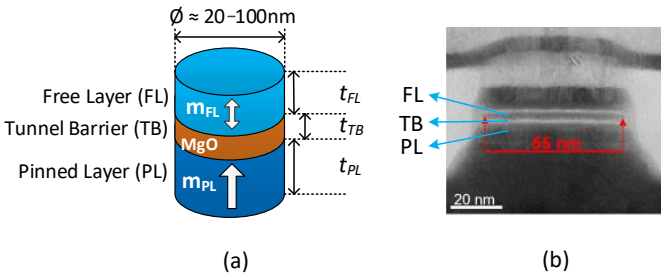

Fig. 1. Perpendicular MTJ device: (a) structure schematic and (b) cross-sectional TEM image.

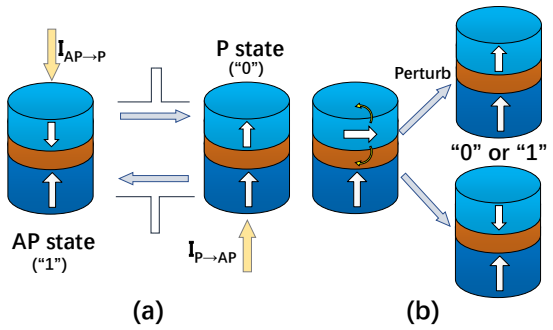

Fig. 2. (a) STT-MTJ's bipolar switching method between AP and P states. (b) Stochastic switching process to either "0" or "1" due to thermal perturbation.

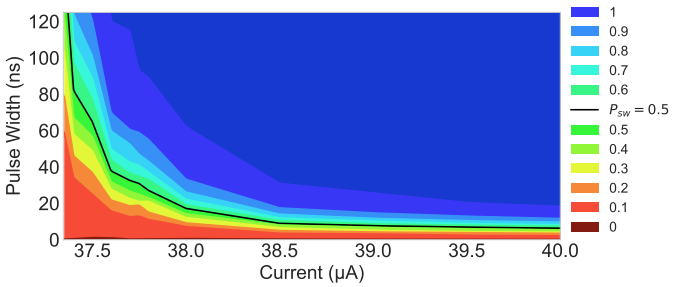

Fig. 3. The AP→P switching probability of STT-MTJ under different write current and pulse width.

mainstream MTJ design in recent years, thanks to its small area and low power consumption. Fig. 1.b shows a cross-sectional transmission electron microscopy (TEM) image of a $\varnothing$ 55 nm MTJ device fabricated at IMEC [31]. At present, Everspin Technologies has commercialized 1 Gb STT-MRAM chips with MTJs as persistent data-storing devices [32]. In addition, many foundries worldwide such as TSMC, Samsung, and GlobalFoundries [33]–[35] have claimed production service ready for cutting-edge MTJ devices.

C. STT Switching Stochasticity

The MTJ state can be switched by several methods such as external magnetic field [36], spin-transfer torque (STT) [37], and spin-orbit torque (SOT) [38]. An MTJ that uses STT effect to switch its binary state is called STT-MTJ. As shown in Fig. 2.a, when a positive pulse is applied across the MTJ in AP state, it drives a current $I_{\mathrm{AP}\rightarrow\mathrm{P}}$ flowing perpendicularly from the FL to the PL. If the pulse's amplitude and width reach certain threshold, the FL's magnetization switches to the opposite direction, typically in 2-100 ns. Similarly, a negative pulse can switch the MTJ from P to AP under the spin-polarized current $I_{\mathrm{P}\rightarrow\mathrm{AP}}$ when it exceeds the critical switching current. Note that STT-MTJ is a bipolar device, meaning that the polarity of the switching current determines the magnetization direction in the FL, which in turn determines the STT-MTJ's resistive state [39].

Due to thermal fluctuation [40], the STT switching process of STT-MTJ is inherently stochastic for a given write pulse. The thermal fluctuation effect causes the magnetizations in the free layer (FL) and pinned layer (PL) to have a random, varying initial angle between them in each cycle, resulting in a random STT switching behavior, as shown in Fig. 2.b. This produces a natural, cycle-to-cycle variance in the switching time. As a result, it is guaranteed that STT-MTJ, operating as an entropy source, can provide a random and independent bit sequence [41].

As an example, Fig. 3 illustrates the simulated switching probability (P_{sw}) from the AP to P state under various write pulse configurations. As shown in the figure, the switching probability increases with current and pulse width (from the lower-left to upper-right corner). The black line ($P_{\mathrm{sw}} = 0.5$) on the graph represents the operating points in which the switching probability is 50%. This indicates that by tuning the write pulse to one of these operating points, the switching result can be considered as a random variable X which obeys the Bernoulli distribution: $X \sim \mathrm{B}(n, 0.5)$.

In summary, the switching stochasticity of STT-MTJ can serve as a high-quality entropy source for TRNG design.

III. Motivation

A. Application Demand for TRNGs

In the era of information technology, the demand for high-quality random numbers is ubiquitous across various applications. Cryptography, for instance, requires good randomness in generating keys to ensure their security against attackers [1], [21]. This randomness is manifested in the form of long periodicity, non-linearity, unpredictability, among other factors. Similarly, in scientific simulations, the Monte Carlo method relies on high-quality random numbers to simulate random behavior in different processes [42], [43]. Poor quality random bit streams can lead to degraded simulation confidence, and the generation rate of random numbers can also become a performance bottleneck in some Monte Carlo simulations. Apart from traditional scientific fields, emerging fields such as AI and stochastic computing also require high-quality random numbers. For instance, neural networks in AI applications are initialized with random numbers to break symmetry and enable faster convergence [2]. Meanwhile, stochastic computing is an emerging computing paradigm that relies on random numbers for bit-wise operations [3]. Therefore, a high-quality TRNG is essential to ensure precise computation in these applications.

B. Limited System Integration with TRNGs

Depending on application requirements and implementation technologies, TRNG hardware can be integrated into different parts of system architecture, from the pipeline, to cache, to memory, or simply as a peripheral connected to the system. However, the existing TRNG designs, regardless of their implementation technologies, rarely consider the effect of TRNG integration on the target applications.

CMOS-based TRNG designs are typically evaluated using SPICE simulations [10], [12] or FPGA prototypes [11], [13]. These works evaluated TRNG hardware's energy consumption, randomness, and area, but they neglected system integration and its effects. For example, in [11], FPGAs have used to prototype cryptographic systems, but after implementing TRNG on FPGA, they do not evaluate the effect of its integration into cryptographic systems. MTJ-based TRNG

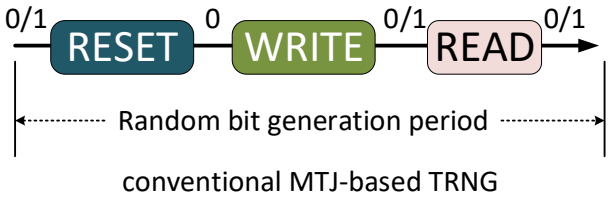

Fig. 4. MTJ-based TRNG design with three phases: reset, write and read.

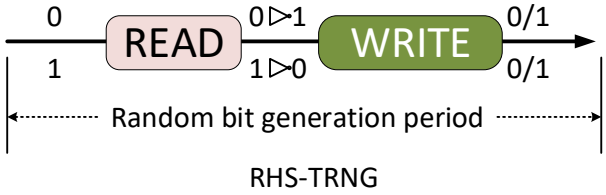

Fig. 5. RHS-TRNG with one cycle consisting of two phases: read and write.

designs are typically evaluated in circuit simulations using MTJ models [16]–[22]. These works have fully analyzed the TRNG circuits composed of MTJ and CMOS, but rarely mentioned the higher level of the system. Even if the individual work evaluates system-level performance [44], it does not consider the acceleration effect of TRNG on the benchmark. Another type of TRNG design is based on commercial DRAM [5]–[7], [45]. Despite these works provide high feasibility of integrating into real systems, they require post processing and encroach memory bandwidth which is already a performance bottleneck in today's computing systems.

C. Shortcomings of Prior MTJ-based TRNGs

There exist several TRNG designs using STT-MTJ as an entropy source in the literature [16]–[22]. However, the performance of existing MTJ-based TRNG designs is significantly limited by their three-phase circuit design for generating a random bit in each cycle, as illustrated in Fig. 4. The random bit generation process starts with the reset phase. This phase is designed to reset the MTJ to a fixed state (e.g., logic 0) from an initial state which can be either logic 0 or 1. This is achieved by applying a large current going through the MTJ, causing it to switch to the P state with 100% probability. The second phase is random write. During this phase, the circuit applies a smaller write current to the MTJ in the opposite direction, which causes it to flip with a 50% probability. As a result, the MTJ will be randomly set to logic 0 or 1 state. After the random write, the circuit goes through a read phase to read out the data stored in the MTJ, i.e., the random bit output during this cycle.

Although the above traditional MTJ-based TRNG design is clear and feasible, it has the following three shortcomings. First, each cycle contains two write operations: reset and random write. Given the fact that MTJ devices feature fast read and slow write, it limits the use of MTJ to design low-latency and high-throughput TRNGs. Second, the reset phase requires a very large current flowing through the MTJ in order to guarantee 100% switching probability (see Fig. 2). This however may lead to device breakdown, limiting the lifetime of the designed TRNG [46]. Third, this simple TRNG design is susceptible to process/voltage/temperature (PVT) variations. It has been demonstrated that process and voltage variations have a large impact on the MTJ's switching behavior [47]. In addition, an increase in the ambient temperature enables switching in both directions with reducing write voltage [48].

While many traditional STT-MTJ TRNG designs require a reset phase, some recent advances in the field have led to new possibilities. For instance, Choi et al. [49] proposed a TRNG that employs probability tracking with two pulse generators, eliminating the need for a reset phase and compensating for output probability fluctuations through counters and software-based real-time probability tracking. Similarly, Oosawa et al. [50] introduced a digitally-controlled probability TRNG that does not rely on a reset phase and evaluated its ability to produce random bit streams with stable probability distribution. However, these designs inevitably utilized post-processing circuits such as correction logics, D/A converters, and digital comparators to enhance the quality of randomness.

In summary, even though many MTJ-based TRNG designs are limited by their dependence on a reset phase, there have been new opportunities due to recent breakthroughs. Our contributions to this advancement are threefold. First, we greatly improved the random number generation speed compared to the prior art. Second, in terms of addressing PVT variations, we avoided the use of post-processing circuits that would significantly increase power and area overheads. Third, we deployed our TRNG design in a RISC-V processor and corroborated its strength in accelerating specific applications.

D. Idea and Goal

To ensure high-quality output bits and improve TRNG output rates to support applications that require acceleration, our ideas are: 1) the write overhead in each cycle of TRNG should be reduced as much as possible; 2) resilience to PVT variations should be obtained through dedicated circuit design to maintain the stability of the output probability of logic 0 and 1 in the generated bit streams; 3) TRNG hardware needs to be seamlessly integrated into existing computing systems as a random number acceleration component. Motivated by these ideas, we aim to design a high-speed and resilient STT-MTJ-based TRNG and integrate it into existing computing platforms to greatly accelerate specific applications such as Monte Carlo simulations.

IV. Circuit-System Codesign

In this section, we introduce the co-design of RHS-TRNG at circuit and system levels. We first elaborate RHS-TRNG circuit design as well as its theoretical benefits. Thereafter, we detail the system and custom instruction design based on RISC-V ISA.

A. RHS-TRNG Design

1) Design Philosophy: Rather than relying on a single switching direction for generating random bit streams, we use current in both switching directions to control the random switch of the STT-MTJ. This enables us to eliminate the reset

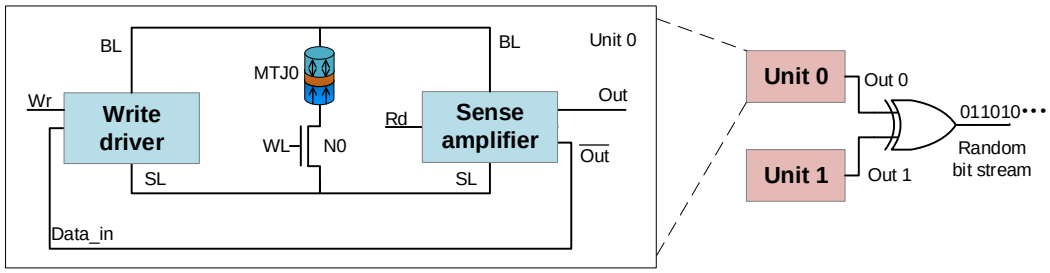

Fig. 6. Circuit design of RHS-TRNG.

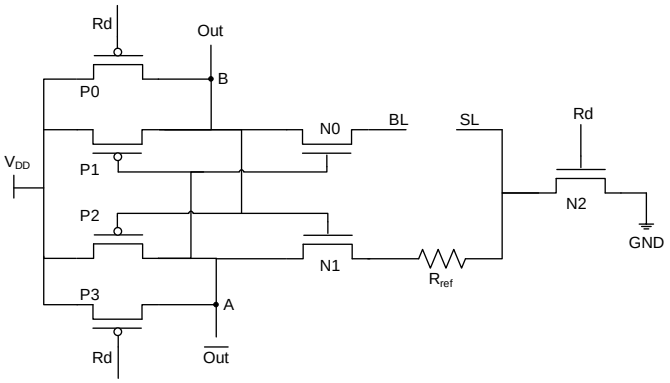

Fig. 7. Sense amplifier circuit design for RHS-TRNG.

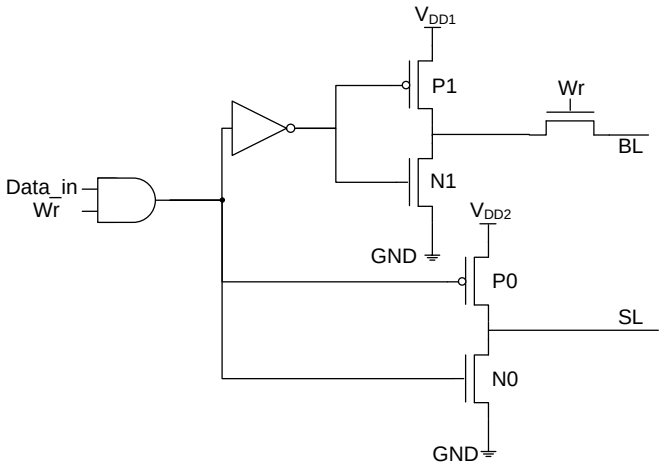

Fig. 8. Write driver circuit design for RHS-TRNG.

phase in each cycle of random bit generation, as depicted in Fig. 5.

At the beginning of each cycle, the MTJ's initial state (logic 0 or 1) is read out by a sense amplifier circuit, then the inverted value is fed to a write driver circuit. In the second phase, a random write operation is performed by the write driver; the write current direction depends on the data received from the previous read phase. This allows us to perform only one write operation in each cycle, which greatly reduces the time it takes to generate a random bit. Next, we will elaborate how to implement this two-phase TRNG design via VLSI circuits.

2) Circuit Design: The circuit design of RHS-TRNG is shown in Fig. 6. Each RHS-TRNG cell consists of two generator units and an XOR gate. The TRNG cell outputs a randon bitstream that is the XOR result of the two identical generator units. In each generator unit, an STT-MTJ device is connected in series with an NMOS transistor as a selector. The on/off state of the NMOS is controlled by a word line (WL). The 1T-1MTJ structure is connected in parallel with a set of write driver and sense amplifier circuits through a bit line (BL) and a source line (SL), which perform write and read operations on the STT-MTJ. When the "Rd" signal is enabled, the sense amplifier can simultaneously read out the logic state (denoted as "Out") of the MTJ and its inverted value "$\overline{\text{Out}}$". The "$\overline{\text{Out}}$" signal is then fed back to the write driver as an input "Data_in". When the "Wr" signal is asserted, the write driver starts programming "Data_in" into the MTJ device by applying a pulse across the BL and SL. It generates a switching current whose direction is determined by the data to be programmed (i.e., "Data_in"), as illustrated in Fig. 2.

The sense amplifier circuit is shown in Fig. 7. We use pre-charge sense amplifier design which compares the currents going through the MTJ cell under sensing and a fixed reference cell [51]. The entire read process is divided into the following three stages [52]. (1) **Pre-charge**. The "Rd" signal is set to 0, which turns on PMOSs P0 and P3 and turns off NMOS N2. After the two nodes A and B are pre-charged to the same potential $V_{\rm DD} - V_{\rm TH}$, N0 and N1 are turned on, whereas P1 and P2 are turned off. (2) **Voltage development**. The "Rd" signal is set to 1, which controls P0 and P3 to be turned off and N2 to be turned on. The two nodes A and B begin to discharge. The reference resistor is a fixed-resistance MTJ, so there is no area overhead of integrated resistors. Since the resistance value of the reference resistor is between $R_{\rm P}$ and $R_{\rm AP}$, A and B will have different discharge rates. This results in a small voltage quickly developed between A and B. (3) **Voltage amplification**. Once the voltage reaches a threshold, it is quickly amplified to the full swing by the cross-coupled inverters (P1, P2, N0, N1). Due to its fast read speed (hundreds of picoseconds) and tiny sensing current, this sense amplifier circuit is ideal for designing high-speed and low-power TRNGs.

The write driver circuit is shown in Fig. 8. When the write control signal "Wr" is enabled and the write data on "Data_in" is at 1, N0 and P1 will be turned on while P0 and N1 will be turned off. This leads to a write current flowing from the BL to the SL. The amplitude and duration of the current determine whether the MTJ will be set to state 1 successfully, as explained in Sec. II-C. Likewise, when 'Data_in" is at 0,

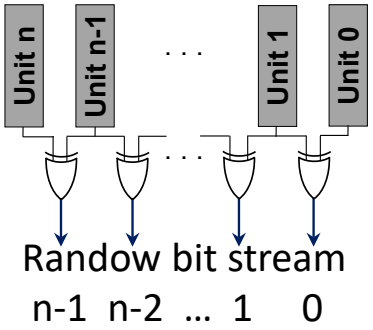

Fig. 9. Multi-cell parallel structure of RHS-TRNG.

a write current with the opposite direction is drawn by the circuit to flip the MTJ's state to 0 with certain probability if it is in state 1. Since the MTJ device has different resistance values in the P and AP states, the required bias voltages V_{DD1} and V_{DD2} are set different to make the currents in the forward and reverse directions have a 50% switching probability under the same pulse width.

Finally, as illustrated in Fig. 9, we configure the RHS-TRNG cells in parallel to increase the random bit throughput. To reduce area and power consumption, adjacent cells share an RHS-SingleUnit. The corresponding area and power results are presented in Section. V.

B. Theoretical Analysis

1) Benefits of Bidirectional Switching Currents: Compared to the MTJ-based TRNG circuit design with three phases per cycle, our two-phase RHS-TRNG provides lower latency of random bit generation. Eliminating the reset phase also prolongs the MTJ's lifetime, since the large reset current to ensure a 100% switching probability is avoided; note that the larger the switching current, the smaller the endurance of the MTJ device. In addition to the above two benefits, our circuit design provides a resilient mechanism to cope with PVT variations to enhance the quality of random number generation.

Fig. 10 shows the MTJ state transition diagram of the TRNG with bidirectional write currents. Initially, the MTJ devices may be in the P state (with the probability P_{P}) or AP state (with the probability P_{AP}). When it is in the P state, we assume that the probability of being switched to the AP state under a write current is P_1. When in the AP state, the probability is P_2 under a write current with the opposite direction. Given these assumptions, we can obtain

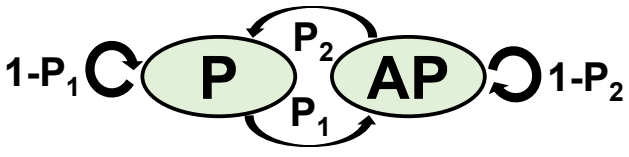

Fig. 10. State transition diagram of an MTJ device controlled by bidirectional switching currents.

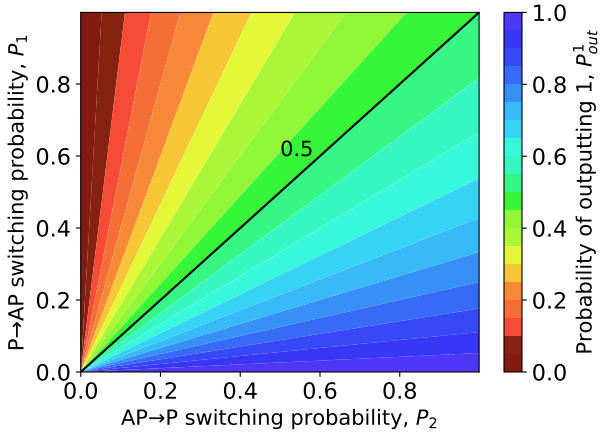

Fig. 11. The impact of AP→P and P→AP switching probabilities on the probability of a random number generator unit outputting logic 1.

the following probability relationships in the state transition diagram:

$$\begin{cases} P_{AP} = P_{AP} \cdot (1 - P_2) + P_P \cdot P_1, \\ P_{AP} + P_P = 1. \end{cases} \tag{1}$$

Assume that each generator unit (see Fig. 6) has a probability P^1_{out} of outputting bit 1 and a probability P^0_{out} of outputting bit 0. P^1_{out} and P^0_{out} can be expressed as follows:

$$\begin{cases} P^1_{\mathrm{out}} = P_{\mathrm{AP}} = \frac{P_1}{P_1+P_2}, \\ P^0_{\mathrm{out}} = P_{\mathrm{P}} = \frac{P_2}{P_1+P_2}. \end{cases} \tag{2}$$

It can be seen that both P^0_{out} and P^1_{out}depend on two probability variables rather than a single variable as found in some previous designs.

Fig. 11 plots P^1_{out} in Equation (2), where the X-axis represents P_2 and the Y-axis represents P_1. When both P_1 and P_2 are equal to 50% ideally, the probability of the random number generator unit outputting "1" is also 50%. The off-center condition is caused by PVT variation. When P_1 and P_2 are shifted in the same direction by the same magnitude, P^1_{out} can remain 50%, as shown with the black solid line in the figure. Recall that several variation sources (e.g., ambient temperature) that may affect the output probability tend to shift P_1 and P_2 along the same direction. Our circuit design allows the TRNG to remain resilient to such variations.

2) Benefits of Two Generator Units: RHS-TRNG uses two identical random number generator units to generate a random bit stream. By means of this redundant mechanism, the quality of random bit generation is further improved by circuit self-stability. Assume that the probability of the generator unit 0 outputting bit 1 is P^1_{out0} and the probability of the generator unit 1 outputting bit 1 is P^1_{out1}, we can mitigate the influence of the shifts in P^1_{out0} and P^1_{out1} to a certain extent via adding an XOR operation on the outputs of the two generator units. When one of them outputs 1 and the other outputs 0, the XOR result is 1; otherwise, the result is 0. Thus, the probabilities

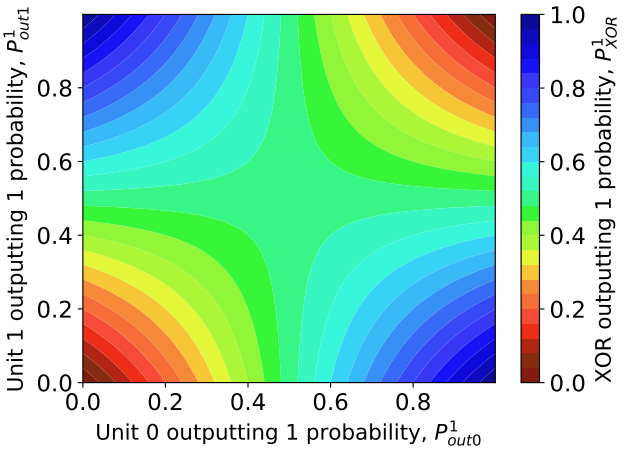

Fig. 12. Probability of outputting "1" after XOR of dual generator units.

of the XOR gate outputting bit 1 ($P^1_{\rm XOR}$) and bit 0 ($P^0_{\rm XOR}$) are calculated as follows:

$$\begin{cases} P^1_{\rm XOR} = P^1_{\rm out0} \cdot (1 - P^1_{\rm out1}) + P^1_{\rm out1} \cdot (1 - P^1_{\rm out0}), \\ P^0_{\rm XOR} = 1 - P^1_{\rm XOR}. \end{cases} \quad (3)$$

Fig. 12 plots $P^1_{\rm XOR}$ as a function of $P^0_{\rm XOR}$ and $P^1_{\rm XOR}$. It can be seen that when $P^1_{\rm out0}$ and $P^1_{\rm out1}$ are both 50% ideally, the output result of the entire TRNG design remains 50%. If these two values slightly drift away from 50%, the output probability $P^1_{\rm XOR}$ still converges to 50% thanks to the XOR mechanism. This design can greatly mitigate the adverse impact of PVT variations, thus keeping the final generated random bit stream stable.

C. System and Custom Instruction Design

To unleash the power of our RHS-TRNG hardware in accelerating practical applications, we embed it to the CPU pipeline as an instruction execution unit. Dedicated instructions thus have to be designed to control its execution. Since RISC-V is an open, royalty-free ISA which features design freedom and flexible architecture extensions, our system design is focused on RISC-V processors.

Fig. 13 shows the architecture of a multi-core RISC-V processor which has integrated our RHS-TRNG as an acceleration component. The left side of the figure is a general-purpose four-core architecture. Each RISC-V core owns a private L1

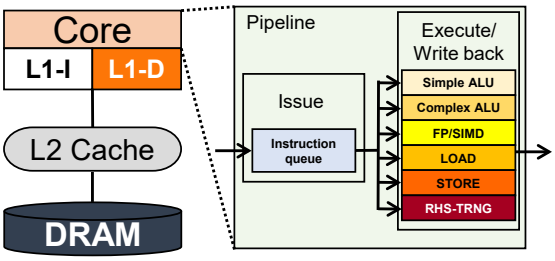

Fig. 13. A system overview of multi-core RISC-V processor with the RHS-TRNG hardware integrated into its pipeline as an instruction execution unit.

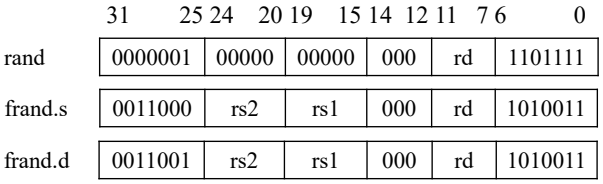

Fig. 14. The structure of custom random number generation instructions.

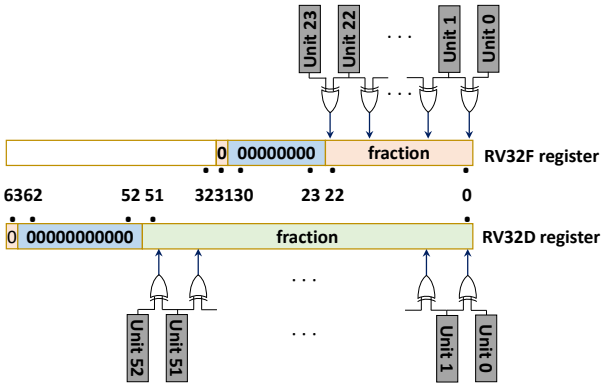

Fig. 15. Hardware structure for converting integer random numbers into floating-point random numbers.

instruction cache (L1-I) and a private L1 data cache (L1-D). All cores share a large-volume L2 cache, which is connected to the main memory (DRAM). On the right side of the figure, we can see a portion of the instruction pipeline in the core. Note that only the issue and execute stages are shown since they are modified and are directly relevant to the proposed RHS-TRNG design.

In contrast to a generic RISC-V core, our architecture places an RHS-TRNG as an instruction execution unit in the execution stage. We design a set of custom instruction to control the RHS-TRNG circuit and read the generated random numbers into a register for use by the program. It should be noted that the integration level of RHS-TRNG into systems depends on many factors such as target application, usage frequency, and lifespan, due to the limited write endurance ($\sim 10^{15}$). Leveraging reliability mitigation schemes such as voltage-reducing and wear-leveling can lead to longer lifetime. But this comes with the cost of longer latency and larger area. Thus, a trade-off between performance and cost has to be made when designing MTJ-based TRNGs. In this paper, we aim to avoid limiting the interface rate for our TRNG. Placing it within the pipeline reduces its read latency and better showcases its performance characteristics.

Fig. 14 shows the design of three custom instructions: rand, frand.s, and frand.d to control the RHS-TRNG acceleration unit. The instruction "rand" controls the generation of an unsigned integer random number between 0 and 32767, it does not require any operand and the generated result is stored in the segment "rd". The other two custom instructions "frand.s" and "frand.d" control the generation of single and double-precision floating-point numbers,which are converted from integer random numbers and are expanded to the range

TABLE I
KEY DEVICE PARAMETERS FOR MTJ COMPACT MODEL.

Parameter	Description	Value
t_{FL}	Thickness of the free layer	1.3nm
$\sigma_{t_{\mathrm{FL}}}$	Standard deviation of t_{FL}	3% of 1.3nm
CD	Critical diameter	32nm
t_{TB}	Thickness of the tunnel barrier	0.85nm
$\sigma_{t_{\mathrm{TB}}}$	Standard deviation of t_{TB}	3% of 0.85nm
TMR	TMR ratio	200%
σ_{TMR}	Standard deviation of TMR	3% of 200%

between the two values stored in rs2 and rs1, respectively. The execution latency of each instruction depends on the random bit generation period at circuit level as well as the system clock cycle.

For integer random numbers, the RHS-TRNG illustrated in Fig. 9 can directly output parallel bit streams to RV32I registers. However, generating floating-point random numbers requires more considerations. Fig. 15 depicts the method of converting integer random numbers into single and double-precision floating-point random numbers. Since the exponent bits should not be uniform, we set all the sign and exponent bits to 0 and actually generate non-normalized floating-point numbers ranging from 0 to 1.

V. EXPERIMENTS AND EVALUATION

A. Experimental Method and Setup

To evaluate the functionality of the proposed self-stabilized RHS-TRNG circuits in this paper, we conducted a series of SPICE circuit simulations. The Verilog-A MTJ compact model in [53] is used in our simulations; it models the TMR effect and STT effect of the MTJ. As a physical model that integrates static, dynamic, and stochastic behaviors, it is currently widely used by researchers in the field of Spintronics. The MTJ device parameters are shown in Table. I; their values are selected in accord with real-world MTJs fabricated at IMEC. We set the ambient temperature to $300\,\mathrm{K}$ (i.e., room temperature) except for the study of temperature variation. The proposed design is simulated using the Cadence Virtuoso tool with GPDK $45\,\mathrm{nm}$ technology.

In our experiments, we first evaluated the timing performance of RHS-TRNG to determine the maximum throughput. Then we obtained a random sequence with a length of 1 million via Monte Carlo simulation to evaluate its statistical randomness. Next, we conducted experiments to analyze the quality of generated random sequences under PVT variations. The area and power consumption of the proposed circuit were then evaluated by the simulation results and layout design results using the Cadence Virtuoso tool. To systematically evaluate the acceleration effect of RHS-TRNG in a general-purpose computing system for the corresponding application, we also integrated it into a RISC-V processor as an instruction execution unit and modeled the system in the gem5 simulator. The architecture that incorporates this unit has significant performance advantages over the general-purpose architecture when running a Monte Carlo option pricing program. The architectures used for comparison are identically configured, as shown in Table. II.

TABLE II
SIMULATED SYSTEM CONFIGURATIONS FOR THE GEM5 SIMULATOR.

Parameter	Value
Gem5 version	21.1.0
Simulation model	SE
CPU Type	MinorCPU
Frequency	2 GHz
Icache size	32 kB
Dcache size	32 kB
L2 cache size	512 kB
Instruction execution latency	8 cycles

To compare the performance of different MTJ-based TRNGs including both the conventional three-stage design and our two-stage design in this work, we implemented four TRNG configurations as follows.

1) **Conv. APtoP**:
   This configuration is a conventional (Conv.) MTJ-based TRNG design (see Fig. 4), where the WRITE phase applies an APtoP switching current flowing through the MTJ device to generate a random bit.
2) **Conv. PtoAP**:
   This configuration is identical to the above Conv.APtoP configration except that the current in the WRITE phase has a probability of 50% to switch the MTJ from the P state to the AP state.
3) **RHS-SingleUnit**:
   This configuration is a single random number generator unit of RHS-TRNG (i.e., Unit 0 or Unit 1 in Fig. 6). Its MTJ cell and peripheral circuits are consistent with that of RHS-TRNG.
4) **RHS-TRNG**:
   This configuration represents a complete RHS-TRNG design.

The names of these configurations will be used directly in the remainder of this paper.

B. SPICE Circuit Simulation

As our RHS-TRNG design consists of two identical RHS-SingleUnits and an XOR gate, we ran transient simulations using the RHS-SingleUnit configuration for the sake of similarity in the Cadence virtuoso tool to evaluate its performance. Fig. 16 shows the transient simulation results, i.e., the waveforms for the "Rd" signal, the "Wr" signal, the state of the MTJ, and the readout voltage at node Out and $\overline{\mathrm{Out}}$ in Fig. 7. Recall that each cycle contains two phases: read and write; the read phase is further divided into three sub-phases: pre-charge, voltage development, and voltage amplification.

First, in the read phase, the "Wr" and "Rd" signals are both set to low potential. After the sense amplifier completes the pre-charge sub-phase (as short as t_{pre}), the "Rd" signal is pulled high. This is followed by the voltage development and amplification sub-phases; the time periods are together denoted as t_{rd} in the figure). After the read phase, the current MTJ state is read out at the node Out. Next is the write phase,

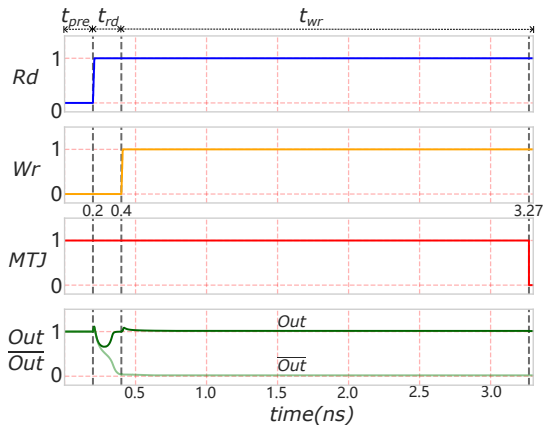

Fig. 16. Waveforms of key signals in a transient circuit simulation of RHS-SingleUnit for a single random bit generation cycle (3.3 ns).

where the "Wr" signal is pulled high to write the opposite state to the MTJ with a 50% probability. In the case shown in the figure, the MTJ switches within t_{wr}. Because of the probabilistic switching behavior, we can achieve random bit generation by setting t_{wr} to the mean of the random switching time distribution.

Through extensive experiments, we observed that t_{pre} and t_{rd} in Fig. 16 can be controlled below $0.2\,\mathrm{ns}$, respectively, and t_{wr} can be controlled at about $2.9\,\mathrm{ns}$. Hence, the random bit generation latency of each RHS-SingleUnit can be controlled at $3.3\,\mathrm{ns/bit}$. In summary, RHS-TRNG can achieve a generation rate of $303\,\mathrm{Mb/s}$ for a single cell, and will provide higher random bit output rates when parallelized as needed.

C. Statistical Randomness Test

The National Institute of Standards and Technology (NIST) SP 800–22 rev.1a Test [54] is used to test whether a set of binary bit sequences satisfies statistical randomness; it is often used to test the random number generation quality of RNGs. The NIST test consists of different test modules that examine the presence of non-random samples in a sequence of bits. For example, the test module "Frequency" checks whether the appearance frequencies of "0" and "1" in a bit sequence are approximately the same. The test module "Runs" is used to test whether the number of consecutive occurrences of the same bit such as "0000" or "111" is as expected. "FFT" is used to detect the peak height after the step-by-step discrete Fourier transform of a sequence, thus detecting the periodicity of the signals under test. Each test gives a P-value which quantifies the difference between a sample bit sequence under test and an ideal random bit sequence. When the P-value is greater than the threshold, the sample can be considered to pass this test. If the sequence length is large enough, some of the tests are executed multiple times and further give a pass rate. We consider the sample to pass the test when both the P-value and the pass rate are greater than their respective thresholds. Generally, the P-value threshold is 0.0001, and the pass rate threshold is 0.91. For a given bit sequence which passes all tests, it is considered to have good distribution characteristics in terms of statistical randomness.

TABLE III
NIST TEST RESULTS ON RANDOM BIT SEQUENCES GENERATED BY RHS-TRNG.

Test module		P-value	Pass rate	Pass/Fail
Frequency		0.911413	10/10	Pass
BlockFrequency		0.911413	10/10	Pass
CumulativeSums	Forward	0.213309	10/10	Pass
	Reverse	0.350485	10/10	Pass
Runs		0.739918	10/10	Pass
LongestRun		0.350485	10/10	Pass
Rank		0.350485	10/10	Pass
FFT		0.991468	10/10	Pass
NonOverlappingTemplate		-	1460/1480	Pass
OverlappingTemplate		0.534146	10/10	Pass
ApproximateEntropy		0.213309	10/10	Pass
Serial	Forward	0.213309	9/10	Pass
	Reverse	0.122325	9/10	Pass
LinearComplexity		0.739918	10/10	Pass

We generated one million random bits for NIST tests and divided them into 10 groups. To generate random bit sequences that avoid artificially inducing favorable results, we simulated the switching of MTJs from AP to P state and P to AP state separately, and then mixed them according to the mechanism proposed in our design, which captures the correlation between periods. The test results are shown in Table. III. It can be seen that we passed all the test modules in the table, which suggests that bit streams generated by RHS-TRNG have excellent statistical randomness.

D. Resilience Experiments Against PVT Variations

To evaluate the resilience enhancement effect of the two self-stabilization mechanisms in RHS-TRNG introduced in Section. IV, we experimentally evaluated its ability of tolerating PVT variations. The experimental results are also compared to that of traditional three-stage MTJ-based TRNG designs.

1) Output Entropy Concept: In our experiments, we use output entropy to quantify the random bit generation quality of TRNG. Entropy describes the chaotic degree of information in a system. The higher the entropy value, the more chaotic the system is. For a bit sequence, a high entropy value indicates that the sequence is well uniformly distributed.

Typically, two types of entropy are widely used, which are Shannon entropy and minimum entropy. For each bit in a sequence, its state is either "0" or "1". Assume that each bit is a random variable X, the Shannon entropy of this sequence is defined as:

$$H_{Shannon}(X) = -\sum_{x} P(x)\log_2 P(x), x \in \{0,1\}, \quad (4)$$

where $P(x)$ is the occurrence probability of x in the bit sequence. The range of $H_{Shannon}(X)$ is $[0, \log_2 m]$, where m is the number of all possible states of X ($m = 2$ in this case). One can easily derive that the maximum Shannon entropy is 1

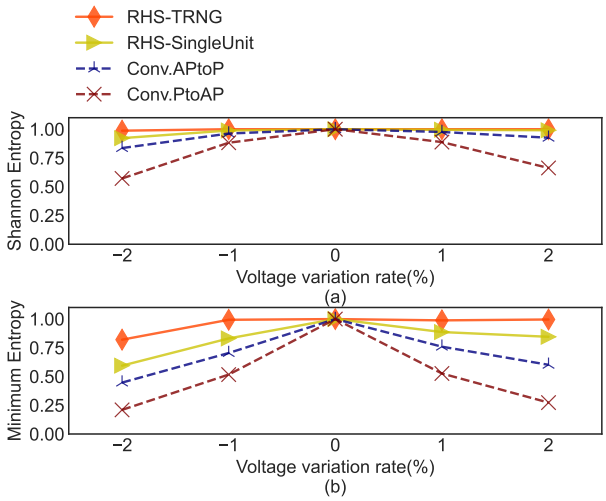

Fig. 17. Entropy changes of the generated random sequences using different TRNG designs under voltage variation.

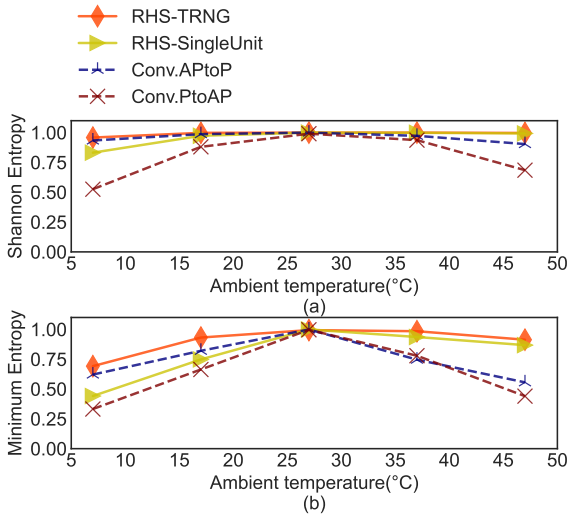

Fig. 18. Output entropy of different TRNG designs at different temperatures.

when the distribution probabilities of "0" and "1" are both 1/2. Similarly, for such a random variable X, its minimum entropy is defined as:

$$H_{Min}(X) = min(-\log_2 P(x)), x \in \{0, 1\}. \quad (5)$$

$H_{Min}(X)$ is also in the range $[0, 1]$. When the distribution of the bit sequence is non-random, the Shannon entropy is less than 1 and the minimum entropy is even smaller. The minimum entropy is the lower bound of entropy and represents the worst distribution of the random variable reflected by a sample. Combined with the Shannon entropy, we can estimate the range of fluctuation in the randomness of the distribution of the random variable.

2) Resilience Against Voltage Variation: In Section. IV-B, we have theoretically evaluated the improved output probabilistic stability using bidirectional switching currents and two generator units. We also performed solid experiments to evaluate the resilience advantages brought by the above-mentioned two mechanisms when compared to traditional TRNG designs. We varied the supply voltages V_{DD1} and V_{DD2} in a stepped manner, as shown in Fig. 8, for four configurations of MTJ-based TRNGs: Conv.APtoP, Conv.PtoAP, RHS-SingleUnit, RHS-TRNG. We calculated the entropy values for the generated random sequences under voltage variation; the results are shown in Fig. 17.a and Fig. 17.b for Shannon entropy and minimum entropy, respectively. When the voltage variation rate is 0, it means that V_{DD1} and V_{DD2} are at nominal values. A positive voltage variation rate presents an increase in the two supply voltages, while a negative value indicates a decrease in the voltages. The y-axis in the figure represents the entropy value in response to voltage variation; the closer to 1, the higher quality of the random sequence.

It can be seen that RHS-SingleUnit has higher Shannon entropy and minimum entropy than the two conventional TRNG designs, indicating that our design using bidirectional switching currents can increase the resilience of the circuit to voltage variation. We can also see that the Shannon entropy and minimum entropy of the sequence output by RHS-TRNG are significantly higher than those of RHS-SingleUnit, which suggests that the two generator unit mechanism can further improve the resilience of the circuit to voltage variation.

3) Resilience Against Process Variation: Similar to transistors, MTJ device parameters also fluctuate around their nominal values due to process variation. We took into account three key parameters t_{FL}, t_{TB}, and TMR in our experiments, where we assigned a Gaussian distribution to them with a variation percentage of 3% according to the MTJ model [53], as shown in Table. I. The experimental results are shown in Table. IV. It can be seen that under the influence of process variation, the minimum entropy of the sequence output by RHS-TRNG is still greater than 0.99. However, the minimum entropy of the other two sequences generated by the conventional MTJ-based TRNGs has dropped to 0.49 and 0.66. Due to the fact that process variations in MTJ devices have a significant impact on resistance, the resultant drop in output entropy is unacceptable without any mitigation designs. With our proposed scheme using bidirectional switching currents and two generator units, RHS-TRNG can effectively tolerate device parameter variations and guarantee high output entropy.

4) Resilience Against Temperature Variation: MTJ-based TRNG circuits may also be affected by various external conditions. This paper takes the ambient temperature variation as an example to study the ability of RHS-TRNG in resisting environmental variations. Fig. 18 shows the impact of ambient temperature on Shannon entropy and minimum entropy for the four different configurations. The temperature is swept from 7 °C to 47 °C, and the center of the x-axis is 27 °C.

TABLE IV
OUTPUT ENTROPY OF DIFFERENT TRNG DESIGNS WHEN PARAMETER VARIATION OF MTJ EQUIPMENT IS CONSIDERED.

	RHS-TRNG	**Conv.APtoP**	**Conv.PtoAP**
Shannon entropy	0.99996	0.94742	0.86682
Minimum entropy	0.99001	0.65707	0.49113

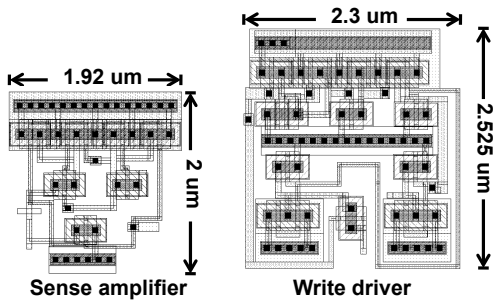

Fig. 19. Layout of the sense amplifier and the write driver for RHS-TRNG.

Fig. 18.a shows that the Shannon entropy of RHS-TRNG is always close to 1 across the entire temperature range. Compared to Conv.APtoP and Conv.PtoAP, the RHS-TRNG output sequence also provides higher minimum entropy. Interestingly, it can be observed that the ability of RHS-TRNG to withstand high-temperature changes is stronger than its ability to withstand low-temperature changes. These results indicate that the RHS-TRNG circuit is suitable for being integrated into a computer system as an acceleration component. We can also note that RHS-SingleUnit performs even less well in low temperatures than Conv.APtoP, which indicates that the bidirectional switching current mechanism does not cope well with low temperature variations. But when combined with two generator units mechanism, we can obtain the optimal output entropy.

E. Area and Power

We built RHS-TRNG circuits in Cadence Virtuoso; the circuit power consumption was evaluated through transient simulation and the on-chip area was evaluated through layout design. Using the GPDK $45\,\mathrm{nm}$ technology, the power consumption of RHS-TRNG is $1.6\,\mathrm{mW}$ for generating a random bit in a generation cycle of $3.3\,\mathrm{ns}$; that is, the power consumption of the TRNG is $5.3\,\mathrm{pJ/bit}$. The area of the MTJ device is based on the results presented by Vincent et al. [55]. Fig. 19 shows the layout design of the sense amplifier and write driver, with an area of $9.64\,\mu\mathrm{m}^2$ together. To form an RHS-SingleUnit, an NMOS transistor and an MTJ have to be added, resulting in a total area of $9.79\,\mu\mathrm{m}^2$. Furthermore, to construct a complete RHS-TRNG, two such units and an XOR gate are required, leading to a total area of $24.29\,\mu\mathrm{m}^2$ (see Fig. 6).

Our RHS-TRNG design also shows great scalability to achieve higher throughput. After further parallel expansion as shown in Fig. 9, the n parallel output bit sequences can be genarated by n+1 RHS-SingleUnits, with a single output bit occupying an area slightly larger than $14.5\,\mu\mathrm{m}^2$ and consuming slightly more than $2.65\,\mathrm{pJ}$. Although RHS-TRNG does not show outstanding performance in terms of power and area, the evaluation results are still within an acceptable range. It is worth noting that the power and area will be greatly reduced using more advanced MTJ and CMOS technologies. Detailed comparisons with other works will be presented in the next section.

F. System-Level Evaluation

By modeling RHS-TRNG in the architecture simulator, we can evaluate the performance acceleration of the benchmark program when it is integrated into the system.

In the extreme case, one random bit generation cycle of our TRNG unit consists of two processes with times of $0.4\,\mathrm{ns}$, and $2.9\,\mathrm{ns}$, respectively, so the highest supported main frequency can reach 3.3GHz. We list the dominant frequencies of the architecture modeled in the simulator in Table. II. Therefore, to adapt to the system clock, the running time of the three stages of TRNG will be relaxed to $0.5\,\mathrm{ns}$, $0.5\,\mathrm{ns}$ and $3\,\mathrm{ns}$, and the delay of each TRNG generation instruction is 8 ticks. We ran the Monte Carlo option pricing program in SE mode using gem5 to evaluate the acceleration effect at the system level.

1) Monte Carlo Option Pricing Benchmark: The Monte Carlo algorithm is widely used in scientific computing, finance, radiology and other fields. It is a representative class of applications that require massive high-quality random numbers. Monte Carlo simulation is often used for option pricing, risk management, and financial modeling in the financial field. It can deal with complex high-dimensional problems that are difficult to solve using traditional analytical methods, but the simulation time has always been a big concern for researchers. Malesevic introduces the background knowledge and corresponding procedures of option pricing using the Monte Carlo method in [56]. The benchmarks used in this paper are derived from the General Monte Carlo Method listed there. We rewrite the program of Hilpisch et al. [57] into a C++ program that can be compiled and run on gem5. The benchmark program can be configured using the following three methods to generate random numbers: 1) the rand function of the C++ stdlib, 2) the lagged_fibonacci1279 function of the Boost lib [58], and 3) the proposed custom RHS-TRNG instructions (see IV-C).

2) Performance Comparison: We ran the option pricing benchmark with the above-mentioned three different configurations on gem5. The system configurations can be found in Table. II.

Fig. 20.a compares the instruction count of the benchmark for the three random number generation methods. We normalized all values to that of RHS-TRNG for the sake of comparison. When the number of Monte Carlo simulations is 1e2, the instruction counts for the benchmark using Boost lib and C++ stdlib are 2 and 1 times larger than that of the RHS-TRNG, respectively. This is because when the number of simulations is small, the random number generation part does not occupy much of the total program runtime. As the Monte Carlo simulation number increases, the advantage of RHS-TRANG-enpowered system starts to stand out. It can be seen that these two multiples will converge at 9.5 and 3.3 times When the simulation time reaches 1e6. This is due to the fact that after reaching the number of simulations that make the results converge, the main overhead of the program is spent on generating random numbers, of which our custom instructions reduce the number of instructions significantly.

Fig. 20.b illustrates the difference of program runtime of the benchmark at different Monte Carlo simulation numbers. We can observe that when the simulation number exceeds 1e5, the speedup effect of our RHS-TRNG on the option pricing

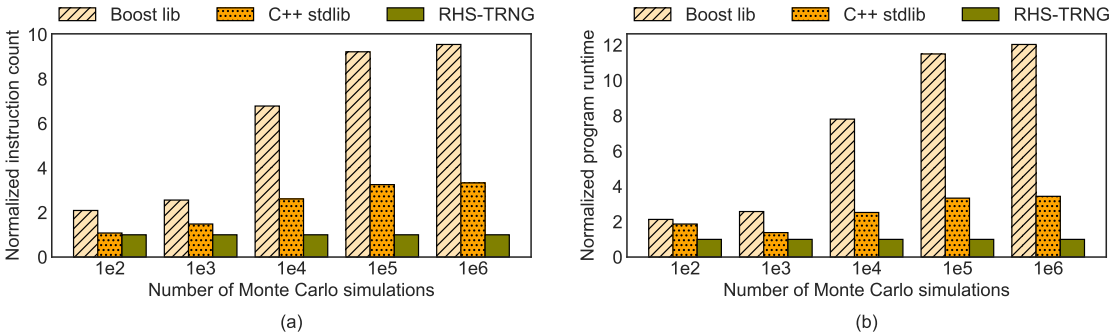

Fig. 20. System performance comparison using option pricing benchmark which generates random numbers by Boost lib, C++ stdlib, and RHS-TRNG.

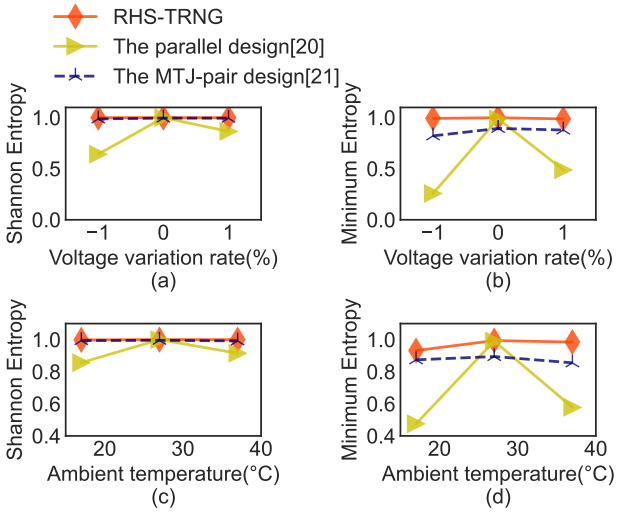

Fig. 21. Comparison of Shannon entropy and minimum entropy between RHS-TRNG, the parallel design in [20], and the MTJ-pair design in [21] under voltage and temperature variations.

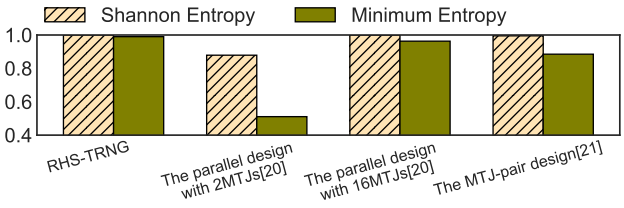

Fig. 22. Comparison of Shannon entropy and minimum entropy under MTJ process variation between RHS-TRNG, the parallel design in [20], and the MTJ-pair design in [21].

program starts to level off. In summary, using RHS-TRNG achieves 3.4–12× performance acceleration in comparison to the other two software-based RNGs. This result is consistent with the improvement in the number of instructions.

VI. Related Work

In this section, we compare our RHS-TRNG design with some state-of-the-art TRNG designs, including DRAM-based and MTJ-based, and highlight the advantages of this work.

Olgun et al. [45] proposed a quadruple-activation (QUAC) TRNG design based on commodity DRAM chips. QUAC-TRNG activates four memory rows that store conflicting data simultaneously through several consecutive DRAM commands that violate timing constraints. This causes the bitline sense amplifiers to non-deterministically converge to random values, which are considered as a random bit sequence. QUAC-TRNG generates 7664 random bits every $1940\,\mathrm{ns}$, equivalent to a throughput rate of $3.44\,\mathrm{Gb/s}$ per DRAM channel. Despite the claimed high throughput, QUAC-TRNG is based on the configuration of bank group parallelism and row clone, which requires the use of a whole DRAM block; it also conflicts with normal DRAM accesses. In contrast, RHS-TRNG obtains a tenth of its throughput with only a single memory cell. By exploiting cell-level parallelism, the throughput of RHS-TRNG can easily outperform that of QUAC-TRNG. Moreover, RHS-TRNG consumes much less energy since it does not require the periodic refresh operations ($\sim 60\,\mathrm{ms}$) in DRAM.

Perach et al. [18] designed an MTJ-based asynchronous TRNG for low-power edge devices. By using the capacitor discharge as the current excitation source for the MTJ, this asynchronous design decouples the random number generation process from the system clock. In terms of entropy generation rate, a random bit generation cycle is divided into a charging phase, a enable phase, and a read phase. They take $66\,\mathrm{ns}$, $10\,\mathrm{ns}$, and $2.8\,\mathrm{ns}$, respectively; thus one cycle is $78.8\,\mathrm{ns}$. In contrast, the entropy generation rate of our RHS-TRNG is nearly $24\times$ higher. Although the entropy generation rate can be improved by parallelizing TRNG cells, the use of capacitors as the excitation source of MTJ limits its scalability to 8 bits.

Amirany et al. [17] designed a TRNG based on a neuromorphic variation-tolerant spintronic structure. It uses 4 MTJs to control the generation of a random bit and uses XNOR as a post-processing circuit to ensure the stability of the output probability. The proposed neuromorphic spin-based TRNG has typical reset-write-read three phases, where the first two phases take $5\,\mathrm{ns}$ and $10\,\mathrm{ns}$, respectively. So the entropy generation rate of our work is about 4.5x higher, and a longer MTJ lifetime is obtained due to less writing.

Qu et al. proposed two variation-tolerant TRNG designs, which are referred to as parallel design [20] and MTJ-pair design [21] hereafter. The former utilizes a parallel structure to mitigate the variation effect, while the latter leverages

TABLE V
COMPARISON OF RHS-TRNG AND OTHER MTJ-BASED TRNG DESIGNS.

TRNG	**Throughput** (Mb/s)	**Energy Consump** (pJ/bit)	**Area** (μm^2)
[59]	0.5-100×10^{-3}	2-20×10^{-3}	2
[18]	7.7-15.1	5.7-13.4	50.6-200.6
[17]	50	1.1	219
[20]	66.7-177.8	0.6-0.8	3.8-7.6
[21]	66.7	0.8	3.84
This work	303	2.65-5.3	14.5-24.29

the symmetry of two MTJs to eliminate correlation. We compared these two designs with our proposed RHS-TRNG, and evaluated their abilities to cope with PVT variations under consistent parameters and simulation conditions. For consistency, we configured the parallel design with two MTJs in parallel, adding a set of 16 MTJs in parallel configurations in the evaluation process variation. The voltage and temperature tolerance of the three designs is shown in Fig. 21. It can be observed that the parallel design exhibits insufficient resilience to voltage and temperature variations, as the MTJs configured in parallel are subject to the same variations under both conditions. Fig. 22 presents a comparison of the three designs under process variations. When only two MTJs are used, the parallel design exhibits a Shannon entropy of only 0.88 and a minimum entropy as low as 0.51 for outputting random sequence under process variations, failing to tolerate process variations. Only by scaling up to 16 MTJs does it achieve a Shannon entropy above 0.99 and a minimum entropy of 0.96, which is slightly lower than our design. The MTJ-pair design can effectively handle all three variations, but its mechanism may lead to errors (such as switching of both MTJs), resulting in lower minimum entropy compared to our design. The generation rates of the two designs are 66.7-177.8 Mb/s and 66.7 Mb/s, respectively, which are much lower than our design (303 Mb/s).

Table. V compares RHS-TRNG with five representative MTJ-based TRNGs in the literature at three aspects: throughout, energy consumption, and area, and ranks them in ascending order of throughput rate. As can be seen, RHS-TRNG provides the highest throughout, while having acceptable area and energy consumption.

VII. DISCUSSION

In this section, we discuss additional topics of interest for future research work. First, emerging spin devices such as VCMA and SOT also have the potential to serve as TRNG entropy sources while offering lower power consumption and switching overheads. Nevertheless, ensuring their reliability may require more complex device and circuit designs, which are topics worth investigating. Second, our design enables the integration of integer and floating-point TRNGs into computing systems. We evaluated the integer TRNG for applications, but the floating-point design compromised precision to some extent. It is worth exploring how to provide high-precision floating-point random numbers for applications that demand such accuracy. Third, TRNGs can offer potential benefits such as high throughput and statistical quality. We evaluated the effect of high throughput on applications using option pricing as an example. However, integrating TRNGs into systems to improve statistical quality for applications such as cryptography still poses a challenge and will be an interesting research direction for future work.

VIII. CONCLUSION

In this paper, we have presented a self-stabilized STT-MTJ-based TRNG: RHS-TRNG. It generates high-quality random bit sequences at the maximum speed of $303\,\mathrm{Mb/s}$, which is higher than all prior works to the best of our knowledge. By exploiting cell-level parallelism, higher random bit throughput can be supplied depending on the need of target applications. RHS-TRNG not only exhibits a strong immunity against PVT variations but only has a longer lifetime, thanks to our circuit design with bidirectional switching currents and dual generator units. We have also integrated RHS-TRNG into a RISC-V processor and demonstrated that it can significantly accelerate programs that have a strong demand for random numbers, e.g., the Monte Carlo option pricing program. With seamless circuit/system co-design, this work also demonstrates that Spintronics can be a great driving force to further boost computing system performance in the post-Moore era.

REFERENCES

[1] L. González-Manzano *et al.*, "Encryption by heart (EbH)-using ECG for time-invariant symmetric key generation," *Future Generation Comput. Syst.*, vol. 77, pp. 136–148, 2017, doi:10.1016/j.future.2017.07.018.

[2] R. Abboud *et al.*, "The surprising power of graph neural networks with random node initialization," *arXiv preprint arXiv:2010.01179*, 2020, doi:10.48550/arXiv.2010.01179.

[3] X. Jia *et al.*, "SPINBIS: Spintronics-based Bayesian inference system with stochastic computing," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 39, pp. 789–802, 2019, doi:10.1109/TCAD.2019.2897631.

[4] J.B. Plumstead, "Inferring a sequence generated by a linear congruence," in *23rd Annual Symp. Foundations Comput. Science*, 1982, pp. 153–159, doi:10.1109/SFCS.1982.73.

[5] M.S. Hashemian *et al.*, "A robust authentication methodology using physically unclonable functions in DRAM arrays," in *IEEE Design Autom. Test Europe Conf.*, 2015, pp. 647–652, doi:10.7873/DATE.2015.0308.

[6] S. Sutar *et al.*, "D-PUF: An intrinsically reconfigurable DRAM PUF for device authentication and random number generation," *ACM Trans. Embedded Comput. Syst.*, vol. 17, pp. 1–31, 2017, doi:10.1145/3105915.

[7] C. Eckert *et al.*, "DRNG: DRAM-based random number generation using its startup value behavior," in *IEEE Int. Midwest Symp. Circuits Syst.*, 2017, pp. 1260–1263, doi:10.1109/MWSCAS.2017.8053159.

[8] C. Pyo *et al.*, "DRAM as source of randomness," *Electron. Lett.*, vol. 45, pp. 26–27, 2009.

[9] J.S. Kim *et al.*, "D-RaNGe: Using commodity DRAM devices to generate true random numbers with low latency and high throughput," in *IEEE Int. Symp. High Performance Comput. Architecture*, 2019, pp. 582–595, doi:10.1049/el:20091899.

[10] E. Kim *et al.*, "8.2 8Mb/s 28Mb/mJ robust true-random-number generator in 65nm CMOS based on differential ring oscillator with feedback resistors," in *IEEE Int. Solid-State Circuits Conf.*, 2017, pp. 144–145, doi:10.1109/ISSCC.2017.7870302.

[11] S. Choi *et al.*, "Analysis of ring-oscillator-based true random number generator on FPGAs," in *IEEE Int. Conf. Electron. Inf. Commun.*, 2021, pp. 1–3, doi:10.1109/ICEIC51217.2021.9369714.

[12] V.B. Suresh *et al.*, "Robust metastability-based TRNG design in nanometer CMOS with sub-vdd pre-charge and hybrid self-calibration," in *IEEE Int. Symp. Quality Electron. Design*, 2012, pp. 298–305, doi:10.1109/ISQED.2012.6187509.

[13] C. Li *et al.*, "A metastability-based true random number generator on FPGA," in *IEEE Int. Conf. ASIC*, 2017, pp. 738–741, doi:10.1109/ASICON.2017.8252581.

[14] N. Liu *et al.*, "A true random number generator using time-dependent dielectric breakdown," in *IEEE Symp. VLSI Circuits - Dig. Tech. Papers*, 2011, pp. 216–217.

[15] J. Stöhr *et al.*, "Magnetism," *Solid-State Sciences. Springer, Berlin, Heidelberg*, vol. 5, p. 236, 2006.

[16] Y. Wang *et al.*, "A novel circuit design of true random number generator using magnetic tunnel junction," in *IEEE/ACM Int. Symp. Nanoscale Architectures*, 2016, pp. 123–128, doi:10.1145/2950067.2950108.

[17] A. Amirany *et al.*, "True random number generator for reliable hardware security modules based on a neuromorphic variation-tolerant spintronic structure," *IEEE Trans. Nanotechnol.*, vol. 19, pp. 784–791, 2020, doi:10.1109/TNANO.2020.3034818.

[18] B. Perach *et al.*, "An asynchronous and low-power true random number generator using STT-MTJ," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 27, pp. 2473–2484, 2019, doi:10.1109/TVLSI.2019.2927816.

[19] E.I. Vatajelu *et al.*, "High-entropy STT-MTJ-based TRNG," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 27, pp. 491–495, 2018, doi:10.1109/TVLSI.2018.2879439.

[20] Y. Qu *et al.*, "A true random number generator based on parallel STT-MTJs," in *IEEE Design Autom. Test Europe Conf.*, 2017, pp. 606–609, doi:10.23919/DATE.2017.7927058.

[21] Y. Qu *et al.*, "Variation-resilient true random number generators based on multiple STT-MTJs," *IEEE Trans. Nanotechnol.*, vol. 17, pp. 1270–1281, 2018, doi:10.1109/TNANO.2018.2873970.

[22] M. Morsali *et al.*, "A process variation resilient spintronic true random number generator for highly reliable hardware security applications," *Microelectronics Journal*, vol. 129, p. 105606, 2022, doi:10.1016/j.mejo.2022.105606.

[23] K.C. Chun *et al.*, "A scaling roadmap and performance evaluation of in-plane and perpendicular MTJ based STT-MRAMs for high-density cache memory," *IEEE J. solid-state circuits*, vol. 48, pp. 598–610, 2012, doi:10.1109/JSSC.2012.2224256.

[24] W.J. Gallagher *et al.*, "Development of the magnetic tunnel junction MRAM at IBM: From first junctions to a 16-Mb MRAM demonstrator chip," *IBM J. Research Develop.*, vol. 50, pp. 5–23, 2006, doi:10.1147/rd.501.0005.

[25] J. Chen *et al.*, "Tunable linear magnetoresistance in MgO magnetic tunnel junction sensors using two pinned CoFeB electrodes," *Applied Physics Lett.*, vol. 100, p. 142407, 2012, doi:10.1063/1.3701277.

[26] J. Chen *et al.*, "Yoke-shaped MgO-barrier magnetic tunnel junction sensors," *Applied Physics Lett.*, vol. 101, p. 262402, 2012, doi:10.1063/1.4773180.

[27] S. Ikeda *et al.*, "Magnetic tunnel junctions for spintronic memories and beyond," *IEEE Trans. Electron Devices*, vol. 54, pp. 991–1002, 2007, doi:10.1109/TED.2007.894617.

[28] S. Yuasa *et al.*, "Materials for spin-transfer-torque magnetoresistive random-access memory," *MRS Bulletin*, vol. 43, pp. 352–357, 2018, doi:10.1557/mrs.2018.93.

[29] J. Mathon *et al.*, "Theory of tunneling magnetoresistance of an epitaxial Fe/MgO/Fe (001) junction," *Physical Review B*, vol. 63, p. 220403, 2001, doi:10.1103/PhysRevB.63.220403/10.1103/PhysRevB.63.220403.

[30] C. Park *et al.*, "Systematic optimization of 1 Gbit perpendicular magnetic tunnel junction arrays for 28 nm embedded STT-MRAM and beyond," in *IEEE Int. Electron Devices Meeting*, 2015, pp. 26.2.1–26.2.4, doi:10.1109/IEDM.2015.7409771.

[31] L. Wu *et al.*, "Defect and fault modeling framework for STT-MRAM testing," *IEEE Trans. Emerging Topics Comput.*, vol. 9, pp. 707–723, 2019, doi:10.1109/TETC.2019.2960375.

[32] J. Sun *et al.*, "Commercialization of 1Gb standalone spin-transfer torque MRAM," in *IEEE Int. Memory Workshop*. IEEE, 2021, pp. 1–4, doi:10.1109/IMW51353.2021.9439616.

[33] Y.D. Chih *et al.*, "13.3 A 22nm 32Mb embedded STT-MRAM with 10ns read speed, 1M cycle write endurance, 10 years retention at 150 °C and high immunity to magnetic field interference," in *2020 IEEE Int. Solid- State Circuits Conf.*, 2020, pp. 222–224, doi:10.1109/ISSCC19947.2020.9062955.

[34] Y.K. Lee *et al.*, "Embedded STT-MRAM in 28-nm FDSOI logic process for industrial MCU/IoT application," in *IEEE Symp. VLSI Technol.*, 2018, pp. 181–182, doi:10.1109/VLSIT.2018.8510623.

[35] V.B. Naik *et al.*, "Manufacturable 22nm FD-SOI embedded MRAM technology for industrial-grade MCU and IOT applications," in *IEEE Int. Electron Devices Meeting*, 2019, pp. 2.3.1–2.3.4, doi:10.1109/IEDM19573.2019.8993454.

[36] B.N. Engel *et al.*, "A 4-Mb toggle MRAM based on a novel bit and switching method," *IEEE Trans. Magnetics*, vol. 41, pp. 132–136, Jan. 2005, doi:10.1109/TMAG.2004.840847.

[37] W.J. Gallagher *et al.*, "22nm STT-MRAM for reflow and automotive uses with high yield, reliability, and magnetic immunity and with performance and shielding options," in *IEEE Int. Electron Devices Meeting*, Dec. 2019, pp. 2.7.1–2.7.4, doi:10.1109/IEDM19573.2019.8993469.

[38] N. Sato, "CMOS compatible process integration of SOT-MRAM with heavy-metal bi-layer bottom electrode and 10ns field-free SOT switching with STT assist," in *IEEE Symp. VLSI Technol.*, Jun. 2020, pp. 1–2, doi:10.1109/VLSITechnology18217.2020.9265028.

[39] A. Khvalkovskiy *et al.*, "Basic principles of STT-MRAM cell operation in memory arrays," *J. Physics D: Applied Physics*, vol. 46, p. 074001, 2013, doi:10.1088/0022-3727/46/7/074001.

[40] L. Wu *et al.*, "Survey on STT-MRAM testing: Failure mechanisms, fault models, and tests," *arXiv preprint*, pp. 1–24, Jan. 2020, arXiv:2001.05463.

[41] T. Seki *et al.*, "Switching-probability distribution of spin-torque switching in MgO-based magnetic tunnel junctions," *Applied Physics Lett.*, vol. 99, p. 112504, 2011, doi:10.1063/1.3637545.

[42] B. Dang *et al.*, "Physically transient true random number generators based on paired threshold switches enabling Monte Carlo method applications," *IEEE Electron. Device Lett.*, vol. 40, pp. 1096–1099, 2019, doi:10.1109/LED.2019.2919914.

[43] P. Ziegenhein *et al.*, "Fast cpu-based monte carlo simulation for radiotherapy dose calculation," *Physics in Medicine & Biology*, vol. 60, p. 6097, jul 2015, doi:10.1088/0031-9155/60/15/6097.

[44] Z. Hou *et al.*, "Reconfigurable and dynamically transformable in-cache-MPUF system with true randomness based on the SOT-MRAM," *IEEE Trans. Circuits and Systems I: Regular Papers*, vol. 69, pp. 2694–2706, 2022, doi:10.1109/TCSI.2022.3168133.

[45] A. Olgun *et al.*, "QUAC-TRNG: High-throughput true random number generation using quadruple row activation in commodity DRAM chips," in *ACM/IEEE Annual Int. Symp. Computer Architecture*, 2021, pp. 944–957, doi:10.1109/ISCA52012.2021.00078.

[46] C. Yoshida *et al.*, "A study of dielectric breakdown mechanism in CoFeB/MgO/CoFeB magnetic tunnel junction," in *IEEE Int. Rel. Physics Symp.*, 2009, pp. 139–142, doi:10.1109/IRPS.2009.5173239.

[47] L. Wu *et al.*, "Characterization, modeling, and test of intermediate state defects in STT-MRAMs," *IEEE Trans. Comput.*, pp. 1–4, 2021, doi:10.1109/TC.2021.3125228.

[48] X. Bi *et al.*, "STT-RAM cell design considering CMOS and MTJ temperature dependence," *IEEE Trans. Magn.*, vol. 48, pp. 3821–3824, 2012, doi:10.1109/TMAG.2012.2200469.

[49] W.H. Choi *et al.*, "A magnetic tunnel junction based true random number generator with conditional perturb and real-time output probability tracking," in *2014 IEEE Int. Electron Devices Meeting*. IEEE, 2014, pp. 12–5, doi:10.1109/IEDM.2014.7047039.

[50] S. Oosawa *et al.*, "Design of an STT-MTJ based true random number generator using digitally controlled probability-locked loop," in *2015 IEEE 13th Int. New Circuits and Syst. Conf. (NEWCAS)*. IEEE, 2015, pp. 1–4, doi:10.1109/NEWCAS.2015.7182089.

[51] W. Zhao *et al.*, "High speed, high stability and low power sensing amplifier for MTJ/CMOS hybrid logic circuits," *IEEE Trans. Magn.*, vol. 45, pp. 3784–3787, 2009, doi:10.1109/TMAG.2009.2024325.

[52] L. Wu, "Testing STT-MRAM: Manufacturing defects, fault models, and test solutions," Ph.D. dissertation, Delft University of Technology, 2021, doi:10.4233/uuid:088a3991-4ea9-48a0-9b92-cc763748868c.

[53] Y. Wang *et al.*, "Compact model of magnetic tunnel junction with stochastic spin transfer torque switching for reliability analyses," *Microelectronics Reliability*, vol. 54, pp. 1774–1778, 2014, doi:10.1016/j.microrel.2014.07.019.

[54] L.E. Bassham III *et al.*, *Sp 800-22 rev. 1a. a statistical test suite for random and pseudorandom number generators for cryptographic applications*. National Institute of Standards & Technology, 2010.

[55] A.F. Vincent *et al.*, "Analytical macrospin modeling of the stochastic switching time of spin-transfer torque devices," *IEEE Trans. Electron Devices*, vol. 62, pp. 164–170, 2014, doi:10.1109/TED.2014.2372475.

[56] G. Malesevic, "Use of the Monte Carlo simulation in valuation of european and american call options," Ph.D. dissertation, Lake Forest College, 2017.

[57] Y. Hilpisch, *Python for Finance: Analyze big financial data*. " O'Reilly Media, Inc.", 2014.

[58] B. Schäling, *The boost C++ libraries*. Boris Schäling, 2011, doi:10.1007/978-1-4419-7719-9_6.

[59] D. Vodenicarevic *et al.*, "Low-energy truly random number generation with superparamagnetic tunnel junctions for unconventional computing," *Physical Review Applied*, vol. 8, p. 054045, 2017, doi:10.1103/PhysRevApplied.8.054045.