Watermarked ReRAM: A Technique to Prevent Counterfeit Memory Chips

Farah Ferdaus, Bashir Mohammad Sabquat Bahar Talukder, and Md Tauhidur Rahman
ECE Department, Florida International University
Miami, USA
{fferd006,bbaha007,mdtrahma}@fiu.edu

ABSTRACT

Electronic counterfeiting is a longstanding problem with adverse long-term effects for many sectors, remaining on the rise. This article presents a novel low-cost technique to embed watermarking in devices with resistive-RAM (ReRAM) by manipulating its analog physical characteristics through switching (*set/reset*) operation to prevent counterfeiting. 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. Experimental results show that our proposed ReRAM watermarking is robust against temperature variation and acceptably fast with $\sim 0.6 bit/min$ of imprinting and $\sim 15.625 bits/s$ of retrieval rates.

CCS CONCEPTS

• **Hardware → Memory and dense storage**; • **Security and privacy → Security in hardware**.

KEYWORDS

Watermarking, ReRAM, Counterfeiting, Supply-chain Security

ACM Reference Format:
Farah Ferdaus, Bashir Mohammad Sabquat Bahar Talukder, and Md Tauhidur Rahman. 2022. Watermarked ReRAM: A Technique to Prevent Counterfeit Memory Chips. In *Proceedings of the Great Lakes Symposium on VLSI 2022 (GLSVLSI '22), June 6–8, 2022, Irvine, CA, USA.* ACM, New York, NY, USA, 6 pages. https://doi.org/10.1145/3526241.3530341

1 INTRODUCTION

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 [2, 5, 6, 9, 12, 14, 15]. Recent studies show that memory and memory integrated ICs (microprocessors, programmable logic devices, etc.) consist of ~50% of the total counterfeit market share [6]. Most counterfeit memory chips suffer from sub-standard quality, poor performance, and shorter lifespan, severely affecting the security and reliability domains [6, 9, 17]. 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, physical inspection-based test, burn-in test, and electrical test [2, 6, 9, 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 [6, 9, 10]. 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 [3].

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, crossbar structure feasibility, excellent reliability at high temperature, high endurance compared to other traditional storage memories. [7, 18, 19]. 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.

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 imprints the watermark by repeatedly stressing the memory cells by alternatively writing '1' and '0'. Repeated stressing through switching operation ('1' → '0' or '0' → '1') gradually decreases the *HRS* resistance, degrading the memory performance and eventually causing endurance failure [1, 13]. 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 imprinting 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. Our proposed technique is irreversible as the impact of cell stressing is immutable. Hence, the imprinted watermark cannot be tampered. Additionally, our proposed technique does not require any hardware modification and can be directly deployed into available commercial products. Furthermore, the embedded watermark is robust against temperature variation as ReRAM is

GLSVLSI '22, June 6–8, 2022, Irvine, CA, USA

ACM ISBN 978-1-4503-9322-5/22/06…$15.00
https://doi.org/10.1145/3526241.3530341

inherently insensitive to temperature [8]. Moreover, our proposed method can be evaluated using standard ReRAM *read/write* operation and only costs ~2% of the total endurance of ReRAM cells. 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 paper is organized as follows. Sec. 2 briefly overviews the ReRAM memory preliminaries. Sec. 3 presents the proposed watermark imprinting and extracting mechanism, including the method for characterization of changes in ReRAM *write* time caused by stress. Sec. 4 explains the experimental setup and exhibits obtained results. Finally, Sec. 5 concludes our work.

2 RERAM PRELIMINARIES

Resistive switching phenomena in a dielectric material is the core mechanism of ReRAM to store logic states [4, 13]. The capacitor-like ReRAM bit cell structure consists of two electrodes ($Electrode_{Top}$ and $Electrode_{Bottom}$) separated by a metal oxide resistive switch material (Fig. 1). 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 [4, 13]. However, different materials result in different device characteristics such as endurance, retention, and scalability [4, 13]. 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 between *HRS*'s resistance to LRS's is required to be large enough to ensure robust *read/write* operation [13]. 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 bit cell, and the measured resistance (by sensing current) determines the stored logic state.

Each switching operation (i.e., changing state from *LRS* to *HRS* or *HRS* to *LRS*) on ReRAM gradually decreases the resistance of *HRS*, wearing-out the device [1]. 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

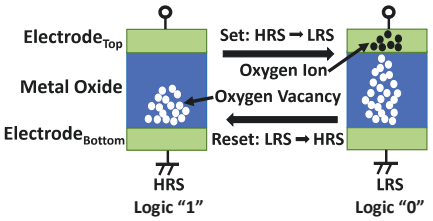

Figure 1: ReRAM cell structure with two logic states [13].

the resistance ratio of *HRS*/*LRS* [1, 13]. To maintain the desired resistance ratio of *HRS*/*LRS*, *set* and *reset* times must be increased for stressed memory cells[1]. In this work, we use this property to distinguish between the fresh and stressed ReRAM cells.

3 PROPOSED WATERMARKING TECHNIQUE

The flowchart in Fig. 2 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. 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.

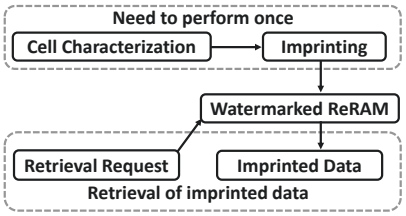

Figure 2: Steps used for ReRAM watermarking.

3.1 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 cells 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 separate the stressed cell

[1]The ReRAM internal control circuit maintains appropriate *set/reset* time by initiating write-verify-write operation sequence [11].

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

Data: $\mathcal{N}_{\mathcal{M}}$: Max rewrite operations (data endurance)
$\mathcal{A}_{\mathcal{S}}$: Set of memory addresses targeted to stress
w_L: Word length
$\mathcal{D}$: Data vector of length w_L, intended to write in target memory cells belong to $\mathcal{A}_{\mathcal{S}}$
t: Timer
Result: $\mathcal{S}_{\mathcal{T}}$: *Set* time of memory cells belongs to $\mathcal{A}_{\mathcal{S}}$
$\mathcal{R}_{\mathcal{T}}$: *Reset* time of memory cells belongs to $\mathcal{A}_{\mathcal{S}}$

// Initialization
1 $\mathcal{S}_{\mathcal{T}} = \{\}$; $\mathcal{R}_{\mathcal{T}} = \{\}$; $\mathcal{D} = Ones(1 \times w_L)$;
2 **foreach** $a \in \mathcal{A}_{\mathcal{S}}$ **do**
3 $\quad write(a, \mathcal{D})$;
4 **end**
// Stressing memory cells
5 **for** $i = 0$ *to* $\mathcal{N}_{\mathcal{M}}$ **do**
6 $\quad$ **foreach** $a \in \mathcal{A}_{\mathcal{S}}$ **do**
7 $\quad\quad \mathcal{D} = Zeros(1 \times w_L)$;
8 $\quad\quad tic = t$;
9 $\quad\quad write(a, \mathcal{D})$; // *Set* operation
10 $\quad\quad toc = t - tic$;
11 $\quad\quad \mathcal{S}_{\mathcal{T}} = \mathcal{S}_{\mathcal{T}} \cup \{toc\}$; // Accumulating *set* time
12 $\quad\quad \mathcal{D} = Ones(1 \times w_L)$;
13 $\quad\quad tic = t$;
14 $\quad\quad write(a, \mathcal{D})$; // *Reset* operation
15 $\quad\quad toc = t - tic$;
16 $\quad\quad \mathcal{R}_{\mathcal{T}} = \mathcal{R}_{\mathcal{T}} \cup \{toc\}$; // Accumulating *reset* time
17 $\quad$ **end**
18 **end**

from the fresh cell reliably. 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.

Note that, according to our observation, the relation between switching characteristics (i.e., *set/reset* time vs. stress count[2]) is almost uniform for all memory 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.

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

3.2 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 [16]. The watermark may include standard device ID, chip-specific unique ID, and other manufacturing-related information [16]. 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 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 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. 3.1). 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.

3.3 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 2 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.

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 RESULTS AND DISCUSSION

4.1 Evaluation Setup and Analysis

The analysis is performed over five *MB85AS8MT*[3] (40nm technology node) 8-bit serial peripheral interfaced (SPI) 8*Mb* memory chips

[3]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

Algorithm 2: 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
$\mathcal{D}$: Data vector of length w_L, intended to write in target memory cells belong to $\mathcal{A}_\mathcal{W}$
t: Timer
Result: $\mathcal{S}_\mathcal{T}$: *Set* time of memory cells belongs to $\mathcal{A}_\mathcal{W}$
$\mathcal{R}_\mathcal{T}$: *Reset* time of memory cells belongs to $\mathcal{A}_\mathcal{W}$

// Initialization
$\mathcal{S}_\mathcal{T} = \{\}$; $\mathcal{R}_\mathcal{T} = \{\}$; $\mathcal{D} = Ones(1 \times w_L)$;
foreach $a \in \mathcal{A}_\mathcal{W}$ **do**
  $write(a, \mathcal{D})$;
end
// Imprinting watermark
for $i = 0$ *to* $\mathcal{N}$ **do**
  foreach $a \in \mathcal{A}_\mathcal{W}$ **do**
    if *wMark[Bit]==1* **then**
      $\mathcal{D} = Zeros(1 \times wS)$;
      $write(a, \mathcal{D})$;
      $\mathcal{D} = Ones(1 \times wS)$;
      $write(a, \mathcal{D})$;
    end
  end
end
// Extracting watermark
foreach $a \in \mathcal{A}_\mathcal{W}$ **do**
  $\mathcal{D} = Zeros(1 \times w_L)$;
  $tic = t$;
  $write(a, \mathcal{D})$; // *Set* operation
  $toc = t - tic$; // Accumulating *Set* time
  $\mathcal{S}_\mathcal{T} = \mathcal{S}_\mathcal{T} \cup \{toc\}$;
  $\mathcal{D} = Ones(1 \times w_L)$;
  $tic = t$;
  $write(a, \mathcal{D})$; // *Reset* operation
  $toc = t - tic$; // Accumulating *Reset* time
  $\mathcal{R}_\mathcal{T} = \mathcal{R}_\mathcal{T} \cup \{toc\}$;
end

manufactured by Fujitsu Semiconductor Limited. We have used our own custom-designed memory controller implemented on *Teensy 4.1* microcontroller development board. 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 imprint a single bit of data. However, the measured *set/reset* time might vary due to the external and internal noise. Therefore, we imprint a single bit data into 256 consecutive addresses of the ReRAM to suppress the impact of noise. During evaluation, we have measured *set/reset* time for each address and computed the average. From now on to the rest of the paper, 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 and hence, reduces overall stressing time. Although the figures (except Fig. 5) we present in this section are based on one ReRAM chip (randomly chosen from five test chips), the observation is valid for all test chips. Additionally, the Fig. 5 summarizes the result from all five test chips.

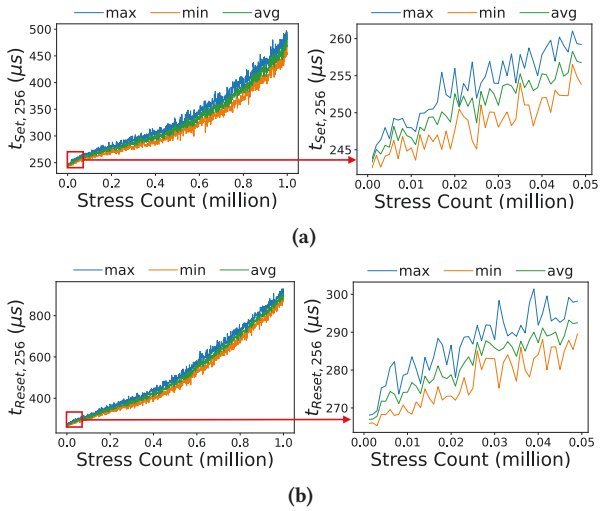

Figure 3: ReRAM cell characterization under stress- (a) $t_{Set,256}$ and (b) $t_{Reset,256}$.

Fig. 3 shows the switching characteristics (*set/reset* time vs. the stress counts) of the ReRAM chips at 25°C. This figure represents the maximum, minimum, and average of $t_{Set,256}$ (Fig. 3a) and $t_{Reset,256}$ (Fig. 3b) as a function of different stress levels (up to maximum possible rewrite operations[4]) over the 2K random address-space. Fig. 3 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. 3a, and 3b 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 ~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 cells and stressed cells with ~12K *set/reset* operations. Although Fig. 3 is constructed with 2K memory addresses, a similar characteristic is valid for the whole address space.

Next, the following steps are performed to verify the feasibility of the proposed watermarking. We have imprinted an arbitrarily chosen 32-bit random data into (256×32) = 8192 memory addresses

much slower than the *MB85AS8MT*. If the reviewers want, we will present data for *MB85AS4MT* chips as well.

[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.

varying the number of switching cycles, $\mathcal{N}$, up to $20K$ times to experimentally demonstrate the watermark imprinting (discussed in Sec. 3.2) and retrieval (discussed in Sec. 3.3) process.

Fig. 4 represents the experimental data from arbitrarily chosen test chips with imprinted data 0xC2F740EB[5]. 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 shows that logic '1' and logic '0' begin to separate at $5K$ stress count (Fig. 4a), and they become well-separated at $10K$ stress count (Fig. 4b). With further stress, the separation between logic '1' and logic '0' further increases (Fig. 4c). Similarly, with $t_{Reset,256}$, logic '1' and logic '0' begin to separate at $10K$ stress count (Fig. 4d) and become well-separated at $15K$ stress count (Fig. 4d). 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.

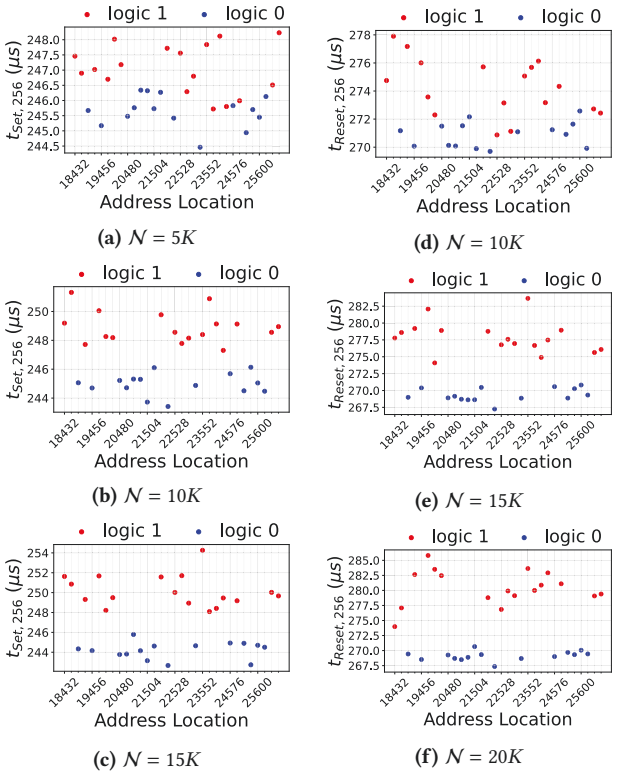

Figure 4: Imprinted data at different stress count- (a)–(c) $t_{Set,256}$ at stress count $5K$, $10K$, and $15K$; (d)–(f) $t_{Reset,256}$ at stress count $10K$, $15K$, and $20K$.

Fig. 5 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. 5 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. 4a), 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 $\left(d(b_0^i, b_1^j)\right) > 0$).

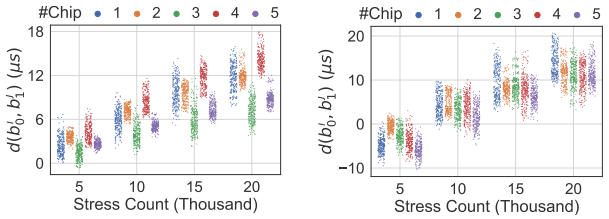

Figure 5: Verifying watermark in test chips, using- (a) $t_{Set,256}$, and (b) $t_{Reset,256}$.

4.2 Robustness Analysis

The watermark should 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. 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 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 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 watermark information is not affected by temperature and remains well-separated (Fig. 6) 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 of *HRS*/*LRS* is relatively temperature insensitive [8]. Note that, ReRAM chips that we have used in our experiment are rated to operate up to $85°C$.

4.3 Performance Analysis

4.3.1 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: $\mathcal{T}_{switch_{pair}} = (5ms + 5ms) = 10ms$,

[5]Also verified for other random data.

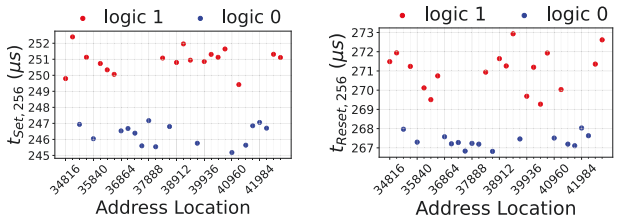

Figure 6: Robustness analysis after high-temperature baking ($80°C$) with- (a) $t_{Set,256}$ (b) $t_{Set,256}$

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[6].

4.3.2 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 $\frac{\mathcal{B}_{WMark}}{\mathcal{T}_{retrieve}} = \frac{32bits}{250\mu s \times 32 \times 256} = 15.625bits/s$.

4.3.3 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

5 CONCLUSION

This paper 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 against temperature variation and does not require any hardware modifications.

6 ACKNOWLEDGMENTS

This work was supported by the National Science Foundation under Grant Number DGE-2114200. We would also like to thank Mr. Tomohiro Kawakubo of Fujitsu Semiconductor Limited for sharing the necessary ReRAM chip information.

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

REFERENCES

[1] Masashi Arita et al. 2015. 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)*, Vol. 5. https://doi.org/10.1038/srep17103

[2] Abhishek Basak and Swarup Bhunia. 2016. P-Val: Antifuse-Based Package-Level Defense Against Counterfeit ICs. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* 35, 7 (2016), 1067–1078. https://doi.org/10.1109/TCAD.2015.2501311

[3] Mike Borza. 2021. Counterfeit Chips 101: Protect Your Next Design. https://blogs.synopsys.com/from-silicon-to-software/2021/11/02/what-are-counterfeit-chips/ 18 November, 2021.

[4] Yangyin Chen. 2020. ReRAM : History, Status, and Future. *IEEE Transactions on Electron Devices* 67, 4 (2020), 1420–1433. https://doi.org/10.1109/TED.2019.2961505

[5] Gustavo K Contreras, Md Tauhidur Rahman, and Mohammad Tehranipoor. 2013. Secure split-test for preventing IC piracy by untrusted foundry and assembly. In *2013 IEEE International symposium on defect and fault tolerance in VLSI and nanotechnology systems (DFTS)*. IEEE, 196–203.

[6] DJ Forte and RS Chakraborty. 2018. Counterfeit Integrated Circuits: Threats, Detection, and Avoidance. In *Conference on Cryptographic Hardware and Embedded Systems*.

[7] Fujitsu Semiconductor Memory Solution. 2019. ReRAM (Resistive Random Access Memory). Retrieved 15 October, 2021 from https://www.fujitsu.com/jp/group/fsm/en/products/reram/

[8] Bogdan Govoreanu et al. 2013. 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 (2013), 2471–2478. https://doi.org/10.1109/TED.2013.2266357

[9] Ujjwal Guin et al. 2014. Counterfeit Integrated Circuits: A Rising Threat in the Global Semiconductor Supply Chain. *Proc. IEEE* 102, 8 (2014), 1207–1228. https://doi.org/10.1109/JPROC.2014.2332291

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

[11] Pulkit Jain et al. 2019. 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)*. 212–214. https://doi.org/10.1109/ISSCC.2019.8662393

[12] Ioannis Karageorgos, Mehmet M Isgenc, Samuel Pagliarini, and Larry Pileggi. 2019. Chip-to-Chip Authentication Method Based on SRAM PUF and Public Key Cryptography. In *Journal of Hardware and Systems Security*, Vol. 3. 382–396. https://doi.org/10.1007/s41635-019-00080-y

[13] Manqing Mao, Yu Cao, Shimeng Yu, and Chaitali Chakrabarti. 2015. Optimizing latency, energy, and reliability of 1T1R ReRAM through appropriate voltage settings. In *2015 33rd IEEE International Conference on Computer Design (ICCD)*. 359–366. https://doi.org/10.1109/ICCD.2015.7357125

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

[15] Jeyavijayan Rajendran et al. 2015. Fault Analysis-Based Logic Encryption. *IEEE Trans. Comput.* 64, 2 (2015), 410–424. https://doi.org/10.1109/TC.2013.193

[16] Sadman Sakib, Aleksandar Milenković, and Biswajit Ray. 2020. Flash Watermark: An Anticounterfeiting Technique for NAND Flash Memories. *IEEE Transactions on Electron Devices* 67, 10 (2020), 4172–4177. https://doi.org/10.1109/TED.2020.3015451

[17] BMS Bahar Talukder et al. 2020. Towards the avoidance of counterfeit memory: Identifying the DRAM origin. In *2020 IEEE International Symposium on Hardware Oriented Security and Trust (HOST)*. IEEE, 111–121.

[18] Yi Wu et al. 2012. Recent progress of resistive switching random access memory (RRAM). In *2012 IEEE Silicon Nanoelectronics Workshop (SNW)*. 1–4. https://doi.org/10.1109/SNW.2012.6243331

[19] Yu Chao Yang et al. 2009. Fully Room-Temperature-Fabricated Nonvolatile Resistive Memory for Ultrafast and High-Density Memory Application. *Nano Letters* 9, 4 (2009), 1636–1643. https://doi.org/10.1021/nl900006g