IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received April 30, 2020, accepted May 10, 2020, date of publication May 20, 2020, date of current version June 5, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.2995891

Extraction of Device Fingerprints Using Built-in Erase-Suspend Operation of Flash Memory Devices

THE-NGHIA NGUYEN[1], (Student Member, IEEE), SUNGHYUN PARK[2], AND DONGHWA SHIN[2], (Senior Member, IEEE)
[1]Department of Software Convergence, Soongsil University, Seoul 06978, South Korea
[2]Department of Smart Systems Software, Soongsil University, Seoul 06978, South Korea

Corresponding author: Donghwa Shin (donghwashin@soongsil.ac.kr)

This work was supported by the National Research Foundation of Korea (NRF) through the Korea Government (MSIT) under Grant NRF-2018R1D1A1B0704944513.

ABSTRACT The reliability and efficiency of a physically unclonable function (PUF) considerably depends on that of the random seed number generation process in the target hardware technology. Among the proposed hardware PUF techniques, flash memory-based approaches have several advantages because of the widespread use of flash memories in electronic devices. The operations of the flash memories such as read, program, and erase have been utilized to generate the random number based on their random process variations. In this work, we propose a random number generation method for the flash memory-based hardware PUF applications that utilize the intrinsic erase-suspend operation in modern flash memory devices. Unlike the conventional methods, the proposed method does not require any modification of the device or additional peripheral circuitry to control the operations. We evaluate the proposed method from the perspective of reliability and efficiency. The experimental results show that the proposed method can generate a 16-bit random number in 1 ms approximately while maintaining greater than 95 % reproductivity of the random number.

INDEX TERMS Flash memory, erase-suspend method, digital fingerprint, physical unclonable function, true random number generator.

I. INTRODUCTION

Computing systems are usually constructed with multiple layers, from the transistor to the application software. A higher layer of abstraction, such as an application or operating system, should be built on a stack of abstracted layers that can be trusted. The initial sources of this trust are known as the roots of trust, are typically hardware features [1]. Unfortunately, in reality, such roots of trust cannot be ensured for all systems, as highly sophisticated counterfeit semiconductor devices have infiltrated the market deeply [2], [3].

The potential hazards of these counterfeit devices have become critical with the increase in the economic importance and popularity of electronic and computing devices. Non-functioning fake devices are being introduced into the market, and consumers are suffering economic losses. Furthermore, such devices can cause serious security problems by stealing private information from devices. Therefore, computing applications with stringent security requirements have been developed, with the ability to identify a particular electronic device uniquely as illustrated in Fig. 1. The security of applications such as personal authentication, personalized services, and digital rights management can be improved significantly using per-device identifiers (fingerprints) that are difficult to clone. Therefore, physically unclonable functions (PUFs) have been proposed as such roots of trust in this context [1], [4]. PUFs usually utilize the uniqueness of each individual device, which originates from the chip-to-chip variations generated during the manufacturing process. A PUF taxonomy has been carefully studied in [5] considering the physical mechanisms of different PUFs.

Flash memory devices are widely used for nonvolatile storage in electronic devices, as valuable private information in the storage would be a desirable target for criminals.

The associate editor coordinating the review of this manuscript and approving it for publication was Cihun-Siyong Gong.

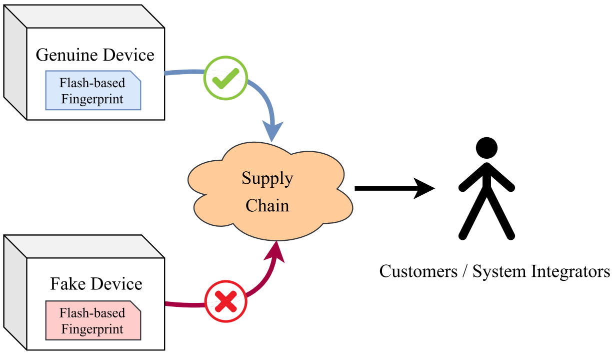


FIGURE 1. General concept of counterfeit detection using device fingerprint.

From the perspective of hardware PUF techniques, attempts have been made to utilize the characteristics of flash memory operations such as read, program, and erase [6]. The chip-to-chip variations in the read, program, and erase characteristics can be utilized to generate random seed numbers. Then, security keys can be generated from these random numbers and helper data.

In this work, we propose a random-number generation method for hardware PUF applications that is based on the intrinsic erase-suspend operation of modern flash memory devices and does not require any modification of the chip or peripheral circuitry. The hardware identifier must be small and secure, by design, to perform security-related functions. The proposed technique does not require any additional circuity or modification of the flash memory chip. Rather, it uses only the intrinsic functions, such as the erase-and-suspend function in modern flash memories. Experimental results show that the proposed technique can generate a 16-bit random number in 1 ms with greater than 95 % reproducibility.

The remainder of the paper is organized as follows. We introduce related studies in Section II focusing on the implementation and evaluation of flash memory PUFs. The operating principles of the proposed method are presented in Section III, and the evaluation results are presented in Section IV. Finally, we conclude the paper and suggest future research directions in Section V.

II. RELATED WORK

Many hardware-based security systems have been investigated to secure sensitive information against attackers. The most important goal of these systems is to find a new source of unclonable randomness originating from the manufacturing process variations of an integrated circuit (IC). These variations or inherent fluctuations of semiconductor devices, which are considered unexpected effects that decrease their effective field, can be exploited by PUF devices as valuable resources to generate uncontrollability, unpredictability, and unclonability. Moreover, silicon PUF embodies its unique physical and intrinsic features into a physical structure that cannot be replicated, even if the manufacturing process used for its fabrication is known.

The very first PUF generation approach was introduced by Pappu *et al.* in [7]. Their work proposed using the light scattering patterns of a three-dimensional microstructure as a physical one-way function. The research of Pappu *et al.* was the baseline for several PUF circuit designs based on the features of hidden timing and delay information of ICs [8]–[10]. Silicon PUFs were implemented either as customized circuits or reconfigurable logic components on field-programmable gate arrays (FPGAs) [11]–[13]. In these approaches, the power-up state of the SRAM memory in FPGAs was used as a source for generating the PUF, to protect the intellectual property core. To enhance the reliability of generated secret keys on FPGA-based PUFs, external helper data such as repetition code and ECC are essentially required [14]. In our proposed method, we leverage the non-volatile characteristic of commercial flash memory devices to mitigate the requirement of the power cycle as well as the requirement of external assisted data.

In [15], Kim et. al exploited the intrinsic variations in programming operations, arising from the statistical fluctuations in the threshold voltages (V_{TH}) of fabricated flash memory devices, to create PUF devices. In this work, a unique programming efficiency of the flash memory was investigated at the unit-cell level. V_{READ} was replaced by V_{PUF}, which was the statistical median V_{TH} of programmed state. V_{PUF} was designed to distinguish the programmed logic states of "0" and "1". Although the memory cells were fabricated using the same processes and equipment, each memory cell had a different program/erase efficiency, which inevitably induced V_{TH} variations. An inhibited region was defined to avoid the errors arising from small V_{TH} values, within the intermediate regions between the "0" and "1" states. The inhibited region is determined by detecting the current level. In our proposed PUF, we only use the intrinsic operations of the modern flash memory devices in order to avoid hardware modification as much as possible.

The PUF circuits were also used as hardware random number generators in [16]. Instances of PUFs were considered as digital signatures in [17], for IC authentication and hardware-based cryptographic key generation. The concept of flash-based PUFs has been attracting considerable attention in the recent years. In [18], Prabhu *et al.* suggested seven techniques to extract unique device fingerprints from NAND flash devices. Among these techniques, the program disturb produced good result in generating unique signatures for different chips according to authors' conclusion. The Pearson correlation was used to measure the robustness of the signatures and identify the individual flash devices uniquely. Although this approach was feasible for distinguishing between different chips with informative signatures, the desired unique signature was obtained only after several thousand programming and reading operations. Moreover, the lifetime of the flash memory was affected significantly by the repeated access operations during the prolonged extraction. In our approach, the intrinsic erase-suspend operation of the modern flash memory devices does not affect much to the lifetime of hardware. Therefore, the digital signature of hardware device can be used for longer time.

In the research of Wang *et al.* [19], the analog characteristic of flash memories, i.e., the program time of each individual bit, was exploited to hide private data. The information could be hidden in the data stored in the memory, without using any additional components. In [20], the authors proposed a method to generate true random numbers by extracting the program disturb and read noise properties of flash memory bits, which were considered fundamental properties of all NAND flash memory arrays. They claimed that their design was cost-effective and tolerant to aging and temperature effects. In addition, their method could be deployed through software updates which can be considered to employ in our further work.

III. DEVICE FINGERPRINT GENERATION BY ERASE-SUSPEND OPERATION

A. ASYMMETRIC ERASE-SUSPEND CHARACTERISTICS OF FLASH MEMORY DEVICES

Different from the volatile memory or delay-based PUFs that have been implemented on FPGAs, the flash-based PUFs generally attempt to utilize the variance in non-volatile cell characteristics. The flash memory changes its cell values by injecting (programming) or releasing (erasing) the charges in the floating gates. All the cells in a block are set to 1 by the erase operation, and the individual cells can be reset to 0 by the program operation. Usually, data in the flash memory can be read and programmed in pages, but they can be erased only at the block level, with multiple pages. The size of a page is typically approximately 4 to 16 kB, and the size of a block is approximately several MB.

Modern flash devices are equipped with a suspend function for the program and erase functionalities, to enhance the chip-level performance [21]. The time required for performing an erase operation in flash devices is generally several orders of magnitude longer than that required for the read or program operations. Therefore, the sector (or block) erase operation can be suspended, on occasion, to enable timing interception by operations with higher priority.

Fig. 2 shows a portion of the erased bits in a sector, over time. The portion of erased bits (P_{erased}) over different durations of suspension is measured using the following equation:

$$P_{\text{erased}} = \frac{\text{number of erased bits}}{\text{number of bits in a sector}} \tag{1}$$

The gradient of P_{erased} (dotted plot in Fig. 2 exhibits a positively skewed aspect over time. The number of erased cells starts increasing rapidly at a certain time. This phenomenon is followed by a long tail of unerased cells. The long tail parts of the operation provide much higher reproducibility than the peak points, as only a small difference in the peak points may cause a substantial change in the result. It is expected that we can obtain the fingerprint more effectively at the stable tail part.

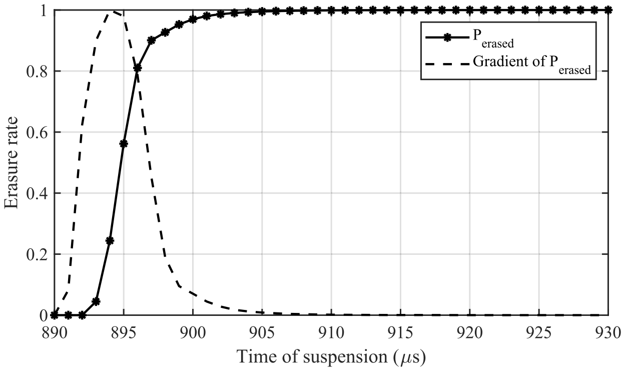


FIGURE 2. Portion of erased bits of SST39VF1601 flash memory device after different suspension time at 258 K.

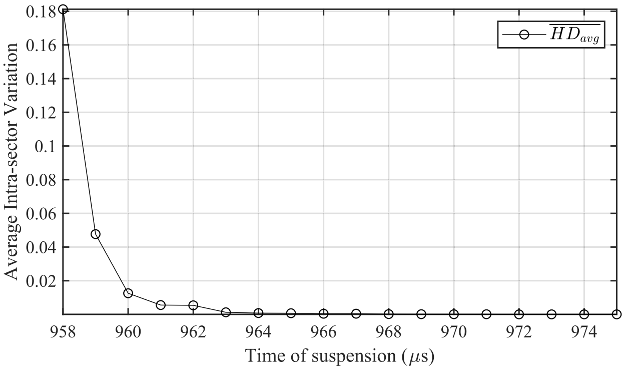


FIGURE 3. Average Hamming distance and reproducibility of fingerprint at the later part of erasure process.

Without loss of generality, a small intra-sector Hamming distance provides better reproducibility among the generated results, as we use the positions of the bits as random fingerprints. We will use the intra-sector Hamming distance ($HD_{intra-sector}$) as a proxy for reproducibility, in this context, as follows:

$$HD_{intra-sector} = \frac{1}{m}\sum_{i=1}^{m}\frac{HD(R_i, R_{i,t})}{n} \tag{2}$$

where R_i is a reference response of challenge i, $R_{i,t}$ is the n-bit response of the same challenge i extracted at a different operating condition, and m is the number of responses collected to calculate the $HD_{intra-sector}$. Note that a smaller $HD_{intra-sector}$ represents better reproducibility.

Fig. 3 shows the averaged Hamming distance for the trials, over time. We can see that the averaged Hamming distance decreases significantly, to lower than 0.01%, after 960 μs at 288 K whereas the value is much higher and highly unpredictable in the earlier phases of the erasure process.

From the perspective of effectiveness and reliability, it can be observed that the later part of the erasure process would be more suitable to extract a device fingerprint. Of course, P_{erased} will change according to the operating conditions but $HD_{intra-sector}$ must be as much small as possible; therefore, it is necessary to handle the variability, even if we find a proper target to generate the fingerprint. We will present the method that was applied to handle the operating-condition-induced variability to enhance the reliability of the proposed method.

During erase and program, the variations in the thickness of the tunneling oxide, charge-trapping layer, and blocking oxide affect the time taken to complete the operation. According to the experimental data, the variation in the tunneling oxide thickness is less than 1%. However, the variations in the thickness of the charge-trapping layer and blocking oxide are approximately 2% [15]. This causes variations in the erase and program times of the device. A minimum operation time is suggested to guarantee completion of the operations for a given set of operating conditions.

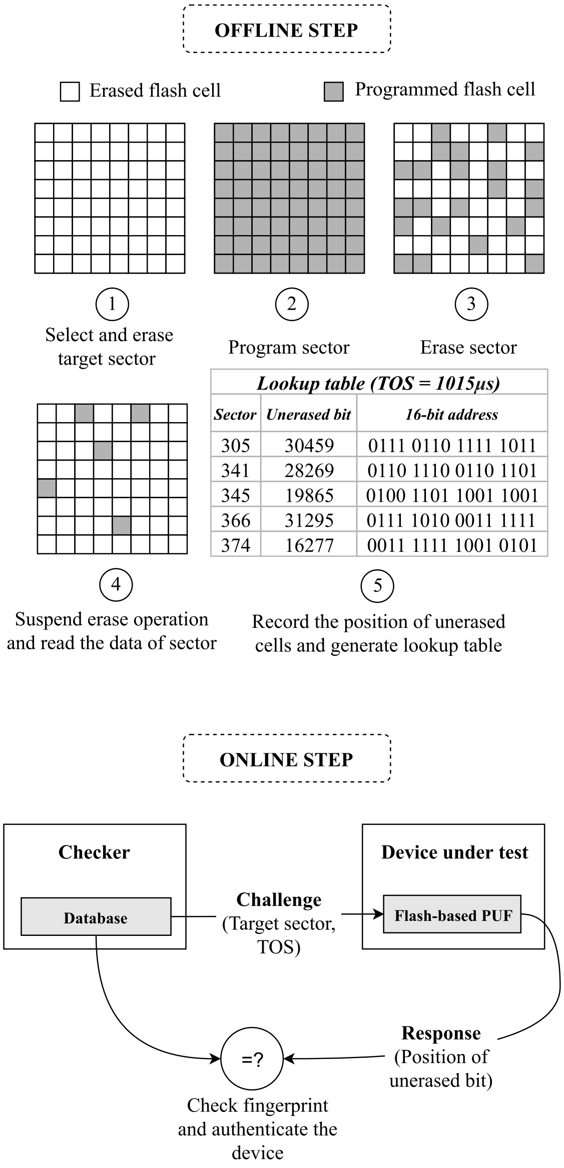
B. GENERATION AND DISCRIMINATION OF FINGERPRINTS CONSIDERING VARIABILITY

1) OFFLINE CHARACTERIZATION AND ONLINE DISCRIMINATION

To generate a random fingerprint, we must know the change in the sector content at each specific time after sending the erase operation. Power failures during flash memory operation can lead to several non-intuitive behaviors, such as data corruption or unreliability in future programming operations [22]. Instead of programming and then suddenly cutting off the power supply to the flash chip to interrupt the erase operation repeatedly, we use the erase-suspend operation of SST39VF1601.

We use the positions of the unerased bits in a sector as random fingerprints. This is done in two separate steps: an offline step and an online step, as presented in Fig. 4. The long tail part of the erase process is characterized in the offline step. In other words, different lengths of erase periods are applied by the erase-suspend operation on target sectors to capture the unerased bits. Once we have the raw data by sampling, we analyze the effect of the erase-suspend operation on entire selected sectors of the flash memory using the Hamming distance. We choose the most desirable bits to make a stable key based on this analysis result. Next, in the online step, we reproduce this key using predetermined suspension timings in the erase-suspend operations, called time of suspension (TOS).

When users want to obtain random fingerprints, the checker runs a program to analyze the flash memory for the target sectors, period of erase-suspend operations, and position of unerased bits. First, the selected sector is completely erased (Fig. 4 ①) and programmed (Fig. 4 ②). Another erase operation is initiated (Fig. 4 ③), and suspended after the predetermined period of an erase-suspend operation (Fig. 4 ④). The checker will record the positions of unerased bits and erase-suspend timings in a table (Fig. 4 ⑤). The recorded positions of the bits are compared with the challenge-response results in the online step.


OFFLINE STEP
□ Erased flash cell
▨ Programmed flash cell
1
Select and erase target sector
2
Program sector
3
Erase sector
4
Suspend erase operation and read the data of sector
5
Record the position of unerased cells and generate lookup table
ONLINE STEP
Checker
Database
Challenge (Target sector, TOS)
Device under test
Flash-based PUF
=?
Response (Position of unerased bit)
Check fingerprint and authenticate the device

Lookup table (TOS = 1015μs)

Sector	*Unerased bit*	*16-bit address*
305	30459	0111 0110 1111 1011
341	28269	0110 1110 0110 1101
345	19865	0100 1101 1001 1001
366	31295	0111 1010 0011 1111
374	16277	0011 1111 1001 0101

FIGURE 4. Offline characterization step and online challenge-response step of the proposed fingerprint generation procedure.

2) EFFECT OF VARIATIONS IN TEMPERATURE

The bit sequences in the sector, created by the erase-suspend operations at the same time of suspension, would ideally be identical. However, in reality, variations will exist because the operating conditions and internal characteristics of the chip and cell result in digital noise (e.g., telegraph noise). The sources of variability in the erase timing include temperature, supply voltage, and aging [20].

For instance, the erase time is known to be inversely proportional to the temperature, in general. Fig. 5 shows the relationship between the process of erasure and the temperature measured in the experimental platform. It is observed that the entire curve is shifted by a temperature change. The ambient temperature has a strong effect on the erase operation. At 263 K, the sector is fully erased after 41 μs (i.e., from 890 μs

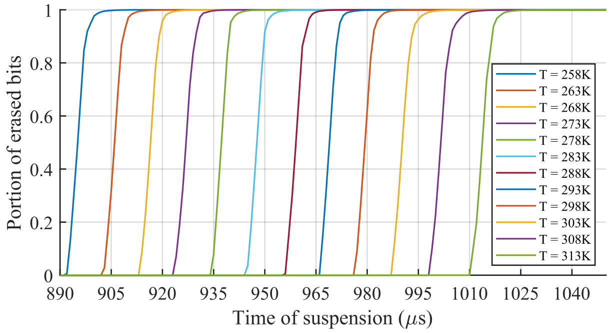


FIGURE 5. **Portion of erased bits at different ambient temperatures.**

to 930 μs). However, if we change the ambient temperature to 273 K, that sector needs only 29 μs (i.e., from 920 μs to 948 μs) to be fully erased.

In this work, we experimentally assess the effects of temperature variations, and attempt to find a way to achieve a stable result, irrespective of such variations. We do not deal with voltage variations in this work. However, as long as the change in P_{erased} follows a trend similar to the shift in temperature, we can apply the same method to handle other variability sources, without loss of generality.

3) STABLE GENERATION OF FINGERPRINT FROM LONG TAILS OF CDF AT DIFFERENT TEMPERATURE

In Section III-A, we mentioned that the latter part of the erasure process showed much higher stability in terms of the positions of unerased bits and, accordingly, the fingerprints. The controller is set to change the number of erase operations with the unit timing of suspension to achieve the predetermined number of unerased bits, which depends on the expected lengths of the fingerprints.

Fig. 6 shows the results of the proposed incremental erase-suspend method, when the controller is set to stop at the last four unerased bits. We implemented the proposed method on a selected sector under different constant temperatures that ranged from 258 K to 313 K. Fifty runs were performed for each temperature.

For statistical purposes, we collected all outcomes of these four bits and organized them in subtables. The numbers in the right green column of each subtable represent the number of times that a specific sequence of four unerased bits occurred. We also extracted their positions in those outcomes and listed them in the blue row on the top of each subtable. These positions are denoted using capital letters from A to K. The real values of the positions can be found at the bottom of table. In addition, the numbers of occurrences of each unerased bit in the entire 50 runs are summarized in the yellow row at the bottom of each subtable. Through this presentation, we can obtain an overview of the experimental results. From the table, we can see that as the length of the sequence decreases, the probability of regeneration increases. Moreover, the sequences of unerased bits differ for different temperatures. However, among the 4 kB bits in a sector, the 30459th bit is always in the unerased state as the temperature changes from 258 to 313 K. With a stable bit, we can obtain 16-bit binary numbers, as a fingerprint, from the position of the unerased bit, as the size of the sector is 4 kB in the target flash device in the experiments.

4) INCREMENTAL ERASURE FOR RELIABILITY ENHANCEMENT OF FINGERPRINTS

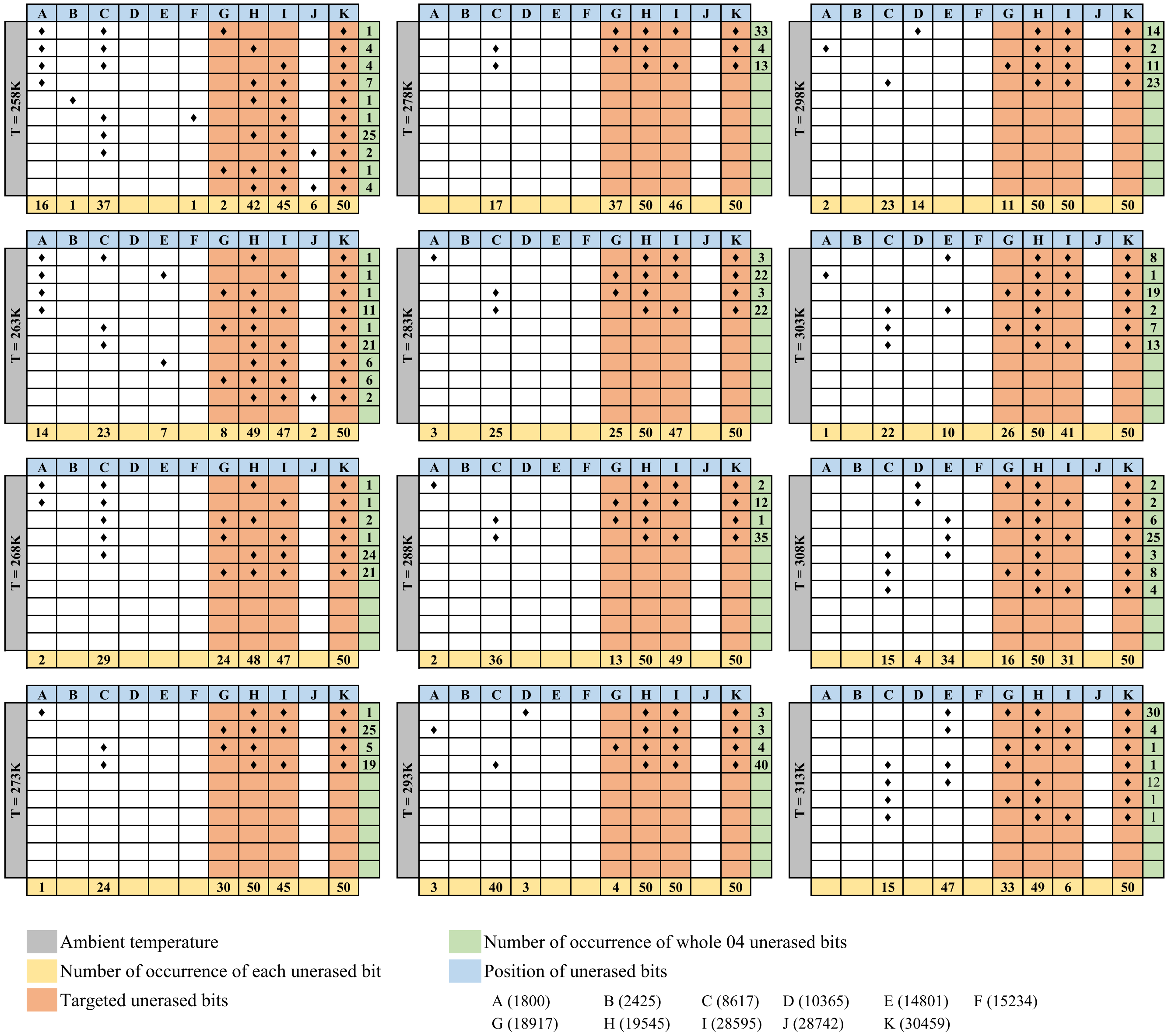
Common ways to improve the reliability of fingerprints in the literature include circuit-level approaches, fuzzy extraction with error-correcting codes, and voting mechanisms [1]. The proposed erase-suspend method is designed not to require additional circuitry, and we do not focus on post processing in this work. Instead, we will adopt an incremental erase-suspend method to enhance the stability.

The proposed incremental erase-suspend operation deals with shifts in the P_{erased} curves in an adaptive manner. As the actual effect of each source of variability is unpredictable, it is not practical to characterize the effect variability in advance. Instead, we proceed with the erase-suspend operation incrementally, where the commercial flash memory devices with erase-suspend feature generally support a "resume" operation as well [21].

As the resumed erase operation can also be suspended, we can handle the temperature-induced timing variations with sequences of intrinsic functions in the flash devices alone. Algorithm 1 shows a flowchart that describes the proposed incremental erase-suspend method. Starting from the predetermined earliest point of the sample, the cycle of **{program – erase – suspend – reset – read}** is repeated until the number of unerased bits reaches the required value.

Algorithm 1 Extracting Erase-Suspend-Based Fingerprint

```
block = presetTargetBlock;
sector = presetTargetSector;
lengthOfFingerprint = presetLengthOfFingerprint;
TOS = presetTOS;
fingerprintFound = False;
EraseBlock(block);
while fingerprintFound <> True do
    EraseEntireSector(sector);
    ProgramEntireSector(sector);
    SuspendErasure(TOS);
    ReadEntireSector(sector);
    unerasedBits = CountUnerasedBits();
    currentP_erased = CalculateP_erased(); if
    unerasedBits <> lengthOfFingerprint then
        TOS = PredictNextTOS(currentP_erased);
    else
        fingerprint:= unerasedBitPositions;
        fingerprintFound = True;
    end
end
```



	A	B	C	D	E	F	G	H	I	J	K	
T = 258K	♦		♦				♦				♦	1
	♦		♦					♦			♦	4
	♦		♦						♦		♦	4
	♦							♦	♦		♦	7
		♦						♦	♦		♦	1
			♦			♦			♦		♦	1
			♦					♦	♦		♦	25
			♦						♦	♦	♦	2
							♦	♦	♦		♦	1
								♦	♦	♦	♦	4
	16	1	37			1	2	42	45	6	50	

	A	B	C	D	E	F	G	H	I	J	K	
T = 278K							♦	♦	♦		♦	33
			♦				♦	♦			♦	4
			♦					♦	♦		♦	13
			17				37	50	46		50	

	A	B	C	D	E	F	G	H	I	J	K	
T = 298K				♦				♦	♦		♦	14
	♦							♦	♦		♦	2
							♦	♦	♦		♦	11
			♦					♦	♦		♦	23
	2		23	14			11	50	50		50	

	A	B	C	D	E	F	G	H	I	J	K	
T = 263K	♦		♦					♦			♦	1
	♦				♦				♦		♦	1
	♦						♦	♦			♦	1
	♦							♦	♦		♦	11
			♦				♦	♦			♦	1
			♦					♦	♦		♦	21
					♦			♦	♦		♦	6
							♦	♦	♦		♦	6
								♦	♦	♦	♦	2
	14		23		7		8	49	47	2	50	

	A	B	C	D	E	F	G	H	I	J	K	
T = 283K	♦							♦	♦		♦	3
							♦	♦	♦		♦	22
			♦				♦	♦			♦	3
			♦					♦	♦		♦	22
	3		25				25	50	47		50	

	A	B	C	D	E	F	G	H	I	J	K	
T = 303K					♦			♦	♦		♦	8
	♦							♦	♦		♦	1
							♦	♦	♦		♦	19
			♦		♦			♦			♦	2
			♦				♦	♦			♦	7
			♦					♦	♦		♦	13
	1		22		10		26	50	41		50	

	A	B	C	D	E	F	G	H	I	J	K	
T = 268K	♦		♦					♦			♦	1
	♦		♦						♦		♦	1
			♦				♦	♦			♦	2
			♦				♦		♦		♦	1
			♦					♦	♦		♦	24
							♦	♦	♦		♦	21
	2		29				24	48	47		50	

	A	B	C	D	E	F	G	H	I	J	K	
T = 288K	♦							♦	♦		♦	2
							♦	♦	♦		♦	12
			♦				♦	♦			♦	1
			♦					♦	♦		♦	35
	2		36				13	50	49		50	

	A	B	C	D	E	F	G	H	I	J	K	
T = 308K				♦			♦	♦			♦	2
				♦				♦	♦		♦	2
					♦		♦	♦			♦	6
					♦			♦	♦		♦	25
			♦		♦			♦			♦	3
			♦				♦	♦			♦	8
			♦					♦	♦		♦	4
			15	4	34		16	50	31		50	

	A	B	C	D	E	F	G	H	I	J	K	
T = 273K	♦							♦	♦		♦	1
							♦	♦	♦		♦	25
			♦				♦	♦			♦	5
			♦					♦	♦		♦	19
	1		24				30	50	45		50	

	A	B	C	D	E	F	G	H	I	J	K	
T = 293K				♦				♦	♦		♦	3
	♦							♦	♦		♦	3
							♦	♦	♦		♦	4
			♦					♦	♦		♦	40
	3		40	3			4	50	50		50	

	A	B	C	D	E	F	G	H	I	J	K	
T = 313K					♦		♦	♦			♦	30
					♦			♦	♦		♦	4
							♦	♦	♦		♦	1
			♦		♦		♦				♦	1
			♦		♦			♦			♦	12
			♦				♦	♦			♦	1
			♦					♦	♦		♦	1
			15		47		33	49	6		50	

Ambient temperature
Number of occurrence of each unerased bit
Targeted unerased bits
Number of occurrence of whole 04 unerased bits
Position of unerased bits
A (1800) B (2425) C (8617) D (10365) E (14801) F (15234)
G (18917) H (19545) I (28595) J (28742) K (30459)

FIGURE 6. Position of non-erased bits after erase-suspend operation at different temperature.

IV. EXPERIMENTS

A. EXPERIMENTAL SETUP

We used the SST39VF1601 flash memory from Microchip in our experiments. Each sector of this flash chip consists of 4 kB, equivalent to 32,768 bits. The SST39VF1601 device provides a typical word-program time of 7 μs and sector-erase time of 18 ms. We applied different duration from the start of erase operation to suspension and then counted the number of unerased bits using a read operation after the soft-reset operation.

We implemented a custom PCB as a piggyback board on an MCU (Arduino Due) board, as shown in Fig. 7 (b), to provide the presented control features. The custom PCB board was designed for a flash memory (Microchip SST39VF1601) to enable the control of the individual signals of the flash memory chip. It consisted of a memory chip and sockets for power (Vdd), flash memory control, and buses. We performed the experiments in a temperature-controlled environment using the temperature chamber shown in Fig. 7 (a).

B. STABILITY OF THE PROPOSED INCREMENTAL ERASE-SUSPEND METHOD AT DIFFERENT TEMPERATURES

We performed the experiments under temperatures ranging from 258 to 308 K to evaluate the stability of the proposed incremental erase-suspend method. Based on the observations presented in Section III-B3,we focused on bits at positions 8617, 19545, 28595, and 30459 because

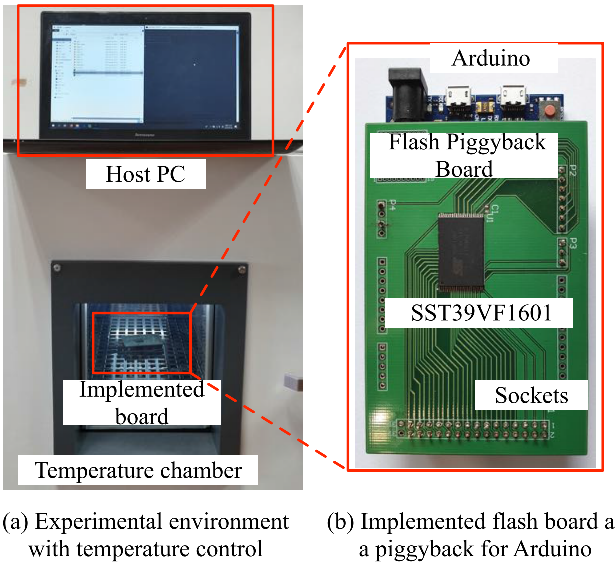

Host PC
Implemented board
Temperature chamber
Arduino
Flash Piggyback Board
SST39VF1601
Sockets
(a) Experimental environment with temperature control
(b) Implemented flash board as a piggyback for Arduino

FIGURE 7. **(a) Experimental setup with temperature chamber and (b) Implemented flash board as a piggyback of Arduino.**

of their frequency of appearance in the experimental results. In this part, we implemented 1000 additional runs, which could provide practical proof of the stability of the fingerprint.

Fig. 8 shows the positions of the unerased bits in the last tail of the erase operation for each trial. We can see that some of the unerased bits show more stable results than the other cells, suggesting that they can be utilized as fingerprints. Table in the bottom of Fig. 8 shows the summarized results of the target cells after 1000 runs. Even with the disturbance in different temperatures, the 30459th bit showed stable results in the last tail of erase operation.

Another important factor to be considered in the proposed method is the effect of aging. The experimental results presented in Fig. 8 show phenomena that can be regarded as effects of aging. As we proceed with further trials, some cells change their states. The aging effect caused by repetitive program/erase cycles can affect the raw data extracted from the flash memory and the stability of the device signature [23].

The relationships among cells have been confirmed to be relatively stable during the lifetime of the flash memory chips [24]. Thus, for the proposed method, the aging factor is expected to stretch the intervals among the unerased bits in the tail part. In Fig. 8, we locate the fixed number of cells in the last tail and observe that the required suspension time is shifted from 1045 to 1060 μs during the 1000 trials.

The exact cause of these unstable result is difficult to confirm through these device-level experiments. In reality, we have to choose only some of the sectors, carefully, in a flash memory, which fluctuate less in the tail part, for fingerprint generation. We can also regenerate the fingerprint considering the aging effect. Every fingerprint should have a limited lifetime, and the users should be advised to change their secret key after a certain time, similar to an online password.

TABLE 1. **Parameter values used in the experiments.**

Variables	Value	Descriptions
T_{sp}	14.336 µs	Time to program a sector
T_{se0}	930 µs	The earliest starting time of erase for given temperature range
T_{ses}	1 µs	Unit timing erase to suspension time in incremental iterations
T_{sr}	70 µs	Time to read the data of a sector
T_{srs}	5 µs	Time to reset the flash memory device in order to enable the read function

C. TIMING OVERHEAD

The time required for the fingerprint generation using the proposed incremental erase-suspend method (T_{overhead}) can be described using the following equation:

$$T_{\text{overhead}} = N_{\text{iter}} * (T_{\text{sp}} + T_{\text{se0}} + T_{\text{ses}} + T_{\text{r}} + T_{\text{srs}}) \qquad (3)$$

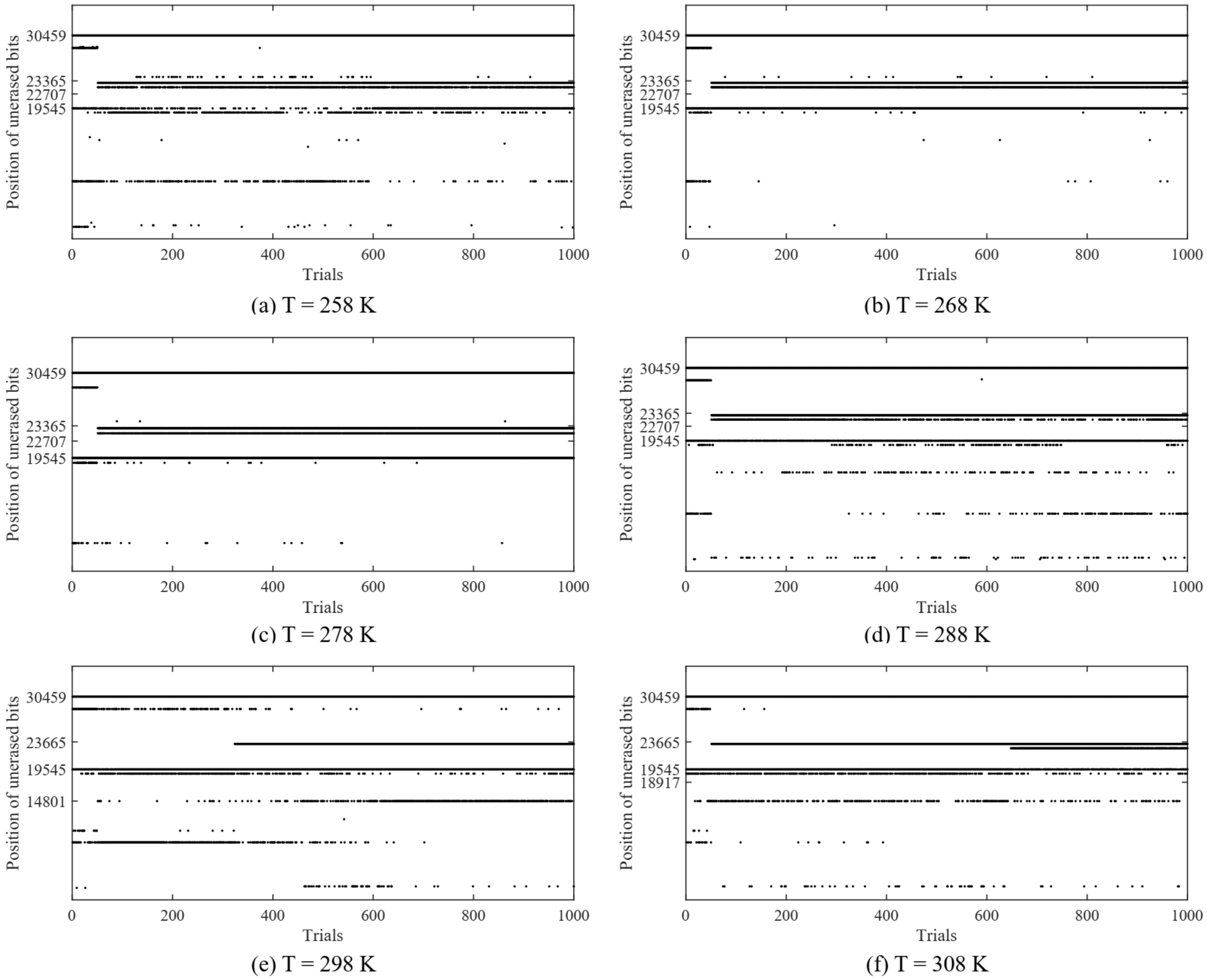
In the offline step, we do not have any information on where the fingerprint is in the P_{erased} curve; therefore, T_{se0} for any sector can be selected by averaging the starting time of erase from general P_{erased} table. T_{ses} changes according to the function PredictNextTOS() in Algorithm 1, while T_r and T_{srs} are constant. N_{iter} is the number of iterations required for adjusting the TOS to pick up the last unerased bit accurately. It depends on the practical characteristics of each sector and the ambient temperature. Finding the fingerprint becomes difficult if a sector has too many bits that have the same charge level in the floating gate at the tail part or if the experiment is executed under low-temperature conditions.

In the online step, T_{se0} is given as a part of a challenge from checker (Fig. 4). N_{iter} will be reduced because of focusing exactly on the tail part (by executing PredictNextTOS()) where the fingerprint can be found. In other words, many steps are not required to examine and observe the entire erasing process.

According to the experimental results, we need, on an average, 35 iterations over approximately 36.4 ms to generate the entire P_{erased} characteristic curve for each sector under a specific ambient temperature. In the offline step, an average of 10 ms is required to generate a 16-bit fingerprint and an average of 50 ms is required to confirm that fingerprint is stable to use. However, an average of only 1 ms is required to regenerate the fingerprint with the given TOS in the online step.

D. UNIQUENESS AND RANDOMNESS IN EXTRACTED FINGERPRINTS

Although we focus our attention on how to achieve the stability (or reliability) of the proposed method with the intrinsic device functions, we have to make sure that the device



(a) T = 258 K

(b) T = 268 K

(c) T = 278 K

(d) T = 288 K

(e) T = 298 K

(f) T = 308 K

Position and occurence of the last unerased bits in tail part of the 1000 sector erase operations under different ambient temperature																			
Position of bit	**1711**	**1800**	**2029**	**2425**	**8617**	**10365**	**12091**	**13787**	**14270**	**14801**	**15234**	**18917**	**19545**	**22707**	**23365**	**24250**	**28595**	**28742**	**30459**
T = 258K	2	20	14	1	264			1	1	5	1	352	556	739					1000
T = 268K		2	1		35					3		40	998	912	949	13	47		1000
T = 278K					34							55	1000	913	949	3	46		1000
T = 288K		2	92		177					136		138	930	522	950		49	1	1000
T = 298K		2	56		335	19	1			414		332	994		677		170		1000
T = 308K			59		24	4				288		403	900	339	950		33		1000
% of occurrence	**0.033**	**0.433**	**3.7**	**0.017**	**14.48**	**0.383**	**0.017**	**0.017**	**0.017**	**14.1**	**0.017**	**22**	**89.63**	**57.08**	**74.58**	**0.267**	**5.75**	**0.017**	**100**

FIGURE 8. Statistical results of error rate at different temperature.

fingerprints are able to satisfy the requirements of high security applications. Therefore, the uniqueness and randomness criteria of generated device fingerprints should be considered thoroughly. The definitions of uniqueness and randomness measurement was introduced in the research of Maiti *et al.* [25]. In this section, we evaluate those criteria with experimental data obtained from our proposed method.

1) UNIQUENESS MEASUREMENT

Uniqueness is defined as the ability of PUFs in generating unique responses from a specific challenge. The uniqueness value of responses among k devices can be estimated by average inter-chip Hamming distance, in this context, as the following equation [25]:

$$HD_{inter-chip} = \frac{2}{k(k-1)} \sum_{i=1}^{k-1} \sum_{j=i+1}^{k} \frac{HD(R_i, R_j)}{n} \times 100\% \tag{4}$$

where R_i and R_j are n-bit responses for a specific challenge from different devices i and j. Our experimental results showed the average $HD_{inter-chip}$ is 48.57%. It is quite close to 50% to indicate the distinguishing feature of device fingerprints.

2) RANDOMNESS MEASUREMENT

Randomness (or uniformity) of device fingerprints can be measured by evaluating the distribution of the number of 0's and 1's in the response bits. The randomness of PUF's responses can be described as follows [25]:

$$Randomness = \frac{1}{n}\sum_{l=1}^{n} R_{i,l} \times 100\% \tag{5}$$

where $R_{i,l}$ is the l-th binary bit of n-bit response i. Our experimental results showed that the randomness of device fingerprints generated by our proposed method is average 47.87% under temperature variation condition. It is quite close to 50% in order to make sure that the device fingerprints are truly random.

V. CONCLUSIONS

The reliability and efficiency of security applications such as the PUF depend greatly on the random seed number (i.e., device fingerprint) generation process in the target hardware device. The operations of flash memories, such as read, program, and erase, have been utilized to generate random numbers based on their random process variation. However, the conventional flash-based fingerprint generation methods require chip or circuit-level modification, which is not easy in real-life applications.

In this work, we proposed a method for generating random device fingerprints based on the intrinsic erase-suspend operation of modern flash memory devices, without modifying the chip or circuits. It was based solely on the intrinsic functions, including erase-and-suspend and soft reset, in modern flash memories. We observed that the long tail part of the erase operation in the flash device could provide a much more stable result and that the effect of environmental variability could be diminished by adopting the incremental erasure using an erase-suspend-reset sequence. The experimental results showed that the proposed method could generate a random fingerprint successfully from the location of the last unerased bit, over a wide range of operating temperatures.

REFERENCES

[1] A. Vijayakumar, V. Patil, and S. Kundu, "On improving reliability of SRAM-based physically unclonable functions," *J. Low Power Electron. Appl.*, vol. 7, no. 1, p. 2, Jan. 2017.

[2] D. Barboza, "In China, knockoff cellphones are a hit," The New York Times, 2009.

[3] *Winning the Battle Against Counterfeit Semiconductor Products*, Semicond. Ind. Assoc., Anti-Counterfeiting Task Force, San Jose, CA, USA, 2013.

[4] A. Braeken, "PUF based authentication protocol for IoT," *Symmetry*, vol. 10, no. 8, p. 352, Aug. 2018.

[5] T. McGrath, I. E. Bagci, Z. M. Wang, U. Roedig, and R. J. Young, "A PUF taxonomy," *Appl. Phys. Rev.*, vol. 6, no. 1, Mar. 2019, Art. no. 011303.

[6] K.-D. Suh, B.-H. Suh, Y.-H. Lim, J.-K. Kim, Y.-J. Choi, Y.-N. Koh, S.-S. Lee, S.-C. Kwon, B.-S. Choi, J.-S. Yum, J.-H. Choi, J.-R. Kim, and H.-K. Lim, "A 3.3 V 32 mb NAND flash memory with incremental step pulse programming scheme," *IEEE J. Solid-State Circuits*, vol. 30, no. 11, pp. 1149–1156, Nov. 1995.

[7] R. Pappu, "Physical one-way functions," *Science*, vol. 297, no. 5589, pp. 2026–2030, Sep. 2002.

[8] B. Gassend, D. Clarke, M. van Dijk, and S. Devadas, "Controlled physical random functions," in *Proc. 18th Annu. Comput. Secur. Appl. Conf.*, 2002, pp. 149–160.

[9] B. Gassend, D. Clarke, M. van Dijk, and S. Devadas, "Silicon physical random functions," in *Proc. 9th ACM Conf. Comput. Commun. Secur. (CCS)*, 2002, pp. 148–160.

[10] J. W. Lee, D. Lim, B. Gassend, G. E. Suh, M. van Dijk, and S. Devadas, "A technique to build a secret key in integrated circuits for identification and authentication applications," in *Symp. VLSI Circuits. Dig. Tech. Papers*, 2004, pp. 176–179.

[11] S. S. Kumar, J. Guajardo, R. Maes, G.-J. Schrijen, and P. Tuyls, "The butterfly PUF protecting ip on every FPGA," in *Proc. IEEE Int. Workshop Hardw.-Oriented Secur. Trust*, 2008, pp. 67–70.

[12] J. Guajardo, S. S. Kumar, G.-J. Schrijen, and P. Tuyls, "FPGA intrinsic PUFs and their use for ip protection," in *Proc. Int. Workshop Cryptograph. Hardw. Embedded Syst.* Springer, 2007, pp. 63–80.

[13] S. Morozov, A. Maiti, and P. Schaumont, "A comparative analysis of delay based PUF implementations on FPGA," *IACR Cryptol. ePrint Arch.*, vol. 2009, p. 629, Dec. 2009.

[14] C. Bösch, J. Guajardo, A.-R. Sadeghi, J. Shokrollahi, and P. Tuyls, "Efficient helper data key extractor on FPGAs," in *Proc. Int. Workshop Cryptograph. Hardw. Embedded Syst.* Springer, 2008, pp. 181–197.

[15] M.-S. Kim, D.-I. Moon, S.-K. Yoo, S.-H. Lee, and Y.-K. Choi, "Investigation of physically unclonable functions using flash memory for integrated circuit authentication," *IEEE Trans. Nanotechnol.*, vol. 14, no. 2, pp. 384–389, Mar. 2015.

[16] C. W. O'donnell, G. E. Suh, and S. Devadas, "PUF-based random number generation," in *MIT CSAIL CSG Technical Memo 481*. 2004, p. 2004. [Online]. Available: http://csg.csail.mit.edu/pubs/memos/Memo-481/Memo-481.pdf

[17] G. E. Suh and S. Devadas, "Physical unclonable functions for device authentication and secret key generation," in *Proc. 44th ACM/IEEE Design Autom. Conf.*, Jun. 2007, pp. 9–14.

[18] P. Prabhu, A. Akel, L. M. Grupp, S. Y. Wing-Kei, G. E. Suh, E. Kan, and S. Swanson, "Extracting device fingerprints from flash memory by exploiting physical variations," in *Proc. Int. Conf. Trust Trustworthy Comput.* Springer, 2011, pp. 188–201.

[19] Y. Wang, W.-K. Yu, S. Q. Xu, E. Kan, and G. E. Suh, "Hiding information in flash memory," in *Proc. IEEE Symp. Secur. Privacy*, May 2013, pp. 271–285.

[20] B. Ray and A. Milenković, "True random number generation using read noise of flash memory cells," *IEEE Trans. Electron Devices*, vol. 65, no. 3, pp. 963–969, Mar. 2018.

[21] *Microchip Flash Memory Device (SST39VF1601)*, 2019. Accessed: Aug. 3, 2019.

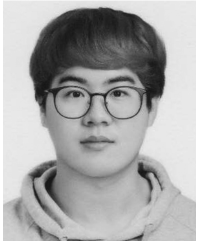
[22] H.-W. Tseng, L. Grupp, and S. Swanson, "Understanding the impact of power loss on flash memory," in *Proc. 48th Design Autom. Conf. (DAC)*, pp. 35–40, ACM, 2011.

[23] 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.

[24] S. Jia, L. Xia, Z. Wang, J. Lin, G. Zhang, and Y. Ji, "Extracting robust keys from NAND flash physical unclonable functions," in *Proc. Int. Conf. Inf. Secur.* Springer, 2015, pp. 437–454.

[25] A. Maiti, V. Gunreddy, and P. Schaumont, "A systematic method to evaluate and compare the performance of physical unclonable functions," Cryptol. ePrint Arch., Tech. Rep. 2011/657, 2011.

THE-NGHIA NGUYEN (Student Member, IEEE) received the M.S. degree in computer engineering from Yeungnam University, South Korea, in 2018. He is currently pursuing the Ph.D. degree with the Department of Software Convergence, Soongsil University, South Korea. His research interests include low-power embedded system application, machine learning, and hardware security application.



SUNGHYUN PARK is currently pursuing degree with the Department of Smart System Software, Soongsil University, South Korea. His research interests include low-power embedded system application, machine learning, and hardware security application.

Donghwa Shin (Senior Member, IEEE) received the B.S. degree in computer engineering and the M.S. and Ph.D. degrees in computer science and electrical engineering from Seoul National University, Seoul, South Korea, in 2005, 2007, and 2012, respectively. He joined the Ming Hsieh Department of Electrical Engineering, University of Southern California, Los Angeles, CA, USA, as a Visiting Scholar, and the Dipartimento di Automatica e Informatica-EDA Group, Politecnico di Torino, Turin, Italy, as a Researcher. He was an Assistant Professor with the Department of Computer Engineering, Yeungnam University, Gyeongsan, South Korea, from 2014 to 2017. He is currently an Assistant Professor with the Department of Smart Systems Software, Soongsil University. His research interests include system-level low-power techniques for embedded systems and hybrid power system design. He is currently focusing on the next-generation computing and energy resources including neuromorphic computing. He serves (and served) as a Reviewer of the IEEE TComputers, TCAD, TVLSI, ACM TODAES, TECS, JETC, the *International Journal of Electrical Power and Energy Systems*, the *Journal of Applied Electrochemistry*, the *Journal of Signal Processing Systems*, and the *International Symposium on Industrial Electronics*. He serves on the Technical Program Committee of the IEEE and ACM technical conferences, including Design Automation and Test in Europe (DATE), International Symposium on Low-Power Electronics and Design (ISLPED), Asia South-Pacific Design Automation Conference (ASP-DAC), ACM Great Lakes Symposium on VLSI (GLSVLSI), International Green and Sustainable Computing Conference (IGSC), and IFIP/IEEE International Conference on Very Large Scale Integration.