This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

True Random Number Generation Using Read Noise of Flash Memory Cells

Biswajit Ray, *Member, IEEE*, and Aleksandar Milenković, *Senior Member, IEEE*

***Abstract*— In this paper, we propose and demonstrate a novel technique for true random number generation using commercial off-the-shelf Flash memory. Flash memory cells are known to exhibit thermal noise and random telegraph noise during sensing of their threshold voltage. In order to extract this inherent noise properties of the Flash memory bits through a standard digital Flash memory interface, we utilize the program disturb and read noise characteristics, which are fundamental properties of all NAND Flash memory arrays. The proposed technique is experimentally demonstrated and evaluated using state-of-art Flash memory chips. The experimental evaluation shows that the proposed technique enables extraction of high quality, high throughput, controllable (or tunable), and temperature- and aging-tolerant random bits. The random bits generated by the proposed technique pass all tests in the National Institute of Standards and Technology statistical test suite. The advantages of the proposed technique are as follows: 1) it is cost-effective as it does not require any special circuitry or hardware modification; 2) it is tolerant to aging and temperature effects; 3) it is easy to implement in software and to deploy through software updates; and 4) it is widely applicable to all electronic devices utilizing modern NAND Flash memory chips.**

***Index Terms*— NAND flash memory, program disturb, random telegraph noise (RTN), true random number.**

I. Introduction

RANDOM numbers are the cornerstone of many cryptographic primitives and secure communication protocols. Pseudorandom number generators, typically used in modern systems due to their high throughput and ease of implementation, cannot provide true randomness and hence are vulnerable to cyber-attacks [1]. True random number generators (TRNGs) can provide true randomness, but their speed is typically low and their implementation remains too complex for many practical applications. Over the last few years, there have been several proposals of TRNGs that rely on some physical processes, such as radioactive decay, single photon optical processes, Brownian motion, clock jitters, noise in electronics devices, and others. However, the complexity involved in randomness extraction from such sources often makes the proposed TRNGs impractical for many emerging applications that rely on resource-constrained systems (e.g., Internet-of-Things and sensor networks). Such systems would greatly benefit from low-cost, easily implementable, robust, high-speed TRNGs.

In this paper, we propose a novel technique to extract true random numbers utilizing the program disturb and read noise characteristics of commercially available Flash memory cells. Read operations in Flash memory involve sensing the threshold voltage of floating gate transistors (or FG-MOSFET), which are either at programmed (high V_t and bit "0") or erased state (low V_t and bit "1"). It is well known that V_t of an FG-MOSFET fluctuates due to noise, such as random telegraph noise (RTN) and thermal noise [2]–[6]. A typical design of Flash memory provides enough voltage margin between the erased and programmed states in order to fight against the noise during a read operation. We utilize the program disturb characteristics of Flash memory in order to override the voltage margin, which ensures that more bits exhibit noisy behavior during read operations. We develop an algorithm for automatic selection of bits with noisy behavior and subsequent conversion of their noise characteristics into a stream of random binary bits.

The notion of electronic hardware-based random number generators is not new and has been extensively studied over the last few years. For example, many FPGA-based TRNGs utilize the logical state unpredictability of metastable flip-flops [7]–[12]. Similarly, DRAM-based TRNGs utilize the memory-specific properties, such as memory cells' remanence effect [13], or access time variation caused by refresh cycles [14]. The other interesting TRNG proposals include an oscillator-based TRNG [15], a technology-independent TRNG [16], a portable TRNG using smartphone camera [17], a TRNG based on hot-carrier injection-balanced metastable sense amplifiers [18], and TRNGs based on switching instability in emerging memory technologies, such as resistive random access memory [19]–[23], spin transfer-torque magnetic memory [24], [25], phase change memory [26], and carbon nanotube-based memory [27]. All the proposed TRNGs represent a great advancement in the field and meet the required application-specific requirements. However, there remains a few important downsides to most of these TRNG constructions. First, some require specific hardware or dedicated circuitry to extract the randomness from the physical entities

Manuscript received September 7, 2017; revised November 8, 2017 and December 14, 2017; accepted January 8, 2018. The review of this paper was arranged by Editor G.-H. Koh. *(Corresponding author: Biswajit Ray.)*

The authors are with the Electrical and Computer Engineering Department, University of Alabama in Huntsville, Huntsville, AL 35899 USA (e-mail: biswajit.ray@uah.edu; milenka@uah.edu).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TED.2018.2792436

U.S. Government work not protected by U.S. copyright.

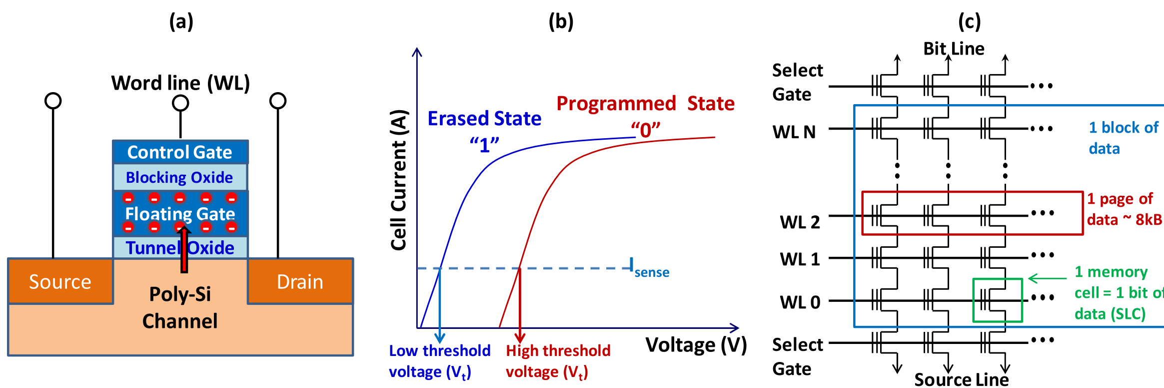

Fig. 1. (a) Floating gate NAND Flash memory cell. (b) Typical current–voltage characteristics of the memory cell. (c) Hierarchical storage in NAND Flash array consisting of kB of memory cells connected through a single WL (called a page of information). A group of word line forms a block of memory. The select gate transistors can be standard MOSFET or floating gate transistors, depending on manufacturers or technology node.

on the device. Second, the throughput of TRNGs is generally relatively low. Third, the implementation of some of the proposed TRNGs requires few system-prohibited or inconvenient operations, such as hardware reset and power on cycle.

Utilizing the commercial off-the-shelf Flash memory as a TRNG has several advantages compared with many existing TRNGs. First, utilizing Flash memory as a TRNG makes the proposal widely accessible in many applications. Currently, Flash memory is widely used in electronics gadgets, such as USB memory sticks, SD memory cards, microcontrollers, smartphones, and solid-state drives of modern laptop and desktop computers. Second, the random number extraction technique outlined in this paper does not require any hardware change or any specific circuit design or any system-prohibited (or privileged) operation, such as hardware reset or power on cycle. Hence, it can be implemented as system software or firmware updates in electronic devices that contain a Flash memory chip. Third, the quality of the random bits from Flash memory will not be compromised at different temperatures or with aging of the device, because read-noise exists at wide temperature ranges and in worn out memory blocks.

The Flash-based TRNG proposal in this paper is not the first of its kind. Wang *et al.* [28] previously proposed Flash-based TRNG that utilizes the RTN. However, our proposal improves the Flash-based TRNG as follows. First, Wang *et al.* utilized partial programming technique to force the memory cells into an unreliable state which can be affected by noise. Partial programming requires precise control of program time by issuing RESET commands to halt the Flash programming at right time that is chip-specific. Our technique hinges upon the utilization of program disturb characteristics by repeated programming of the same page. The number of repeated program cycles can be easily controlled and calibrated by software for all Flash technologies or manufacturers. Second, the proposal by Wang *et al.* requires periodic refresh operation for identifying RTN bits due to finite data retention property of flash memory bits. Our technique does not require any periodic refresh as it is not dependent on a fixed set of RTN bits. Third, throughput from our technique is much higher than from the method proposed by Wang *et al.* as our proposal requires minimal setup time. The repeated programming is done only once. Once the memory is set up to the unreliable state, it remains unreliable for more than data-retention time limit which is more than ten years at room temperature (RT). Over the time, some specific noisy bits may disappear, however, new bits appear to show noisy characteristics so that the total number of noisy bits remains unchanged. Hence, unlike the previous method, we do not need to refresh periodically the programmed bits.

The key contributions in this paper are as follows.

1) We introduce a novel TRNG that utilizes the program disturb characteristics of Flash memory to upset the memory states to an unreliable one in order to facilitate the extraction of noise from the Flash bits.
2) The proposed TRNG is: a) cost-effective, as it does not require any special circuitry or hardware modification; b) easy to implement in software and deploy through software or firmware updates; and c) widely applicable to many electronic gadgets which already use Flash memory for data storage.
3) The proposed method is found to be robust against temperature and aging.

The rest of this paper is organized as follows. Section II provides the background on the Flash memory cells with emphasis on read noise and program disturb characteristics. Section III outlines the step-by-step procedure to generate random numbers. Section IV discusses the experimental results for the proposed technique on Flash chips by evaluating the randomness of the generated random bits using the National Institute of Standards and Technology (NIST) test suite. We also provide program disturb and noise characteristics data at various temperature and aging conditions to support the robustness of our algorithm. Finally, Section V concludes this paper and discusses different application scenarios.

II. Fundamentals of Flash Memory

A. Device Structure and Operation of Flash Memory Cell

The device structure of a floating gate Flash memory cell (FG-MOSFET) is shown in Fig. 1(a) and its I–V characteristics in Fig. 1(b). It is essentially to have an MOSFET

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

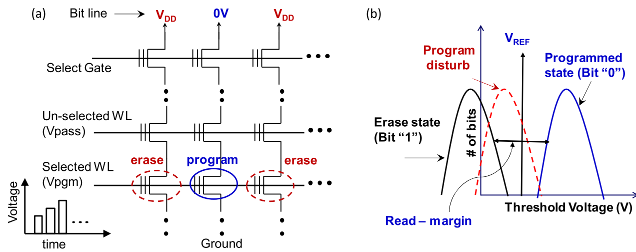

Fig. 2. (a) Biasing scheme for the NAND array during program pulse. Program pulses (V_{pgm}) are applied on the WL of the page selected for programming. WLs of the remaini ng pages in the block are biased at a high voltage (V_{pass}). BL voltage corresponding to the programmed cell is grounded, whereas the BLs for the erased cell is biased at higher voltage. (b) Threshold voltage distribution of memory cells on selected WLs (solid lines). The red dashed line represents the upshifted erase state distribution with repeated program disturb.

transistor with an additional conducting or charge trap layer insulated between the control gate and the channel. Information or binary bits are stored in the form of electric charges on the floating gate (or the charge trap layer). Flash memory offers three basic operations: program, erase, and read. During program operation, charges are injected into the floating gate from the transistor channel by Fowler–Nordheim tunneling through the tunnel oxide. After programming, electron density or negative charges rises on the floating gate, increasing the threshold voltage of the FG-MOSFET. This programmed state of the memory cell represents the binary bit "0" in single bit per cell (SLC) technology. The cells are erased by applying a high positive voltage on the substrate of the transistors while the control gate is grounded. This erased state will have lower V_t and it represents logic "1." During read operation, a fixed "read reference voltage" (V_{REF}), which is lower than the programed voltage, is applied on the control gate. Based on V_t of the transistor, whether the cell conducts current or not will be sensed as bit "1" or "0," respectively, by the sense circuitry of the Flash memory.

The memory cells are organized in a memory chip as multiple 2-D arrays known as Flash blocks as shown in Fig. 1(c). The cells in each row of a block are electrically connected through a single word line (WL), which acts as the control gate of individual FG transistors. Each column in a block is connected to a different bit line (BL). Each block consists of multiple WLs (typically 32–64 per block) which correspond to different page addresses. Read and program operations are performed at the page granularity, whereas erase is performed at the block granularity.

B. Program Disturb

Program disturb is the phenomena of unintentional programming of the erased cells during the program operation. In a NAND memory array, the program operation is performed on all the bits in a page at the same time. However, based on the user data, some cells will be programmed and some cells will remain erased. The distinction between programmed and erased cells is done through different BL voltages as shown in Fig. 2(a). BLs of the programmed cells remain grounded during programming, pulling down their channel potential to 0 V, which enhances programming efficiency. On the contrary, high voltage is applied on the BLs corresponding to the erased cells, which turns OFF the select transistors of the NAND chain and forces the channel potential to float up at higher voltage through coupling with voltages on unselected WLs. This methodology is called "self-boosting" which inhibits programming of the erased cells.

Even with channel boosting, unintentional (weak) programming may take place on erased cells due to loss of boosting potential through any leakage path of the FG-transistor network (e.g., fabrication preexisting defects). If we repeatedly program the same page with the same data, the program disturb effect will add up cumulatively on the erased cells and it will show up as error bits after a certain number of repeated programming cycles. In this proposal, we utilize this effect in order to bring the Flash memory into an unreliable state so that we can extract noise property from the error bits.

C. Read Noise

Read operation involves sensing of threshold voltage. During sensing, a "sense voltage" is applied on the BLs and V_{REF} is applied on the control gate (or WL) of the selected page. All other unselected WLs are biased at high voltage. Current will flow from the BL to the ground through the bits whose $V_t < V_{\text{REF}}$. As shown in Fig. 2(b), V_{REF} is chosen in between the erased and programmed state distributions, so that there is enough margin to correctly identify the bit states.

During the sensing operation, cell current fluctuates due to thermal noise or RTN, which may cause false identification of the erased cell as programmed cell or vice versa. In this paper, we refer to "read noise" as a combination of thermal noise and RTN, since both coexist in all practical Flash memory chips. Thermal noise is white noise generated by the thermal agitation of the charge carriers and it exists in nearly all electronic devices. RTN is usually caused by the random trapping and detrapping of charge carriers at semiconductor–oxide interfaces. In Flash memory, the defects that cause RTN are located in the tunnel-oxide near or at the interface of Si-channel surface through which read current flows. The amplitude of noise fluctuation in current/voltage due to RTN is inversely proportional to the gate area. As Flash memory cells scale down, RTN effects become relatively stronger and their impact on the V_t distribution of memory cells can be significant as reported in the recent literature.

III. Algorithm for Random Number Generation

In order to observe the noise characteristics of Flash memory bits using digital interface, the absolute V_t of the cell needs to be close enough to the read reference voltage. In a typical memory operation, there remains large enough voltage margin, which hides such V_t fluctuation due to noise. However, in this paper, we perturb the absolute V_t's of the erased cells by utilizing program disturb characteristics, which in effect, shift up V_t of the erased cell close to the read reference voltage. This allows us to observe the noisy behavior of the Flash memory bits during multiple read operations of the same page.

Fig. 3(a) illustrates our proposal for the extraction of randomness from the SLC Flash memory cell's physical

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

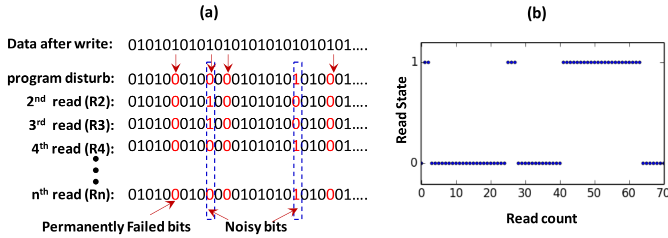

Fig. 3. (a) Illustration of noise extraction algorithm. First row is the checkerboard data pattern written on a memory page. Second row represents the same data after repeated program disturb. (b) State of a noisy bit during "*N*" recurrent read count. Clearly, the bit changes its state randomly during the repeated read operation.

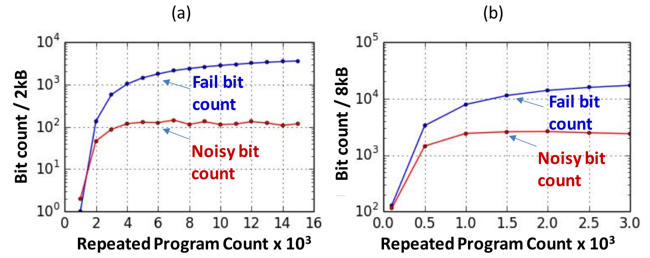

Fig. 4. Program disturb characterization (*y*-axis) as a function of the number of repeated programming operations (*x*-axis) for Micron's (a) SLC and (b) MLC chips. The blue curve represents the FBC (or the erase to program fails) in a given page due to repeated program stress on the same page. The red curve represents the number of noisy bits identified in the corresponding program disturbed page.

properties. The same algorithm with a slight modification is applicable to the multi level cells (MLCs) Flash memory technology and is discussed further at the end of this section. The first step in our algorithm is to erase a Flash block and write the checker board data pattern to a selected page of the block. The programming operation is repeated on the same page (the same data) without erase operation to cause program disturb on the erased cells. Hence, some of the erased cells will flip their state ("1" $\rightarrow$ "0") which will increase the fail bit count (FBC) for that page. These bit flips are illustrated by arrows between the first and the second row in Fig. 3(a). With this unreliable memory state [second row in Fig. 3(a)] if we keep reading the page data repeatedly (e.g., N times), we find that some bits remain at the failed state for all N read operations. These permanently failed bits experience much worse program disturb, so that their $V_t \gg V_{\text{REF}}$. There is another group of bits that remain at their correct state for all N read operations. Both of these groups of bits are not useful for noise extraction. However, there is a third group of bits which changes their state during read operations due to noise [marked with dashed rectangle in Fig. 3(a)]. Physically, these bits' $V_t \sim V_{\text{REF}}$. Hence, slight noise (both RTN and thermal) affects their state during read operations and causes their state fluctuation. Fig. 3(b) shows the measured read-state fluctuation of one such noisy bit as a function of continuous read number. In the following, we describe the steps (including preconditioning steps) to generate true random numbers using SLC Flash memory.

1) Erase a Flash memory block.
2) Program a page of the block with the checkerboard data pattern.
3) Perform repeated programming on the same page with the same data pattern till the FBC reaches a desired threshold value. This step ends the preconditioning part.
4) Read the page N times and identify the noisy bits. One identification scheme is to sum the read values of individual bits. Discard the bits whose sum value (S) is either 0 or N. Bits with sum value, $0 < S < N$, are the noisy bits.
5) From each noisy bit, extract bits for the true random number. In this paper, we extract "0" if S is even, else extract "1."
6) Perform Von Neumann debiasing.

The steps 4–6 are repeated as many times as new random bits are needed. For MLC Flash memory, the above-mentioned algorithm remains the same, except steps 2 and 3. For MLC Flash memory, every WL is shared by two pages (MSB-page containing MSB bits and LSB-page containing LSB bits). Since two logical pages are physically tied with the same WL, repeated programming on one page affects the bits of the other page. In this paper, we first program both the LSB and MSB pages of the same WL and perform program stressing only on LSB-page. Program stress on the LSB page causes program disturb on the MSB page. The remaining steps for noise extraction remain the same as mentioned previously with unreliable memory bits on the MSB page.

IV. Experimental Results and Discussion

A commercial Flash memory evaluation board is used to program and read the Flash chips. The board contains a socket to hold a Flash chip under test, an ARM microprocessor to issue commands and receive data from the Flash chip, and a serial interface. The proposed algorithm has been ported on the development board. The setup represents typical embedded platforms, such as USB flash drives and sensor nodes.

A. Program Disturb Characterization

Program disturb in a Flash memory chip is characterized by the number of erased to programmed bit flips (or FBC) due to repeated programming operation of the page. Fig. 4 shows the program disturb characteristics (blue curve) of both SLC [Fig. 4(a)] and MLC [Fig. 4(b)] Flash memory chips as a function of the number of repeated programming operations on the same page. Page size is 2 kB for the SLC chip and 8 kB for the MLC chip. Due to program disturb, programmed cells are not typically affected; however, some of the erased cells get programmed causing "1"$\rightarrow$ "0" bit flips, which is measured as FBC. As illustrated in the figure, we find that the MLC Flash chip requires fewer program operations than the SLC chip to disturb the memory states. This is expected, since the voltage margin between the erased and programmed states in the MLC chip is significantly lower than in the SLC chip. In addition, the FBC due to program disturb on the MLC chip is higher than in the SLC chip. A higher FBC on the MLC chip is partly due to larger page size and partly due to different modes of programming in the SLC and MLC chips. For both, we observe that FBC saturates for very high number of repeated programming operations which is due to the limited

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

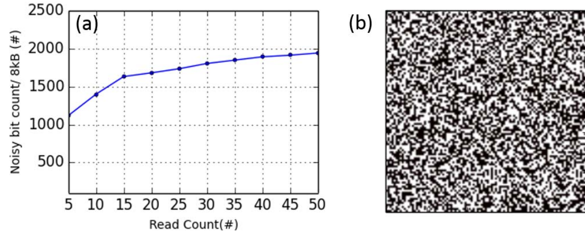

Fig. 5. (a) Count of the noisy bits (NBC) in a memory page is plotted with the number of recurrent reads ("N") of that page. NBC saturates with very high number of reads. In this paper, we choose $N = 20$, for noisy bit identification. (b) 2-D image representation of generated random bit stream. Black color represents bit 0 and white is bit 1. We find almost equal number of 1's and 0's in the generated random bit stream.

number of erased bits in a page. In general, the FBC count as a function of program stress is highly dependent on the manufacturing process and it will vary for Flash chips from different vendors.

The red curves in Fig. 4(a) and (b) show the noisy bit count in the same memory page. The number of noisy bits per page is a very important quantity for the TRNG application. The higher the number of noisy bits in a page, the higher is the throughput. As illustrated in Fig. 4(a) and (b), the number of noisy bits in a memory page increases as the number of repeated programming operations increases. This can be explained as follows. The repeated programming stresses the erased state V_t distribution, so its median moves up in the vicinity of read reference voltage [see Fig. 2(b)]. In this situation, there is a large number of erased bits with V_t so close to the read reference voltage that noise can affect their sensed state.

Finally, we generate a stream of random bits of length 1 000 000, using the noisy bits' toggling characteristics as illustrated in algorithm steps described in Section III. For the noisy bit identification, we perform "N" repeated read operations of the same page. In this paper, we choose $N = 20$, since from Fig. 5(a), we find that the noisy bit count almost saturates after $N = 20$. We utilize the same memory page repeatedly to generate a large stream of random bits. More specifically, if we identify m noisy bits in N consecutive read operations of the same page, then we get m random bits in that trial. We repeat the read operations on that page for another N times to get the next m (or close to m) random bits. Note that the exact location of noisy bits may change between the first and the second trial, since new noisy bits can appear as we read in the subsequent trials and at the same time, few previously identified noisy bits may disappear as well. We present the generated true random bits as a 2-D image in Fig. 5(b) for better illustration. We find almost the same number of zeros and ones in the generated random bit stream.

B. Randomness Analysis Using NIST Test Suite

In order to evaluate the randomness of the bits produced by the Flash-TRNG, we utilize the NIST Test Suite, a statistical package consisting of different types of tests to evaluate the randomness of binary sequences. Table I describes the test results for the random numbers generated by our algorithm. Each statistical test calculates a P-value that shows the randomness of the given sequences based on that test. If P-value $\geq$ 0.0001, then the sequence can be considered as uniformly distributed [29]. Note that some of the tests, such as nonoverlapping template, random excursions, and random excursions variant, consists of several individual tests. We report the least P-value out of several internal tests for them. We have generated five different random sequences each consisting of 10^6 random bits. As shown in Table I (proportion), the random sequences pass all the NIST tests indicating excellent randomness property of the proposed TRNG.

TABLE I
RANDOMNESS EVALUATION RESULTS USING NIST STATISTICAL TEST SUITE

NIST Test	P-value	Proportion
Frequency	0.7588	5/5
Block frequency	0.9603	5/5
Runs	0.8190	5/5
Longest runs	0.8116	5/5
Rank	0.2947	5/5
FFT	0.5896	5/5
Overlapping template	0.1498	5/5
Non-overlapping template	0.9999	5/5
Universal	0.9994	5/5
Linear complexity	0.7361	5/5
Serial	0.6159	5/5
Approximate entropy	0.8626	5/5
Cumulative Sums	0.6670	5/5
Random Excursion	0.0807	5/5
Random Excursions Variant	0.0494	5/5

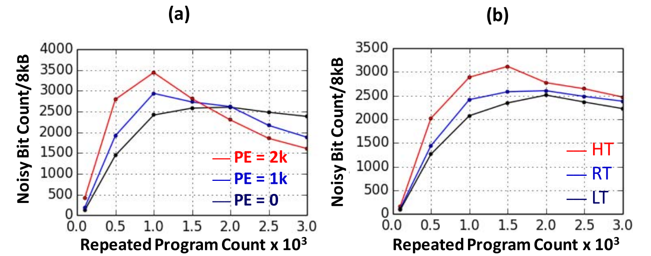

Fig. 6. Effect of (a) aging and (b) temperature on the noise characteristics. For emulating aging condition, we perform PE cycling whose count is shown as different legends for the three curves in (a). In (b), we utilize three different temperature conditions. Red curve: HT (85 °C). Blue curve: RT (25 °C). Black curve: LT (−10 °C). This experiment is done on MLC chip of page size of 8 kB.

C. Aging and Temperature Effect

We analyze the effects of aging [typically quantified by program–erase (PE) cycle count] on the noise characteristics of Flash bits in Fig. 6(a) which shows that the noisy bit count in a page increases with an increase in the number of PE operations. In general, the RTN of a Flash memory bit depends on the defect or trap density in the tunnel oxide or at the interface between the tunnel oxide and the channel Si [2]–[6]. With higher number of PE cycling, the defect density increases which causes higher bits to show noisy behaviors. We also explore the effects of temperature in the range of 85 °C [high temperature (HT)] to −10 °C [low temperatures (LTs)] on the noise characteristics in Fig. 6(b). Interestingly, the noisy bit

count is not significantly affected by temperature lowering as shown in Fig. 6(b). In fact, previous studies [30] show that the RTN behavior, which is one of the main noise sources during read, continues even at extremely LTs. The results in Fig. 6(b) thus ensure that the proposed TRNG is robust against the temperature variation.

D. Throughput Analysis

The throughput from the proposed TRNG depends on flash memory's read time (t_R) of a page and the number of noisy bits (B_N) identified (or available) in that page during the N read operations of the same page as discussed in the algorithm section. As such, the following formula can be used for throughput estimation:

$$Throughput = \frac{B_N}{N \times t_R}.$$

For, Micron MLC chip, we found $t_R = 100\mu s$. From our characterization, we found $B_N > 2000/page$ for page size 8 kB and $N = 20$ is used in our algorithm. Thus, using above-mentioned formula, we can see that TRNG throughput exceeds 1 $Mbits/s$, which is significantly higher than the previous work [28].

V. Conclusion

In this paper, we propose a new method for hardware true random number generation using the program disturb and read noise characteristics of unmodified commercial off-the-shelf Flash memory. The method outlined in this paper can be implemented with standard digital interface without any need of customized hardware design. Since Flash memory is widely used, the proposed method can be utilized in many of the existing electronic devices. The generated random bits are tested with the NIST statistical tests and the results clearly show that the Flash-TRNG is a promising candidate for cryptographic systems. We also tested the aging effects of memory on the TRNG performance and found the technique to be robust with aging. The proposed technique also remains unaffected under measured (85 °C to −10 °C) temperature changes.

References

[1] T. Ristenpart and S. Yilek, "When good randomness goes bad: Virtual machine reset vulnerabilities and hedging deployed cryptography," in *Proc. Netw. Distrib. Secur. Symp. (NDSS)*, 2010, pp. 1–18.

[2] S.-M. Joe *et al.*, "Threshold voltage fluctuation by random telegraph noise in floating gate NAND flash memory string," *IEEE Trans. Electron Devices*, vol. 58, no. 1, pp. 67–73, Jan. 2011.

[3] C. M. Compagnoni, M. Ghidotti, A. L. Lacaita, A. S. Spinelli, and A. Visconti, "Random telegraph noise effect on the programmed threshold-voltage distribution of flash memories," *IEEE Electron Device Lett.*, vol. 30, no. 9, pp. 984–986, Sep. 2009.

[4] A. Ghetti, C. M. Compagnoni, A. S. Spinelli, and A. Visconti, "Comprehensive analysis of random telegraph noise instability and its scaling in deca–nanometer flash memories," *IEEE Trans. Electron Devices*, vol. 56, no. 8, pp. 1746–1752, Aug. 2009.

[5] S.-H. Bae *et al.*, "The $1/f$ noise and random telegraph noise characteristics in floating-gate nand flash memories," *IEEE Trans. Electron Devices*, vol. 56, no. 8, pp. 1624–1630, Aug. 2009.

[6] H. Kurata *et al.*, "Random telegraph signal in flash memory: Its impact on scaling of multilevel flash memory beyond the 90-nm node," *IEEE J. Solid-State Circuits*, vol. 42, no. 6, pp. 1362–1369, Jun. 2007.

[7] G. Taylor and G. Cox, "Behind Intel's new random-number generator," *IEEE Spectr.*, Aug. 2011. [Online]. Available: http://spectrum.ieee.org/computing/hardware/behind-intels-new-randomnumber-generator

[8] H. Hata and S. Ichikawa, "FPGA implementation of metastability-based true random number generator," *IEICE Trans. Inf. Syst.*, vol. E95-D, no. 2, pp. 426–436, Feb. 2012.

[9] A. P. Johnson, R. S. Chakraborty, and D. Mukhopadyay, "An improved DCM-based tunable true random number generator for Xilinx FPGA," *IEEE Trans. Circuits Syst. II, Exp. Briefs*, vol. 64, no. 4, pp. 452–456, Apr. 2017.

[10] C. Tokunaga, D. Blaauw, and T. Mudge, "True random number generator with a metastability-based quality control," in *IEEE Int. Solid-State Circuits Conf. (ISSCC) Dig. Tech. Papers*, Feb. 2007, pp. 404–611.

[11] S. Srinivasan *et al.*, "2.4 GHz 7mW all-digital PVT-variation tolerant true random number generator in 45 nm CMOS," in *Proc. Symp. VLSI Circuits*, Jun. 2010, pp. 203–204.

[12] P. Z. Wieczorek and K. Golofit, "Dual-metastability time-competitive true random number generator," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 61, no. 1, pp. 134–145, Jan. 2014.

[13] F. Tehranipoor, W. Yan, and J. A. Chandy, "Robust hardware true random number generators using DRAM remanence effects," in *Proc. IEEE Int. Symp. Hardw. Oriented Secur. Trust (HOST)*, May 2016, pp. 79–84.

[14] C. Pyo, S. Pae, and G. Lee, "DRAM as source of randomness," *Electron. Lett.*, vol. 45, no. 1, pp. 26–27, Jan. 2009.

[15] T. Amaki, M. Hashimoto, and T. Onoye, "A process and temperature tolerant oscillator-based true random number generator with dynamic 0/1 bias correction," in *Proc. IEEE Asian Solid-State Circuits Conf. (A-SSCC)*, Nov. 2013, pp. 133–136.

[16] M. T. Rahman, K. Xiao, D. Forte, X. Zhang, J. Shi, and M. Tehranipoor, "TI-TRNG: Technology independent true random number generator," in *Proc. 51st ACM/EDAC/IEEE Design Autom. Conf. (DAC)*, Jun. 2014, pp. 1–6.

[17] X. Zhang, L. Qi, Z. Tang, and Y. Zhang, "Portable true random number generator for personal encryption application based on smartphone camera," *Electron. Lett.*, vol. 50, no. 24, pp. 1841–1843, 2014.

[18] M. Bhargava, K. Sheikh, and K. Mai, "Robust true random number generator using hot-carrier injection balanced metastable sense amplifiers," in *Proc. IEEE Int. Symp. Hardw. Oriented Secur. Trust (HOST)*, May 2015, pp. 7–13.

[19] Y. Wang, W. Wen, H. Li, and M. Hu, "A novel true random number generator design leveraging emerging memristor technology," in *Proc. 25th Ed. Great Lakes Symp. VLSI*, New York, NY, USA, 2015, pp. 271–276.

[20] S. Gaba, P. Sheridan, J. Zhou, S. Choi, and W. Lu, "Stochastic memristive devices for computing and neuromorphic applications," *Nanoscale*, vol. 5, no. 13, pp. 5872–5878, 2013.

[21] S. Balatti, S. Ambrogio, Z. Wang, and D. Ielmini, "True random number generation by variability of resistive switching in oxide-based devices," *IEEE J. Emerg. Sel. Topics Circuits Syst.*, vol. 5, no. 2, pp. 214–221, Jun. 2015.

[22] S. Balatti *et al.*, "Physical unbiased generation of random numbers with coupled resistive switching devices," *IEEE Trans. Electron Devices*, vol. 63, no. 5, pp. 2029–2035, May 2016.

[23] C.-Y. Huang, W. C. Shen, Y.-H. Tseng, Y.-C. King, and C.-J. Lin, "A contact-resistive random-access-memory-based true random number generator," *IEEE Electron Device Lett.*, vol. 33, no. 8, pp. 1108–1110, Aug. 2012.

[24] A. Fukushima *et al.*, "Spin dice: A scalable truly random number generator based on spintronics," *Appl. Phys. Exp.*, vol. 7, no. 8, p. 083001, 2014.

[25] W. H. Choi *et al.*, "A magnetic tunnel junction based true random number generator with conditional perturb and real-time output probability tracking," in *IEDM Tech. Dig.*, Dec. 2014, pp. 12.5.1–12.5.4.

[26] E. Piccinini, R. Brunetti, and M. Rudan, "Self-heating phase-change memory-array demonstrator for true random number generation," *IEEE Trans. Electron Devices*, vol. 64, no. 5, pp. 2185–2192, May 2017.

[27] W. A. G. Rojas *et al.*, "Solution-processed carbon nanotube true random number generator," *Nano Lett.*, vol. 17, no. 8, pp. 4976–4981, Aug. 2017.

[28] Y. Wang, W.-K. Yu, S. Wu, G. Malysa, G. E. Suh, and E. C. Kan, "Flash memory for ubiquitous hardware security functions: True random number generation and device fingerprints," in *Proc. IEEE Symp. Secur. Privacy*, May 2012, pp. 33–47.

[29] A. Rukhin *et al.*, "A statistical test suite for random and pseudorandom number generators for cryptographic applications," NIST, Gaithersburg, MD, USA, Special Pub. 800-22, 2010.

[30] J. H. Scofield, N. Borland, and D. M. Fleetwood, "Temperature-independent switching rates for a random telegraph signal in a silicon metal–oxide–semiconductor field-effect transistor at low temperatures," *Appl. Phys. Lett.*, vol. 76, no. 22, pp. 3248–3250, 2000.

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

Biswajit Ray (S'12–M'16) received the Ph.D. degree in electrical and computer engineering from Purdue University, West Lafayette, IN, USA, in 2013.

He was developing 3-D NAND flash memory technology with SanDisk Corporation, Milpitas, CA, USA. He is currently an Assistant Professor of electrical and computer engineering with the University of Alabama in Huntsville, Huntsville, AL, USA, where he is involved in security and reliability of electronic devices.

Aleksandar Milenković (M'96–SM'10) received the Dipl. Ing., M.S., and Ph.D. degrees in computer engineering and science from the University of Belgrade, Belgrade, Serbia, in 1994, 1997, and 1999, respectively.

He is currently a Professor of electrical and computer engineering with the University of Alabama in Huntsville, Huntsville, AL, USA, where he leads the LaCASA Laboratory.