FLORIDA INTERNATIONAL UNIVERSITY

Miami, Florida

FACILITATING EMERGING NON-VOLATILE MEMORIES FOR ENSURING SECURITY AND DEVICE TRUST

A dissertation submitted in partial fulfillment of the

requirements for the degree of

DOCTOR OF PHILOSOPHY

in

ELECTRICAL AND COMPUTER ENGINEERING

by

Farah Ferdaus

2022

To: Dean John L. Volakis
College of Engineering and Computing

This dissertation, written by Farah Ferdaus, and entitled Facilitating Emerging Non-volatile Memories for Ensuring Security and Device Trust, having been approved in respect to style and intellectual content, is referred to you for judgment.

We have read this dissertation and recommend that it be approved.

Kemal Akkaya

Ou Bai

Ananda Mohan Mondal

Md Tauhidur Rahman, Major Professor

Date of Defense: November 04, 2022

The dissertation of Farah Ferdaus is approved.

Dean John L. Volakis
College of Engineering and Computing

Andrés G. Gil
Vice President for Research and Economic Development
and Dean of the University Graduate School

Florida International University, 2022

© Copyright 2022 by Farah Ferdaus

All rights reserved.

DEDICATION

To my mother, Shahanara Begum, without whose unfailing support and continuous encouragement, I would not achieve my goals.

And my fourteen months old son, Zayyan B. Talukder, who re-birthed me, supporting me spiritually, the source of my strength and inspiration to accomplish my targets.

ACKNOWLEDGMENTS

First, I would like to express my deepest thanks to my committee members for their valuable suggestions, insightful comments, and constructive technical feedback. I want to express my very profound gratitude to my fellow lab-mate and mentor, Dr. Bashir Mohammad Sabquat Bahar Talukder, for stimulating discussions and helping me during critical times. Finally, I would also like to acknowledge the support provided by National Science Foundation; this dissertation is mostly based upon the work supported by National Science Foundation under the grant number DGE-2114200.

ABSTRACT OF THE DISSERTATION

FACILITATING EMERGING NON-VOLATILE MEMORIES FOR ENSURING SECURITY AND DEVICE TRUST

by

Farah Ferdaus

Florida International University, 2022

Miami, Florida

Professor Md Tauhidur Rahman, Major Professor

Emerging non-volatile memory (NVM) technologies such as Magneto-resistive random access memory (MRAM), Resistive RAM (ReRAM), Ferroelectric RAM (FeRAM), and Phase Change Memory (PCM) have shown great potential for building next-generation computing systems because of their near-zero leakage, unlimited endurance, scalability, fast speed, and high-density characteristics. While the MRAM has the potential to replace static RAM (SRAM) in large-scale and low-power on-chip caches and dynamic RAM (DRAM) in energy-efficient main memory, the ReRAM is a promising candidate for low-power and large-scale main memory and storage systems. Emerging cost-effective, low latency, high performance, and low energy memories can be used as a stand-alone memory chip or integrated at the same process nodes of microcontrollers, system-on-chip, and FPGAs; thus becoming an excellent candidate for applications requiring high security and efficient NVM embedded in semiconductors such as Internet of Things (IoT), wearables, tablets, smartphones, consumer electronics, artificial intelligence, industrial, automotive, and medical. While these memories play essential roles in performance, they can also be used as core components to ensure system and supply-chain security. Higher levels of security can be achieved by implementing security primitives or digital signatures within integrated circuit semiconductor hardware and using them in

secure communication, authentication, and other cryptographic operations. On the other hand, these inherent properties of emerging memories can be used to protect cloning and other means of counterfeiting. This thesis will explore the prospect of emerging memory chips to ensure system and supply-chain security issues. First, a new idea of hiding information in commercial off-the-shelf (COTS) ReRAM is presented for secure and covert data storage. Second, this work proposes a low-cost, non-invasive, robust watermarking technique using COTS ReRAM to identify major counterfeiting types, such as remarked, overproduced, and cloned. Third, COTS MRAM-based security primitive, i.e., True random number generator or TRNG, is proposed and demonstrated for low-cost alternatives to generate random keys, cryptographic nonces, session keys, one-time-pad, etc., for the secure operation of electronic devices and systems.

TABLE OF CONTENTS

CHAPTER PAGE

1. INTRODUCTION 1
1.1 Motivation 1
1.2 Research Problem 3
1.3 Research Objectives 6
1.3.1 Research Objective 1: Using ReRAM for Device Watermarking and Data Hiding 7
1.3.2 Research Objective 2: Using MRAM as a True Random Number Generator (TRNG) 8
1.4 Research Approach 10
1.4.1 Hiding Information in ReRAM 11
1.4.2 Preventing Counterfeit ReRAM Devices 11
1.4.3 MRAM-based TRNG 12
1.5 Dissertation Contributions 12
1.6 Organization of the Dissertation 15

2. Emerging Non-volatile Memories 16
2.1 MRAM 17
2.2 ReRAM 20

3. HIDING INFORMATION IN ReRAM 22
3.1 Motivation 24
3.2 Proposed Hiding Technique 25
3.2.1 Cell Characterization 26
3.2.2 Information Hiding Technique 28
3.2.3 Hidden Information Retrieval Technique 29
3.2.4 Encoding/Decoding Secret Information 29
3.3 Performance Evaluation 32
3.3.1 Evaluation Setup 32
3.3.2 Cell Characterization 34
3.3.3 Appropriate Replica Size Selection 34
3.3.4 Retention Characteristics 36
3.3.5 Performance Analysis 37
3.3.6 Initial Stress Tolerance 39
3.3.7 Post-Hiding Stress Tolerance 40
3.3.8 Robustness Analysis 45
3.3.9 Security Analysis 48
3.3.10 Evaluation Summary 53
3.3.11 Discussion 54
3.4 Conclusion 55

4. IDENTIFY COUNTERFEIT RERAM DEVICES . . . 57
4.1 Motivation . . . 58
4.2 Proposed Watermarking Technique . . . 60
4.2.1 Imprinting Scheme . . . 60
4.2.2 Retrieval Scheme . . . 61
4.3 Performance Evaluation . . . 63
4.3.1 Robustness Analysis . . . 64
4.3.2 Performance Analysis . . . 67
4.3.3 Security Analysis . . . 68
4.3.4 Discussion . . . 70
4.4 Conclusion . . . 71

5. MRAM-BASED ROBUST TRNG . . . 72
5.1 Motivation . . . 73
5.2 Background Preliminaries . . . 75
5.2.1 MRAM Operation . . . 75
5.2.2 Entropy Source . . . 76
5.3 Generating Random Numbers using COTS MRAM . . . 78
5.3.1 Appropriate Reduced Time Selection . . . 79
5.3.2 MRAM Cells Characterization . . . 80
5.3.3 Appropriate Cell Location Selection . . . 81
5.3.4 Low-overhead Post-processing . . . 82
5.4 Experimental Results . . . 83
5.4.1 Selection of t_W . . . 84
5.4.2 Characterization of Temporally Unbiased Cells . . . 85
5.4.3 Evaluation . . . 88
5.4.4 Robustness Analysis . . . 88
5.4.5 Throughput Analysis . . . 90
5.5 Discussion . . . 92
5.6 Conclusion . . . 93

6. CONCLUSION AND FUTURE WORK . . . 94

BIBLIOGRAPHY . . . 98

VITA . . . 106

LIST OF TABLES

TABLE PAGE

3.1 ReRAM Chip Specifications [Fuj19]. 33

3.2 Post-hiding stress tolerance statistics using all test chips. 44

5.1 MRAM Chip Specifications [Evea]. 84

5.2 Cell statistics after applying cell characterization algorithm. 85

5.3 The worst-case NIST test results at the reduced t_W. 87

5.4 The worst-case NIST test results verify the robustness of our proposed TRNG. 89

LIST OF FIGURES

FIGURE PAGE

1.1 Threat model for modern horizontal supply chain [FC18]. 4

1.2 (a) Threat models for user-to-server communication protocol, (b) A TRNG-based solution to improve security. 6

2.1 (a) Toggle MRAM cell structure with MTJ. (b) Schematic representation of Gaussian resistance (R_{Low} and R_{High} states) distribution of larger-sized MTJ array [ADS16]. (c) Principle of tunneling magnetoresistance (TMR) [TDK]. 18

2.2 ReRAM cell structure with two logic states [MCYC15]. 21

3.1 Operation steps used to hide information. 31

3.2 ReRAM cell characterization under stress- (a) $t_{Set,256}$ and (b) $t_{Reset,256}$. . 35

3.3 Influence of replica size on the hidden information, using- (a) $t_{Set,[32,256]}$, and (b) $t_{Reset,[32,256]}$. 36

3.4 Retention characteristics of the hidden information using $t_{Set,256}$. 37

3.5 Retention characteristics of the hidden information using $t_{Reset,256}$. . . . 37

3.6 Influence of initial stress ($50K$) on the hidden information using- (a) $t_{Set,256}$ and (b) $t_{Reset,256}$. 40

3.7 Hidden data (hiding stress count, $\mathcal{N} = 15K$) at different post-hiding stress level- (a)–(b) $t_{Set,256}$ at stress count $130K$, and $140K$; (c)–(d) $t_{Reset,256}$ at stress count $40K$, and $50K$. 41

3.8 Post-hiding stress tolerance of concealed data at different hiding stress levels, using- (a)–(c) $t_{Set,256}$, and (d)–(f) $t_{Reset,256}$. 43

3.9 Post-hiding stress tolerance of concealed data considering maximum stressing using- (a) $t_{Set,256}$ and (b) $t_{Reset,256}$. 44

3.10 Robustness analysis (a)–(c) before (d)–(f) after high-temperature baking ($80°C$) with- $t_{Set,256}$. 46

3.11 Robustness analysis (a)–(c) before (d)–(f) after high-temperature baking ($80°C$) with- $t_{Reset,256}$. 47

3.12 Retrieved data with an incorrect initial address at ($15K$) stressing used to hide information with- (a) $t_{Set,256}$ (b) $t_{Reset,256}$. 50

3.13 Data hiding with enhanced security: replication + left circular rotation. The start bit is marked with yellow to track the rotation. 52

3.14 Retrieved data with an correct key with $\mathcal{N} = 15K$ stressing used to hide information with- (a) $t_{Set,256}$ (b) $t_{Reset,256}$. 52

3.15 Retrieved data with an incorrect key with $\mathcal{N} = 15K$ stressing used to hide information with- (a) $t_{Set,256}$ (b) $t_{Reset,256}$. 53

4.1 Steps used for ReRAM watermarking. 60

4.2 Imprinted data at different stress count- (a)–(c) $t_{Set,256}$ at stress count $5K$, $10K$, and $15K$; (d)–(f) $t_{Reset,256}$ at stress count $10K$, $15K$, and $20K$. 65

4.3 Verifying watermark in test chips, using- (a) $t_{Set,256}$, and (b) $t_{Reset,256}$. . 66

4.5 (a) Bypassing watermarking scheme (b) Secured watermarking. 69

5.1 Write enable ($\overline{W}$) controlled write cycle of MRAM chip. 75

5.2 Different state of MRAM cell. Red arrow defines the magnetic orientation of free layer, blue arrow defines the magnetic orientation of fixed layer. (a) parallel magnetic orientation in low resistance state (R_{Low}), (b) anti-parallel magnetic orientation of high resistance state (R_{High}), and (c) abnormal magnetic orientation of free layer when the write current is applied for insufficient amount of time while changing the magnetic orientation. Here, the magnetic orientation of free layer is perpendicular to magnetic orientation of fixed layer and resistance will be in-between R_{Low} and R_{High}. 78

5.3 The characteristics of memory cells of $C4$: most of the cells are purely invariant (stuck at ‘0’/‘1’). 86

CHAPTER 1

INTRODUCTION

1.1 Motivation

The field of cybersecurity has predominantly focused on the security of the software and the communication network due to the assumption that the underlying hardware is trustworthy and reliable. However, emerging attack vectors on hardware originating from the untrusted global supply chain, along with unintentional design and integration-caused hardware vulnerabilities, questions the well-regarded notion of hardware 'root-of-trust'. Furthermore, the connected ecosystem of numerous sensor-enabled smart systems around us has introduced unprecedented hardware-assisted attack vectors. In this thesis, we are unlocking the potential of emerging memories in cybersecurity before and after the deployments of memory chips in the system. During the post-deployment, data in transit and data in motion are the primary targets of an attacker. Current solutions are either expensive or require additional hardware of different types. This research focuses on using a single hardware component that is already in the system to ensure cybersecurity, which is possible due to the intrinsic properties of emerging memory chips. In this research, we use emerging memory chips to ensure low-cost data and communication security by hiding data using the intrinsic properties of emerging memory chips and generating high-quality random numbers, respectively. To ensure supply chain authenticity, we use the same memory chips to watermark the manufacturer information, which can be imprinted in both commercial stand-alone or embedded memory chips.

In past two decades a lot of research has been done on memory-based security solutions. Memory-based security solutions are cheaper to implement as they do not require any new hardware, and most electronic devices are already equipped

with memory components. However, conventional memory-based security primitives or anti-counterfeiting solutions have at least one of the following limitations [KPH+19, GKST07, FBTR22] - (i) take longer time to evaluate, (ii) require a new power cycle, (iii) power inefficiency, and (iv) unreliable under adverse operating conditions. Fortunately, emerging memories are inherently free from such limitations (faster, power-efficient, and robust against adverse operating conditions). Therefore, emerging memories are more preferred candidates for generation security primitives and anti-counterfeiting solutions. Additionally, conventional memories (dynamic random access memory or DRAM, static random access memory or SRAM, NAND flash, etc.) are also expected to be replaced with emerging memories in all future electronic devices for better performance and potential gains in manufacturing costs. Unfortunately, very limited research has been done on emerging memories from the security perspective; and even though there are few, those research are mostly performed in simulated environments or need strict requirements on operating conditions (e.g., precise control over current/voltage pulse width/magnitude/waveform) [YDW+18, VDNP16, PWG+20, YAM+21]. In this dissertation, we focus on mitigating this research gap and exploring various security solutions using commercially available emerging technologies. Among many different emerging memory technologies, MRAM (magneto-resistive random-access memory) and ReRAM (resistive random-access memory) are the most promising candidate due to their high density, scalability, speed, compatibility with existing CMOS technology, non-volatility, low leakage power, asymmetric read/write latency, and asymmetric read/write energy. SRAM- or DRAM-similar read latency and energy with cell size between DRAMs and SRAMs under the same capacity makes MRAM a promising replacement of SRAM for large-size and low-power on-chip caches as well as an energy-efficient alternative to DRAM. ReRAM has higher densities compared to DRAM, better

endurance compared to NAND flash, and is capable of achieving DRAM-similar read latency and energy. Therefore, these features make them potential candidates for building large-size and low-power main memory and high-performance storage. Hence, we mainly focus on these two types of emerging memories while exploring new security solutions.

1.2 Research Problem

Existing memory-based security solutions can be broadly categorized into three domains- (i) memory-based security primitives, such as, PUF (physical unclonable function) and TRNG (true random number generator) [SD07, GKST07, MJS+16, RXF+14, KPH+19, WYW+12, KLK17], (ii) memory-based anti-counterfeiting solutions such as, PUF and watermarking [TMR+20, GDT14, SD07, GKST07], and (iii) enhance security of storage device by data hiding [WYX+13]. Therefore, this desertion reevaluates the following security problems by discussing these three security domains from the perspective of emerging memory-

1. Semiconductor supply chain vulnerability: Fabricating chips in untrusted facilities is increasing worldwide, paving the way for counterfeit chips to the supply chain. Especially it has become a global concern since the introduction of the horizontal supply chain (Fig. 1.1). In a horizontal supply chain, each step of the chip manufacturing process is taken care by different parties [FC18]. Without establishing a root-of-trust among these fabrication facilities, chips can be counterfeited at any stage of this chip manufacturing process. For example, the design can be cloned while flowing among different facilities, and the fabrication and assembly facility can overproduce and distribute to the market. Moreover, a counterfeiter can recycle chips from old PCB boards and sell them as new. Among different types of counter-

feiting, cloned, remarked, and recycled ICs are the most common type and usually occupy >80% of the total counterfeit IC market share [FC18]. Furthermore, memory and memory-based integrated devices (microprocessor, microcontroller, FPGA, etc.) are the most targeted by the counterfeiters as they are the most expensive component of a system. Therefore, the memory-based anti-counterfeiting solution can protect memory and memory-integrated ICs against such counterfeitings. Among different memory-based anti-counterfeiting solutions, memory-based PUF and watermarking techniques are the most effective and can prevent a wide range of counterfeitings, including recycled and cloned ICs. However, the PUF-based technique usually requires maintaining a centralized database which is expensive to maintain. Therefore, this dissertation aims to design an emerging memory-based chip watermarking protocol that does not require any special lab facility or hardware modification.

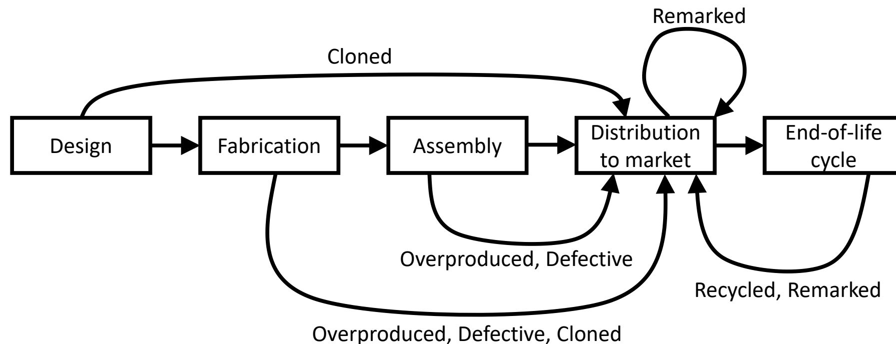


Figure 1.1: Threat model for modern horizontal supply chain [FC18].

2. Security vulnerability of storage devices: The recent progress in computer hardware enables one to perform a large computation within a reasonable amount of computation time. Therefore, an attacker can perform a brute-force attack on encrypted data within a reasonable time period using such modern computation devices. One way to prevent such attacks is to increase the key length arbitrarily; however, increasing the key length is not always possible if there is a resource constraint in the victim device. In addition, the conventional encryption mecha-

nism cannot provide any security against ransom attacks even with an arbitrarily large encryption key [MP17]. However, it is well demonstrated that the data hiding technique can be an excellent mechanism against such brute-force attacks and ransom attacks. Data hiding is a relatively new technique that is analogous to steganography. Data hiding techniques can safeguard data by concealing the secret information so that the secret message's existence cannot be revealed by regular read/write operation, which is the key advantage of data hiding over standard cryptographic techniques. Additionally, in the particular event of a ransom attack, even if the ransomware encrypts the whole storage device, the hidden data in memory can still be recovered as data hiding uses different properties from regular read/write operations. Therefore this dissertation also aims to design a data hiding technique that can enhance security of critical user data against brutefore and ransom attack.

3. On-device security primitives: Memory-based TRNGs/PUFs are widely used in many application, including generating device signature, statistics, simulation, gaming, etc. However, the most significant use of security primitive is securing communication between a server and an individual. Fig. 1.2a shows the simplest communication protocol, where both server and user communicate with each other without using any encryption technique. Such communication is not secure and can be easily eavesdropped on by simply tapping the communication channel. However, introducing a random number generator can easily defend such attacks (Fig. 1.2b). In this protocol, the user first sends a request to the server to initiate the communication. In reply, the server sends its public key (K_{public}) to the user. The user generates a random number (K_{RNG}) and encrypts with the public key. Then he sends the encrypted random number ($K_{public}(K_{RNG})$) to the server. The server decrypts the random number using its private key ($K_{private}$). Once both the user and server have the K_{RNG}, all further communications are encrypted with K_{RNG}.

Therefore, K_{RNG} plays the role of establishing a secure communication channel which is also known as the session key. The communication protocol presented in Fig. 1.2b is secure as long as the K_{RNG} generated at user end has the sufficient entropy. This random number-based protocol is widely used as a part of modern HTTPS protocol. Although TRNGs can be implemented as a separate hardware module, memory-based security primitives are preferred as memories are readily available in most electronic devices. In this dissertation, we propose a high-speed TRNG with high entropy, which can be applicable to futuristic devices equipped with emerging memories.

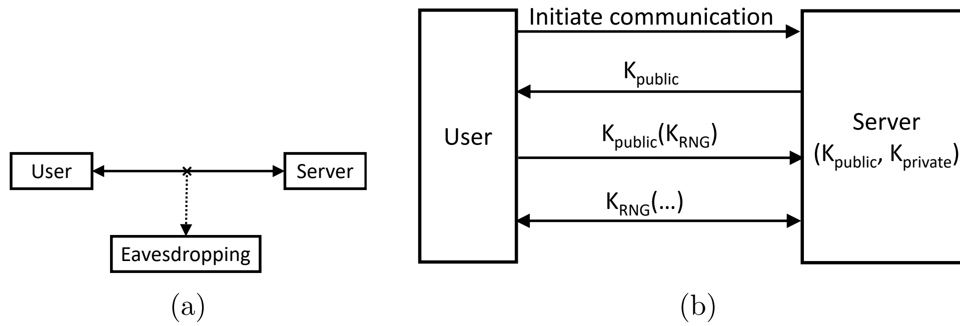


Figure 1.2: (a) Threat models for user-to-server communication protocol, (b) A TRNG-based solution to improve security.

1.3 Research Objectives

Below are the research objectives (RO) and corresponding research questions (RQ) that we explore in this dissertation.

1.3.1 Research Objective 1: Using ReRAM for Device Watermarking and Data Hiding

Watermarking and data hiding are usually done on top of stored data. The current ReRAM is primarily used as storage memory; therefore, we explored the possible opportunity to watermark/hide data on ReRAM. We answer the following questions to achieve this objective —

- **RQ1:** Which property of ReRAM is appropriate for watermarking and data hiding? What is the physics behind it?
- **RQ2:** How robust is the watermark data? What would be the appropriate protocol to verify the watermark data? What is the imprinting/reading speed?
- **RQ3:** How robust is the data hiding technique? How long can it sustain, considering the normal usage of ReRAM?
- **RQ4:** How secure the hidden data is? Is there any scope to improve security?

We explored the following goals to answer the above research questions —

1. **Robustness and performance analysis of watermarking technique:** We analyzed the robustness of our proposed watermarking technique by varying the operating conditions (voltage, temperature). We also analyzed the performance of our watermarking technique by experimentally evaluating the imprinting/reading speed of watermark data. We also demonstrated a simple protocol to verify the watermark data securely.
2. **Robustness and performance analysis of data hiding technique:** Similar to the watermarking technique, we analyzed the data hiding technique's robustness by varying operating conditions. We evaluated encoding and retrieval time as a basic data hiding scheme's performance metric. We also verified the im-

pact of the typical read/write operation before and after hiding information on ReRAM.

3. **Security analysis of data hiding technique:** We performed a detailed security analysis of our proposed data hiding scheme and demonstrated a possible security improvement in order to make the data more invisible to the attacker.

1.3.2 Research Objective 2: Using MRAM as a True Random Number Generator (TRNG)

Throughput (speed) of TRNG is an important consideration for most applications. Due to low access time of MRAM, we aim to explore possible opportunities to generate random numbers using MRAM. We answer the following research questions in order to attain this research objective.

- **RQ5:** Which property of MRAM is suitable for generating random numbers? What is the physics behind it?
- **RQ6:** How robust will it be if we can use MRAM as a TRNG? How can we verify if the generated sequence is truly random?
- **RQ7:** What would be the TRNG throughput (speed)? Is it comparable with existing conventional memory-based solutions?

To answer above questions, this dissertation explores the following goals —

1. **Developing TRNG technique:** We explore the physical characteristics of MRAM and determine the impact of different timing parameters on MRAM. We characterized the data error induced by the reduced value of timing parameters. We also quantified the temporal variation of the error pattern and determined if

this bit error can generate true random numbers. We used the NIST test suite[1] to verify the randomness of the error pattern. The NIST test suite is a standardized tool to check the mathematical properties of a random sequence.

2. **Source of entropy:** The limited source of entropy of existing memory-based TRNG is the key basis of exploring proposed memory-based TRNG. The startup state-based SRAM TRNG requires minor modifications to the conventional SRAM array to improve the instability of SRAM cells and ensure the randomness and quality of TRNGs in the context of device stress and long-term aging. Hence, the SRAM-based TRNG suffers from limited entropy, the throughput is considerably lower, and it needs a new power cycle to generate the random number. Similarly, the throughput of the Flash-based TRNG, which utilizes the program disturb characteristics, is very low. In addition, Flash-based TRNG, which utilizes partial programming, requires precise control of program time, which is chip-specific. This technique also requires periodic refresh operations for identifying noisy bits due to flash memory's finite data retention property. Furthermore, DRAM-based TRNG can be generated either by varying DRAM Latency parameters (for example, retention/activation/precharge latency) or using the startup condition. Likewise, SRAM, the limitation of the DRAM startup-based TRNG, is the requirement of a new power cycle to generate the random number, which is not feasible for a run-time system. On the other hand, DRAM latency-based TRNG requires precise control over specific memory addresses, which is hard to achieve due to the granular structure of DRAM. In contrast, the main hypothesis of MRAM-based TRNG is that if the MRAM cell resistance is approximately halfway between the low and high resistance states (metastable state),

[1] NIST test suite is the widely accepted standard to verify randomness of a sequence. It checks fifteen standardized mathematical properties of random number generator.

the corresponding output should be random due to the random thermal noise. By reducing timing parameters during the write operation, we can reach such metastable states, and as a result, the MRAM cell demonstrates very indeterministic or chaotic behavior. Therefore, the magnetic orientation can not settle into one particular direction that gives a strong '1' or strong '0' and we have used this phenomenon to generate random numbers. The main advantages of our proposed MRAM-based TRNG are- it does not require any new power cycle, throughput is considerably high, and timing parameters are easily controllable due to the inherent architecture of MRAM.

3. **Analyzing the robustness and the resiliency of the proposed TRNG:** Since we are using physical phenomena of MRAM chips, the quality of random numbers can be affected by external influences. Furthermore, an attacker can change the operating condition to influence the TRNG output. For example, a high voltage can be applied to reduce entropy, which has been demonstrated in other hardware-based TRNG [XRF+14]. Therefore, once we adopt the TRNG algorithm, we induce practical variation in operating conditions (temperature, external magnetic field) to verify the robustness of the proposed TRNG.

4. **Analyzing the performance of the proposed TRNG:** We also evaluated the throughput (speed) of our proposed TRNG from experimental data and compared it with existing conventional memory-based solutions.

1.4 Research Approach

This research aims to ensure hardware-based security and trust for in-transit and rested data using emerging memory technologies. Emerging technologies offer noise sources that can be utilized to design robust, high-throughput, and high-quality

security primitives (i.e., true random number generator or TRNG). Besides, novel approaches, e.g., data hiding or digital imprinting, can prevent IC piracy and securely store information, strengthening the integration of emerging memories in the modern computing system. The following subsections describe our research methodologies through this dissertation proposal.

1.4.1 Hiding Information in ReRAM

First, to hide the information, we perform repeated switching operations (alternatively writing 0's and 1's) to change the physical properties of commercial ReRAM; therefore, the time to write '0' (t_{w0}) and '1' (t_{w1}) of stressed cells deviates from the fresh cells. We observe that both t_{w0} and t_{w1} increase due to the repeated switching operation. After a certain number of switching operations, the stressed cells become completely separable from fresh cells. Each switching operation gradually degrades the resistance of high resistance states, which are permanent; thus cannot be reversed. Using this unique property, we demonstrated that ReRAM could be an excellent storage medium to hide information. Our proposed cost-effective technique can be implemented using a software program in a low-level memory interface.

1.4.2 Preventing Counterfeit ReRAM Devices

Our proposed cost-effective anti-counterfeit solution, watermarking technique, imprints the watermark through repeatedly stressing the memory cells by alternatively writing '1' and '0'. As mentioned in Sect. 1.4.1, repeatedly stressing the ReRAM cell increases its write time (for both logic '0' and '1'). To this extent, our technique imprints logic '0' and '1' by representing the fresh and stressed memory cells, respectively. Later, we retrieve the imprinted sequence by observing the write time of

corresponding memory cells. Thus, our proposed technique can be directly deployed into available commercial products.

1.4.3 MRAM-based TRNG

We propose a technique of generating consumer-off-the-shelf (COTS) MRAM-based TRNG that relies on the bit errors obtained due to reduced write latency. Memory cells show indeterministic characteristics at the reduced latency and generate erroneous bits due to unreliable write operation; therefore, appropriate reduced time selection is essential for generating robust, high-quality TRNG using MRAM. Only a few memory cells, hence addresses are error-prone at a specific suboptimal write latency. Hence, memory cells are classified as persistent and noise-prone cells through error characterization, and a subset of erroneous memory cells is determined to generate TRNG. Noisy cells are excellent candidates to generate TRNG. However, to generate a robust TRNG, we need a cell selection algorithm to filter biased cells from those noise-prone cells because all cells do not provide the same entropy level. Our proposed cell selection algorithm filter unbiased cells by determining an appropriate threshold.

1.5 Dissertation Contributions

The dissertation contributions can be summarized as follows —

1. We proposed a low-cost technique for hiding data in commercially available ReRAM chips to conceal sensitive data preventing the critical and bothersome data privacy problem.

- The impact of repeated stressing on ReRAM *write (set/reset)* time is characterized experimentally, and we observe that the ReRAM *set/reset* time increases monotonically with the stressing level.

- In our hiding scheme we define fresh ReRAM cells as logic '0' and partially worn-out memory cell as logic '1'. We retrieve the hidden data by observing ReRAM *write (set/reset* time through standard digital interfaces.

- The silicon results verify the robustness, performance, stress tolerance (typical memory usage), and security of our proposed technique in multiple COTS ReRAM chips.

2. Our proposed low-cost technique of embedding watermarks in devices with ReRAM manipulates ReRAM's analog physical characteristics through repeated switching (*set/reset*) operations to prevent perennial electronic counterfeiting problems.

   - Imprinting is performed through repeated switching (i.e., flipping ReRAM cell content) operations to only those ReRAM addresses, which are supposed to hold the logic '1' of the watermarked data. The switching operations are repeated until sufficient differences are developed in the *set/reset* time between fresh and stressed memory cells. Retrieval of imprinted watermarked data is performed through the standard digital interface by observing ReRAM *set/reset* time.

   - The throughput and robustness of our watermarking technique in multiple commercial ReRAM chips validate the direct deployment of the proposed scheme into available commercial products.

   - As the technique is irreversible, the imprinted watermark can not be easily removed or destroyed, making the watermark tamper-proof.

3. We demonstrate a true random number generation technique from energy-efficient COTS MRAM chips by manipulating the write latency of MRAM.

    - We extensively characterize the errors obtained from reducing the write time from the manufacturer's recommended value during the write operation. Some cells' errors are entirely random, whereas some cells show deterministic behavior. Furthermore, we propose an algorithm to select the most suitable memory cells that exhibit true randomness to generate robust and high-quality random numbers.
    - The system throughput and robustness analysis of the proposed TRNG in multiple COTS Everspin toggle MRAM chips under a wide range of operating conditions validate the direct deployment of the MRAM-based TRNG into available commercial products.

In summary, this dissertation paves the way to replace conventional memories with emerging memories in various security applications that are unexplored in previous research works. My dissertation demonstrates the capability of emerging memories in security applications and might be used as a reference for all futuristic devices where conventional memories are unavailable. Therefore, the dissertation statement of this work is:

> An effective and efficient framework by characterizing data obtained from varying latency parameters or manipulating analog physical characteristics of emerging memories can strengthen hardware security for any low-powered and low-cost devices.

1.6 Organization of the Dissertation

The rest of this dissertation is organized as follows. Chapter 2 provides the necessary background on different emerging memory technologies such as MRAM, ReRAM, PCM, FRAM, etc. Chapter 3 presents the proposed information hiding and extracting algorithms, including the method for characterizing the changes that occurred in commercial ReRAM *write* time due to memory stressing, and exhibits the effectiveness of the proposed scheme and security perspective through obtained results. Then, Chapter 4 discusses the proposed watermark imprinting and extracting mechanism and exhibits obtained results. Next, Chapter 5 describes the proposed technique of generating true random numbers, including cell characterization and suitable bit-selection algorithm, and demonstrates obtained results to verify the quality and robustness of the proposed TRNG. Finally, Chapter 6 concludes this dissertation.

CHAPTER 2

EMERGING NON-VOLATILE MEMORIES

The main components of the conventional memory hierarchy are on-chip caches (i.e., SRAM), off-chip main memory (i.e., DRAM), and storage (i.e., solid-state drive (SSD) and/or hard disk drive (HDD). Continual scaling down in technology introduces enormous challenges such as substantial process variation, crucial transistors' sensitivity to different operating conditions, significant power consumption, etc., to the existing memory chips. Current mainstream volatile memories (i.e., SRAM and DRAM) suffer from scalability, density, memory persistency, performance, reliability, and leakage issues. On the other hand, existing non-volatile memory (NVM)(i.e., NAND Flash) suffers from performance, endurance, high write energy, and reliability problems. Due to these limitations, existing memories are incompetent in delivering ever-increasing demands of power-efficient, smaller, and high-performance multi-core and large-scale computing systems [MDB+02].

In recent years, academic and industry researchers have made a lot of effort in the research and development of these emerging NVM technologies (e.g., Magnetoresistive RAM (MRAM), Resistive RAM (RRAM), Phase Change Memory (PCM), and Ferroelectric RAM (FeRAM)) to turn them into promising alternatives to conventional memory technologies to integrate them into different levels of the memory hierarchy [LNG+10, ZZYZ09, WYGW12, MDH+01]. Due to their propitious scopes, such as zero leakage, high density, scalability, CMOS compatibility, and high endurance, industries such as Everspin, Fujitsu, Intel/Micron, and Cypress have already commercialized MRAM, ReRAM, PCM, and FRAM chips, respectively. Intel and Micron's 3D XPoint memory technology is a remarked example of the adoption of NVM as a cache for SSD and promises 1000× lower latency as well as higher endurance than NAND flash [Int]. In addition, emerging NVMs have been

investigated to a great extent to integrate into low-power applications, such as the Internet of Things (IoT), wearable devices (e.g., smartwatch, smart glasses), tablets, smartphones, automobiles, and medical devices (e.g., hearing aids). However, before reaching maturity, it is essential to understand their opportunities and drawbacks and leverage them efficiently to improve next-generation computing systems' security, performance, power, and reliability.

Although PCM and FRAM have been successfully commercialized in the recent past, the popularity of PCM and FRAM is diminishing due to the following limitations- (i) high-temperature sensitivity, (ii) low endurance, and (iii) poor scalability at lower technology node [LLW+12, FR21]. However, MRAM and ReRAM are inherently free from such limitations, and as a result, industry attention is shifted to MRAM and ReRAM. Thus, this dissertation mainly focuses on the potential usages of MRAM and ReRAM in security applications. This chapter briefly discusses the architecture and operating principle of MRAM and ReRAM cells.

2.1 MRAM

The core element of MRAM is the magnetic tunnel junction (MTJ) that uses the Savtchenko switching (Toggle MRAM) [ADS16] or spin-transfer torque (spin-transfer torque or STT MRAM) [ADS16, FR21] property to store both data states (high and low). The 1T-1MTJ MRAM bit cell comprises two ferromagnetic layers separated by a thin dielectric tunnel oxide (AlOx or MgO) layer (Fig. 2.1a). One layer's magnetic orientation is permanently fixed, commonly referred to as the reference (or fixed) magnetic layer (RML). In contrast, the other layer's magnetization can freely be oriented depending on the magnetic field, known as the free magnetic layer (FML). The FML is composed of NiFe synthetic antiferromagnet (SAF). The substantially higher magnetic anisotropy of RML compared to FML ascertains the

stable magnetization direction of FML during memory read/write operation. The resistance states determine the bit that will be stored in the memory array. When both the FML and RML are aligned in the same direction (a large current passed from SelectLine or SL to BitLine or BL through the barrier layer), the MTJ produces low electrical resistance. On the other hand, when their magnetic field orientation is opposite, the resistance becomes extremely large; hence, almost no current or weak current flows through the barrier layer. Therefore, the MTJ exhibits high electrical resistance.

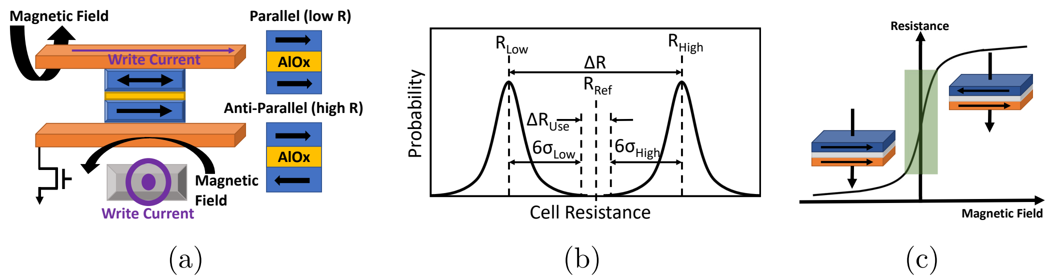


Figure 2.1: (a) Toggle MRAM cell structure with MTJ. (b) Schematic representation of Gaussian resistance (R_{Low} and R_{High} states) distribution of larger-sized MTJ array [ADS16]. (c) Principle of tunneling magnetoresistance (TMR) [TDK].

Writing a bit in the magnetic field-driven toggle MRAM or spin polarized current-driven STT MRAM array requires passing a high write current (I_w) for changing FML's magnetic orientation [ADS16]. The applied I_w to the write lines, placed on top and bottom of the MTJ devices (see Fig. 2.1a), creates an auxiliary magnetic field that changes the FML direction in the required position. Contrastingly, the RML's direction is strongly coupled with an anti-ferromagnet [ADS16]. During the write operation, the memory circuit performs a pre-read operation to determine the state of the target bit and execute a toggle pulse (if required) to change the state of the bit if the desired state is not the same as the targeted state. Consequently,

it reduces the overall power consumption and improves power efficiency. However, this increases the total write cycle time (including an additional read operation).

A small bias voltage (far below the device's breakdown voltage) is applied across the MRAM cell during the read cycle. Depending on parallel (R_{Low}) or anti-parallel (R_{High}) magnetic orientation, a current sensing circuitry that is attached with the MRAM cell experiences different amounts of current and latches the appropriate logic ('0' or '1') compared with the reference resistance (R_{Ref}) shown in Fig. 2.1b. The random resistance variation effect of a larger-sized MRAM array read circuitry is illustrated in Fig. 2.1b. The accepted bits are those whose statistical separation is greater than 5σ from the mean, where σ is the standard deviation. The accuracy of the read circuitry entirely depends on determining the actual resistance state in the tail region (useable resistance change, ΔR_{Use}) of the distribution. For robust operation, less noise-sensitive, and high-speed read operation with normal process variation, large ΔR_{Use}, and significantly more than 12σ separation are crucial [ADS16]. Note that the width of resistance distribution varies from cell to cell because of manufacturing process variations. Besides, the quality, size, and level of in-homogeneity of the MTJ tunnel barrier significantly impact larger relative bit-to-bit resistance variation [ADS16]. Therefore, a thicker tunnel barrier ($\sim 1nm$) is essential to maintain the resistance level of the MTJ in the kilo Ω range for minimizing the series resistance effect from the isolation transistor [ADS16], where Ω is the SI unit of resistance. However, Fig. 2.1c shows that the change in resistance due to the change in the induced magnetic field is steep in a certain region (the green region on Fig. 2.1c). In this region, a slight change in the induced magnetic field may cause a drastic change in resistance states and alter the decision of the read circuitry. Therefore, manufacturers define timing parameters for memory chips to support reliable write/read operation against a wide range of operating conditions.

2.2 ReRAM

Resistive switching phenomena in a dielectric material is the core mechanism of ReRAM to store logic states [Che20, MCYC15]. The capacitor-like ReRAM cell structure consists of two electrodes ($Electrode_{Top}$ and $Electrode_{Bottom}$) separated by a metal oxide resistive switch material (Fig. 2.2). Studies show that various metal oxide materials can be used to build the resistive switch layer, such as Al_2O_3, NiO, SiO_2, Ta_2O_5, ZrO_2, TiO_2, HfO_2, and Nb_2O_5 [Che20, MCYC15]. However, depending on the interfacial properties, different materials result in different device characteristics such as endurance, retention, and scalability [Che20, MCYC15]. Whenever a voltage is applied to the $Electrode_{Top}$, the metal oxide breakdown process is initiated and produces oxygen vacancies in the oxide layer. Consequently, these oxygen vacancies form a conductive filament between two electrodes and produce the low resistance state (LRS or logic '0' state). A voltage with opposite polarity is applied across the metal oxide to eliminate the conductive filament, representing the high resistance state (HRS or logic '1' state) of the ReRAM cell. The ratio of HRS's resistance to LRS's ($\frac{R_{HRS}}{R_{LRS}}$) is required to be large enough to ensure robust *read/write* operation [MCYC15]. The switching operations from HRS (LRS) to LRS (HRS) is known as *set* (*reset*) operation, and the time required for switching is known as the *set* (*reset*) time. In summary, the ReRAM *read/write* operation is performed as follows-

- The *write* operation ensures appropriate voltage magnitude and polarity across the ReRAM cell; as a result, the ReRAM cell obtains the appropriate resistance state (LRS for logic '0' and HRS for logic '1').
- During the *read* operation, a small voltage is applied across the ReRAM cell, and the measured resistance (by sensing current) determines the stored logic.

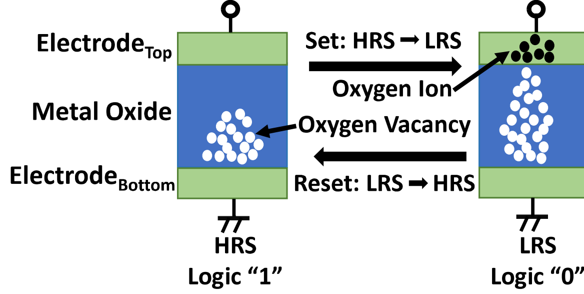


Figure 2.2: ReRAM cell structure with two logic states [MCYC15].

Each switching operation (i.e., changing state from $LRS \rightarrow HRS$ or $HRS \rightarrow LRS$) on ReRAM gradually decreases the resistance of HRS, wearing-out the device [ATO+15]. Each write operation gradually induces crystal defect (also known as a dielectric breakdown[LHJ+19]) and introduces trapped charges. Those trapped charges act like charge carriers, increasing the conductivity of metal oxide and reducing the HRS's resistance (R_{HRS}). Hence, fresh memory cells possess distinctly different analog properties from the stressed cells (i.e., cells that undergo repeated switching operations). For example, the reduction of resistance of HRS due to the wear-out process degrades the resistance ratio ($\frac{R_{HRS}}{R_{LRS}}$) [ATO+15, MCYC15]. To maintain the desired resistance ratio, *set* and *reset* times must be increased for stressed memory cells[1].

[1]The ReRAM internal control circuit maintains appropriate *set/reset* time by initiating the write-verify-write operation sequence [JAS+19]. In this mechanism, the write operation continue to write data on a memory cells till able to verify the data by reading it back.

CHAPTER 3

HIDING INFORMATION IN RERAM

This chapter introduces a novel, low-cost technique for hiding data in commercially available ReRAM chips. The data is kept hidden in ReRAM cells by manipulating its analog physical properties through switching (*set/reset*) operations. This hidden data, later, is retrieved by sensing the changes in cells' physical properties (i.e., *set/reset* time of the memory cells). The proposed system-level hiding technique does not affect the normal memory operations and does not require any hardware modifications. Furthermore, the proposed hiding approach is robust against temperature variations and the aging of the devices through normal read/write operation. Our proposed data hiding technique is acceptably fast, and the hidden message is unrecoverable without the knowledge of the secret key, which is used to enhance the security of hidden information.

The advantage of embedding secret information in a concealed manner in ReRAM cells by leveraging its analog characteristics is the confidential information remains invisible from the normal memory content viewpoint. Our proposed technique can be easily implemented on commercial off-the-shelf (COTS) ReRAM chips enabling secure and covert data storage. The advantage of the information hiding technique is — (i) the hidden data can not be retrieved due to watermarking if the storage device is lost or stolen [FBTR22], and (ii) the attacker can not read or copy the ciphertext, ensuring an additional protection layer over a typical encryption engine.

Technically, ReRAM is analogous to a two-terminal passive variable resistor where two resistance states, high resistance state (HRS) and low resistance state (LRS), represent the binary data values. Our technique embeds the hidden information by repeatedly stressing the memory cells by writing '1' and '0' alternatively. Repeated stressing through switching operation ('1' $\rightarrow$ '0' or '0' $\rightarrow$ '1') gradually

decreases the HRS resistance, degrading the memory performance and eventually causing endurance failure [ATO+15, MCYC15]. Our experiment indicates that repeatedly stressing the ReRAM cell increases its *write* time (for both logic '0' and '1'). To this extent, we propose a technique of encoding logic '0' and '1' by representing the fresh and stressed memory cells, respectively. Later, we retrieve the encoded sequence by observing the *write (set/reset)* time of corresponding memory cells. Our proposed technique is irreversible as the impact of cell stressing is immutable; hence, the hidden message cannot be tampered. Additionally, our proposed technique does not require hardware modification and can be directly deployed into available commercial products. Moreover, the embedded message is robust against temperature variation as ReRAM is inherently insensitive to temperature [GCR+13]. Furthermore, our proposed method can be evaluated using standard ReRAM *read/write* operation and only costs ~3% of the total endurance of ReRAM cells. The information concealing technique can enable several interesting applications, such as watermarking, secure and covert data storage, data integrity, and covert communication. The key contributions of this chapter are as follows.

- We present a new idea of hiding information in ReRAM by storing logic '0' bit in fresh ReRAM cells and logic '1' in stressed ReRAM cells. We experimentally show that the hidden data can be retrieved by observing ReRAM *write* time.
- We demonstrate the robustness, performance, retention characteristics, normal memory usage tolerance, and security analysis of our proposed technique in multiple COTS ReRAM chips.

The rest of the chapter is organized as follows. Section 3.1 presents the motivation of our work. Section 3.2 presents the proposed information hiding and extracting algorithms, including the method for characterization of changes in ReRAM *write* time caused by stress. Section 3.3 explains the experimental setup and ex-

hibits the effectiveness and security perspective through obtained results. Finally, Section 3.4 concludes the chapter.

3.1 Motivation

This work correlates two research areas — steganography with hardware security. Due to the rapid evolution in digital multimedia and information technology, hiding information within digital content, e.g., documents, videos, audios, images, text, etc., is gaining significant attention to protect content and intellectual property [CCCM10, WYX$^+$13, PAK99, PH03]. Prior works on typical digital steganography techniques either use unused meta-data fields or exploit noise in the digital content where the hidden information is usually tied to the digital file itself or the file's content. In [SDB11], firmware defines the hard disk drive's physical locations (sectors), which contains the hidden information, as unusable; hence, the operating system (OS) can not access those sectors, making the recovery process difficult and complicated. In contrast, our proposed scheme depends on intentionally applied stressing to conceal information in inherent analog physical characteristic variations of ReRAM (i.e., *write* time), decoupling the concealed information from the digital memory content and coupling it to physical objects. Retrieving hiding information from physical properties requires detailed and time-consuming measurement and analysis, making detection, copy, and erasure difficult for attackers. Therefore, the key benefits of our proposed technique over digital steganography are as follows.

- Our proposed covert technique has no impact on normal memory usage or memory content. The hidden information is entirely uncorrelated with the memory content. Therefore, an attacker can not reveal the hidden information by inspecting the memory content or performing memory operations. Our

experimental results reveal that the hidden information remains intact even after thousands of normal memory operations.

- Unlike digital steganography, hidden information can not be copied or stored for future analysis as our technique manipulates analog physical characteristics. For example, suppose an attacker gains access to the ReRAM without knowledge of the location of the hidden bit. In that case, it will not be possible to reveal confidential information by only measuring the *set/reset* time of the memory cells. Performing a brute force attack is not feasible as it requires a vast amount of time.

Besides, other steganographic methods modify the physical layer protocol to hide data in the noise of optical and wireless transmission instead of encoding in digital objects [PFKW09, SM16, MLP08]. These techniques require either special tools or modifications to existing protocols. In contrast, our proposed technique is analogous to physical steganography, where physical objects hide confidential information [JJ98, WYX$^+$13]. The key benefit of our proposed cost-effective ReRAM-based technique is — the embedding/retrieval of the concealed information can be performed without any circuit/hardware modification or subject-matter experts [Bor21]. Furthermore, our easily applicable scheme can be implemented using a software program in a low-level memory interface and works with standard digital interfaces with COTS memory chips.

3.2 Proposed Hiding Technique

This section describes the concealing and retrieval techniques of our information hiding scheme, along with the cell characterization performed to understand the analog physical characteristics of ReRAM cells.

3.2.1 Cell Characterization

Repeated switching operations (alternatively writing 0's and 1's) change the physical properties of ReRAM; therefore, the *set/reset* timing of stressed cells deviates from the fresh cells. The degree of deviation depends on the number of switching operations performed on stressed cells. Our proposed technique imprints logic '1' with stressed cells and '0' with fresh cells. Later, we retrieve the data by separating the fresh and stressed cells based on their switching time. However, ReRAM stressing reduces cell endurance. Therefore, we want to keep the stress level as little as possible and simultaneously ensure that fresh and stressed cells are reliably separable with *set/reset* time.

To this extent, we propose Algorithm 1 to understand the ReRAM cell characteristics and the impact of switching operation on *set/reset* timing. This algorithm allows us to determine the minimum number of switching operations required to reliably separate the stressed cell from the fresh cell. It also builds a relationship between ReRAM switching time and corresponding stressing level. The sequence of operations for this algorithm is as follows. We initiate our algorithm by writing all '1' data patterns to selected memory addresses (line 2 through line 4 of Algorithm 1). Then, all '0' and all '1' data patterns are written alternatively to those addresses (line 5 through line 18 of Algorithm 1). The switching times are captured and stored as *set/reset* times accordingly. We repeat the switching operation until the target memory cells are fully worn-out (i.e., no longer able to store data reliably). We observe that both the *set* and *reset* times increase due to the repeated switching operation, and after a certain number of switching operations, the stressed cells completely become separable from fresh cells. The degradation of set/reset time is linearly proportional to the number of trap charges induced by the dielectric breakdown process, which is also linearly dependent on the electrical stress[LHJ$^+$19] (i.e.,

Algorithm 1: Pseudo-code for characterizing memory cells using repeated switching operation.

```
Data: 𝒩_ℳ: Max rewrite operations (data endurance)
      𝒜_𝒮: Set of stressed memory addresses
      w_L: Word Length
      𝒟: (1 × w_L) matrix containing data intended to write each memory cells
      t: Timer
Result: 𝒮_𝒯: Set time of each memory address belongs to 𝒜_𝒮
        ℛ_𝒯: Reset time of each memory address belongs to 𝒜_𝒮
   // Initialization
 1 𝒮_𝒯 = {}; ℛ_𝒯 = {}; 𝒟 = Ones(1 × w_L);
 2 foreach a ∈ 𝒜_𝒮 do
 3 |   write(a, 𝒟);
 4 end
   // Stressing memory cells
 5 for i = 0 to 𝒩_ℳ do
 6 |   foreach a ∈ 𝒜_𝒮 do
 7 |   |   𝒟 = Zeros(1 × w_L);
 8 |   |   tic = t;
 9 |   |   write(a, 𝒟); // Set operation
10 |   |   toc = t − tic;
11 |   |   𝒮_𝒯 = 𝒮_𝒯 ∪ {toc}; // Accumulating set time
12 |   |   𝒟 = Ones(1 × w_L);
13 |   |   tic = t;
14 |   |   write(a, 𝒟); // Reset operation
15 |   |   toc = t − tic;
16 |   |   ℛ_𝒯 = ℛ_𝒯 ∪ {toc}; // Accumulating reset time
17 |   end
18 end
```

the number of switching operations). Therefore, the set/reset time of the ReRAM cell is linearly dependent on the stress count applied to the memory cell. In this work, we use this property to distinguish between the fresh and stressed ReRAM cells. Note that, according to our observation, the relation between switching characteristics (i.e., *set/reset* time vs. stress count[1]) is almost uniform for all memory

[1]One 'stress' means a pair of *set-reset* operation.

chips sharing the same part-number. Therefore, it should be sufficient to sample a small set of memory chips from each part-number and perform cell characterization over those chips.

3.2.2 Information Hiding Technique

Our proposed technique conceals information in the *write* (*set/reset*) time of ReRAM bit cells. The *set* and *reset* time increase monotonically with stress levels, making it possible to retain a hidden message and retrieve it through a proper threshold value that separates the stressed and fresh memory cells [FBTR22]. The authorized entity performs the proposed information concealing technique in the memory to encode secret information in ReRAM. In the proposed technique, we choose a set of addresses for the secret message; the number of addresses depends on the length of the secret message. Initially, all memory cells possess perfect or near-perfect analog properties since they are fresh. To encode a secret message, (i) initially, logic '1' is written to those chosen addresses (line 2 through line 4 of Algorithm 2), and (ii) repeated switching (*set* and *reset*) operations are performed (line 5 through line 14 of Algorithm 2) to only those ReRAM addresses, which are supposed to hold the logic '1' of the secret message. The switching operations are repeated until sufficient differences are developed in the *set/reset* time between fresh and stressed memory cells. Each switching operation gradually degrades the resistance of HRS, which are permanent; thus cannot be reversed. However, the number of repeated switching cycles, $\mathcal{N}$, used to encode the secret message is determined empirically through cell characterization for given memory chips to ensure proper separation without causing excessive stress [FBTR22]. From an encoding perspective, minimizing $\mathcal{N}$ is desirable because the encoding time of the secret message is directly proportional

to the number of switching cycles. However, higher $\mathcal{N}$ enhances the accuracy by distinguishing fresh and stressed memory cells more perfectly even after thousands of memory operations. The silicon results show that a few thousand rewrite operations are sufficient to hide information securely.

3.2.3 Hidden Information Retrieval Technique

In order to retrieve concealed information, the physical properties of memory cells are extracted (in our case, *set/reset* times) to distinguish between fresh and stressed memory cells. Line 15 to 26 of Algorithm 2 outlines the required steps for extracting the *set* and *reset* times from the addresses containing concealed information. We observe that both *set* and *reset* time change with stress counts, and both can be used to hide the secret message. In practice, we observe an apparent gap between the average write times of the fresh and stressed memory cell clusters with sufficient switching operations. As a result, it is straightforward to define the threshold value of *set/reset* time after encoding the secret message, which can be used to differentiate between fresh and stressed memory cell clusters. It is worth mentioning that *set/reset* characteristics of ReRAM cells appear to be uniform across all ReRAM chips that we have tested.

3.2.4 Encoding/Decoding Secret Information

The flowchart in Fig. 3.1 shows the steps of the information hiding process, i.e., encoding and decoding of information in ReRAM chronologically. The information hiding operation steps are pretty straightforward. We hide the secret message in analog physical characteristics of ReRAM by repeatedly stressing the memory cells. To this end, first, we characterize a few memory cells to understand the analog phys-

Algorithm 2: Pseudo-code for encoding and decoding secret message.

Data: $\mathcal{N}$: Number of stress count (i.e. *set-reset* pairs)
$\mathcal{A}_{\mathcal{M}}$: Set of memory addresses containing secret message.
w_L: Word Length
$\mathcal{S}_{msg}$: Secret message
$\mathcal{D}$: $(1 \times w_L)$ matrix containing data intended to write each memory cells
t: Timer

Result: $\mathcal{S}_{\mathcal{T}}$: *Set* time of each memory address belongs to $\mathcal{A}_{\mathcal{M}}$
$\mathcal{R}_{\mathcal{T}}$: *Reset* time of each memory address belongs to $\mathcal{A}_{\mathcal{M}}$

```
   // Initialization
1  S_T = {}; R_T = {}; D = Ones(1 × w_L);
2  foreach a ∈ A_M do
3  |  write(a, D);
4  end
   // Encoding secret message
5  for i = 0 to N do
6  |  foreach a ∈ A_M do
7  |  |  if S_msg[Bit]==1 then
8  |  |  |  D = Zeros(1 × wL);
9  |  |  |  write(a, D);
10 |  |  |  D = Ones(1 × wL);
11 |  |  |  write(a, D);
12 |  |  end
13 |  end
14 end
   // Decoding secret message
15 foreach a ∈ A_M do
16 |  D = Zeros(1 × w_L);
17 |  tic = t;
18 |  write(a, D); // Set operation
19 |  toc = t − tic; // Accumulating Set time
20 |  S_T = S_T ∪ {toc};
21 |  D = Ones(1 × w_L);
22 |  tic = t;
23 |  write(a, D); // Reset operation
24 |  toc = t − tic; // Accumulating Reset time
25 |  R_T = R_T ∪ {toc};
26 end
```

ical characteristics of ReRAM cells at different stressing levels up to the maximum endurance [FBTR22]. Later, the authorized personnel performs the retrieval operation to extract the hidden message bits from the analog physical characteristics through standard digital interfaces when required. The initial address of the stored information, replica size, and the shift sequence of each hidden bit are used as a hiding key during both hiding and recovery operations. Adding the error-correcting code (ECC)[2] to the secret message can improve the information recovery process by correcting obtained bit errors.

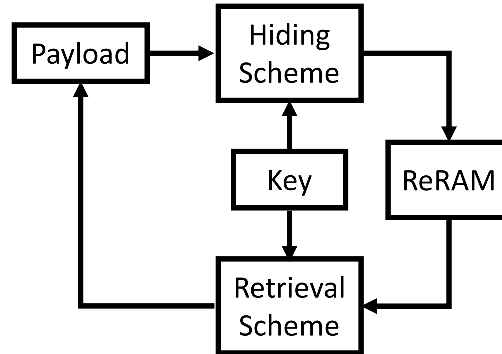


Figure 3.1: Operation steps used to hide information.

We assume that an attacker gets temporary access to the memory that contains hidden information so that they can check, inspect and manipulate the hidden information through the normal memory operations using a standard digital interface. We also assume that the attacker is aware of the information hiding technique so that they can also examine the analog physical characteristics through the standard digital interface. Therefore, our target is to develop the hiding scheme in such a way that it takes a sufficiently long time and effort to detect the existence and retrieve or remove the hidden information.

[2]Detail ECC algorithm is out of this chapter's scope.

3.3 Performance Evaluation

In this section, we evaluate our proposed technique through experiments on COTS ReRAM chips. In addition to validating the correct operation of the information hiding and retrieval scheme, we also analyze the robustness, performance, retention characteristics, recovery without the hiding key, and normal memory usage tolerance.

3.3.1 Evaluation Setup

The analysis is performed over five *MB85AS8MT* (40nm technology node) 8-bit serial peripheral interfaced (SPI) $8Mb$ memory chips manufactured by Fujitsu Semiconductor Limited. Table 3.1 captures the key features of the memory chips. We have used our own custom-designed memory controller implemented on *Teensy 4.1* microcontroller development board to issue commands and receive data from the memory chip. These multiple chip samples are used to verify the proposed technique and determine its feasibility. The *MB85AS8MT* ReRAM chips are byte-addressable. Therefore, a single byte is the smallest unit for which we can measure *set/reset* time. As a result, we need at least a one-byte storage area in the ReRAM to encode a single bit of data. However, the measured *set/reset* time might vary due to the external and internal noise, mostly due to the delay uncertainty between the memory controller and the memory. Therefore, for simplicity, we encode single-bit data into 256 consecutive addresses of the ReRAM to suppress the impact of noise (i.e., single bit is replicated over 256 addresses). Section 3.3.3 discusses the impact of using different replica sizes. Moreover, to make the detection scheme more complex, it is also possible to encode data into non-consecutive addresses (discussed in section 3.3.9).

we have measured the *set/reset* time for each address and computed the average for the evaluation. From now on to the rest of this chapter and chapter 4, we denote the average *set/reset* time over 256 addresses as $t_{Set,256}$, and $t_{Reset,256}$, respectively. Note that the *write buffer* size of our tested ReRAMs is also 256, which enables us to stress 256 addresses with a single *write* command, reducing overall stressing time. Although the figures presented in this section are based on a single ReRAM chip (randomly chosen from five test chips), the observation is valid for all test chips. Additionally, Tab. 3.2 summarizes the post-hiding stress tolerance results from all five test chips.

The following steps are performed to verify the feasibility of the proposed information hiding technique. We have embedded an arbitrarily chosen 32-bit random data (0xECE3038B[3]) into $(256 \times 32) = 8192$ memory addresses, varying the number of switching cycles, $\mathcal{N}$, up to $45K$ times to demonstrate the information hiding (discussed in section 3.2.2) and retrieval (discussed in section 3.2.3) technique experimentally. To study whether the proposed technique can reliably hide and recover bits in the *write* (*set/reset*) time characteristics, we use the separation between logic '0' and '1' as the evaluation metric.

Table 3.1: ReRAM Chip Specifications [Fuj19].

Parameter	Value
Capacity	8, 12 Mbits
Supply Voltage	1.6 - 3.6 V
Operating Frequency	10 MHz
Data Endurance	5×10^5 *times*/4 *bytes*
Data Retention	10 years
AC Standby/Sleep Current	65 μA/6 μA
AC Active Current (Read/Write)	0.15 mA/1.5 mA

[3] Also verified for other random data.

3.3.2 Cell Characterization

The switching characteristics (*set/reset* time vs. the stress counts) of the ReRAM chips at $25°C$ are shown in Fig. 3.2. These figures represent the maximum, minimum, and average of $t_{Set,256}$ (Fig. 3.2a) and $t_{Reset,256}$ (Fig. 3.2b) as a function of different stress levels (up to maximum possible rewrite operations[4]) over the 2K random address-space. Fig. 3.2 demonstrates that both the $t_{Set,256}$ and $t_{Reset,256}$ increase monotonically with stress levels, making it possible to distinguish between stressed and fresh memory cells. For example, the right-side zoomed plot of Fig. 3.2a, and Fig. 3.2b represents *set/reset* time up to $50K$ stress count, which demonstrates that the minimum value of $t_{Set,256}$ and $t_{Reset,256}$ at stressed count $\sim 12K$ is larger than the maximum value of $t_{Set,256}$ and $t_{Reset,256}$ at fresh condition. Therefore, a proper threshold value of $t_{Set,256}$ or $t_{Reset,256}$ can reliably identify fresh and stressed cells with $\sim 12K$ *set/reset* operations. Although Fig. 3.2 is constructed with $2K$ memory addresses, a similar characteristic is valid for the whole address space.

3.3.3 Appropriate Replica Size Selection

Fig. 3.3 illustrates the impact of replica size (i.e., consecutive addresses used to encode single-bit data). This figure represents the distribution of $d(b_0, b_1)$ at different replica sizes, where $d(b_0, b_1)$ represents the distance between logic '0' bits (b_0) and logic '1' bits (b_1). Each dot in Fig. 3.3 represents $d(b_0^i, b_1^j)$ for each possible (b_0^i, b_1^j). A negative value of $d(b_0, b_1)$ implies the overlap between logic '0' and logic '1', which is not expected. The distance between logic '0' bits and logic '1' bits improves with respect to replica size (i.e., most convergence of (b_0^i, b_1^j) occurs at replica size 256 for

[4]Maximum rated endurance for *MB85AS8MT* ReRAM chip is $1M$ rewrite cycles (i.e., $500K$ *set-reset* pairs). However, we observe that most memory cells can endure more rewrite operations than the rated endurance. In our experiment, we stress memory cells with up to $1M$ *set-reset* pairs.

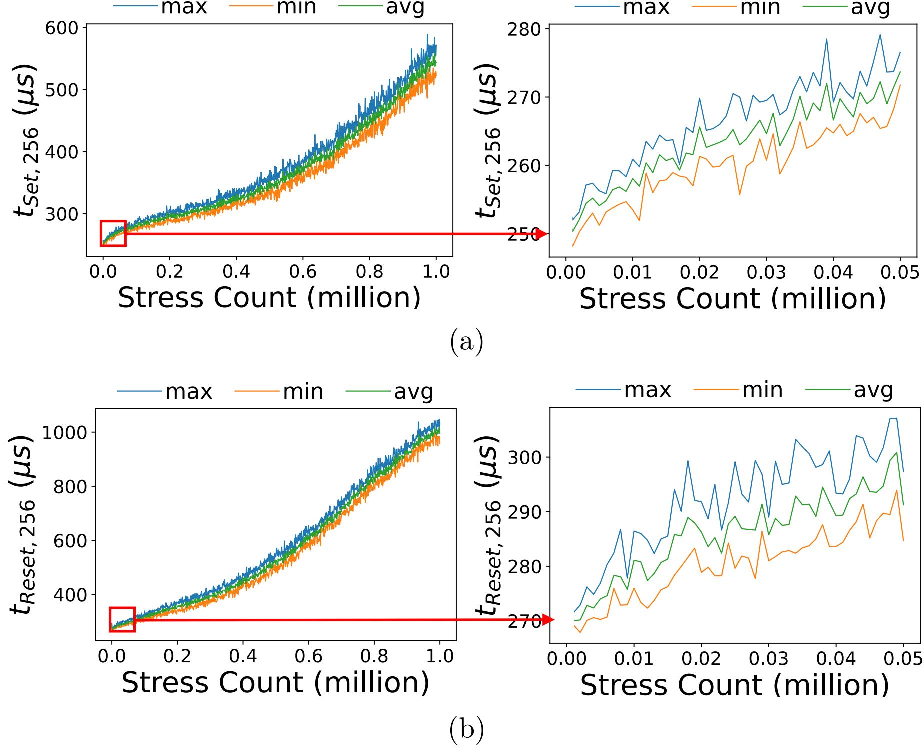


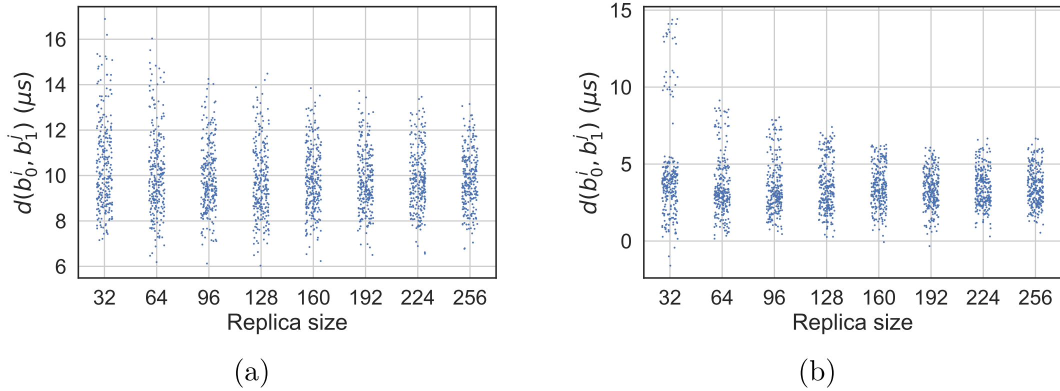
Figure 3.2: ReRAM cell characterization under stress- (a) $t_{Set,256}$ and (b) $t_{Reset,256}$.

both *set* and *reset* operations). Fig. 3.3a shows that, while using *set* time, the bit '0' and bit '1' are well separable (i.e., $(b_0^i, b_1^j) > 0$, for all (i, j)) with the smallest possible replica size of 32. On the other hand, *reset* time only distinguishes the bit '0' and '1' with the minimum replica size of 224 (Fig. 3.3b). So, in our proposed data hiding technique, we can use any value above 32 while using *set* time and any value above 224 while using *reset* time. The impact of noise in the *set* time is smaller than the *reset* time, which is expected due to the higher magnitude of the *set* time.

In addition to that, we take another consideration while selecting the replica size. All of our test ReRAM chips come with an internal data buffer that can hold data for up to 256 addresses. This buffer is an unmodifiable hardware component (a set of 8-bit registers). These ReRAM also comes with dedicated instruction, which enables us to perform the following operations at once —

- Send data from the memory controller to the memory chip for 256 addresses.
- Temporarily hold those 256 address data in the buffer.
- Write those 256 address data in parallel.

Therefore by choosing a replica size of 256 and taking advantage of this data buffer, we can simplify our implementation and significantly improve the imprinting speed.



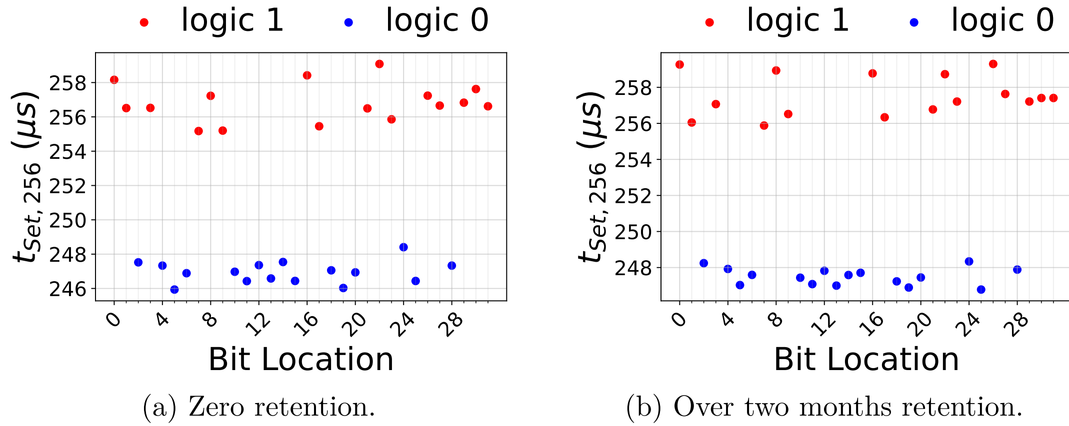
(a) (b)

Figure 3.3: Influence of replica size on the hidden information, using- (a) $t_{Set,[32,256]}$, and (b) $t_{Reset,[32,256]}$.

3.3.4 Retention Characteristics

The retention characteristics of the proposed hiding technique are also studied and shown in Fig. 3.4 and Fig. 3.5. Note that since each retrieval performs one *set-reset* operation, these retention characteristics include impacts from additional *set-reset* operations in addition to the time between information hiding and retrieval. The red and blue dots in Fig. 3.4 and Fig. 3.5 represent the imprinted logic 1's and 0's, respectively. Fig. 3.4a and Fig. 3.5a show that logic '1' and logic '0' are well separated at stress level $15K$, which is used to hide information considering both $t_{Set,256}$ and $t_{Reset,256}$, respectively. After over two months of retention, the logic '1'

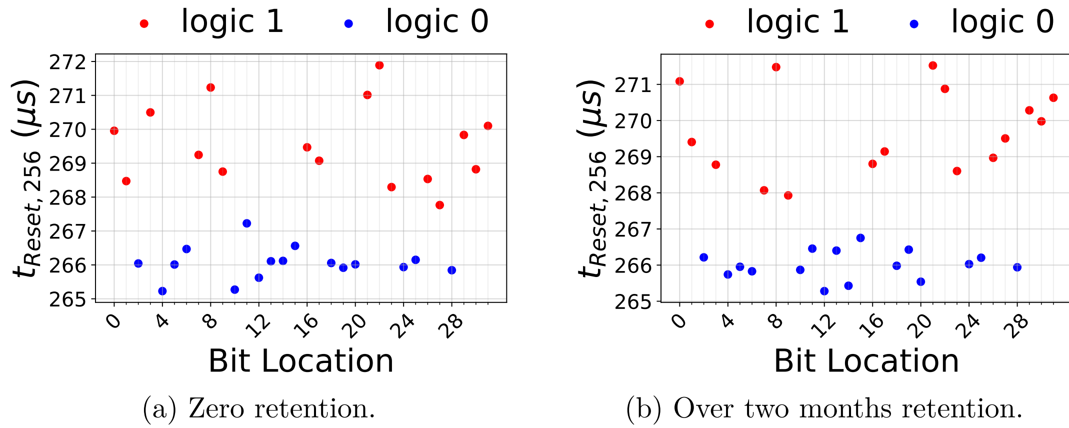
and logic '0' remain separated for both $t_{Set,256}$ (Fig. 3.4b) and $t_{Reset,256}$ (Fig. 3.5b). The results verify that the retention time has little or no impact over bit separation.



(a) Zero retention.

(b) Over two months retention.

Figure 3.4: Retention characteristics of the hidden information using $t_{Set,256}$.



(a) Zero retention.

(b) Over two months retention.

Figure 3.5: Retention characteristics of the hidden information using $t_{Reset,256}$.

3.3.5 Performance Analysis

In this subsection, we use the encoding and retrieval time as well as encoding cost as the metric for measuring performance.

Encoding Time

The proposed technique for information hiding relies on the repeated switching operation of ReRAM cells. Thus, the time required to encode the hidden information is directly proportional to the number of stress counts, $\mathcal{N}$. The estimated time to hide information is $\mathcal{T}_{imprint} = (\mathcal{N} \times \mathcal{B}_{Msg} \times \mathcal{T}_{switch_{pair}})$, where $\mathcal{T}_{switch_{pair}} = (\mathcal{T}_{set} + \mathcal{T}_{reset})$ represents stressing time (*set-reset* pair) for 256 addresses (switching resistance state with single *write* command), and $\mathcal{B}_{Msg}$ represents the number of bits that need to conceal. The chip used for our experimental evaluation has the following timing parameters: $\mathcal{T}_{switch_{pair}} = (5ms + 5ms) = 10ms$, and $\mathcal{B}_{Msg} = 32$. Thus, the baseline implementation requires $((5ms + 5ms) \times 32 \times 15k) = 4800s$ for $15K$ switching operations to conceal secret message. Therefore, the throughput for the encoding process is $\frac{32bits}{4800s} = 0.4bit/min$. The hiding throughput will also be higher if $\mathcal{N}$ is smaller. Besides, it is worth mentioning that the encoding time of our proposed technique heavily depends on the *write* speed of the ReRAM chips. Fortunately, in the past few years, the *write* speed of ReRAM chips significantly improved and will continue to improve in the future. For example, the *write* speed of *MB85AS8MT* ReRAM chips is improved $>3X$ over its previous generation *MB85AS4MT* ReRAM chips[5].

Retrieval Time

Unlike the encoding procedure, the extraction procedure is significantly fast. The estimated time to retrieve the hidden information can be calculated by—$\mathcal{T}_{retrieve} = (\mathcal{T}_{switch} \times \mathcal{B}_{Msg} \times \mathcal{N}_{rep})$, where $\mathcal{T}_{switch}$ is the average value of $t_{Set,256}$ or $t_{Reset,256}$, and $\mathcal{N}_{rep}$ represents the number of addresses used to encode single bits. After $15K$ stressing, the average value of $t_{Set,256}$ is $\sim 250\mu s$, and we used $\mathcal{N}_{rep} = 256$ in our im-

[5] *MB85AS8MT* and *MB85AS4MT* chips were launched in 2019 and 2016, respectively.

plementation. Therefore, the throughput for the retrieval is $\frac{\mathcal{B}_{Msg}}{\mathcal{T}_{retrieve}} = \frac{32bits}{250\mu s \times 32 \times 256} = 15.625bits/s$. The average value $t_{Set,256}$ includes program data transfers from the microcontroller to the host computer and microcontroller overhead. Besides, the retrieval throughput will be higher if the hiding technique uses a smaller replica size (i.e., the number of memory addresses to encode each hidden bit), as discussed in section 3.3.3.

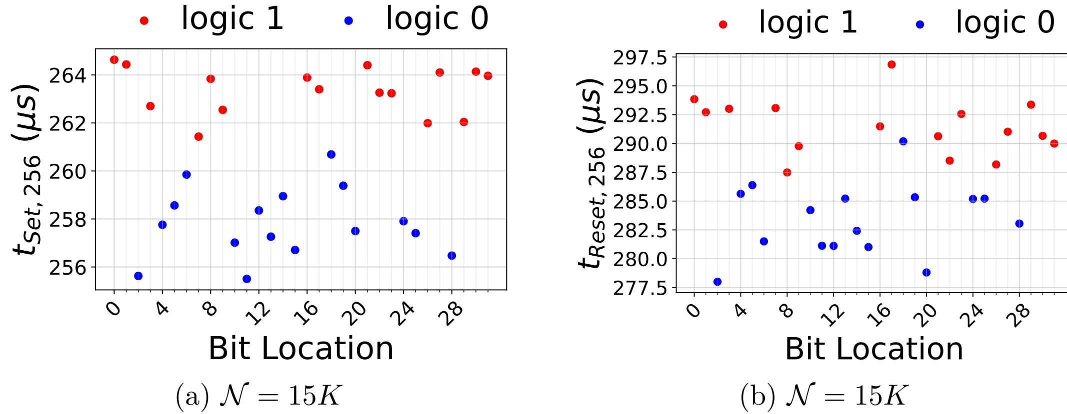
Encoding Cost

Our proposed technique requires a minimum of $15K$ *set-reset* operations (i.e., $30K$ rewrite cycles) to make a distinguishable separation between logic '0' and '1' of the concealed information (using both $t_{Set,256}$ and $t_{Reset,256}$). However, the rated endurance of ReRAM chips is $1M$. Therefore, our proposed technique costs only 3% of the rated endurance of encoded addresses.

3.3.6 Initial Stress Tolerance

The effectiveness of the proposed technique on moderately used memory chips is also examined. The influence of the initial stress count (i.e., the number of stress that occurred due to normal usage of memory before hiding information) is shown in Fig. 3.6. To simulate the normal usage of the storage device, we write random data patterns to the memory for the initial stressing. The occurrence of random data patterns appears according to Ref. [WDZ+16] to make more realistic stressing on the memory cells incurred from memory usage. For example, the separation between logic '0' and '1' at the initial stress level of $50K$ switching cycles shows the bit separation when bits are hidden using $15k$ *set-reset* operation after performing $50K$ normal memory usage operations. We observe that bit separation decreases, i.e., bit

error starts to occur (Fig. 3.6b) as the initial stress level increases. However, a higher initial stress count can be tolerated by increasing the stress level in the encoding process. Besides, the *set* operation can tolerate higher initial stress than the *reset* operation. Note that the error rate observed using $t_{Reset,256}$ is still manageable (3.125%) even after thousands of normal memory operations.



(a) $\mathcal{N} = 15K$ (b) $\mathcal{N} = 15K$

Figure 3.6: Influence of initial stress ($50K$) on the hidden information using- (a) $t_{Set,256}$ and (b) $t_{Reset,256}$.

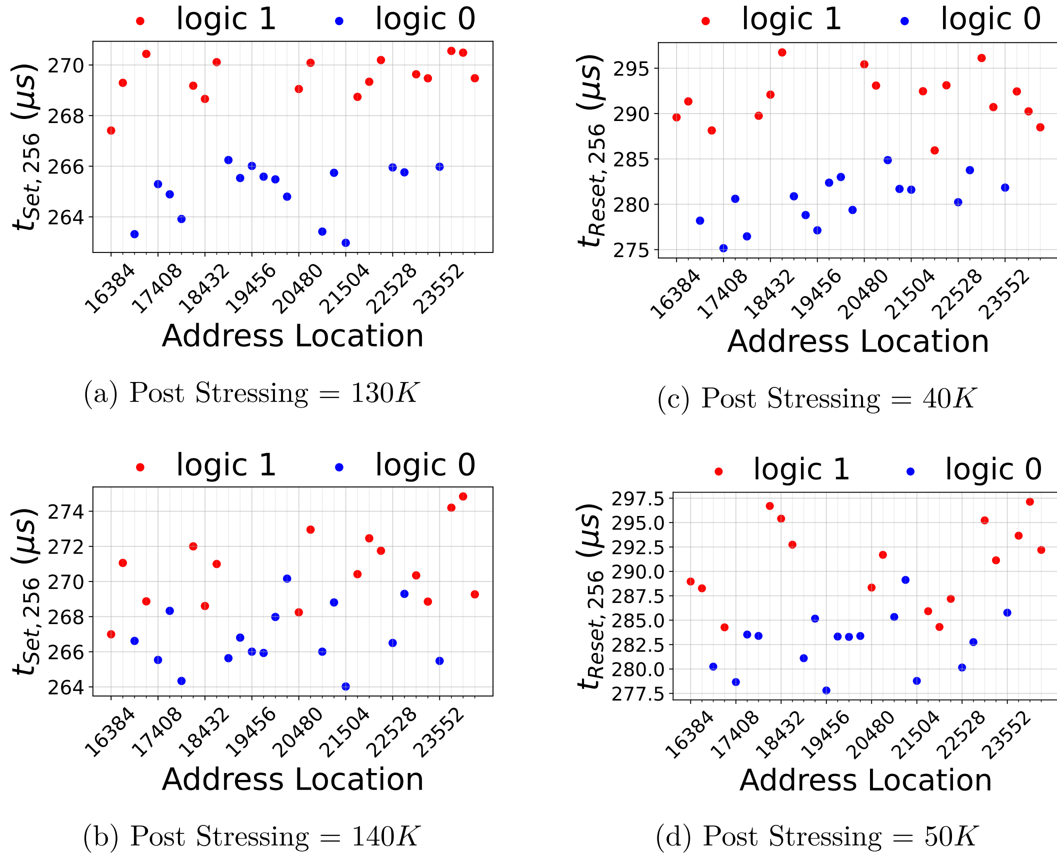
3.3.7 Post-Hiding Stress Tolerance

To test the post-hiding stress tolerance of our proposed technique, we deliberately stress the memory chip after hiding information on the chip. We write random data patterns to the memory to emulate the post-hiding stressing. The occurrence of random data patterns appears according to Ref. [WDZ+16] to make more realistic stressing on the memory cells incurred from memory usage. Fig. 3.7 and Fig. 3.8 illustrate the influence of post-hiding stressing (i.e., the number of stresses performed after hiding information).

Fig. 3.7 represents the post-hiding stress tolerance of the hidden data where a minimum ($\mathcal{N} = 15K$) stress count is used to hide the information[6]. We conceal

[6]Also verified for other stressing levels (up to $\mathcal{N} = 45K$) used to hide information.

the data in a confidential memory location using the key. The red and blue dot represents the hidden logic 1's and 0's, respectively. Fig. 3.7 shows that logic '1' and logic '0' remain separated up to 130K stress count (Fig. 3.7a). They become inseparable at 140K stress count (Fig. 3.7b). Similarly, with $t_{Reset,256}$, logic '1' and logic '0' remain separated up to 40K stress count (Fig. 3.7c) and become inseparable at 50K stress count (Fig. 3.7d).



(a) Post Stressing = 130K

(c) Post Stressing = 40K

(b) Post Stressing = 140K

(d) Post Stressing = 50K

Figure 3.7: Hidden data (hiding stress count, $\mathcal{N}$ = 15K) at different post-hiding stress level- (a)–(b) $t_{Set,256}$ at stress count 130K, and 140K; (c)–(d) $t_{Reset,256}$ at stress count 40K, and 50K.

In addition, Fig. 3.8 represents the distribution of $d(b_0, b_1)$ at different post-hiding stress levels, where $d(b_0, b_1)$ represents the distance between logic '0' bits (b_0) and logic '1' bits (b_1). Each dot in Fig. 3.8 represents $d(b_0^i, b_1^j)$ for each possible

(b_0^i, b_1^j). The distance should be positive for well-separated logic '0' and '1'. A larger value of $d(b_0^i, b_1^j)$ is more desirable as it provides better separation between logic '0' and logic '1' bits. However, if the maximum value of *set/reset* time of logic '0' bits is larger than the minimum value of *set/reset* time of logic '1' bits (similar to Fig. 3.7b), then logic '0' bits and logic '1' bits cannot be separated with 100% accuracy. In such a scenario, the $d(b_0^i, b_1^j)$ can be negative for a few pairs of (b_0^i, b_1^j). The figure demonstrates that the separation between logic '0' bits and logic '1' bits degrade with respect to post-stress count. However, the separation between logic '0' bits and logic '1' bits is quite reasonable, even after thousands of post-hiding stresses. For example, the logic '0' bits (b_0) and logic '1' bits (b_1) are clearly separable up to $110K$, $140K$, and $230K$, post-hiding stress levels using $t_{Set,256}$ with $\mathcal{N} = 15K$ (Fig. 3.8a), $30K$ (Fig. 3.8b), and $45K$ (Fig. 3.8c) hiding stress count used to conceal information, respectively (i.e., $\min\left(d(b_0^i, b_1^j)\right) > 0$). On the other hand, the logic '0' bits (b_0) and logic '1' bits (b_1) are clearly separable up to $40K$, $70K$, and $140K$, post-hiding stress levels using $t_{Reset,256}$ with $\mathcal{N} = 15K$ (Fig. 3.8d), $30K$ (Fig. 3.8e), and $45K$ (Fig. 3.8f) hiding stress count, respectively. However, note that if we tolerate one or two-bit errors, then our proposed scheme can survive a few more thousands of post-hiding stress levels.

Table 3.2 summarizes the post-hiding stress tolerance statistics for all test chips. The first two columns show the stress level used to hide information and the switching (*set* or *reset*) operation considered while performing post-hiding stressing. Finally, column three to seven represents the post-hiding stress levels that different chips can endure. We observe that more stress in the hiding process increases the write time difference between bits hiding '1's and '0's. If we use lower stress levels (even lower than $15K$) to encode information, the hidden information can survive fewer thousands of normal memory usage operations. Besides, the *set* operation can

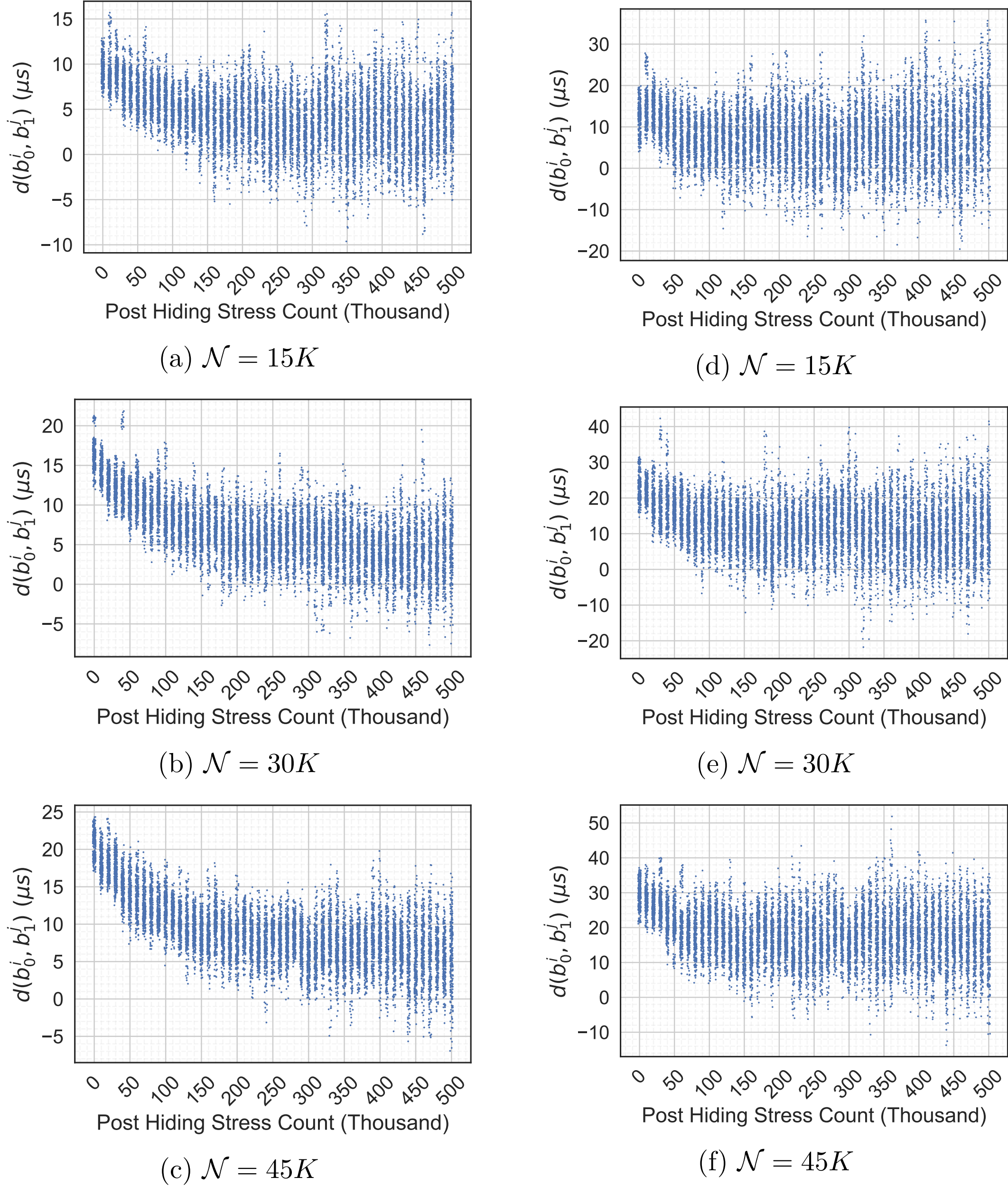


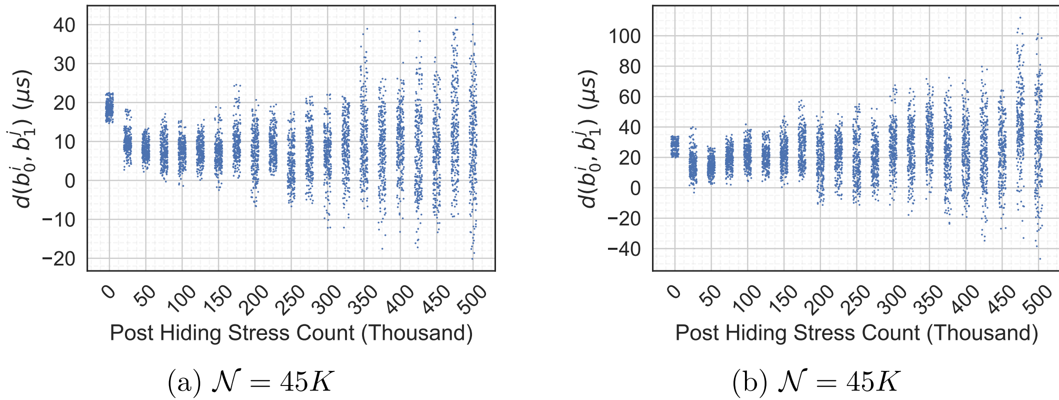
Figure 3.8: Post-hiding stress tolerance of concealed data at different hiding stress levels, using- (a)–(c) $t_{Set,256}$, and (d)–(f) $t_{Reset,256}$.

endure a higher post-hiding stress count than the *reset* operation. Therefore, depending on the application requirement, one can choose the stressing level to encode information to sustain the desired level of memory usage.

Table 3.2: Post-hiding stress tolerance statistics using all test chips.

Stress Count, $\mathcal{N}$	Op.	Chip1	Chip2	Chip3	Chip4	Chip5
15K	Set	110K	130K	100K	100K	130K
	Reset	60K	40K	50K	30K	40K
30K	Set	150K	150K	140K	150K	140K
	Reset	70K	100K	100K	90K	80K
45K	Set	180K	190K	190K	200K	230K
	Reset	180K	180K	200K	130K	140K

Furthermore, to apply maximum post-hiding stress, we write '1' → '0' → '1' to each memory cell after hiding information on the memory chip. The influence of maximum post-hiding stress on the separation between logic bits vs. the number of post-hiding stresses performed after hiding information is shown in Fig. 3.9. From the figure, we can see that the distance between logic '1' and '0' decreases as the post-hiding stress level increases. However, the distance between logic '1' and '0' of hidden information is quite reasonable for both $t_{Set,256}$ and $t_{Reset,256}$, even after thousands of post-hiding stress count. For example, with $45K$ switching operations used to hide information can endure $200K$ post-hiding stress levels for both $t_{Set,256}$ (Fig. 3.9a) and $t_{Reset,256}$ (Fig. 3.9b).



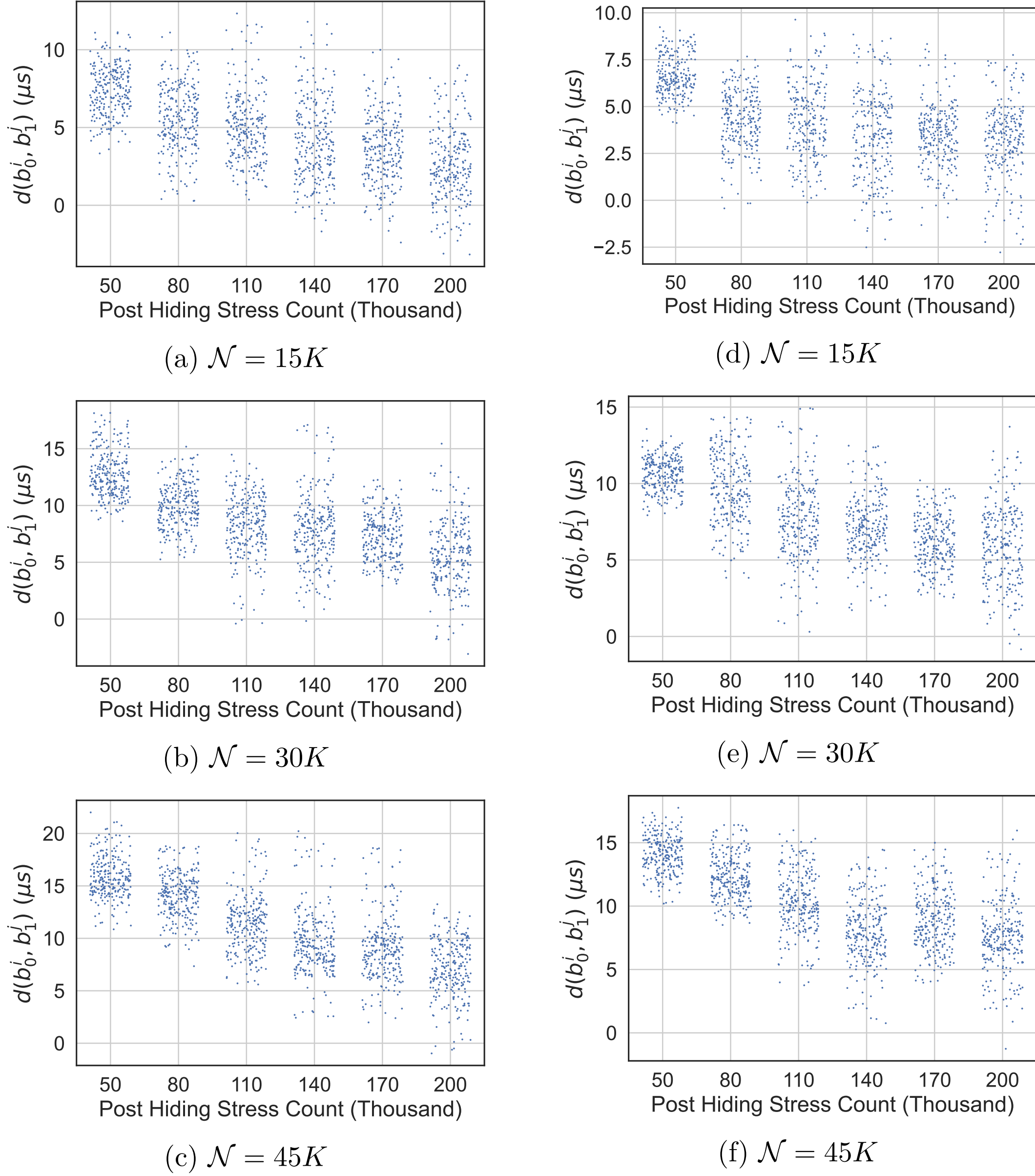
(a) $\mathcal{N} = 45K$ (b) $\mathcal{N} = 45K$

Figure 3.9: Post-hiding stress tolerance of concealed data considering maximum stressing using- (a) $t_{Set,256}$ and (b) $t_{Reset,256}$.

3.3.8 Robustness Analysis

The hidden message should be resilient to the variation of operating conditions, i.e., it will not be possible to modify or change the hidden information with localized heating or operating voltage. Inherently, all modern ICs are resilient to small variations in operating voltage as they are usually integrated with a voltage regulator. Voltage regulators are capable of retaining the operating voltage within a valid range of supply voltage. However, to verify the robustness of our encoding technique against the temperature with post-hiding stressing, first, we conceal information in a confidential address space with $15K$, $30K$, and $45K$ stress levels, respectively. Then we write random data patterns to the memory for $50K$ to $200K$ times with $30K$ intervals to examine the robustness of the proposed hiding scheme. The occurrence of random data patterns appears according to Ref. [WDZ+16] to make more realistic stressing on the memory cells incurred from memory usage. After every $30K$ memory operations, we isolated the memory chip from the system and baked it at $80°C$ for 1 day. Lastly, we have evaluated the $t_{Set,256}$ and $t_{Reset,256}$ while maintaining the chip temperature of $80°C$ for all three stress levels (in our case, $\mathcal{N} = 15K$, $30K$, and $45K$, respectively) used to conceal information.

Fig. 3.10 and Fig. 3.11 illustrate the influence of high-temperature baking. These figures represent the distribution of $d(b_0, b_1)$ at different post-hiding stress levels, where $d(b_0, b_1)$ represents the distance between logic '0' bits (b_0) and logic '1' bits (b_1). As discussed in section 3.3.7, a larger positive value of $d(b_0^i, b_1^j)$ is desirable to provide better separation between logic '0' and logic '1' bits. However, in Fig. 3.11a and Fig. 3.11d, the $d(b_0^i, b_1^j)$ is negative for a few pairs of (b_0^i, b_1^j); hence, logic '0' bits and logic '1' bits cannot be separated with 100% accuracy. Here, Figs. 3.10a, 3.10b, and 3.10c show $d(b_0^i, b_1^j)$ using $t_{Set,256}$ before performing high-temperature system-level operation. Similarly, Figs. 3.10d, 3.10e, and 3.10f show $d(b_0^i, b_1^j)$ using



(a) $\mathcal{N} = 15K$ (d) $\mathcal{N} = 15K$

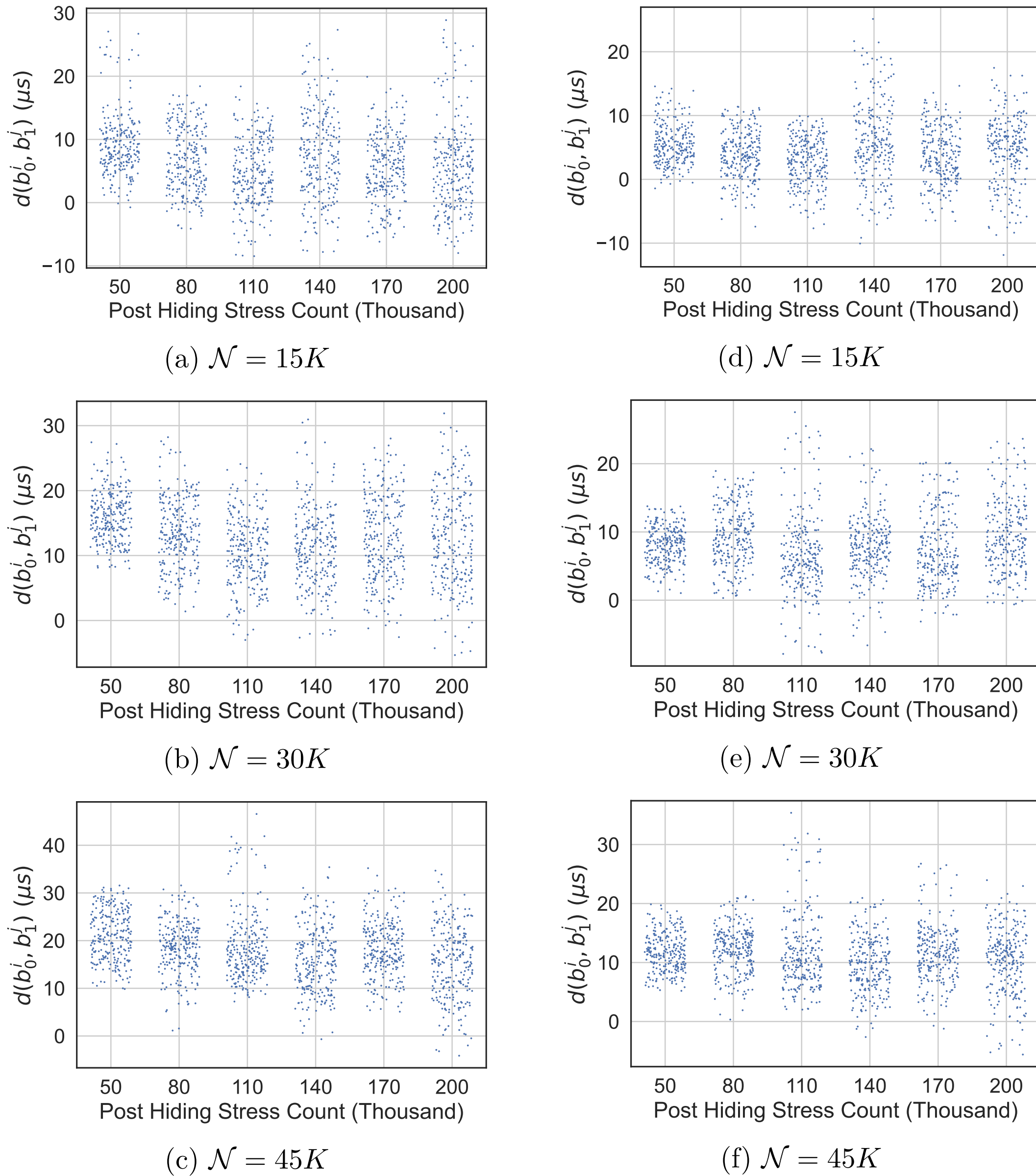
(b) $\mathcal{N} = 30K$ (e) $\mathcal{N} = 30K$

(c) $\mathcal{N} = 45K$ (f) $\mathcal{N} = 45K$

Figure 3.10: Robustness analysis (a)–(c) before (d)–(f) after high-temperature baking ($80°C$) with- $t_{Set,256}$.

$t_{Set,256}$ after performing high-temperature baking and high-temperature system-level operation. Similar observations are valid for $t_{Reset,256}$ (Fig. 3.11). We have observed that the hidden information is not significantly affected by temperature and remains well-separated after the high-temperature baking and high-temperature system-level

operation (considering both $t_{Set,256}$ and $t_{Reset,256}$). Such behavior of ReRAM is expected as the resistance ratio ($\frac{HRS}{LRS}$) is relatively temperature insensitive [GCR+13]. Note that the ReRAM chips used in our experiment are rated to operate up to 85°C.



(a) $\mathcal{N} = 15K$

(d) $\mathcal{N} = 15K$

(b) $\mathcal{N} = 30K$

(e) $\mathcal{N} = 30K$

(c) $\mathcal{N} = 45K$

(f) $\mathcal{N} = 45K$

Figure 3.11: Robustness analysis (a)–(c) before (d)–(f) after high-temperature baking (80°C) with- $t_{Reset,256}$.

3.3.9 Security Analysis

Data hiding in memory is analogous to steganography in digital information, i.e., adding new information on top of existing information without any trace (e.g., without requiring additional storage space). Like steganography, data hiding techniques can be used to make robust watermarks (can be recovered after using memory for a long period of time), second-layer protection over encrypted data, law enforcement, military communication, etc. Additionally, as hidden data is not visible through regular read/write memory operation; thus, it can be considered safe against ransomware attacks. In a ransom attack, the attacker introduces malware in the system, which encrypts all available information in storage memory. Such an attack cannot be prevented even by encrypting the storage memory. However, data hiding technique can be very useful in such scenarios. In the particular event of a ransom attack, even if the whole storage device is encrypted by the ransomware, the hidden data in ReRAM can still be recovered as data hiding uses different properties (in our case, *set/reset* time) from regular read/write operation (ReRAM cell resistance).

Therefore, in this section, we perform the security analysis of our proposed technique. The previous section showed how the *write* time is manipulated to store hidden information reliably. Contrarily, in this section, we discuss if an attacker gains access to the memory containing hidden information then whether they can detect hidden information or its existence.

Retrieval with Inaccurate Initial Address

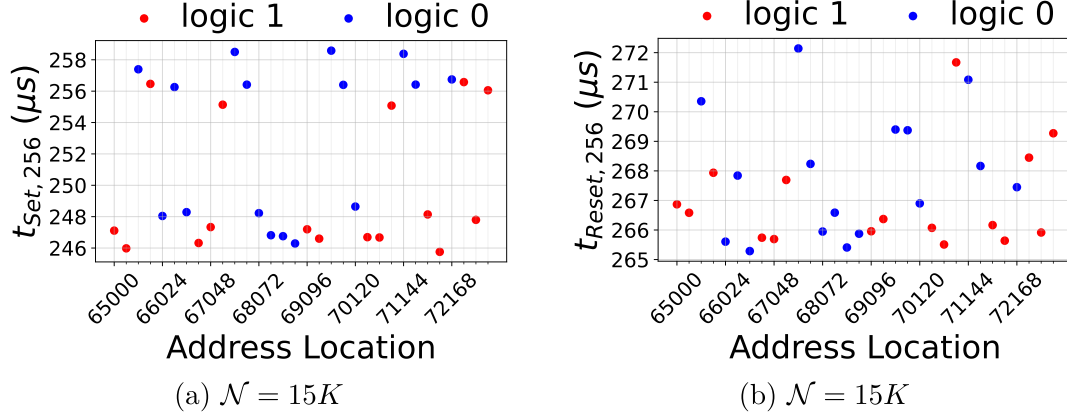
Encoding one bit of secret information in a group of addresses rather than each memory address improves security. If the attacker does not know the hiding key, he or she can not accurately identify the hidden information. Grouping addresses with an inaccurate key will contain both fresh and stressed memory cells; therefore, even

the correct threshold value cannot distinguish the fresh and stressed memory cells of the inaccurate group.

Fig. 3.12 illustrates the retrieved data with an incorrect initial address where the minimum ($\mathcal{N} = 15k$) stress level is used to hide information. The red and blue dot represents the imprinted logic 1's and 0's, respectively. In this experiment, we selected the initial address in three different ways. For example, we used $45K, 30K$, and $15K$ stressing to hide information starting with 32768, 49152, and 65536 memory addresses, respectively. Next, we choose incorrect initial addresses 28672, 32000, and 36864 for $45K$; 45056, 48500, and 53248 for $30K$; and 61440, 65000, and 69632 for $15K$ stress levels, respectively. The reason for choosing the initial addresses in such a way is that —

- Case 1: The incorrect group overleaps the maximum portion of the correct group, or
- Case 2: The incorrect initial address resides inside the correct group, or
- Case 3: The incorrect initial address is an integer multiple of replica size (in our case, replica size is 256).

Although Fig. 3.12 is constructed with the ($\mathcal{N} = 15K$) stress count, a similar characteristic is valid for a higher stress count (i.e., up to $45K$). Besides, Fig. 3.12 illustrates case 3 only; a similar characteristic is valid for the other two cases. The figure shows that there is no clear separation between the fresh and stressed memory cells. Therefore, the value of hidden bits can not be retrieved through thresholding. This result depicts that it is difficult for attackers to retrieve hidden data without the correct initial address because each group is likely to have both fresh and stressed memory cells.



(a) $\mathcal{N} = 15K$ (b) $\mathcal{N} = 15K$

Figure 3.12: Retrieved data with an incorrect initial address at $(15K)$ stressing used to hide information with- (a) $t_{Set,256}$ (b) $t_{Reset,256}$.

Hiding Data with Enhanced Security

In our proposed data hiding technique, we use *set/reset* time to imprint hidden data in ReRAM. However, measuring *set/reset* time is a noisy process, mostly due to the delay uncertainty between the memory controller and the memory. To avoid this situation, we replicate the hidden data to multiple copies and imprint all copies to ReRAM. However, if we replicate the data with the same address interval, the hidden data can easily be unveiled by averaging *set/reset* time over that address interval. In the previous sections, we hide 1st bit of hidden data in 256 consecutive addresses, then the 2nd bit in another 256 consecutive addresses, and so on. However, the security of our proposed data hiding scheme can be enhanced in many ways. For example, we can perform the hiding through replication and random rotation to strengthen security by making the hidden data untraceable. Fig. 3.13 demonstrates this method by using 8-bit data (payload). For simplicity, we assume that replicating the payload only 16 times is sufficient to recover it without any error. However, instead of saving each replica directly to the memory, we rotated (circular shift) each replica with a random displacement ($\mathcal{D}$). The displacement should be uniform within the possible range to maximize security (in this case, $7 \leq \mathcal{D} \leq 0$). Here, the $\mathcal{D}$ for

each replica can be considered as the key. While retrieving the hidden information, the key is required to reverse the rotation in order to find the appropriate set of *set/reset* time for each information bit. Therefore the security of the information can be guaranteed as long as the key is uncompromised. If we circularly shift each copy of the replicated data with $\mathcal{D}$ (as shown in Fig. 3.13), then a simple mean of *set/reset* time would not reveal any data. In addition to that, after hiding useful data in the specific address space, one can imprint random unmeaningful data over the unused address space. This will equalize the *set/reset* time over the whole address space and make it harder to notice the change in the *set/reset* time over a specific address space. Note that any uniform pseudo-random number generator (such as a linear congruential generator or LCG) can be used to generate random displacements for circular shifting. Therefore, instead of using the whole set of random displacements as the key, one can simply use the seed of the pseudo-random number generator as the key (rotational displacements need to be derivated from this key).

The authorized entity is solely responsible for generating and distributing the keys. The key distribution protocol of data hiding completely depends on the applications. If the main purpose of the data hiding is to keep the data untraceable from regular read/write operations, then we do not need to keep the key secret, and the key can be any known standardized value. In such applications, the key will serve the purpose of shuffling each copy of hidden data and keeping the set/reset time uniform over the whole address space. However, if the main purpose of this data hiding is to keep the data completely secret, then we need to use a standard key distribution protocol, such as the Diffie-Hellman key exchange protocol.

Fig. 3.14 is constructed with the correct key using a $\mathcal{N} = 15K$ stress level to hide information (with 256 replicas). A similar characteristic is valid for a higher stress count (i.e., up to $45K$). The figure shows that there is a clear separation between

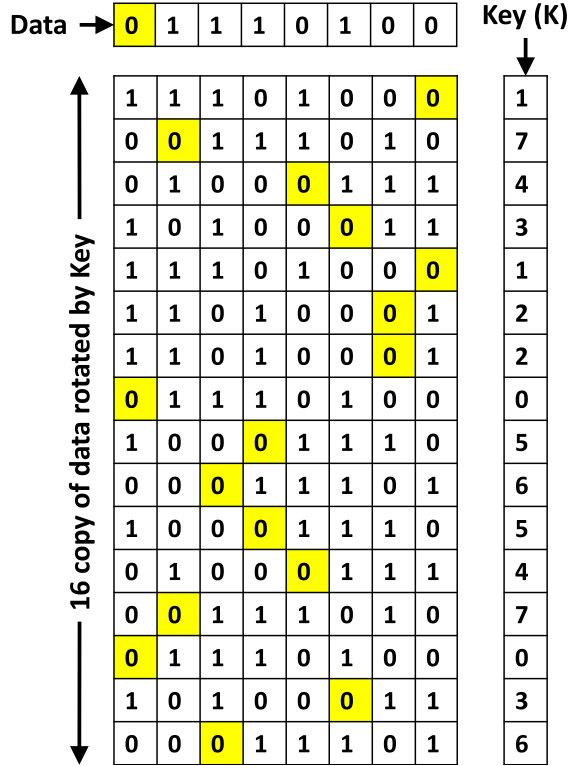
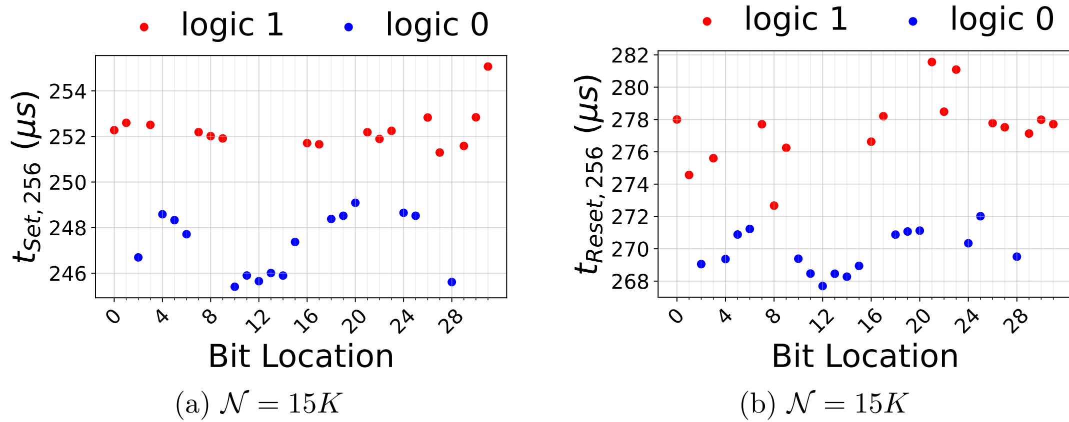


Figure 3.13: Data hiding with enhanced security: replication + left circular rotation. The start bit is marked with yellow to track the rotation.

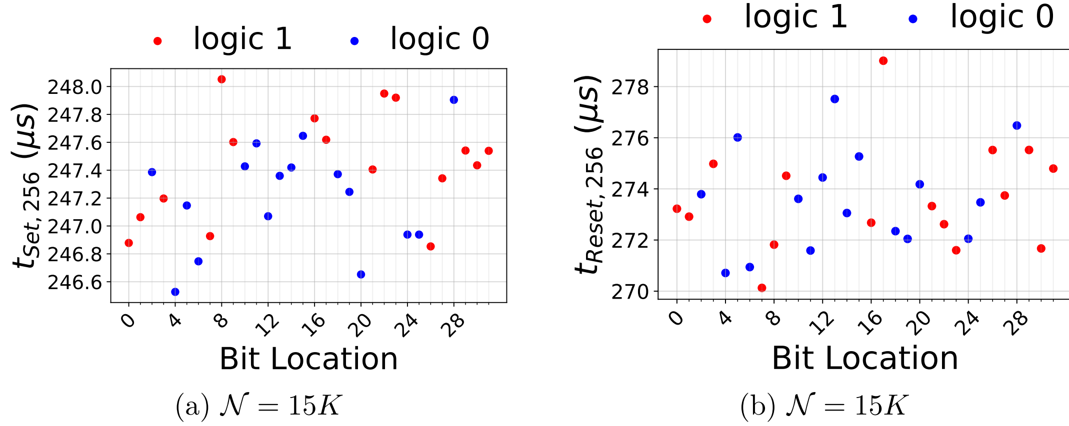


(a) $\mathcal{N} = 15K$ (b) $\mathcal{N} = 15K$

Figure 3.14: Retrieved data with an correct key with $\mathcal{N} = 15K$ stressing used to hide information with- (a) $t_{Set,256}$ (b) $t_{Reset,256}$.

the fresh and stressed memory cells. On the other hand, Fig. 3.15 is constructed with an incorrect key with the same $15K$ stress level to hide information. However, there is no clear separation between the fresh and stressed memory cells. Therefore,

the value of hidden bits can not be retrieved through thresholding. This result depicts that attackers find it difficult to retrieve hidden data without the correct key because each group is likely to have both fresh and stressed memory cells.



(a) $\mathcal{N} = 15K$ (b) $\mathcal{N} = 15K$

Figure 3.15: Retrieved data with an incorrect key with $\mathcal{N} = 15K$ stressing used to hide information with- (a) $t_{Set,256}$ (b) $t_{Reset,256}$.

3.3.10 Evaluation Summary

Overall, we draw the following main conclusions from the results.

1. The uniform switching characteristics of memory chips sharing the same part-number make it possible to sample a small set of memory chips from each part-number and perform cell characterization over those chips only. Additionally, we only need to characterize once to understand the switching properties.

2. Our silicon result shows that the imprinting and retrieval throughput is $0.4 bit/min$ and $15.625 bits/s$, respectively. The throughput will be even higher if the hiding scheme uses a smaller replica size.

3. Retention time has little or no impact over bit separation.

4. The *set* operation can endure higher initial and post-hiding stress levels than the *reset* operation.

5. The hidden information is not significantly affected by high-temperature baking and high-temperature system-level operation.
6. The Hiding key is a security parameter that can be adjusted to provide greater or lesser protection against brute force attacks and ransom attack.

3.3.11 Discussion

Hiding Multi-bit Data

Like real application data, multi-bit hidden data can be hidden using multiple memory addresses. For example, if we hide the first bit of hidden data in the first n address location, then we can hide the next bit of hidden data in the next n address location. Hidden data does not have any impact on regular application data.

The regular application data is stored in the device by utilizing the low resistance state (LRS) and high resistance state (HRS) of the ReRAM cell. On the other hand, our hidden data is stored by utilizing the *set/reset* time variation among the relatively fresh cells and worn-out (stressed) ReRAM cells. That is, a ReRAM cell can simultaneously hold the real application data bit by its resistance state and a hidden data bit by its *set/reset* time.

Applicability to Other NVM Technologies

Our proposed data hiding technique embeds the hidden information by repeatedly stressing the memory cells by writing '1' and '0' alternatively. Repeated stressing through switching operation ('1" → '0' or '0' → '1') gradually decreases the high resistance state (HRS) resistance. Stressing reduces cell endurance; hence, it is essential to keep the stress level as little as possible and simultaneously ensure that fresh and stressed cells are reliably separable. The maximum rated endurance

for commercial off-the-shelf (COTS) ReRAM chip is 1M rewrite cycles (i.e., $500K$ *set-reset* pairs). Our proposed method only costs 3% ($500K \times 3\% = 15K$ *set-reset* operations) of the total endurance of ReRAM cells. We believe our proposed technique might also be applicable for PCM memories as the endurance of PCM is $< 1M$ cycles [Eveb]. So, it might be possible to skew some of the physical properties with a realistic number of write operations, which might be utilized to hide data. However, the maximum rated endurance for some other emerging memories, such as MRAM, is greater than 10^{14} rewrite cycles [Eveb]. Therefore, one might require a large number of write operations to get an observable difference between fresh and stressed cells, which might not be practical[7]. Hence, we believe our technique is suitable only for those emerging memories with lower endurance levels.

3.4 Conclusion

This chapter demonstrated a cost-effective technique to hide information using the memory cells' *set/reset* time of commercially available ReRAM chips. *Write* (*set/reset*) time of ReRAM is an analog physical characteristic that has no relation with the stored digital content and the normal memory usage. Besides, the stored information can be survived successfully even after thousands of memory operations. In our proposed technique, we utilize repeated switching operations to change the physical properties of the memory cells. Without adequate knowledge about the hiding key, analog physical characteristics measurement will not help reveal the hidden information. The effectiveness of the proposed technique is evaluated by metrics of interest, i.e., the bit separation and hiding cost. Additionally, our proposed tech-

[7]We did not observe any noticalble difference in any physical property of MRAM cell after stressing it for 10^{10} times.

nique is robust against temperature variation and does not require any hardware modifications.

CHAPTER 4

IDENTIFY COUNTERFEIT RERAM DEVICES

In chapter 4, we discusses a novel low-cost technique to embed watermarking in devices with ReRAM by manipulating its analog physical characteristics through switching (*set/reset*) operation to prevent longstanding electronic counterfeiting problems. We develop a system-level framework to control memory cells' physical properties for imprinting irreversible watermarks into commercial ReRAMs that will be retrieved by sensing the changes in cells' physical properties. Our proposed ReRAM watermarking is robust against temperature variation and acceptably fast.

The emerging ReRAM has several advantages: architectural simplicity, high scalability, ultra-low power operation, high density, cross-bar structure feasibility, excellent reliability at high temperature, high endurance compared to other traditional storage memories, etc. [WYGW12, YPL+09, Fuj19]. Therefore, ReRAM has been investigated to a great extent to integrate into low-power applications, such as the Internet of Things (IoT), wearable devices (e.g., smartwatch, smart glasses), tablets, smartphones, automobiles, and medical devices (e.g., hearing aids). Such elevated use of ReRAMs makes it a lucrative target to counterfeiters. Our aim is to prevent counterfeiting of such chips by embedding watermarks in ReRAM cells by leveraging analog characteristics of ReRAM. The major contributions of this work are as follows.

- We characterize the impact of repeated stressing on ReRAM *write* time experimentally and show that the ReRAM *write* time increases monotonically with respect to the stress count.
- We present a novel idea of ReRAM watermarking by storing logic '0' bit in fresh ReRAM cells and logic '1' in stressed ReRAM cells. We experimentally

show that the imprinted data can be retrieved by observing ReRAM *write* time.

- We demonstrate the system throughput and verify the robustness of our proposed watermarking technique in multiple commercial off-the-shelf (COTS) ReRAM chips.

The rest of the chapter is organized as follows. Section 4.1 discusses the motivation of this work. Section 4.2 presents the proposed watermark imprinting and extracting mechanism. Section 4.3 explains the experimental setup and exhibits obtained results. Finally, Section 4.4 concludes the chapter.

4.1 Motivation

Fabricating chips in untrusted facilities is increasing worldwide, which paves the way for an easy entrance of counterfeit chips into the supply chain in different formats, such as recycled, remarked or forged documentation, tampered, cloned, reverse-engineered, out-of-spec/defective, and overproduced [KIPP19, GHD+14, BB16, RZZ+15, RT21]. However, among different types of counterfeiting, cloned, remarked, and recycled ICs are the most common type and usually occupy $> 80\%$ of the total counterfeit IC market share. Recent studies show that memory and memory integrated ICs (microprocessors, programmable logic devices, etc.) consist of $\sim$50% of the total counterfeit market share [GHD+14] as well as they are the most expensive component of a system. Most counterfeit memory chips suffer from sub-standard quality, poor performance, and shorter lifespan, severely affecting the security and reliability domains [GHD+14]. To date, there have been several anti-counterfeiting solutions to avoid fake chips, such as hardware metering, secured split testing (SST), on-chip sensor, split manufacturing, electronic chip ID, IC camouflaging, DNA marking, physi-

cal inspection-based test, burn-in test, and electrical test [GHD+14, BB16, RZZ+15]. Unfortunately, all of these techniques suffer at least one of the following limitations- (i) focused on a single counterfeit type (e.g., only identifying remarked chips), (ii) requires hardware modification, (iii) involves complex supply chain management, (iv) requires help from the subject-matter of experts, (v) suffers from low test accuracy, and (vi) requires expensive lab facility [GHD+14]. In contrast, watermarking is considered a cost-effective anti-counterfeit solution because watermark imprint-/extraction can be performed without circuit modification, subject-matter experts, or extensive testing [Bor21]. Watermarking is analogous to the digital signature to verify IC origin and can easily protect memory and memory-integrated ICs from such counterfeiting by watermarking memory components. Our proposed technique is applicable to all future electronic devices integrated with ReRAM memory.

This article focuses on preventing counterfeit ReRAM chips or chips with embedded ReRAM by watermarking technique. The emerging ReRAM has several advantages: architectural simplicity, high scalability, ultra-low power operation, high density, cross-bar structure feasibility, excellent reliability at high temperature, high endurance compared to other traditional storage memories, etc. [WYGW12, YPL+09, Fuj19]. Therefore, ReRAM has been investigated to a great extent to integrate into low-power applications, such as the Internet of Things (IoT), wearable devices (e.g., smartwatch, smart glasses), tablets, smartphones, automobiles, and medical devices (e.g., hearing aids). Such elevated use of ReRAMs makes it a lucrative target to counterfeiters. Our aim is to prevent counterfeiting of such chips by embedding watermarks in ReRAM cells by leveraging analog characteristics of ReRAM.

4.2 Proposed Watermarking Technique

The flowchart in Fig. 4.1 shows the steps of imprinting watermark chronologically. At first, we characterize a few memory cells to understand the analog physical characteristics of ReRAM cells at different stressing levels up to the maximum endurance (discussed in Sec. 3.2.1). Second, we imprint watermarks through repeated stressing the memory cells. These two steps are required to be performed only once. Finally, in the retrieval step, the end-user or manufacturer extracts the physical properties of the memory cells through standard digital interfaces.

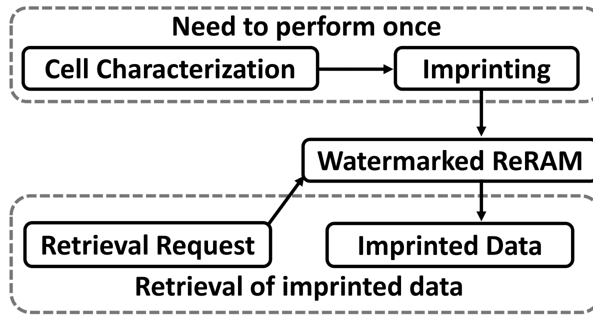


Figure 4.1: Steps used for ReRAM watermarking.

4.2.1 Imprinting Scheme

After characterization, our next step is to imprint watermarks in ReRAM. Chip manufacturers perform the proposed watermark imprinting technique into the memory during the die-sort testing phase [FBTR22]. The watermark may include standard device ID, chip-specific unique ID, and other manufacturing-related information [FBTR22]. In the proposed technique, we reserve a set of addresses for the watermark; the number of addresses depends on the length of the watermark. Initially, all memory cells possess perfect or near-perfect analog properties since they are fresh. To imprint watermarks, (i) initially, logic '1' is written to those reserved ad-

dresses (line 2 through line 4 of Algorithm 3), and (ii) repeated switching (*set* and *reset*) operations are performed (line 5 through line 14 of Algorithm 3) to only those ReRAM addresses, which are supposed to hold the logic '1' of target watermark. The switching operations are repeated until sufficient differences are developed in the *set/reset* time between fresh cells and stressed memory cells. Each switching operation gradually degrades the resistance of HRS, which are permanent; thus cannot be reversed. However, the number of repeated switching cycles, $\mathcal{N}$, used to imprint the watermark must be determined through the cell characterization phase for given memory chips (see Sec. 3.2.1). From an imprinting perspective, it is desirable to minimize $\mathcal{N}$ because the imprinting time of the watermark is directly proportional to the number of switching cycles. However, higher $\mathcal{N}$ enhances the accuracy by distinguishing fresh and stressed memory cells more perfectly.

4.2.2 Retrieval Scheme

System designers read watermarks to verify the chips' authenticity before incorporating them into the products or verify later in the product life-cycle. In order to retrieve watermarks and imprinted status information, the physical properties of memory cells are extracted (in our case, *set/reset* times) to distinguish between fresh and stressed memory cells. Line 15 to 26 of Algorithm 3 outlines the required steps of extracting the *set* and *reset* times from the watermarked addresses. We observe that both *set* and *reset* time change with stress counts, and both can be used to imprint watermarks. For example, the manufacturer can define a threshold value of *set/reset* time after imprinting the watermark, which can be used to differentiate between fresh and stressed memory cells.

Algorithm 3: Pseudo-code for imprinting and extracting watermarks.

Data: $\mathcal{N}$: Number of stress count (i.e. *set-reset* pairs)
$\mathcal{A}_{\mathcal{W}}$: Set of memory addresses containing watermark.
w_L: Word Length
$wMark$: Watermark
$\mathcal{D}$: $(1 \times w_L)$ matrix containing data intended to write each memory cells
t: Timer

Result: $\mathcal{S}_{\mathcal{T}}$: *Set* time of each memory address belongs to $\mathcal{A}_{\mathcal{W}}$
$\mathcal{R}_{\mathcal{T}}$: *Reset* time of each memory address belongs to $\mathcal{A}_{\mathcal{W}}$

```
   // Initialization
1  S_T = {}; R_T = {}; D = Ones(1 × w_L);
2  foreach a ∈ A_W do
3  |   write(a, D);
4  end
   // Imprinting watermark
5  for i = 0 to N do
6  |   foreach a ∈ A_W do
7  |   |   if wMark[Bit]==1 then
8  |   |   |   D = Zeros(1 × wS);
9  |   |   |   write(a, D);
10 |   |   |   D = Ones(1 × wS);
11 |   |   |   write(a, D);
12 |   |   end
13 |   end
14 end
   // Extracting watermark
15 foreach a ∈ A_W do
16 |   D = Zeros(1 × w_L);
17 |   tic = t;
18 |   write(a, D); // Set operation
19 |   toc = t − tic; // Accumulating Set time
20 |   S_T = S_T ∪ {toc};
21 |   D = Ones(1 × w_L);
22 |   tic = t;
23 |   write(a, D); // Reset operation
24 |   toc = t − tic; // Accumulating Reset time
25 |   R_T = R_T ∪ {toc};
26 end
```

It is worth mentioning that *set/reset* characteristics of ReRAM cells appear to be uniform across all ReRAM chips that we have tested. Therefore, the manufacturer can define a fixed standard set of addresses for all memory chips for watermarking. Such arrangement should simplify the evaluation process. For example, the manufacturer can make the addresses that are used for watermarking publicly available. Anyone with this information should be able to access the watermark data and verify the chip authenticity.

4.3 Performance Evaluation

The analysis is also performed over five *MB85AS8MT*[1] (40nm technology node) 8-bit serial peripheral interfaced (SPI) $8Mb$ memory chips manufactured by Fujitsu Semiconductor Limited. The following steps are performed to verify the feasibility of the proposed watermarking. We have imprinted an arbitrarily chosen 32-bit random data into $(256 \times 32) = 8192$ memory addresses varying the number of switching cycles, $\mathcal{N}$, up to $20K$ times to experimentally demonstrate the watermark imprinting (discussed in Sec. 4.2.1) and retrieval (discussed in Sec. 3.2.3) process.

Fig. 4.2 represents the experimental data from arbitrarily chosen test chips with imprinted data 0xC2F740EB. We have also verified with other random data. We imprint the data in a random memory location. The red and blue dot represents the imprinted logic 1's and 0's, respectively. Fig. 4.2 shows that logic '1' and logic '0' begin to separate at $5K$ stress count (Fig. 4.2a), and they become well-separated at $10K$ stress count (Fig. 4.2b). With further stress, the separation between logic '1' and logic '0' further increases (Fig. 4.2c). Similarly, with $t_{Reset,256}$, logic '1' and logic

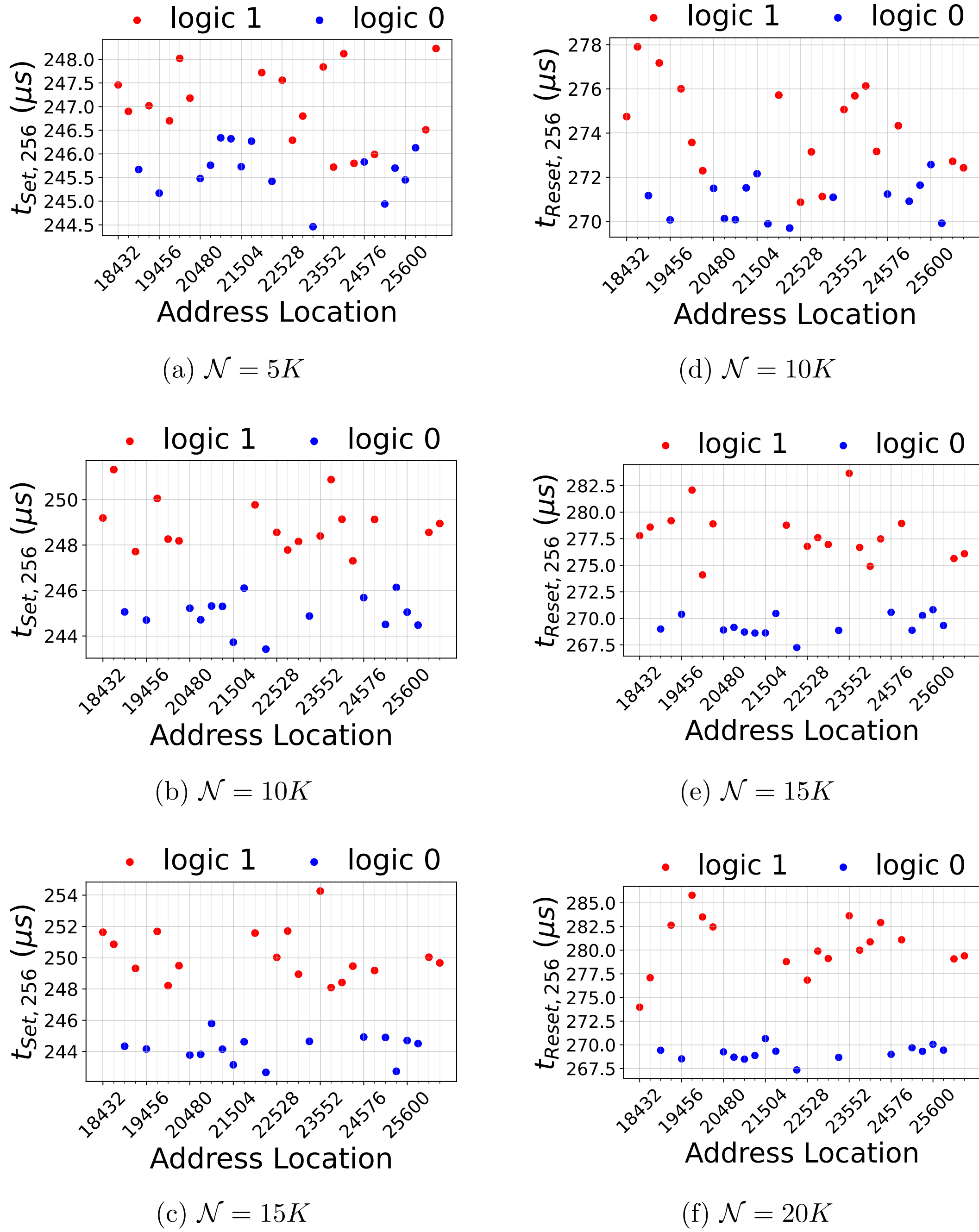
[1]We have also verified our proposed technique with *MB85AS4MT* ReRAM chips produced by the same manufacturer. However, the Fujitsu *MB85AS4MT* (180nm technology node) ReRAM chip is commercially discontinued, and the *read/write* operation is much slower than the *MB85AS8MT*.

'0' begin to separate at $10K$ stress count (Fig. 4.2d) and become well-separated at $15K$ stress count (Fig. 4.2d). Therefore, with a proper threshold value of $t_{Set,256}$ (at $10K$ stress) or $t_{Reset,256}$ (at $15K$ stress), one can easily separate logic '0' and logic '1' bits.

Fig. 4.3 verifies the watermark data imprinted in all five test memory chips. This figure represents the distribution of $d(b_0, b_1)$ at a different level of stresses, where $d(b_0, b_1)$ represents the distance between logic '0' bits (b_0) and logic '1' bits (b_1). Each dot in Fig. 4.3 represents $d(b_0^i, b_1^j)$ for each possible (b_0^i, b_1^j). For well-separated logic '0' and '1', the distance should be positive. A larger value of $d(b_0^i, b_1^j)$ is more desirable as it provides better separation between logic '0' and logic '1' bits. However, if the maximum value of *set/reset* time of logic '0' bits is larger than the minimum value of *set/reset* time of logic '1' bits (similar to Fig. 4.2a), then logic '0' bits and logic '1' bits cannot be separated properly. In such a scenario, the $d(b_0^i, b_1^j)$ can be negative for a few pairs of (b_0^i, b_1^j). The figure demonstrates that the separation between logic '0' bits and logic '1' bits improves monotonically with respect to stress count. For all test chips, the logic '0' bits (b_0) and logic '1' bits (b_1) are clearly separable after $10K$ stresses with $t_{Set,256}$ and $15K$ stresses with $t_{Reset,256}$ (i.e., $\min\left(d(b_0^i, b_1^j)\right) > 0$).

4.3.1 Robustness Analysis

The watermark should also be resilient to the variation of operating conditions, i.e., it will not be possible to modify or change the watermark information with localized heating or operating voltage. To verify the robustness of our imprinting technique against the temperature, first, we have watermarked a fixed address-space with $15K$ stress. Then we have isolated watermarked memory chip from the



(a) $\mathcal{N} = 5K$ (d) $\mathcal{N} = 10K$

(b) $\mathcal{N} = 10K$ (e) $\mathcal{N} = 15K$

(c) $\mathcal{N} = 15K$ (f) $\mathcal{N} = 20K$

Figure 4.2: Imprinted data at different stress count- (a)–(c) $t_{Set,256}$ at stress count $5K$, $10K$, and $15K$; (d)–(f) $t_{Reset,256}$ at stress count $10K$, $15K$, and $20K$.

system and baked it at $80°C$ for 3 hours. Lastly, we have evaluated the $t_{Set,256}$ and $t_{Reset,256}$ while maintaining the chip temperature of $80°C$. We have observed that the

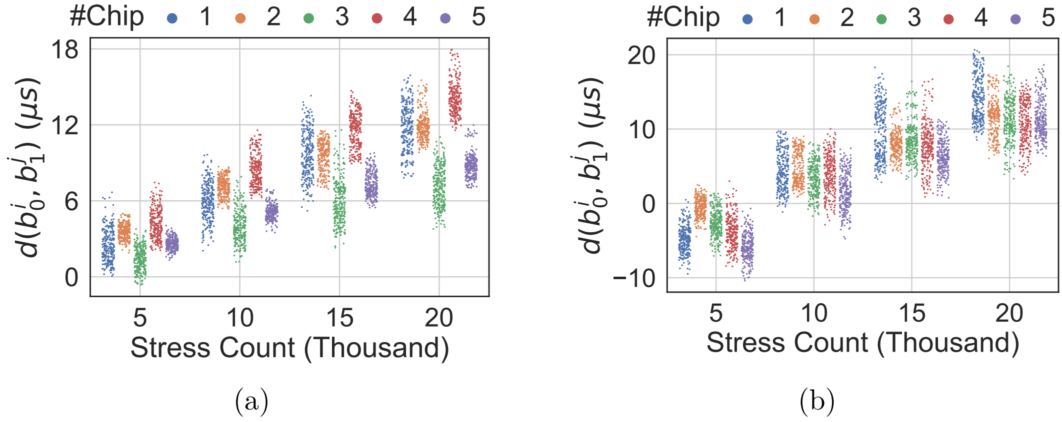


Figure 4.3: Verifying watermark in test chips, using- (a) $t_{Set,256}$, and (b) $t_{Reset,256}$.

watermark information is not affected by temperature and remains well-separated (Fig. 4.4) after the high-temperature baking and high-temperature system-level operation (considering both $t_{Set,256}$ and $t_{Reset,256}$).

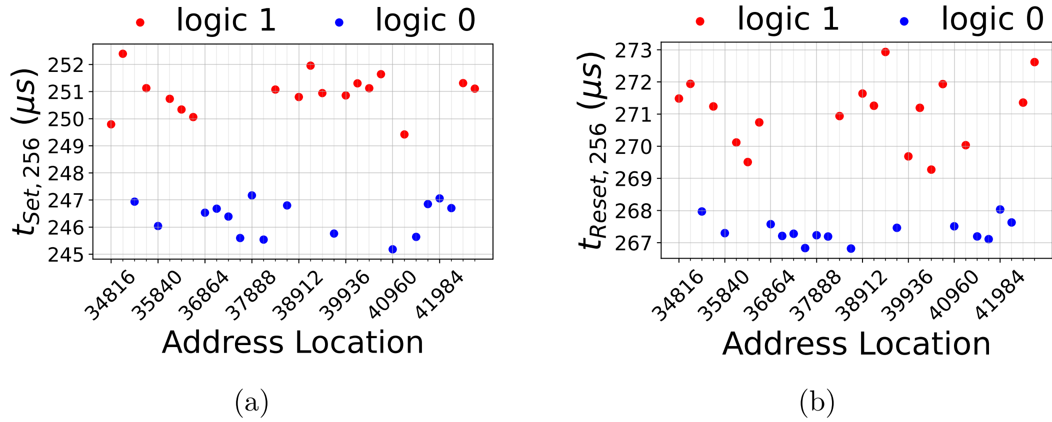


Figure 4.4: Robustness analysis after high-temperature baking (80°C) with- (a) $t_{Set,256}$ (b) $t_{Set,256}$

4.3.2 Performance Analysis

Imprinting Time

The proposed technique for imprinting watermarks relies on repeatedly switching state of ReRAM cells. Thus, the time required to imprint the watermark is directly proportional to the number of stress count, $\mathcal{N}$. The estimated time to imprint watermark is, $\mathcal{T}_{imprint} = (\mathcal{N} \times \mathcal{B}_{WMark} \times \mathcal{T}_{switch_{pair}})$; where $\mathcal{T}_{switch_{pair}} = (\mathcal{T}_{set} + \mathcal{T}_{reset})$ represents stressing time (*set-reset* pair) for 256 addresses (switching resistance state with single *write* command), and $\mathcal{B}_{WMark}$ represents the number of imprinted bits. The chip used for our experimental evaluation has the following timing parameters: $\mathcal{T}_{switch_{pair}} = (5ms + 5ms) = 10ms$, and $\mathcal{B}_{WMark} = 32$. Thus, the baseline implementation requires $((5ms + 5ms) \times 32 \times 10k) = 3200s$ for $10K$ switching operations to imprint the watermark. Therefore, the throughput for the watermark imprinting is $\frac{32bits}{3200s} = 0.6bit/min$. It is worth mentioning that the imprinting time of our proposed technique heavily depends on the *write* speed of the ReRAM chips. Fortunately, in the past few years, the *write* speed of ReRAM chips significantly improved and will continue to improve in the future. For example, the *write* speed of *MB85AS8MT* ReRAM chips is improved $>3X$ over its previous generation *MB85AS4MT* ReRAM chips.

Retrieval Time

Unlike the imprinting procedure, the extraction procedure is significantly fast. The estimated time to retrieve the watermark can be calculated by- $\mathcal{T}_{retrieve} = (\mathcal{T}_{switch} \times \mathcal{B}_{WMark} \times \mathcal{N}_{rep})$; where $\mathcal{T}_{switch}$ is the average value of $t_{Set,256}$ or $t_{Reset,256}$; and $\mathcal{N}_{rep}$ represents the number of addresses used to imprint single bits. After $10K$ stressing, the average value $t_{Set,256}$ is $\sim 250\mu s$, and we used $\mathcal{N}_{rep} = 256$ in our implementation.

Therefore, the throughput for the watermark retrieval is $\frac{\mathcal{B}_{WMark}}{\mathcal{T}_{retrieve}} = \frac{32bits}{250\mu s \times 32 \times 256} = 15.625bits/s$.

Watermarking Cost

Our proposed technique only requires $10K$ *set-reset* operations (i.e., $20K$ rewrite cycles) to make a distinguishable separation between logic '0' and '1' of the imprinted watermark (using $t_{Set,256}$). However, the rated endurance of ReRAM chips is $1M$. Therefore, our proposed technique costs only 2% of the rated endurance of imprinted addresses.

4.3.3 Security Analysis

In this section, we perform the security analysis of our proposed technique. We use the key to enhance watermarking security. For watermarking, we use *set/reset* time to differentiate between logic '0' and logic '1'. Therefore, an attacker can exhaustively measure *set/reset* time over the whole memory address space and can identify memory cells containing the hidden information. To prevent such issues, we need some countermeasures. Fig. 4.5a shows that if we imprint plain watermark in the memory chip, counterfeiter can extract the plain watermark and imprint the original watermark in his counterfeit memory chip. Therefore, as a prevention to avoid cloning and secure watermark data, we propose an asymmetric encryption algorithm as of Fig. 4.5b. In this algorithm, we use the key similarly to the digital signature. While imprinting data (done by the chip manufacturer), the manufacturer can add a silicon-generated salt to the data and encrypt it with a private key. Now, this encrypted data can be imprinted as secure watermarked data. On the other hand, the end users evaluate the authentication phase by decoding the watermark data.

In the authentication phase, the encrypted watermark data need to be decrypted with the public key, and then, the same salt data need to be subtracted. This will result in the plain watermark data. As we propose this watermarking technique for chip authentication, the chip manufacturer needs to generate the private-public key pair and make the public key available to the user. Note that silicon-generate salt can be any physical property that is constant over the next few read/write cycles (as watermark data is expected to verify only once when the chip is new) and varies from chip to chip. For example, the read latency of the ReRAM cell is fairly constant over a few read/write cycles and slightly varies from one memory cell to another due to random process variation in the fabrication process. Therefore, the read latency can be used as a salt.

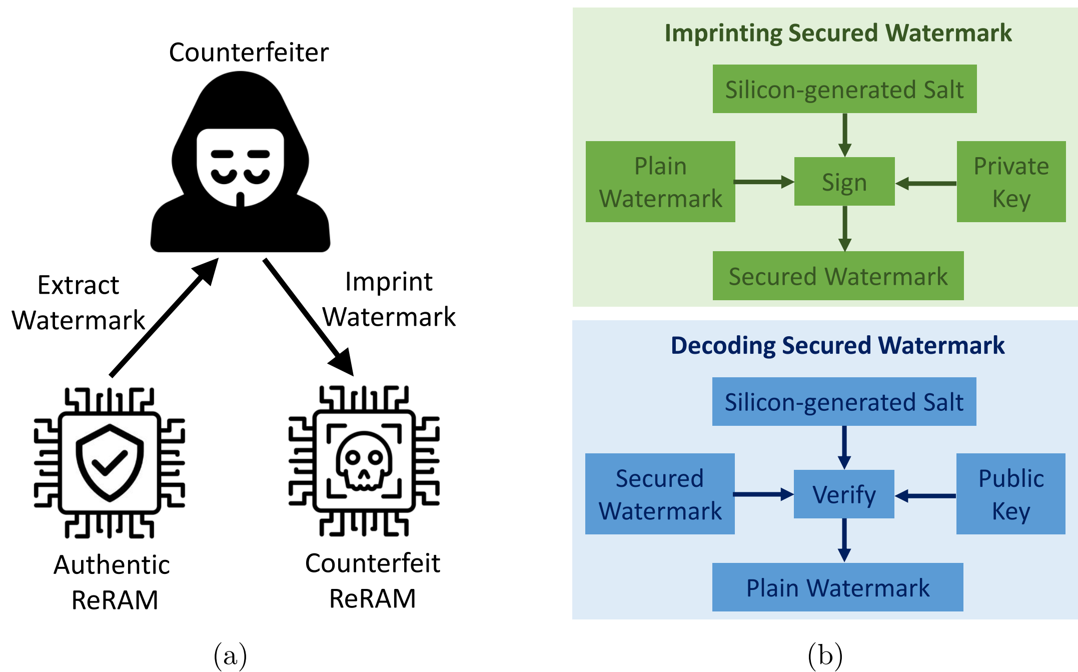


Figure 4.5: (a) Bypassing watermarking scheme (b) Secured watermarking.

Under this system (Fig. 4.5b), a pair of keys is used to encrypt and decrypt information (asymmetric encryption). A private key is used for imprinting the

watermark, and a public key is used for extracting the watermark. The manufacturer is solely responsible for generating this private-public key pair and making the public key available for end-user usage. Keys can be generated as a single process using a random bit generator and an approved set of rules. The public key can be provided to any upon request and does not need any secure distribution protocol; as long as the private key is not known to the attacker, the watermark data cannot be cloned and imprinted to a counterfeit chip. Therefore, our proposed watermark technique is secure as long as the private key is secure (Fig. 4.5b). Without the knowledge of the private key, the attacker will not be able to encrypt the watermark data properly. However, if the private key is compromised, the attacker will be able to imprint the watermark on a counterfeit chip, and therefore, our proposed technique will become ineffective.

4.3.4 Discussion

Watermarking and data hiding (chapter 3) uses the same electrical property of ReRAM cell (*set/reset* time). However, the data hiding algorithm is significantly different from the watermarking algorithm. The both watermarking and data hiding algorithm require to fulfill some additional requirements —

- Watermark data can be an open secret. As long as watermarked data is unmodifiable, it can serve the purpose. On the other hand, hidden data need to be as stealthy as possible.
- While watermarking, we need to ensure the imprinted data can be retrieved at least once to verify the device's authenticity. Once the authenticity is verified, we no longer need the watermark data. On the other hand, ideally, the hidden data is supposed to be sustained forever. However, in practice, hidden data

can become noisy due to the continuous usage of memory over time. Therefore, we aim to keep the hidden data alive as long as possible. For example, hidden data in flash memory can be sustained till $10K$ regular write operation (which is also typical endurance of flash memory); on the contrary, with our proposed technique with ReRAM, the hidden data can endure at least $100K$ regular write operation without inducing any bit error.

- Usually, watermarking is less expensive than data hiding. For example, in the worst case (reset operation), watermarking only takes $15K$ switching operations to imprint the data. On the other hand, data hiding takes at least $45K$ switching operations to keep the imprinted data alive for the next $200K$ cycles. Therefore, data hiding is a more time-consuming process than the watermarking technique.

4.4 Conclusion

This chapter demonstrated a cost-effective watermark imprinting and extraction technique using commercially available ReRAM chips. In our proposed technique, we utilize repeated switching operations to change the physical properties of the memory cells. The effectiveness of the proposed technique is evaluated by metrics of interest, i.e., the bit separation, imprinting throughput, extraction time, and imprinting cost. Additionally, our proposed technique is robust aging temperature variation and does not require any hardware modifications.

CHAPTER 5

MRAM-BASED ROBUST TRNG

This chapter demonstrates an effective technique of generating random numbers from energy-efficient consumer-off-the-shelf (COTS) MRAM chips. In the proposed scheme, the inherent (intrinsic/extrinsic process variation) stochastic switching behavior of magnetic tunnel junctions (MTJs) is exploited by manipulating the write latency of COTS MRAM chips. Subsequent NIST SP-800-22 suite test reveal that the proposed latency-based TRNG is acceptably fast and robust over a wide range of operating conditions. In this work, we propose a technique of generating random numbers that meets the aforementioned requirements by exploiting write latency variations of COTS MRAM chips. In summary, the major contributions of this work are as follows.

- We reduce the write enable ($\overline{W}$) time from the manufacturer recommended value during the write operation to introduce errors. Errors from some of the cells at the reduced timing parameter are completely random and can be used as a source of randomness. However, some of the cells exhibit deterministic behavior. Therefore, we further propose an algorithm to select the most suitable memory cells that exhibit proper randomness in order to generate robust and high-quality random numbers.
- We demonstrate the system throughput and robustness of our proposed TRNG in multiple COTS Everspin toggle MRAM chips ([Evea]) under a wide range of operating conditions.

The rest of the chapter is organized as follows. Section 5.1 discusses the motivation of this work. Section 5.2.1 briefly overviews the operating principle of MRAM chips. Section 5.3 presents the proposed technique of generating true random num-

bers, including cell characterization and suitable bit-selection algorithm. Section 5.4 explains the experimental setup and exhibits obtained results to verify the quality and robustness of the proposed TRNG. Finally, section 5.6 concludes the chapter.

5.1 Motivation

True random number generator (TRNG) plays an important role in cryptographic applications such as random key generation, cryptographic nonces, session keys, one-time-pad, initial seeds of pseudo-random number generator (PRNG), challenge for authentication, hardware metering, etc. [MJS+16, RXF+14, GHD+14, KPH+19]. A TRNG translates random physical phenomena (i.e., physical entropy) into digital sequences. the process variation during integrated circuits (ICs) fabrication is also responsible for random noise [LBB+14]. In most cryptographic applications, the quality of the security of a system relies on the quality of random numbers. A poor TRNG can always be a target to an attacker for attacking the whole system. A TRNG can be a discrete or an integral part of the system (e.g., on-chip TRNG). Usually, an on-chip TRNG has several advantages such as low-overhead (area and energy), non-deterministic, high-throughput, simple design, robust against a wide range of operating conditions, etc.

Although there are many dedicated hardware designs for TRNG, memory-based TRNG are most popular as memory is a ubiquitous component of all computing devices, and therefore, it eliminates the requirement of a new hardware component. Unfortunately, all previous studies of memory-based TRNG are mainly focused on conventional memories, such as DRAM, SRAM, NAND Flash, etc., which have several limitations, including (i) low throughput, (ii) power inefficiency, (iii) lack of robustness, etc. In this dissertation, we proposed MRAM-based TRNG

for future devices which is free from such limitations. In addition, MRAM can turn into a dominant universal memory (cache and main memory) technology due to its promising scopes such as high density and reliability, non-volatility, scalability, thermal robustness, unlimited endurance, high speed, and fast read access, ultralow-power operation, CMOS compatibility, and radiation hardness [KYK+08]. Because of these advantages, most of the systems are expected to include MRAM chips. Therefore, MRAM can be an attractive candidate for low-power TRNG. There have been several high-quality and robust memory-based TRNGs, but most of them suffer from high-overhead and low-throughput [WYW+12, MJS+16, KLK17, YDW+18, HST+12, FTSR21]. MRAM-based TRNGs have gained attention because of their capability of generating significantly high quality and robust random numbers [KCL+21, YDW+18, VDNP16]. However, the existing MRAM-based TRNGs are mostly simulation-based or need modification in standard MRAM structures [KCL+21, VDNP16, YDW+18, FTSR21]. Furthermore, some MRAM-based TRNGs can not be used in the computer system because of their incapability of reconfiguration [KCL+21, FTSR21]. The previous contributions inspirit the need for real memory implementation and build the foundation of proposed MRAM-based TRNGs that can be generated from COTS MRAM chips, require minimal or no additional hardware, are robust against environmental fluctuations, and considerably high throughput.

5.2 Background Preliminaries

5.2.1 MRAM Operation

Storing data in a magnetic state has several benefits over charge-based storage such as non-destructive read operation, unlimited read/write endurance, no leakage during magnetic polarization, no wear-out due to no movement of electrons/atoms during the switching process of magnetic polarization [FTSR21], etc. Moreover, Savtchenko switching based MRAM arrays possesses several important performance characteristics such as lower write error rate and fast read/write cycle ($35ns$). They are also less sensitive to external fields, and therefore they are less sensitive to manufacturing process variations [EAB+05]. The write operation of the MRAM chip can be governed by three different control parameters: write enable ($\overline{W}$), chip enable ($\overline{E}$), and upper/lower byte enable (($\overline{UB}$)/($\overline{LB}$)) signals. A simplified version of the write enable ($\overline{W}$) controlled write operation of the MRAM chip is shown in Fig. 5.1.

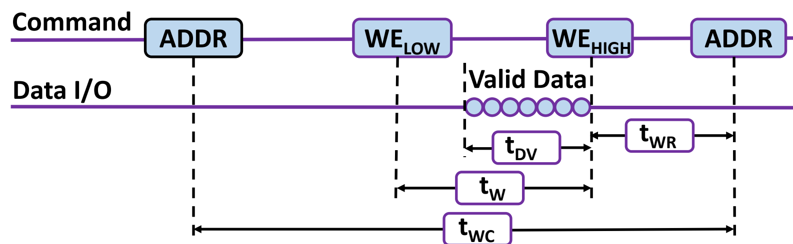


Figure 5.1: Write enable ($\overline{W}$) controlled write cycle of MRAM chip.

Here,

t_{WC} = *write cycle time*, i.e., the time period to complete full write operation in a particular address.

t_W = *write pulse width*, i.e., the time period for which the $\overline{W}$ pin is kept activated.

t_{WR} = *write recovery time*, i.e., the time to complete the write operation after the $\overline{W}$ pin is deactivated.

t_{DV} = *valid data to end of write*, i.e., the time for which the valid data need to be available in the data I/O before the $\overline{W}$ pin is deactivated.

If the output enable ($\overline{G}$) becomes active at the same time, or after $\overline{W}$ is activated, the output will remain in the high impedance state. After all three write control parameters ($\overline{E}$, $\overline{W}$, or $\overline{UB}/\overline{LB}$) become disabled, the $\overline{G}$ signal must remain in the steady-state high for at least $2ns$. Reducing any of these timing parameters can improve the speed and reduce power consumption but may lead to faulty operation. The write timing parameter t_W is manipulated in this work to introduce errors during $\overline{W}$ controlled write operation.

5.2.2 Entropy Source

When the magnetic orientation of the free layer and the fixed layer are in the same direction (i.e., parallel, Fig. 5.2a), it presents low resistance state (R_{Low}). On the other hand, the antiparallel orientation of the free layer and fixed layer (Fig. 5.2b) represents a high resistance state (R_{High}). The sensing circuit compares the cell resistance with the reference resistance ($R_{Ref} = \frac{(R_{Low}+R_{High})}{2}$) and determines if the cell is storing '0' or '1'. To change the stored data in MRAM, we need to change the magnetic orientation of the free layer. However, the magnetic orientation of the free layer can only be changed by applying the write current for an appropriate amount of time which is known as the write time. An insufficient write time may fail to orient the magnetic polarity of the free layer in the appropriate direction (Fig. 5.2c), and the cell resistance may become halfway between R_{Low} and R_{High} (metastable). If the cell resistance is exactly halfway between R_{Low} and R_{High},

the logic sensing circuit will provide random output. The randomness of the cell output mostly depends on random thermal noise (also known as Johnson–Nyquist noise), which is the major entropy source of our proposed TRNG [Che21]. Thermal noise is generated from the random motion of free electrons in the material (in our case, the dielectric material between two electrodes). The random motion of the electron is a direct result of the random thermal vibration of atoms in the material lattice. The magnitude of the thermal vibration of atoms is directly proportional to the temperature [Ste19]. Therefore the random noise in an electronic device is also directly proportional to the temperature. The random noise is random as the motions of atoms are completely independent of one another, and each atom may have a different kinetic energy profile (they can transfer heat to free particles like electrons). Therefore, the thermal vibration of an atom and corresponding thermal noise is completely random, and the resulting output sequence of our TRNG is also random. In our proposed algorithm, we have experimentally chosen a suitable value of write time which causes maximum temporal variation on cell output. We experimentally observe that fewer MRAM cells produce random sequences at low temperatures than at high temperatures. Such dependency on thermal noise arises a potential attack surface on TRNG, i.e., creating a deterministic sequence by reducing the operating temperature. Thus, for our proposed TRNG, we recommend selecting a subset of metastable cells which produce random sequences at the lowest possible operating temperature (worst possible case) and use them over the entire operating temperature range.

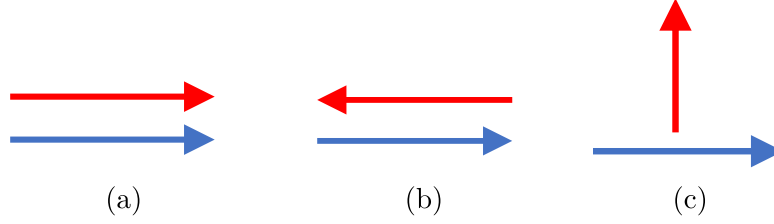


Figure 5.2: Different state of MRAM cell. Red arrow defines the magnetic orientation of free layer, blue arrow defines the magnetic orientation of fixed layer. (a) parallel magnetic orientation in low resistance state (R_{Low}), (b) anti-parallel magnetic orientation of high resistance state (R_{High}), and (c) abnormal magnetic orientation of free layer when the write current is applied for insufficient amount of time while changing the magnetic orientation. Here, the magnetic orientation of free layer is perpendicular to magnetic orientation of fixed layer and resistance will be in-between R_{Low} and R_{High}.

5.3 Generating Random Numbers using COTS MRAM

In our proposed methodology, we exploit the random Savtchenko switching at the reduced (manufacturer-recommended) timing parameter for generating true random numbers. When the *write pulse width*, t_W, of toggle MRAM (see Fig. 5.1) is reduced, it does not get sufficient time and write current to toggle into the desired stable state. Due to the process variation and the non-uniform distribution of current pulse within the chip, random variations are created in the MTJ storage element. Therefore, all of the memory cells are not capable of performing an appropriate write operation. That is the reason that a manufacturer specifies a set of timing parameters for reliable read/write operations. Violation in any of these manufacturer-recommended timing parameters may cause erroneous/faulty outputs during the read/write operation.

If the t_W is not sufficient, there is a chance that FML is not aligned perfectly with the RML (either the same or in the opposite direction) and might settle on an intermediate position. This arrangement may lead the cell resistance to be halfway between R_{Low} and R_{High} [TDK]. Therefore, at reduced t_W, if the resultant cell resistance falls around the ΔR_{Use} region of the resistance distribution (see Fig.

2.1b) curve, the cell will show indeterministic characteristics and generate random bits.

In our proposed scheme, several steps are involved in generating true random numbers. At the reduced t_W, MRAM chips create errors, and the total number of errors differ at different reduced t_W values. At first, we select the most suitable reduced t_W value. This selected t_W aims to maximize the number of cells that can be used for TRNGs. Second, we propose a cell selection algorithm to characterize all of the temporally unbiased MRAM cells from a set of measurements to identify the most appropriate memory cells for generating robust random numbers. The mentioned two steps must be performed only once to choose the appropriate number of random MRAM cells. Finally, we collect data from multiple measurements from the selected MRAM cells and use a low-overhead post-processing technique to generate high-quality random numbers.

5.3.1 Appropriate Reduced Time Selection

The experimental results show that some of the memory cells provide erroneous outputs if the data is written at the reduced timing parameters. The number of these error-prone cells varies within the write pulse activation time range t = $[0,t_W]$. We change the t_W and count the total number of erroneous bit cells. Our main objective is to find a suitable t_W for which the maximum number of erroneous bits is achieved. The number of erroneous cells is calculated from all achievable reduced write timing parameters in the next step. Finally, we propose an algorithm to characterize the memory cells among those erroneous cells to generate random numbers for any system using the timing parameter for which the maximum number of random cells is obtained. The cell characterization technique is described in section 5.3.3.

5.3.2 MRAM Cells Characterization

Our experimental result manifests that all of the memory cells are not suitable to generate robust random numbers. To locate these random cells, at first, we characterize MRAM memory cells by writing different intuitive (solid) and non-intuitive (random, checkerboard, and striped) input data patterns to the entire memory cells at the reduced write enable time, t_W, and read back the full memory contents with appropriate timing parameters a total of N times. Larger N provides better characterization results but increases the computation time for characterization.

Theoretically, reduced write operation reduces the current flowing through the MTJ storage elements [BOET14]. Hence, the magnetic orientation switching (*parallel (P)* $\rightarrow$ *anti-parallel (AP)* or vice versa) time increases significantly [BOET14]. Switching from P to AP is more vulnerable to reduced write operation due to enhanced switching delay, leading to the write failure. Our experimental results also manifest that the write operation at the reduced t_W produces erroneous data. Moreover, these error patterns depend on the input data pattern to be written and vary with different memory chips. Based on the error patterns from the sample measurements, the MRAM cells can be classified into the following two categories-

Persistent Cells: These cells produce stable output from measurement to measurement. These stable cells are excellent candidates to generate memory-based Physically Unclonable Function [VDNP16, KCL+21] but not competent for true random number generation because of manifesting consistent behavior at the reduced t_W.

Noise-prone Cells: These cells provide inconsistent output for different measurements. However, we also observe that most noise-prone cells are biased toward '0' or '1'. Therefore, to avoid producing deterministic random numbers, we propose a cell selection technique to exclude those biased cells from the noise-prone cells.

5.3.3 Appropriate Cell Location Selection

To generate a robust TRNG, unbiased cells need to be filtered because all cells do not provide the same amount of entropy. At first, we discover all erroneous cells at the reduced t_W from (N) measurements. At the reduced t_W, some cells will not create any errors; we define them as the correct state $(\mathcal{S}_C)$. On the other hand, the other cells will create erroneous outputs; we define them as the error state $(\mathcal{S}_E)$. Next, we record the change of state $(\mathcal{S}_C \rightarrow \mathcal{S}_E$ or $\mathcal{S}_E \rightarrow \mathcal{S}_C)$ or flip of all cells comparing to the two consecutive measurements from each of N measurements. This forms a $(1 \times M)$ array containing the total number of flips in each cell location from N measurements, where M is the total number of memory cells. Second, to select the random cells, we need to determine the appropriate threshold, Th. Theoretically, the expected value of Th is $p \times (N-1)$, where p $(= \frac{1}{2})$ is the probability of state change (flip). However, in reality, a fixed Th might not provide sufficient random cells. Hence a specified bound $(Th_L \leq Th \leq Th_U)$ needs to be defined, where Th_L and Th_U are the lower- and upper-bound of the threshold range. Cells within this boundary are considered as TRNG candidates. The silicon results show that the obtained random cells above Th_U are significantly negligible. Therefore, we only choose those cells for which the $(1 \times M)$ array contents are above Th_L. The locations of these unbiased noisy cells are stored in data-set $\mathcal{F}_C$. The step-by-step procedure is shown in Algorithm 4. A true random number must be highly temporal variant, which is the basis of our proposed algorithm. Therefore, the selected random cells with the proposed algorithm are capable of generating high-entropy random numbers.

Algorithm 4: Pseudo-code for Random Cell Location Selection

```
procedure rand_cell_loc(N, input_data, Th_L)
 1: /* N = Number of total measurements */
 2: /* input_data = (1 × num_cell) matrix containing input data stored to each
    memory cell at reduced t_W */
 3: /* Th_L = Lower threshold bound to choose true random cells */
 4: num_cell = Total number of memory cells
 5: /* flip_count matrix stores total number of state change (flip) from
    consecutive N measurements */
 6: flip_count = zeros(1 × num_cell)
 7: for i = 1 to N − 1 do
 8:     x = bitwise_xor(input_data(i), input_data(i + 1))
 9:     flip_count = bitwise_add(x, flip_count)
10: end for
11: /* rand_loc matrix stores random cell locations */
12: rand_loc = zeros(1 × num_cell)
13: num_randcell = 0 /* Total number of random cells */
14: for i = 1 to num_cell do
15:     if flip_count(i) ≥ Th_L then
16:         rand_loc(i) = 1
17:         num_randcell + +
18:     end if
19: end for
20: return rand_loc
end procedure
```

5.3.4 Low-overhead Post-processing

However, the raw random sequence that apparently seems unbiased might provide biased results under extreme operating conditions. Therefore, several post-processing techniques such as Von Neumann corrector, XORing multiple bits, cryptographic hash function, etc. are used to generate a fully non-deterministic random sequence [KEC+11]. Secure Hash Algorithm (SHA)-256 is area efficient, fast, and fewer input bits are required to generate the same entropy level [GYG12, FTSR21]. Therefore, we chose SHA-256 as a post-processing technique and applied over the bit sequence obtained from $\mathcal{F}_C$. To generate a random number of the required

length, at first, we accumulate the obtained random cells at the selected reduced t_W. The value of the measurements is a function of the required length and the used post-processing technique. Next, the bit sequence is split into appropriate chunks to feed into the SHA-256 hash algorithm. Finally, the multiple output chunks gathered from the SHA-256 hash function are concatenated to produce truly unbiased random numbers and are denoted as $\mathcal{H}_C$.

5.4 Experimental Results

The primary analysis is performed over ten (2 chips from each *MR0A16ACYS35, MR0A16AYS35, MR1A16AYS35 MR2A16ACYS35, MR2A16AYS35* models) 16-bit parallel interfaced differently sized ($1Mb - 4Mb$) memory chips manufactured by Everspin technologies. Table 5.1 captures the key features of the memory chips. Among them, five chips are selected randomly to perform extensive analysis for TRNG. We have used our own custom memory controller implemented on *Xilinx Artix 7 (XC7A35T-1C)* FPGA to manipulate different timing latency of a couple of emerging memories [Raj]. As discussed in section 5.3.2, the generated error is pattern dependent at reduced operation. Hence to determine the suitable pattern that is capable of generating high-entropy true random numbers, we collected a total of 5-set measurement data with seventeen different intuitive (solid) and non-intuitive (random, checkerboard, and striped) 16-bit input data patterns: ($0xFFFF$, $0xAAAA$, $0x5555$, $0x0000$) from each ten memory chips. We observed that the *solid* $0x0000$ pattern produces comparatively high erroneous bits than other patterns. Therefore, we can conclude that *parallel (anti-parallel)* configuration is the logic state '1' (state '0'). Next, to characterize the MRAM cells (discussed in section 5.3.2), we collected a total of 50-set measurement data with only *solid* $0x0000$ input pattern

from the selected five memory chips. We chose the smallest possible achievable (due to experimental limitation) value of t_W, 16.6% of the recommended t_W, for this work. However, our selected value of t_W can generate a sufficient number of incorrect outputs to generate high-quality random numbers.

Table 5.1: MRAM Chip Specifications [Evea].

Parameter	Standard Value
Capacity	1 - 4 Mbits
Supply Voltage	3.3 V
Read/Write Cycle (t_{WC})	35 ns
Write Pulse Width (t_W)	15 ns
Write Recovery Time (t_{WR})	12 ns
Valid Data to end of Write (t_{DV})	10 ns
Address/Data Bus Length	16 - 18/16
Retention Time	>20 years
AC Stand by Current	18–28 mA
AC Active Current (Read/Write)	55–80 mA/105–165 mA

5.4.1 Selection of t_W

To compare the behavior of the faulty/erroneous outputs at different reduced t_W values using *solid* $0x0000$ data pattern, an analysis is performed to determine the cell types (i.e., noisy or persistent). We reduce the t_W value from $15ns$ (manufacturer's recommended) to $10ns$, $5ns$, and $2.5ns$, respectively. Due to the experimental set-up limitations, we are incapable of reducing the t_W value any further. At $t_W = 5ns$ and $10ns$, the obtained total failed bits are almost negligible ($< 5\%$ and $< 1\%$, respectively) for all ten chips. However, at $t_W = 2.5ns$, the total number of failed bit count falls within $25.59\% - 37.30\%$, which is sufficient for TRNG analysis. Hence, we choose $t_W = 2.5ns$ (considering the number of failed bit count) to characterize erroneous cells.

5.4.2 Characterization of Temporally Unbiased Cells

The MRAM cell characterization is performed according to section 5.3.3 with $N = 50$ measurements. Table 5.2 shows a summary of random MRAM addresses and cells after performing cell characterization. The results show that the total number of random cells obtained at reduced t_W ($2.5ns$) varies from chip to chip. The results also show that different memory modules may have different thresholds. We also observe that only a few addresses hold these random cells. In Table 5.2, the first row shows the cell selection threshold, Th_L, used for different chips. As we need to perform characterization so different thresholds for different models will be acceptable- the reason for choosing different thresholds to ensure higher throughput. However, the same threshold is used for the same model, i.e., C1 & C2 are from the same model. We can determine a unique threshold for simplicity for all models considering the highest threshold value (in our case, 23); however, at that point, we will get lower throughput as a lower threshold provides higher throughput (see section 5.4.5). The second row represents the percentage of addresses that contain random cell(s). Note that we only consider those addresses which have at least one cell that lies into $\mathcal{F}_C$. As the percentage of random addresses is very small ($\sim 1\%$) regardless of memory size, we will need to store only those few memory cells' information. Finally, the last row presents the average number of random bits per random addresses.

Table 5.2: Cell statistics after applying cell characterization algorithm.

Sample Chip[1]	C1	C2	C3	C4	C5
Threshold (Th_L)	16	16	15	23	21
#(Rand Addr) (%)	1.16	1.5	1.23	0.63	0.94
#(Rand Bits)/#(Rand Addr)	9.71	10.71	13.19	11.39	12.76

[1] C1&C2: MR0A16ACYS35, C3: MR0A16AYS35, C4: MR2A16ACYS35, C5: MR2A16AYS35

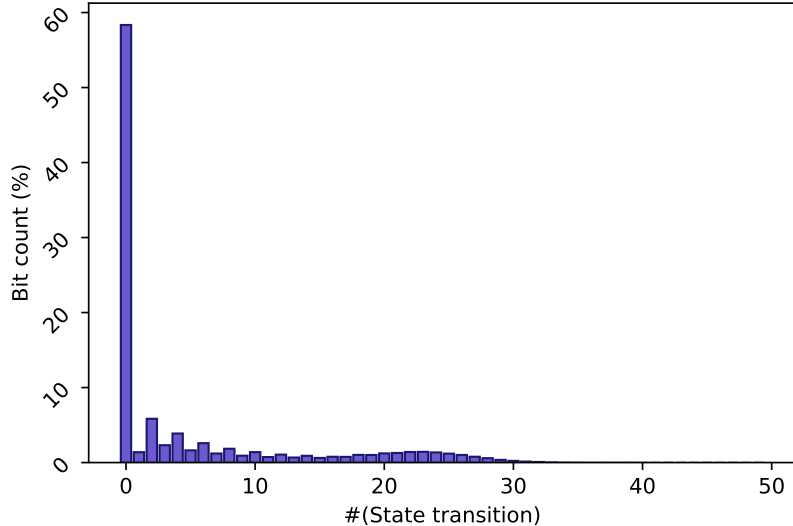


Figure 5.3: The characteristics of memory cells of *C4*: most of the cells are purely invariant (stuck at '0'/'1').

Fig. 5.3 illustrates the number of flips ($\mathcal{S}_C \rightarrow \mathcal{S}_E$ or $\mathcal{S}_E \rightarrow \mathcal{S}_C$) of memory cells with $N = 50$ temporal measurements for a randomly chosen memory chip. *Solid* $0x0000$ is used as the write data pattern. Note that to erase the trace of previously written data, we reset the entire memory with the *solid* $0xFFFF$ data pattern before every measurement. The results show that a larger number of cells ($\sim 40-60\%$) are purely invariant (stuck at '0'/'1'logic state), which is not desirable for high-entropy random number generator. To filter out those temporally persistent cells using the proposed algorithm (described in section 5.3.3), we only chose those cells (set $\mathcal{F}_C$) that are above the lower threshold range $Th_L = [15, 23]$ (see Table 5.2) for different chips used in the experiment. Although the number of eligible cells decreases after performing our proposed cell selection algorithm, the filtered cells are enough to generate high-quality random numbers.

Table 5.3: The worst-case NIST test results at the reduced t_W.

Sample Chip	C1		C2		C3		C4		C5	
Result Type	$\mathcal{P}-val.$	*Prop.*	$\mathcal{P}-val.$	*Prop.*	$\mathcal{P}-val.$	*Prop.*	$\mathcal{P}-val.$	*Prop.*	$\mathcal{P}-val.$	*Prop.*
Frequency	0.122325	14/14	0.637119	20/20	0.141256	21/21	0.534146	11/12	0.991468	20/20
BlockFrequency	0.350485	14/14	0.350485	20/20	0.980883	21/21	0.534146	12/12	0.834308	20/20
CumulativeSums	0.066882	14/14	0.275709	20/20	0.105618	21/21	0.350485	11/12	0.534146	20/20
Runs	0.213309	14/14	0.637119	20/20	0.311542	21/21	0.534146	12/12	0.437274	20/20
LongestRun	0.066882	14/14	0.534146	19/20	0.392456	21/21	0.739918	12/12	0.122325	20/20
Rank	0.534146	14/14	0.275709	19/20	0.875539	21/21	0.739918	12/12	0.534146	20/20
FFT	0.534146	14/14	0.275709	20/20	0.105618	21/21	0.350485	12/12	0.534146	19/20
NonOverlappingTemplate	0.017912	12/14	0.122325	18/20	0.311542	19/21	0.213309	11/12	0.350485	18/20
OverlappingTemplate	0.213309	14/14	0.275709	20/20	0.021262	21/21	0.534146	12/12	0.275709	20/20
Universal	0.017912	14/14	0.739918	20/20	0.311542	21/21	0.213309	12/12	0.637119	18/20
ApproximateEntropy	0.035174	14/14	0.637119	20/20	0.585209	21/21	0.213309	12/12	0.017912	20/20
RandomExcursions	0.991468	10/12	0.213309	11/12	0.534146	11/12	—-	7/7	0.006196	13/13
RandomExcursionsVariant	0.017912	11/12	0.017912	11/12	0.122325	11/12	—-	6/7	0.048716	12/13
Serial	0.350485	13/14	0.739918	19/20	0.105618	21/21	0.350485	12/12	0.437274	19/20
LinearComplexity	0.350485	14/14	0.534146	20/20	0.689019	21/21	0.739918	12/12	0.534146	20/20

*NB. —- test not performed due to insufficient data [FTSR21].

5.4.3 Evaluation

Data collected from different FPGAs verify that the memory controllers do not influence the randomness of the generated random number from MRAM chips. Furthermore, to evaluate the quality, randomness, and effectiveness of the obtained binary sequence, set $\mathcal{H}_C$ (after performing low-overhead post-processing technique, as discussed in section 5.3.4, over the test data sequence), the most frequently used and well-accepted NIST statistical test suite (STS) [FTSR21] is used. Tables 5.3 and 5.4 show the worst-case (i.e., from multiple similar test categories, the worst one is exhibited) NIST test results considering different memory models and extreme operating conditions. In the tables, the higher p-value ($\mathcal{P}-val.$) (calculated from the chi-squared (χ^2) test) indicates a purely random sequence and vice versa. Besides, $Prop.$ is the proportion of the binary sequence that passes the corresponding test. However, for passing the randomness test, the minimum value of $\mathcal{P}-val.$ should be 0.0001, and $Prop.$ needs to be greater than a specified value, which depends on the number of sample sizes (i.e., minimum of 18 tests need to be passed for 20 binary test sequences). The proposed MRAM-based binary sequences pass all (15) of the NIST tests; thus, it can be considered purely random.

5.4.4 Robustness Analysis

Chip Variations: The silicon results from four different memory models of two different sizes show that our proposed TRNG is robust. However, the statistics of random cells are different for different memory models, shown in Table 5.2. These sources of variations come from architectural as well as both inter- and intra-chip dissimilarities. As the random process variation is the key source of any memory chips' randomness, the proposed scheme can generate random numbers.

Table 5.4: The worst-case NIST test results verify the robustness of our proposed TRNG.

Sample Chip	C3						C4					
Operating Condition	$T_{High}(65°C)$		$T_{Low}(20°C)$		M-Field ($8mT$)		$T_{High}(65°C)$		$T_{Low}(20°C)$		M-Field ($8mT$)	
Result Type	$\mathcal{P}-val.$	$Prop.$	$\mathcal{P}-val.$	$Prop.$	$\mathcal{P}-val.$	$Prop.$	$\mathcal{P}-val.$	$Prop.$	$\mathcal{P}-val.$	$Prop.$	$\mathcal{P}-val.$	$Prop.$
Frequency	0.788728	21/21	0.585209	21/21	0.242986	21/21	0.739918	12/12	0.534146	12/12	0.122325	11/12
BlockFrequency	0.311542	20/21	0.585209	21/21	0.392456	21/21	0.534146	12/12	0.739918	11/12	0.066882	12/12
CumulativeSums	0.078086	21/21	0.105618	21/21	0.311542	21/21	0.213309	12/12	0.002043	12/12	0.017912	10/12
Runs	0.689019	21/21	0.875539	21/21	0.392456	21/21	0.122325	12/12	0.739918	12/12	0.213309	12/12
LongestRun	0.311542	21/21	0.242986	21/21	0.242986	20/21	0.213309	12/12	0.008879	12/12	0.066882	12/12
Rank	0.311542	21/21	0.186566	21/21	0.057146	21/21	0.122325	11/12	0.534146	12/12	0.534146	12/12
FFT	0.585209	21/21	0.392456	21/21	0.186566	21/21	0.911413	12/12	0.000439	11/12	0.122325	12/12
NonOverlappingTemplate	0.311542	19/21	0.186566	19/21	0.242986	19/21	0.008879	11/12	0.017912	11/12	0.122325	11/12
OverlappingTemplate	0.242986	21/21	0.392456	20/21	0.585209	21/21	0.350485	12/12	0.213309	12/12	0.008879	11/12
Universal	0.689019	20/21	0.875539	20/21	0.242986	21/21	0.350485	11/12	0.739918	12/12	0.534146	12/12
ApproximateEntropy	0.141256	20/21	0.141256	20/21	0.029796	21/21	0.534146	12/12	0.213309	11/12	0.911413	12/12
RandomExcursions	0.004301	14/14	0.066882	12/12	0.066882	11/12	—-	9/9	—-	6/7	—-	5/5
RandomExcursionsVariant	0.213309	13/14	0.066882	11/12	0.213309	11/12	—-	9/9	—-	6/7	—-	4/5
Serial	0.392456	21/21	0.392456	20/21	0.186566	21/21	0.350485	12/12	0.213309	11/12	0.350485	12/12
LinearComplexity	0.311542	20/21	0.057146	21/21	0.186566	21/21	0.534146	11/12	0.035174	12/12	0.213309	12/12

*NB. —- test not performed due to insufficient data [FTSR21].

Environmental Variations: Robustness against a wide range of operating conditions is one of the requirements of high-quality TRNGs. To verify the robustness of our proposed random number under temperature variation and external magnetic field (M-Field), we collected four sets of test data sequence at different operating conditions: i) room temperature ($26°C$), ii) high temperature ($65°C$), and iii) low temperature ($20°C$) without external M-Field. The fourth set is collected at room temperature with an $\sim 8mT$ external M-Field. The temperature range is chosen within the manufacturer's recommended value, which is $[0°C - 70°C]$ for all memory models. The total number of random cells is observed comparatively less for low temperatures than the other operating conditions. The write latency of MRAM increases significantly at the lower temperature, which results in the reduction of the number of random cells at the reduced write operation [Sla09]. Besides, we applied a constant rare earth magnetic source (generated from the permanent magnet) in six different orientations of 3D coordinates to observe the effect of external M-Field. However, we did not notice any significant change in the total number of random cells with ($8mT$) external M-Field. Therefore, we can conclude that our proposed random numbers are robust against extreme operating conditions. To further evaluate the robustness of the cell selection threshold, we deliberately select those two models (*C3* & *C4*) with the lowest and highest Th_L values (see Tables 5.2 and 5.4). Table 5.4 shows the worst-case NIST STS results in extreme operating conditions, signifying that our proposed TRNG can produce high-quality random numbers.

5.4.5 Throughput Analysis

Cell characterization and registration are performed once during a full life cycle. Therefore, the registration phase is not considered during throughput calculation.

The TRNG throughput of our proposed technique is the function of the average time required to perform read/write operation from one memory cell at the reduced t_W and the average time required to execute the SHA-256 hash function. The throughput of our proposed TRNG is calculated as follows:

$$\mathcal{T}_{avg,worst} = \frac{\mathcal{D}_{len}}{\boldsymbol{t}_{RW,avg} + \boldsymbol{t}_{hash,avg}} \tag{5.1}$$

where,

$$\boldsymbol{t}_{RW,avg} = \frac{\boldsymbol{t}_{RW} \times \mathcal{B}_{len}}{(\frac{\#RandBits}{\#RandAddr})} \tag{5.2}$$

Here, $\mathcal{B}_{len}(= 512\text{-}bit)$ and $\mathcal{D}_{len}(= 256\text{-}bit)$ are the length of the input and hashed output (message digest) block size of the hash function, respectively, $\boldsymbol{t}_{hash,avg}$ is the average time required to hash the input bit sequence of length $\mathcal{B}_{len}$, $\boldsymbol{t}_{RW,avg}$ is the average time required to generate raw random bits of size $\mathcal{B}_{len}$, $\boldsymbol{t}_{RW}$ is the average time required to perform a complete read/write operation from one memory address, and '*#(Rand Bits)/#(Rand Addr)*' (see Table 5.2) is the average number of random bits per random addresses. The cryptographic hash function, SHA-256, is used in this work due to the low-overhead post-processing. Nowadays, almost all modern processors have dedicated instruction set architecture to provide hardware support for performing the secure hash algorithm [GYG12]. We found $\boldsymbol{t}_{hash,avg} = 802.6ns$ using *Intel i7-8700* processor. Note that we use a high-level language (*Python API*) to hash the complete 512-bit block message; hence, the obtained average time ($802.6ns$) includes the function overhead. Ideally, MRAM has a comparatively fast ($35ns$) read/write cycle considering the nominal operation [Evea]. However, using our evaluation board, the obtained $\boldsymbol{t}_{RW}$ considering reduced write operation is $239.76ns$ (much higher than $70ns$), which signifies the inclusion of the communication overhead between the FPGA interfaced with a computer to acquire the data from memory for analysis. Furthermore, Table 5.2 shows the different '*#(Rand*

Bits)/#(Rand Addr)' values for different memory chips. According to Eq. 5.1, our system-level worst-case throughput values are around 18.17, 19.95, 24.12, 21.10, and 23.47$Mbit/s$ for $C1 - C5$ chips, respectively, considering all of the communication and function overhead. The obtained throughput values are significantly higher compared with the performance of many popular (non) volatile memory-based TRNGs [KPH+19, MDH+01, WYW+12]. An efficient implementation of a memory controller can further improve the overall performance of our proposed TRNG.

5.5 Discussion

Our proposed TRNG takes advantage of the metastable state of MRAM memory cell (i.e., cell resistance is halfway between high resistance state and low resistance state). At metastable state, the temporal variation of cell output largely depends on thermal noise. Unfortunately, the magnitude of the thermal noise is directly proportional to the temperature. Therefore, if the attacker has control over the temperature, he/she can introduce determinism into the TRNG output by lowering the temperature. To avoid the situation, we characterize MRAM cell metastability at a lower temperature. Our experimental result shows that metastable cells showing randomness at a lower temperature (i.e., with small thermal noise) also show randomness at a higher temperature (i.e., with higher thermal noise). Thus, for our proposed TRNG, we recommend selecting a subset of metastable cells which produce random sequences at the lowest possible operating temperature (worst possible case) and use them over the entire operating temperature range. Note that a small fluctuation in external voltage does not have any impact on our proposed TRNG, as all modern ICs are equipped with internal voltage controllers. This voltage con-

troller can keep the operating voltage constant if the external voltage is in the valid range.

5.6 Conclusion

This chapter demonstrated an efficient technique to generate high-throughput and high-quality true random numbers from non-volatile COTS MRAM chips by utilizing its internal write latency variation. The NIST SP-800-22 suite results validate that our proposed technique is purely random and robust at extreme operating conditions. The throughput is also considerably higher than most of the available TRNG techniques implemented using existing or emerging volatile or NVM chips.

CHAPTER 6

CONCLUSION AND FUTURE WORK

This dissertation introduces comprehensive frameworks to establish device security and trust using emerging memory technologies without requiring any hardware modification or new hardware component. In this research, we present three novel ideas, which include- (i) developing an anti-counterfeiting technique by watermarking ReRAM, (ii) introducing a data hiding technique in ReRAM, and (iii) demonstrating a high-speed true random number generator using MRAM. Although we have demonstrated the results with commercially available standalone MRAM and ReRAM chips, our proposed method should also be applicable for embedded ReRAM and MRAM as the underlying memory architecture is the same for embedded and standalone ReRAM and MRAM memory modules.

In our proposed watermarking technique, we repeatedly stress the ReRAM cell to make an irreversible change in its physical property (i.e., increases *set* and *reset* times). We have demonstrated that this irreversible change in ReRAM's physical property can be utilized to watermark devices with ReRAM. Towards this end, we propose an efficient protocol to verify the device's authenticity at the user's end using this watermarking technique. Unlike other existing anti-counterfeiting solutions, our proposed method does not require any- (i) new hardware or hardware modification, (ii) special lab facility or subject-matter expert, (iii) centralized database, and (iv) exhaustive communication between manufacturer and user. Additionally, our proposed anti-counterfeiting solution can identify a wide range of device counterfeiting, including recycled, remarked, cloned, overproduced, etc. Therefore, our proposed low-cost technique is well-suited to verify the device's authenticity before deploying them in a security sensitive application.

For hiding information in ReRAM, we induce a similar irreversible change in the physical property (as of watermarking technique) to imprint the hidden data. However, in contrast with watermarking, we perform additional characterization on ReRAM cells and deduce an algorithm so that the hidden data can be alive even after using the device for a long period of time. We also provide a simple framework to enhance the security of hidden data. Our proposed data hiding technique does not interfere with regular memory operations and is robust against external variations in operating conditions (i.e., temperature and voltage). We have also experimentally verified that hidden data with our proposed technique can be recovered even after 100K regular write cycle without introducing any bit error. Our proposed data hiding framework makes the hidden data untraceable through regular *read/write* operations. Therefore, it can be used as a very effective defense mechanism against ransom attacks and the second layer of protection on encrypted data against brute-force attacks.

Next, our work on MRAM-based TRNG entirely relies on the bit errors obtained in suboptimal write latency. The variations among obtained errors originated due to not receiving sufficient current and time during the write operation to toggle into the intended stable resistance state. Hence, appropriate reduced time identification is essential to use MRAM for different security solutions. The experimental result manifests that only a few memory cells, hence addresses are error-prone at a specific reduced write latency. To this end, memory cells are characterized and observed that a subset of erroneous memory cells can be used as a TRNG. As all noisy cells do not possess the same amount of entropy; therefore, a cell selection algorithm is proposed to filter the unbiased cells from the noise-prone cells with the help of an appropriate threshold to generate robust, high-quality TRNG.

In summary, this dissertation demonstrates three critical security solutions using commercially available MRAM and ReRAM chips, which can be evaluated in any embedded platform. Therefore, I hope that this dissertation will be beneficial and inspirational to emerging NVMs for secure computing with trust in mind.

Hereafter, we present some important focus areas of future direction of this research work —

- Our proposed data hiding technique demonstrated that stressing changes the physical property of ReRAM. Hence, an adversary may utilize this irreversible physical property of the memory cells to recover the deleted content using standard digital interfaces. This way, extensive analysis of the analog physical property (i.e., *write* time) of the memory cells in a discarded memory can reveal the original data. We expect that data retrieval from aged ReRAM will endanger sensitive information such as the firmware/software code, encryption keys, etc., and pose a severe threat to intellectual property protection. Therefore we will investigate that experimentally in our future work and propose a technique to ensure data remains unrecoverable after deletion.
- The latest generation ReRAM does not require any access transistor; thus, it supports a cross-point architecture, reduces cost, and theoretically achieves the smallest cell size of $4F^2$ [XNM+15, KAI+13]. Therefore, these ReRAMs use a bidirectional diode as a selector or even no selector at all, thus, requiring a specially designed peripheral circuit and memory organization. These new peripherals can also introduce attack vectors through which adversaries can leak sensitive data as well as modify data by either injecting faults or inducing Trojans. This mandates further research to detect potential security threats caused by emerging memories and proposes countermeasures against existing and emerging malicious attacks.

- Storing data in a magnetic state has several benefits over charge-based storage; however, the memory content is sensitive to external fields. Proper embedded magnetic shielding can ensure data integrity and guarantee reliable memory operation (even memory itself) against the external field, radiation effect, etc. Memory content can be tempered to introduce different attack methodologies, i.e., fault injection attack or stealing sensitive information through hampering data persistency by varying the alternating current (AC) magnetic field's frequency, magnitude, and shape. The impact will be more detrimental than the DC magnetic field effect and introduce read, write, and retention failure. In the future, we will explore MRAM's reliabilities and fundamental security vulnerabilities and propose techniques to overcome or mitigate the impact of the external AC magnetic field.

[ADS16] Dmytro Apalkov, Bernard Dieny, and Jon M Slaughter. Magnetoresistive random access memory. *Proceedings of the IEEE*, 104(10):1796–1830, 2016.

[ATO+15] Masashi Arita, Akihito Takahashi, Yuuki Ohno, Akitoshi Nakane, Atsushi Tsurumaki-Fukuchi, and Yasuo Takahashi. Switching operation and degradation of resistive random access memory composed of tungsten oxide and copper investigated using in-situ tem. In *2015 33rd IEEE International Conference on Computer Design (ICCD)*, volume 5, 2015.

[BB16] Abhishek Basak and Swarup Bhunia. P-val: Antifuse-based package-level defense against counterfeit ics. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 35(7):1067–1078, 2016.

[BOET14] Rajendra Bishnoi, Fabian Oboril, Mojtaba Ebrahimi, and Mehdi B Tahoori. Avoiding unnecessary write operations in stt-mram for low power implementation. In *Fifteenth International Symposium on Quality Electronic Design*, pages 548–553, 2014.

[Bor21] Mike Borza. Counterfeit chips 101: Protect your next design, nov 2021. 18 November, 2021.

[CCCM10] Abbas Cheddad, Joan Condell, Kevin Curran, and Paul Mc Kevitt. Digital image steganography: Survey and analysis of current methods. *Signal Processing*, 90(3):727–752, 2010.

[Che20] Yangyin Chen. ReRAM : History, status, and future. *IEEE Transactions on Electron Devices*, 67(4):1420–1433, 2020.

[Che21] Chih-Hung Chen. Thermal noise measurement and characterization for modern semiconductor devices. *IEEE Instrumentation & Measurement Magazine*, 24(2):60–71, 2021.

[EAB+05] B.N. Engel, J. Akerman, B. Butcher, R.W. Dave, M. DeHerrera, M. Durlam, G. Grynkewich, J. Janesky, S.V. Pietambaram, N.D. Rizzo, J.M. Slaughter, K. Smith, J.J. Sun, and S. Tehrani. A 4-mb toggle mram based on a novel bit and switching method. *IEEE Transactions on Magnetics*, 41(1):132–136, 2005.

[Evea] Everspin. Everspin 16-bit parallel interface mram. Available: `https://www.everspin.com/parallel-interface-mram`. Accessed: 22 April 2021.

[Eveb] Everspin. Spin-transfer torque mram technology. Available: `https://www.everspin.com/spin-transfer-torque-mram-technology`. Accessed: 22 April 2022.

[FBTR22] Farah Ferdaus, Bashir Mohammad Sabquat Bahar Talukder, and Md Tauhidur Rahman. Watermarked reram: A technique to prevent counterfeit memory chips. In *Proceedings of the 2022 on Great Lakes Symposium on VLSI*, GLSVLSI '22, New York, NY, USA, 2022. Association for Computing Machinery.

[FC18] DJ Forte and RS Chakraborty. Counterfeit integrated circuits: Threats, detection, and avoidance. In *Conference on Cryptographic Hardware and Embedded Systems*, 2018.

[FR21] Farah Ferdaus and Md Tauhidur Rahman. *Security of Emerging Memory Chips*, pages 357–390. Springer International Publishing, Cham, 2021.

[FTSR21] Farah Ferdaus, BMS Bahar Talukder, Mehdi Sadi, and Md Tauhidur Rahman. True random number generation using latency variations of commercial mram chips. In *2021 22nd International Symposium on Quality Electronic Design (ISQED)*, pages 510–515, 2021.

[Fuj19] Fujitsu Semiconductor Memory Solution. Reram (resistive random access memory), 2019. (Accessed: 15 Oct 2021).

[GCR+13] Bogdan Govoreanu, Sergiu Clima, Iuliana P Radu, Yang-Yin Chen, Dirk J Wouters, and Malgorzata Jurczak. Complementary role of field and temperature in triggering on/off switching mechanisms in Hf/HfO_2 resistive ram cells. *IEEE transactions on electron devices*, 60(8):2471–2478, 2013.

[GDT14] Ujjwal Guin, Daniel DiMase, and Mohammad Tehranipoor. A comprehensive framework for counterfeit defect coverage analysis and detection assessment. *Journal of Electronic Testing*, 30(1):25–40, 2014.

[GHD+14] Ujjwal Guin, Ke Huang, Daniel DiMase, John M Carulli, Mohammad Tehranipoor, and Yiorgos Makris. Counterfeit integrated circuits: A

rising threat in the global semiconductor supply chain. *Proceedings of the IEEE*, 102(8):1207–1228, 2014.

[GKST07] Jorge Guajardo, Sandeep S. Kumar, Geert-Jan Schrijen, and Pim Tuyls. Fpga intrinsic pufs and their use for ip protection. In Pascal Paillier and Ingrid Verbauwhede, editors, *Cryptographic Hardware and Embedded Systems - CHES 2007*, pages 63–80, Berlin, Heidelberg, 2007. Springer Berlin Heidelberg.

[GYG12] Jim Guilford, Kirk Yap, and Vinodh Gopal. Fast sha-256 implementations on intel architecture processors. *IA Architects*, 2012. Accessed: 05 Nov 2020.

[HST+12] Chien-Yuan Huang, Wen Chao Shen, Yuan-Heng Tseng, Ya-Chin King, and Chrong-Jung Lin. A contact-resistive random-access-memory-based true random number generator. *IEEE Electron Device Letters*, 33(8):1108–1110, 2012.

[Int] Intel. Intel optane memory series. Available: `https://ark.intel.com/content/www/us/en/ark/products/97544/intel-optane-memory-series-16gb-m-2-80mm-pcie-3-0-20nm-3d-xpoint.html`. Accessed: 7 Sep 2022.

[JAS+19] Pulkit Jain, Umut Arslan, Meenakshi Sekhar, Blake C. Lin, Liqiong Wei, Tanaya Sahu, Juan Alzate-vinasco, Ajay Vangapaty, Mesut Meterelliyoz, Nathan Strutt, Albert B. Chen, Patrick Hentges, Pedro A. Quintero, Chris Connor, Oleg Golonzka, Kevin Fischer, and Fatih Hamzaoglu. 13.2 a 3.6Mb 10.1Mb/mm^2 embedded non-volatile ReRAM macro in 22nm FinFET technology with adaptive forming/set/reset schemes yielding down to 0.5v with sensing time of 5ns at 0.7v. In *2019 IEEE International Solid- State Circuits Conference - (ISSCC)*, pages 212–214. IEEE, 2019.

[JJ98] Neil F. Johnson and Sushil Jajodia. Exploring steganography: Seeing the unseen. *Computer*, 31(2):26–34, 1998.

[KAI+13] Akifumi Kawahara, Ryotaro Azuma, Yuuichirou Ikeda, Ken Kawai, Yoshikazu Katoh, Yukio Hayakawa, Kiyotaka Tsuji, Shinichi Yoneda, Atsushi Himeno, Kazuhiko Shimakawa, Takeshi Takagi, Takumi Mikawa, and Kunitoshi Aono. An 8 mb multi-layered cross-point reram macro with 443 mb/s write throughput. *IEEE Journal of Solid-State Circuits*, 48(1):178–185, 2013.

[KCL+21] Mohammad Nasim Imtiaz Khan, Chak Yuen Cheng, Sung Hao Lin, Abdullah Ash-Saki, and Swaroop Ghosh. A morphable physically unclonable function and true random number generator using a commercial magnetic memory. *Journal of Low Power Electronics and Applications*, 11(1):5, 2021.

[KEC+11] Siew-Hwee Kwok, Yen-Ling Ee, Guanhan Chew, Kanghong Zheng, Khoongming Khoo, and Chik-How Tan. A comparison of post-processing techniques for biased random number generators. In *IFIP International Workshop on Information Security Theory and Practices*, pages 175–190. Springer, 2011.

[KIPP19] Ioannis Karageorgos, Mehmet M Isgenc, Samuel Pagliarini, and Larry Pileggi. Chip-to-chip authentication method based on sram puf and public key cryptography. In *Journal of Hardware and Systems Security*, volume 3, page 382–396, 2019.

[KLK17] Eunhwan Kim, Minah Lee, and Jae-Joon Kim. 8.2 8mb/s 28mb/mj robust true-random-number generator in 65nm cmos based on differential ring oscillator with feedback resistors. In *2017 IEEE International Solid-State Circuits Conference (ISSCC)*, pages 144–145. IEEE, 2017.

[KPH+19] Jeremie S Kim, Minesh Patel, Hasan Hassan, Lois Orosa, and Onur Mutlu. D-range: Using commodity dram devices to generate true random numbers with low latency and high throughput. In *2019 IEEE International Symposium on High Performance Computer Architecture (HPCA)*, pages 582–595. IEEE, 2019.

[KYK+08] T. Kishi, H. Yoda, T. Kai, T. Nagase, E. Kitagawa, M. Yoshikawa, K. Nishiyama, T. Daibou, M. Nagamine, M. Amano, S. Takahashi, M. Nakayama, N. Shimomura, H. Aikawa, S. Ikegawa, S. Yuasa, K. Yakushiji, H. Kubota, A. Fukushima, M. Oogane, T. Miyazaki, and K. Ando. Lower-current and fast switching of a perpendicular tmr for high speed and high density spin-transfer-torque mram. In *2008 IEEE International Electron Devices Meeting*, pages 1–4. IEEE, 2008.

[LBB+14] Juergen Lorenz, Eberhard Bär, Alex Burenkov, Peter Evanschitzky, Asen Asenov, Liping Wang, Xingsheng Wang, Andrew R Brown, Campbell Millar, and David Reid. Simultaneous simulation of systematic and stochastic process variations. In *2014 International Conference on Simulation of Semiconductor Processes and Devices (SISPAD)*, pages 289–292. IEEE, 2014.

[LHJ+19] Ethan S. Lee, Luis Hurtado, Jungwoo Joh, Srikanth Krishnan, Sameer Pendharkar, and Jesús A. Del Alamo. Time-dependent dielectric breakdown under ac stress in gan mis-hemts. In *2019 IEEE International Reliability Physics Symposium (IRPS)*, pages 1–5, 2019.

[LLW+12] David Loke, TH Lee, WJ Wang, LP Shi, R Zhao, YC Yeo, TC Chong, and SR Elliott. Breaking the speed limits of phase-change memory. *Science*, 336(6088):1566–1569, 2012.

[LNG+10] Jing Li, Patrick Ndai, Ashish Goel, Sayeef Salahuddin, and Kaushik Roy. Design paradigm for robust spin-torque transfer magnetic ram (stt mram) from circuit/architecture perspective. *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, 18(12):1710–1723, 2010.

[MCYC15] Manqing Mao, Yu Cao, Shimeng Yu, and Chaitali Chakrabarti. Optimizing latency, energy, and reliability of 1t1r ReRAM through appropriate voltage settings. In *2015 33rd IEEE International Conference on Computer Design (ICCD)*, pages 359–366, 2015.

[MDB+02] Jack A Mandelman, Robert H Dennard, Gary B Bronner, John K DeBrosse, Rama Divakaruni, Yujun Li, and Carl J Radens. Challenges and future directions for the scaling of dynamic random-access memory (dram). *IBM Journal of Research and Development*, 46(2.3):187–212, 2002.

[MDH+01] T. Mikolajick, C. Dehm, W. Hartner, I. Kasko, M.J. Kastner, N. Nagel, M. Moert, and C. Mazure. Feram technology for high density applications. *Microelectronics Reliability*, 41(7):947–950, 2001.

[MJS+16] Sanu K. Mathew, David Johnston, Sudhir Satpathy, Vikram Suresh, Paul Newman, Mark A. Anders, Himanshu Kaul, Amit Agarwal, Steven K. Hsu, Gregory Chen, and Ram K. Krishnamurthy. μ rng: A 300–950 mv, 323 gbps/w all-digital full-entropy true random number generator in 14 nm finfet cmos. *IEEE Journal of Solid-State Circuits*, 51(7):1695–1704, 2016.

[MLP08] Ankur M. Mehta, Steven Lanzisera, and Kristofer S. J. Pister. Steganography in 802.15.4 wireless communication. In *2008 2nd International Symposium on Advanced Networks and Telecommunication Systems*, pages 1–3, 2008.

[MP17] Savita Mohurle and Manisha Patil. A brief study of wannacry threat: Ransomware attack 2017. *International Journal of Advanced Research in Computer Science*, 8(5):1938–1940, 2017.

[PAK99] F.A.P. Petitcolas, R.J. Anderson, and M.G. Kuhn. Information hiding-a survey. *Proceedings of the IEEE*, 87(7):1062–1078, 1999.

[PFKW09] Paul R. Prucnal, Mable P. Fok, Konstantin Kravtsov, and Zhenxing Wang. Optical steganography for data hiding in optical networks. In *2009 16th International Conference on Digital Signal Processing*, pages 1–6, 2009.

[PH03] N. Provos and P. Honeyman. Hide and seek: an introduction to steganography. *IEEE Security Privacy*, 1(3):32–44, 2003.

[PWG+20] Yachuan Pang, Huaqiang Wu, Bin Gao, Bohan Lin, Jianshi Tang, Zhen Li, Shuguang Cui, and He Qian. A rram-based data hiding technique utilizing the impact of form condition on set performance. In *2020 IEEE International Memory Workshop (IMW)*, pages 1–4, 2020.

[Raj] Justin Rajewski. Alchitry au fpga development board. Available: `https://alchitry.com/products/alchitry-au-fpga-development-board`. Accessed: 22 April 2021.

[RT21] M Tauhidur Rahman and Bashir Mohammad Sabquat Bahar Talukder. Systems and methods for identifying counterfeit memory, October 5 2021. US Patent 11,139,043.

[RXF+14] Md Tauhidur Rahman, Kan Xiao, Domenic Forte, Xuhei Zhang, Jerry Shi, and Mohammad Tehranipoor. Ti-trng: Technology independent true random number generator. In *2014 51st ACM/EDAC/IEEE Design Automation Conference (DAC)*, pages 1–6. IEEE, 2014.

[RZZ+15] Jeyavijayan Rajendran, Huan Zhang, Chi Zhang, Garrett S Rose, Youngok Pino, Ozgur Sinanoglu, and Ramesh Karri. Fault analysis-based logic encryption. *IEEE Transactions on Computers*, 64(2):410–424, 2015.

[SD07] G Edward Suh and Srinivas Devadas. Physical unclonable functions for device authentication and secret key generation. In *2007 44th ACM/IEEE Design Automation Conference*, pages 9–14. IEEE, 2007.

[SDB11] Iain Sutherland, Gareth Davies, and Andrew Blyth. Malware and steganography in hard disk firmware. *Journal in computer virology*, 7(3):215–219, 2011.

[Sla09] JM Slaughter. Materials for magnetoresistive random access memory. *Annual Review of Materials Research*, 39:277–296, 2009.

[SM16] Krzysztof Szczypiorski and Wojciech Mazurczyk. Steganography in ieee 802.11 ofdm symbols. *Security and Communication Networks*, 9(2):118–129, 2016.

[Ste19] Michael Steer. *Microwave and RF design*. NC State University, 2019.

[TDK] TDK. Tmr sensors. Available: `https://product.tdk.com/en/techlibrary/productoverview/tmr-angle-sensors.html`. Accessed: 22 April 2021.

[TMR+20] BMS Bahar Talukder, Vineetha Menon, Biswajit Ray, Tempestt Neal, and Md Tauhidur Rahman. Towards the avoidance of counterfeit memory: Identifying the dram origin. In *2020 IEEE International Symposium on Hardware Oriented Security and Trust (HOST)*, pages 111–121, 2020.

[VDNP16] Elena Ioana Vatajelu, Giorgio Di Natale, and Paolo Prinetto. Security primitives (puf and trng) with stt-mram. In *2016 IEEE 34th VLSI Test Symposium (VTS)*, pages 1–4. IEEE, 2016.

[WDZ+16] Debao Wei, Libao Deng, Peng Zhang, Liyan Qiao, and Xiyuan Peng. Nrc: A nibble remapping coding strategy for nand flash reliability extension. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 35(11):1942–1946, 2016.

[WYGW12] Yi Wu, Shimeng Yu, Ximeng Guan, and H.-S. Philip Wong. Recent progress of resistive switching random access memory (rram). In *2012 IEEE Silicon Nanoelectronics Workshop (SNW)*, pages 1–4, 2012.

[WYW+12] Yinglei Wang, Wing-kei Yu, Shuo Wu, Greg Malysa, G Edward Suh, and Edwin C Kan. Flash memory for ubiquitous hardware security functions: True random number generation and device fingerprints. In *2012 IEEE Symposium on Security and Privacy*, pages 33–47. IEEE, 2012.

[WYX+13] Yinglei Wang, Wing-kei Yu, Sarah Q. Xu, Edwin Kan, and G. Edward Suh. Hiding information in flash memory. In *2013 IEEE Symposium on Security and Privacy*, pages 271–285, 2013.

[XNM+15] Cong Xu, Dimin Niu, Naveen Muralimanohar, Rajeev Balasubramonian, Tao Zhang, Shimeng Yu, and Yuan Xie. Overcoming the challenges of crossbar resistive memory architectures. In *2015 IEEE 21st international symposium on high performance computer architecture (HPCA)*, pages 476–488. IEEE, 2015.

[XRF+14] Kan Xiao, Md. Tauhidur Rahman, Domenic Forte, Yu Huang, Mei Su, and Mohammad Tehranipoor. Bit selection algorithm suitable for high-volume production of sram-puf. In *2014 IEEE International Symposium on Hardware-Oriented Security and Trust (HOST)*, pages 101–106, 2014.

[YAM+21] Binbin Yang, Daniel Arumí, Salvador Manich, Álvaro Gómez-Pau, Rosa Rodríguez-Montañés, Mireia Bargalló González, Francesca Campabadal, and Liang Fang. Serial rram cell for secure bit concealing. *Electronics*, 10(15), 2021.

[YDW+18] Kaiyuan Yang, Qing Dong, Zhehong Wang, Yi-Chun Shih, Yu-Der Chih, Jonathan Chang, David Blaauw, and Dennis Svlvester. A 28nm integrated true random number generator harvesting entropy from mram. In *2018 IEEE Symposium on VLSI Circuits*, pages 171–172. IEEE, 2018.

[YPL+09] Yu Chao Yang, Feng Pan, Qi Liu, Ming Liu, and Fei Zeng. Fully room-temperature-fabricated nonvolatile resistive memory for ultrafast and high-density memory application. *Nano Letters*, 9(4):1636–1643, 2009.

[ZZYZ09] Ping Zhou, Bo Zhao, Jun Yang, and Youtao Zhang. A durable and energy efficient main memory using phase change memory technology. In *Proceedings of the 36th Annual International Symposium on Computer Architecture*, ISCA '09, page 14–23, New York, NY, USA, 2009. Association for Computing Machinery.

VITA

FARAH FERDAUS

2015	B.Sc., Electrical and Electronic Engineering Bangladesh University of Engineering and Technology Dhaka, Bangladesh
2016 - 2017	Electrical Safety Engineer Stichting Bangladesh Accord Foundation Dhaka, Bangladesh
2018	M.Sc., Electrical and Computer Engineering University of New Hampshire Durham, New Hampshire, USA
2022	Ph.D., Electrical and Computer Engineering Florida International University Miami, Florida, USA

PUBLICATIONS AND PRESENTATIONS

[C1] Farah Ferdaus, B. M. S. Bahar Talukder, Mehdi Sadi, and Md Tauhidur Rahman. True Random Number Generation by Write Latency Variation Using Commercial MRAM chips. International Symposium on Quality Electronic Design (ISQED), 2021, pp. 510-515.

[C2] Farah Ferdaus, B. M. S. Bahar Talukder, and Md Tauhidur Rahman. Watermarked ReRAM: A Technique to Prevent Counterfeit Memory Chips. In Proceedings of the Great Lakes Symposium on VLSI (GLSVLSI), 2022, pp. 21-26.

[B1] Farah Ferdaus, and Md Tauhidur Rahman. Security of Emerging Memory Chips. Emerging Topics in Hardware Security, Springer, Cham, 2021, pp. 357-390.

[J1] Farah Ferdaus, B. M. S. Bahar Talukder, and Md Tauhidur Rahman. Approximate MRAM: High-performance and Power-efficient Computing with MRAM Chips for Error-tolerant Applications. IEEE Transactions on Computers, 2022.

[J2] B. M. S. Bahar Talukder, Farah Ferdaus, and Md Tauhidur Rahman. Memory-based PUFs are Vulnerable as Well: A Non-invasive Attack against SRAM PUFs. IEEE Transactions on Information Forensics and Security, vol. 16, pp. 4035-4049, Jul. 2021.

[J3] Md Imtiaz Rashid, Farah Ferdaus, BMS Bahar Talukder, Paul Henny, Aubrey N. Beal, and Md Tauhidur Rahman. True Random Number Generation Using Latency Variations of FRAM. IEEE Transactions on Very Large Scale Integration

(VLSI) Systems, vol. 29, no. 1, pp. 14-23, Jan. 2021.

[J4] B. M. S. Bahar Talukder, Farah Ferdaus, and Md Tauhidur Rahman. A Non-invasive Technique to Detect Authentic/Counterfeit SRAM Chips. arXiv e-prints, 2021. Available: https://arxiv.org/abs/2107.09199.

[P1] Farah Ferdaus, and Md Tauhidur Rahman. Opportunities of emerging memories in the computing system. CRA-WP Grad Cohort Workshop for Women, Apr. 2021.

[P2] Farah Ferdaus, B. M. S. Bahar Talukder, and Md Tauhidur Rahman. Attesting SRAM manufacturer: Toward the avoidance of counterfeit SRAM. The Third Workshop for Women in Hardware and Systems Security (WISE), May 2019.