Hiding Information for Secure and Covert Data Storage in Commercial ReRAM Chips

Farah Ferdaus, *Member, IEEE,* B. M. S. Bahar Talukder, *Member, IEEE,* and Md Tauhidur Rahman, *Senior Member, IEEE*

***Abstract*—This article introduces a novel, low-cost technique for hiding data in commercially available resistive-RAM (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 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. The silicon results show that our proposed data hiding technique is acceptably fast with $\sim 0.12 bit/s$ of encoding and $\sim 3.26 Kbits/s$ of retrieval rates, and the hidden message is unrecoverable without the knowledge of the secret key, which is used to enhance the security of hidden information.**

***Index Terms*—Privacy, Steganography, Watermarking.**

I. Introduction

THE most widely used standalone data storage media, Flash, faces enormous performance, scaling, reliability, retention, and cycling challenges in small process nodes. Furthermore, the NAND Flash program and erase operation is slow and can not be performed in small granularity. Therefore, classical security solutions that are elegant and reliable consume a significant amount of energy and can be vulnerable to physical attacks [1]–[3]. However, emerging technology-based devices can provide new and robust security primitives and potentially stronger protocols than conventional, complementary metal oxide semiconductor (CMOS) device-based security solutions [4], [5]. These emerging devices have inherent process variations and exhibit nonlinear input-output relationships. To this end, the most promising emerging resistive RAM (ReRAM) technology is gaining attention due to its erase-free simpler architecture, crossbar structure feasibility, fast write performance, lower read latency, more straightforward storage controller design, excellent reliability at high temperature, high endurance, high capacity, high scalability, high performance, ultra-low energy consumption, CMOS compatibility, reduced background memory operations, and reliable storage capability [6], [7], which make them the most viable alternative compared to other traditional storage memories [6], [7], hence, an ideal candidate to integrate into low-power applications.

The authors are with the Department of Electrical and Computer Engineering, Florida International University, Miami, FL 33174 USA (e-mail: fferd006@fiu.edu; bbaha007@fiu.edu; mdtrahma@fiu.edu).

This article demonstrates a technique to embed secret information in a concealed manner in ReRAM cells by leveraging its analog characteristics so that confidential information remains invisible from the normal memory content viewpoint. The advantage of the proposed information hiding technique on commercial off-the-shelf (COTS) ReRAM is — (i) the hidden data can not be retrieved due to watermarking if the storage device is lost or stolen [8], 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 alternatively writing '1' and '0'. 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 [9], [10]. 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* 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 with. 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 [11]. Furthermore, our proposed method can be evaluated using standard ReRAM *read/write* operation and only costs $\sim 3\%$ of the total endurance of ReRAM cells.

Note that our prior work [8] focuses only on watermarking, a traditional application of information hiding, to prevent piracy. Compared to [8], this version presents the information concealing technique, enabling several interesting applications other than watermarking, such as secure and covert data storage, data integrity, and covert communication [1]. The key contributions of this article 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 from ReRAM *write* time.

- We demonstrate our proposed technique's robustness, performance, retention characteristics, normal memory usage tolerance, and security analysis in multiple COTS ReRAMs.

The rest of the paper is organized as follows. Sect. II presents the motivation of our work. Sect. III briefly overviews the organization and operating principle of ReRAMs. Sect. IV presents the proposed information hiding and extracting algorithms. Sect. V to Sect. VII explain the experimental setup and demonstrate our proposed technique's effectiveness and security perspective. Sect. VIII discusses the future research direction. Finally, Sect. IX concludes the article.

II. Motivation

Due to the rapid evolution in digital multimedia and information technology, hiding information within digital content, e.g., documents, videos, audio, images, text, etc., is gaining significant attention to protect content and intellectual property [12]–[16]. 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 its content. In [17], 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. Moreover, the natural aging process introduces significant alterations in the analog domain that change the power-on state of the Static RAM (SRAM) cells, which can also be used for message hiding [18]. Furthermore, exploiting the overprovisioning of solid-state drives (SSDs) and Flash Translation Layer (FTL) can be a medium to hide information in the physical layer of an SSD NAND flash memory [19]. Introducing multiple security levels can avoid sensitive hidden data overwriting while handling data in FTL [18]. However, detecting, copying, and rewriting the hidden data at a lower security level is possible in the worst case [20]. 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. Extracting hidden information from physical properties requires complex and time-intensive measurement and analysis, rendering detection, replication, and erasure challenging for potential attackers. Therefore, the key benefits of our proposed technique over digital steganography are as follows.

- Our proposed covert technique does not impact 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, making it less feasible to brute-force attack.

Besides, other steganographic methods modify the physical layer protocol to hide data in optical and wireless transmission noise instead of encoding in digital objects [21]–[23]. These techniques require either special tools or modifications to existing protocols. In contrast, our proposed technique is analogous to steganography, where a physical object or digital information hides a confidential message [13], [16], [24]. 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. Furthermore, our easily applicable scheme can be implemented using a software program in a low-level memory interface.

III. ReRAM Preliminaries

Resistive switching phenomena in a dielectric material is the core mechanism of ReRAM to store logic states [10], [25]. The capacitor-like ReRAM cell structure consists of two electrodes ($Electrode_{Top}$ and $Electrode_{Bottom}$) separated by a metal oxide resistive switch material (Fig. 1). Whenever a voltage is applied to the $Electrode_{Top}$, the metal oxide breakdown process is initiated, producing 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 must be large enough to ensure robust *read/write* operation [10]. 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.

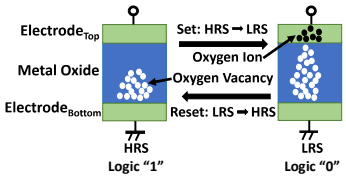


Fig. 1: ReRAM cell structure with two logic states [10].

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 [9]. Hence, fresh

memory cells possess distinctly different analog properties from 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 ($\frac{R_{HRS}}{R_{LRS}}$) [9], [10]. To maintain the desired resistance ratio, *set* and *reset* times must be increased for stressed memory cells[1]. This work uses this property to distinguish between the fresh and stressed ReRAM cells.

IV. 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.

A. 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, it's required to keep the stress level as little as possible so that fresh and stressed cells are reliably separable with *set/reset* time.

To this extent, we examine the ReRAM cell characteristics and the impact of switching operations on *set/reset* timing. This allows us to determine the minimum number of switching operations required to separate the stressed and fresh cells reliably. It also builds a relationship between ReRAM switching time and corresponding stressing level. We write all '1' data patterns to selected memory addresses. Then, all '0' and all '1' data patterns are written alternatively to those addresses. The switching times are captured and stored accordingly as *set/reset* times. 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.

B. Information hiding Technique

Our proposed technique conceals information in the *write* (*set/reset*) time of ReRAM bit cells. The *set* and *reset* time increases 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 [8]. We assume that an authorized entity is responsible for encoding 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 1), and (ii) repeated switching (*set* and *reset*) operations are performed (line 5 through line 14 of Algorithm 1) 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

Algorithm 1: 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 target memory addresses.
w_L: Word Length.
$\mathcal{S}_{msg}$: Message to be hidden in address $\mathcal{A}_{\mathcal{M}}$.
$\mathcal{D}$: $(1 \times w_L)$ vector containing data intended to write each memory address.
t: Timer.

Result: $\mathcal{S}_{\mathcal{T}}$, $\mathcal{R}_{\mathcal{T}}$: *Set* and *reset* time of all memory addresses $\in \mathcal{A}_{\mathcal{M}}$.

// Initialization
1 $\mathcal{S}_{\mathcal{T}} = \{\};\ \mathcal{R}_{\mathcal{T}} = \{\};\ \mathcal{D} = Ones(1 \times w_L);$
2 **foreach** $a \in \mathcal{A}_{\mathcal{M}}$ **do**
3 $\quad write(a, \mathcal{D});$
4 **end**
5 **for** $i = 0$ *to* $\mathcal{N}$ **do** // Encoding secret message
6 $\quad$ **foreach** $a \in \mathcal{A}_{\mathcal{M}}$ **do**
7 $\quad\quad$ **if** $\mathcal{S}_{msg}$*[Bit]==1* **then**
8 $\quad\quad\quad \mathcal{D} = Zeros(1 \times wL);$
9 $\quad\quad\quad write(a, \mathcal{D});$
10 $\quad\quad\quad \mathcal{D} = Ones(1 \times wL);$
11 $\quad\quad\quad write(a, \mathcal{D});$
12 $\quad\quad$ **end**
13 $\quad$ **end**
14 **end**
15 **foreach** $a \in \mathcal{A}_{\mathcal{M}}$ **do** // Decoding secret message
16 $\quad \mathcal{D} = Zeros(1 \times w_L);$
17 $\quad tic = t;$
18 $\quad write(a, \mathcal{D});$ // *Set* operation
19 $\quad toc = t - tic;$
20 $\quad \mathcal{S}_{\mathcal{T}} = \mathcal{S}_{\mathcal{T}} \cup \{toc\};$ // Accumulating *set* time
21 $\quad \mathcal{D} = Ones(1 \times w_L);$
22 $\quad tic = t;$
23 $\quad write(a, \mathcal{D});$ // *Reset* operation
24 $\quad toc = t - tic;$
25 $\quad \mathcal{R}_{\mathcal{T}} = \mathcal{R}_{\mathcal{T}} \cup \{toc\};$ // Accumulating *reset* time
26 **end**

[1]The ReRAM internal control circuit maintains appropriate *set/reset* time by initiating the write-verify-write operation sequence [8].

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

between fresh and stressed memory cells. Each switching operation gradually degrades the resistance of HRS, which is permanent and, thus, cannot be reversed. However, the number of repeated switching cycles, $\mathcal{N}$, used to encode the secret message is determined empirically through the cell characterization phase to ensure proper separation without causing excessive stress [8]. 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 $\mathcal{N} = 15K$ is sufficient to hide information securely.

C. 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. The line 15 to 26 of Algorithm 1 outlines the required steps for extracting the *set* and *reset* times from the addresses containing concealed information. In line 20, we accumulate all the *set/reset* times in a single 'bag' (or set) from the addresses containing hidden information. Therefore, if the imprinted value is 32-bit long, we will have 32 individual *set/reset* times in the bag. 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. Based on the threshold value of *set/reset* time, we identify the logic level (logic '0' or '1') of the memory cells that are bagged previously in line 20. It is worth mentioning that *set/reset* characteristics of ReRAM cells appear to be uniform across all ReRAMs that we have tested.

D. Encoding/Decoding Secret Information

The flowchart in Fig. 2 shows the steps of the information hiding process, i.e., chronologically encoding and decoding information in ReRAM. 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 physical characteristics of ReRAM cells at different stressing levels up to the maximum endurance [8]. 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) to the secret message can improve the information recovery process by correcting obtained bit errors.

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

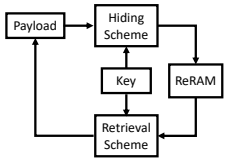


Fig. 2: Steps to hide information in ReRAM.

V. Experimental Results

A. Evaluation Setup

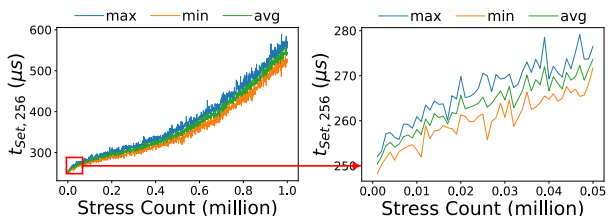
The analysis is performed over five *MB85AS8MT* (40nm technology node) 8-bit serial peripheral interfaced (SPI) $8Mb$ ReRAM memory chips manufactured by Fujitsu Semiconductor Limited. We have used our own custom-designed memory controller implemented on *Teensy 4.1* microcontroller development board 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* ReRAMs 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. 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). Sect. V-C discusses the impact of using different replica sizes. Moreover, it is possible to encode data into non-consecutive addresses to make the detection scheme more complex (discussed in Sect. VII-B). Therefore, to increase the complexity of the retrieval process, instead of encoding single-bit data into 256 consecutive addresses, we can use any stream cipher ([26]) to select the memory addresses for each message bit. The memory addresses and replica selections are based on the "hiding key" (Fig. 2) in a way that cannot be predicted without the key.

We have measured the *set/reset* time for each address and computed the average for the evaluation. 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, reducing overall implementation complexity. Although the figures presented in this section are based on a single ReRAM (randomly chosen from five test chips), the observation is valid for all test chips.

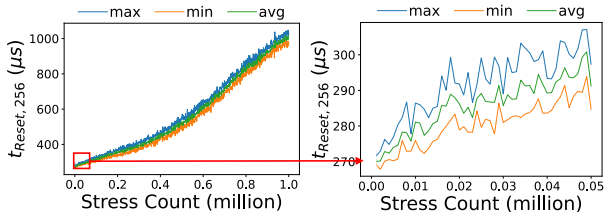
The following steps are performed to verify the feasibility of the proposed information-hiding technique. We have embedded an arbitrarily chosen 32-bit random data (0xECE3038B[3]) into $(256\times32) = 8192$ memory addresses, varying the number of switching cycles, $\mathcal{N}$, up to $45K$ times to demonstrate the information hiding (discussed in Sect. IV-B) and retrieval (discussed in Sect. IV-C) 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.

B. Cell Characterization

The switching characteristics (*set/reset* time vs. the stress counts) of the ReRAMs at $25°C$ are shown in Fig. 3. These figures represent 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 $t_{Set,256}$ 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 Fig. 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 $\sim12K$ 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 $\sim12K$ *set/reset* operations. Although Fig. 3 is constructed with $2K$ memory addresses, a similar characteristic is valid for the whole address space.



(a)



(b)

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

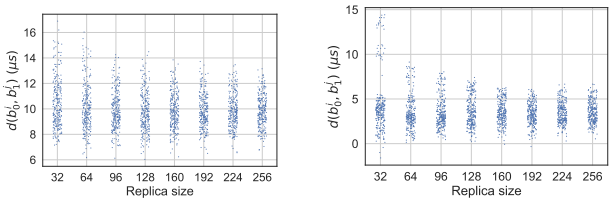
[3]Verified for other random data as well.

[4]We observed that most memory cells of *MB85AS8MT* can endure more rewrite operations (up to $2M$ rewrite cycles used to test in this work) than the rated endurance, which is $1M$ rewrite cycles (i.e., $500K$ *set-reset* pairs).

C. Appropriate Replica Size Selection

Fig. 4 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. 4 represents $d(b_0^i, b_1^j)$ for each possible (b_0^i, b_1^j). The figure demonstrates that the separation between logic '0' bits and logic '1' bits improves with respect to replica size. While using *set* time, the bit '0' and bit '1' are well separable (i.e., $d(b_0, b_1) > 0$, for all (i, j)) with the smallest possible replica size of 32 (Fig. 4a). On the other hand, *reset* time only distinguishes the bit '0' and '1' with the minimum replica size of 224 (Fig. 4b). Therefore, 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. A smaller replica size will increase the performance by reducing the imprinting and retrieval time. In our implementation, we have chosen the replica size 256 for the following two reasons–

- For both t_{Set} and t_{Reset}, $d(b_0, b_1) > 0$ with the replica size of 256.
- All of our test ReRAMs have an internal data buffer to hold data for up to 256 addresses. This buffer is an unmodifiable hardware component (a set of 8-bit registers). These ReRAMs also come with dedicated instructions, enabling us to simultaneously send data for 256 addresses from the memory controller to the memory chip and write them in parallel. Therefore, choosing 256 replica sizes simplified the experimental setup and reduced the implementation complexity.

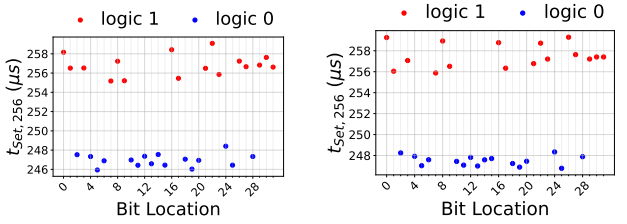


(a) (b)

Fig. 4: Replica size vs. the hidden information with (a) $t_{Set,[32,256]}$ and (b) $t_{Reset,[32,256]}$.

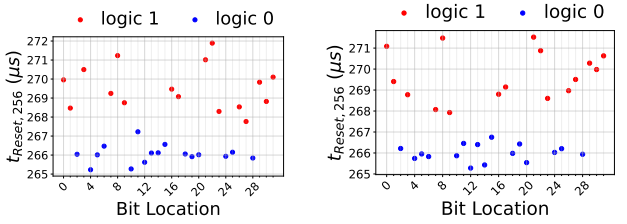
D. Retention Characteristics

The retention characteristics of the proposed hiding technique are also studied and shown in Figs. 5 and 6. 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 Figs. 5 and 6 represent the imprinted logic 1's and 0's, respectively. Fig. 5a and Fig. 6a 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. 5b) and $t_{Reset,256}$ (Fig. 6b). The results verify that the retention time has little or no impact on bit separation.



(a) Zero retention. (b) Over two months retention.

Fig. 5: Retention characteristics of the hidden information using $t_{Set,256}$.



(a) Zero retention. (b) Over two months retention.

Fig. 6: Retention characteristics of the hidden information using $t_{Reset,256}$.

VI. Performance Analysis

This section analyzes the proposed scheme's performance, normal memory usage tolerance, and robustness. We use the encoding and retrieval time and encoding cost as the metric for measuring performance.

A. Encoding Time

The proposed technique for information hiding relies on 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 $\mathcal{T}_{encode} = (\mathcal{N} \times \mathcal{B}_{Msg} \times \mathcal{T}_{switch_{pair}})$, 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[5]– $\mathcal{T}_{switch_{pair}} = (265\mu s + 285\mu s) = 550\mu s$, and $\mathcal{B}_{Msg} = 32$. Thus, the baseline implementation requires $(550\mu s \times 32 \times 15K) = 264s$ for $15K$ switching operations to conceal secret message. Therefore, the throughput for the encoding process is $\frac{32bits}{264s} = 0.12bit/s$. The hiding throughput will 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 ReRAMs. Fortunately, in the past few years, the *write* speed of ReRAMs significantly improved and will continue to improve in the future. For example, the *write* speed of *MB85AS8MT* ReRAMs is improved $>3X$ over its previous generation *MB85AS4MT* ReRAMs.

[5]Average *set/reset* time from first 60K write cycle (Fig. 3).

B. Retrieval Time

Unlike the encoding procedure, the extraction procedure is significantly faster. 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 *set/reset* time, and $\mathcal{N}_{rep}$ represents the number of addresses used to encode single bits. Therefore, $(\mathcal{T}_{switch} \times \mathcal{N}_{rep})$ is the average value of $t_{Set,256}$ or $t_{Reset,256}$. The worst case[6] average value of $t_{Set,256}$ is $\sim 300\mu s$. Hence, the throughput for the retrieval is $\frac{\mathcal{B}_{Msg}}{\mathcal{T}_{retrieve}} = \frac{32bits}{300\mu s \times 32} = 3.26Kbits/s$. The average value $t_{Set,256}$ includes program data transfers between the microcontroller and 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 Sect. V-C.

C. 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 $t_{Reset,256}$): only 3% of the rated endurance of encoded addresses (the rated endurance of ReRAMs is $1M$).

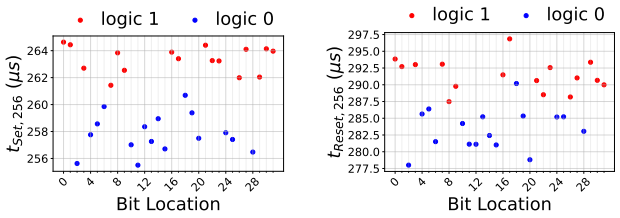
D. Initial Stress Tolerance

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. 7. To simulate the normal usage of the storage device, we write random data patterns to the memory for the initial stressing. Random data patterns appear according to [27] 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. 7b) as the initial stress level increases. However, increasing the stress level in the encoding process can tolerate a higher initial stress count. Besides, the *set* operation can tolerate higher initial stress than the *reset* operation. We found that the error rate observed using $t_{Reset,256}$ is still manageable (3.125%) even after thousands of normal memory operations.

E. 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. Random data patterns appear according to [27] to make more realistic stressing on the memory cells incurred from memory usage. Fig. 8

[6]Considering post-hiding stressing (discussed in Sect. VI-E)

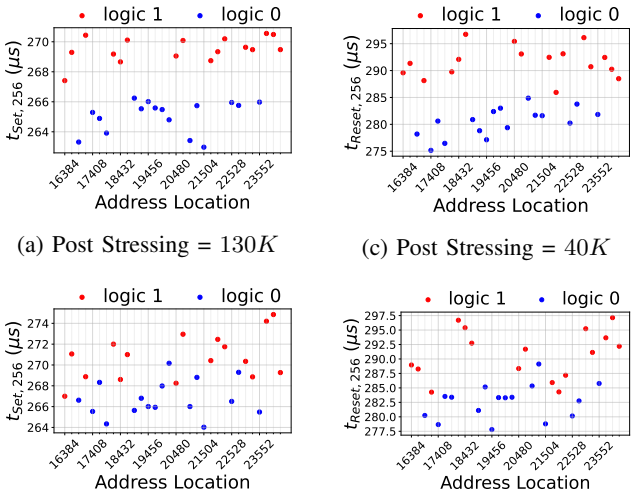


(a) $\mathcal{N} = 15K$ (b) $\mathcal{N} = 15K$

Fig. 7: Influence of initial stress ($50K$) on the hidden information using- (a) $t_{Set,256}$ and (b) $t_{Reset,256}$.

and Fig. 9 illustrate the influence of post-hiding stressing (i.e., the number of stresses performed after hiding information).

Fig. 8 represents the post-hiding stress tolerance of the hidden data where a minimum ($\mathcal{N} = 15K$) stress count is used to hide the information[7]. The red and blue dots represent the hidden logic 1s and 0s, respectively. Fig. 8 shows that logic '1' and logic '0' remain separated up to $130K$ stress count (Fig. 8a). They become inseparable at $140K$ stress count (Fig. 8b). Similarly, with $t_{Reset,256}$, logic '1' and logic '0' remain separated up to $40K$ stress count (Fig. 8c) and become inseparable at $50K$ stress count (Fig. 8d).

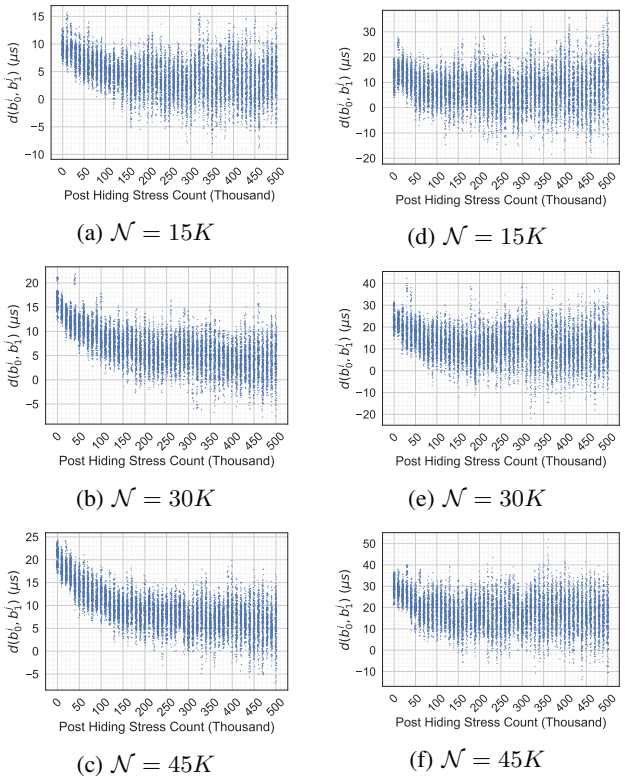


(a) Post Stressing = $130K$ (c) Post Stressing = $40K$

(b) Post Stressing = $140K$ (d) Post Stressing = $50K$

Fig. 8: Hidden data (stress counts $\mathcal{N} = 15K$) at different post-hiding stress levels- (a)–(b) $t_{Set,256}$ at stress count $130K$ & $140K$; (c)–(d) $t_{Reset,256}$ at $40K$ & $50K$.

In addition, Fig. 9 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. 9 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. 8b), 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 while using $t_{Set,256}$ with $\mathcal{N} = 15K$ (Fig. 9a), $30K$ (Fig. 9b), and $45K$ (Fig. 9c), 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 while using $t_{Reset,256}$ with $\mathcal{N} = 15K$ (Fig. 9d), $30K$ (Fig. 9e), and $45K$ (Fig. 9f), respectively. However, note that if we tolerate one or two-bit errors, our proposed scheme can survive thousands of post-hiding stress levels (Table II).



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

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

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

Fig. 9: Post-hiding stress tolerance of concealed data at different hiding stress levels, using- (a)–(c) $t_{Set,256}$, and (d)–(f) $t_{Reset,256}$.

Table I summarizes all test chips' post-hiding stress tolerance statistics. 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, columns three to seven represent 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 endure

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

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. Note that the *set* operation can tolerate higher initial and post-hiding stress. In ReRAM, the *set* operation means transitioning from HRS to LRS, whereas the *reset* operation means transitioning from LRS to HRS (Fig. 1). Repeated stressing on ReRAM gradually reduces the oxide layer integrity and introduces irreversible crystal defects (dielectric breakdown). Due to this crystal defect, the oxide layer contains more charge carriers than when it was new. Therefore, the resistance of the HRS reduces over time if we keep the same operating condition as the fresh ReRAM cell. Additionally, the oxide layer of ReRAM represents its highest resistance when there is no crystal defect. As the crystal defect introduced by device usage is random, the *reset* operation becomes noisier over time compared to the *set* operation ($2\times$ noisier [28]). Therefore, the *reset* operation tolerates less initial stress and less post-hiding stress.

TABLE I: Post-hiding stress tolerance (0-bit error).

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

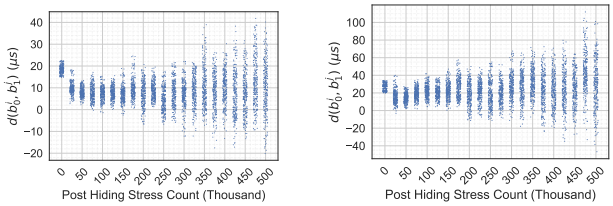
How long the confidential information must be concealed entirely relies on the application. If we use higher stress levels to hide the data, those hidden messages can tolerate more post-hiding stress. For example, if we want to keep the hidden message up to the full endurance level of the memory, we need to stress a little higher than $45K$ (which we demonstrated in Fig 9c). If we hide the date with $45K$ stressing, it can tolerate $180K$ post-hiding stressing (36% of the rated endurance of ReRAMs) without ECC (Table I) and $430K$ post-hiding stressing (86%[8] of the rated endurance) with ECC (Table II), respectively. In addition, if the secret message needs not to be kept for a higher endurance level, we can consider a lesser stressing level to hide the data to make the secret message unrecoverable after 20% (Table I) of the rated endurance of ReRAMs. Furthermore, to keep the hidden data longer with fewer stressing counts, we can continue stressing those cells before the hidden data disappears.

TABLE II: Worst-case post-hiding stress tolerance (considering *set* operation, $t_{Set,256}$).

Stress Count, $\mathcal{N}$	Bit Error Count		
	0	1	2
15k	100K	150K	200K
30k	140K	250K	310K
45k	180K	400K	430K

[8]We can avoid ECC since with higher stress and without ECC, the hidden message remains concealed for 36% of the rated endurance of ReRAM.

Besides, Table II represents the minimum error tolerance of our proposed data-hiding scheme. The 1st column represents the number of stresses used to hide the data (considering *set* operation, $t_{Set,256}$). The 2nd – 4th columns represent the minimum number of allowable post-hiding stresses considering $0-, 1-$, and $2-$bit errors, respectively. Table II shows that if $15K$ stress level is used to hide the data, those secret messages can tolerate $100K$ (20% of the rated endurance), $150K$ (30% of the rated endurance), and $200K$ (40% of the rated endurance) post-hiding stressing with $0-, 1-$, and $2-$bit error(s), respectively. On the other hand, if $45K$ stress level is used to hide the data considering *set* operation ($t_{Set,256}$), those secret messages can tolerate $180K$ (36% of the rated endurance), $400K$ (80% of the rated endurance), and $430K$ (86% of the rated endurance) post-hiding stressing with 0–, 1–, and 2–bit error(s), respectively.



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

Fig. 10: Post-hiding stress tolerance with maximum stressing using- (a) $t_{Set,256}$ and (b) $t_{Reset,256}$.

Furthermore, to apply maximum post-hiding stress, we write '1' $\rightarrow$ '0' $\rightarrow$ '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. 10. The figure shows 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 counts. For example, with $45K$ switching operations used to hide information can endure $200K$ post-hiding stress levels for both $t_{Set,256}$ (Fig. 10a) and $t_{Reset,256}$ (Fig. 10b).

Note that, as the *set/reset* time increases monotonically with respect to usage (Fig. 3), the *set/reset* time of the imprinted cells should also shift upward with regular usage. In this scenario, the bit '0' and bit '1' can be differentiated by either of the following two techniques-

- **Using clustering algorithm:** As we determine the bits '0' and '1' based on the distance in *set/reset* time, a simple clustering algorithm such as k-mean clustering is extremely effective in clustering all '0' bits and '1' bits. Therefore, we do not need to specify any threshold value explicitly.
- **Threshold shifting:** If the clustering algorithm is not available due to the device limitation, the user can monitor the *set/reset* time of a set of memory cells (reference cells) where the hidden data in not imprinted. Now, based on the *set/reset* time on the reference cell, one can easily determine the current usage level of the ReRAMs and shift the threshold accordingly. Ideally, the *set/reset* time of these

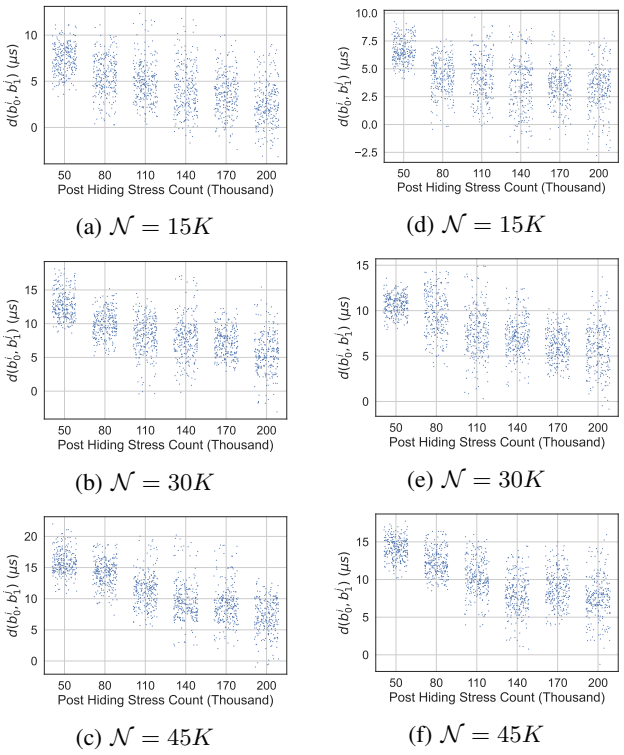
reference cells should be in a similar range as of the logic '0' imprinted memory cells.

Lastly, in our experiment, we emulated the worst-case scenario of memory stress (i.e., toggling memory bit in the write cycle). However, a memory bit does not toggle on every write operation. For example, if we overwrite a random byte, '00101000', with another random byte, '00110101', only four bits will be toggled, and others will not experience any stress. Therefore, we expect our hidden data to last for many more write cycles than we demonstrated in the above discussion. Additionally, in a practical scenario, memory cell usually experiences a specific usage pattern that has been extensively explored in previous work ([27]). Under regular usage, the most significant bits (MSB) usually experience less stress than the least significant bits (LSB), whereas the usage pattern over the entire memory address follows almost a uniform distribution. Our proposed technique measures the *set/reset* time for the whole byte; therefore, the usage deviation from MSB to LSB should not impact our proposed technique.

F. Robustness Analysis

The hidden message should be resilient to the variation of operating conditions (i.e., temperature or operating voltage). Inherently, all modern ICs are resilient to small variations in operating voltage as they are usually integrated with a voltage regulator. 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. Random data patterns appear according to [27] to make more realistic stressing on the memory cells incurred from memory usage. After every $30K$ memory operation, we isolated the memory chip from the system and baked it at $80°C$[9] 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 ($\mathcal{N} = 15K$, $30K$, and $45K$, respectively) used to conceal information.

Figs. 11 and 12 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 Sect. VI-E, a larger positive value of $d(b_0^i, b_1^j)$ is desirable to better separate logic '0' and logic '1' bits. However, in Figs. 12a and 12d, 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. 11a, 11b, and 11c show $d(b_0^i, b_1^j)$ using $t_{Set,256}$ before performing high-temperature system-level operation. Figs. 11d, 11e, and 11f show $d(b_0^i, b_1^j)$ using $t_{Set,256}$ after performing high-temperature baking and high-temperature system-level operation. Similar observations are valid for $t_{Reset,256}$ (Fig. 12). It's observed that the hidden information is not significantly affected by temperature and remains well-separated at high temperatures (considering both $t_{Set,256}$ and $t_{Reset,256}$). Such behavior of ReRAM is expected as the resistance ratio ($\frac{R_{HRS}}{R_{LRS}}$) is relatively temperature insensitive [11].

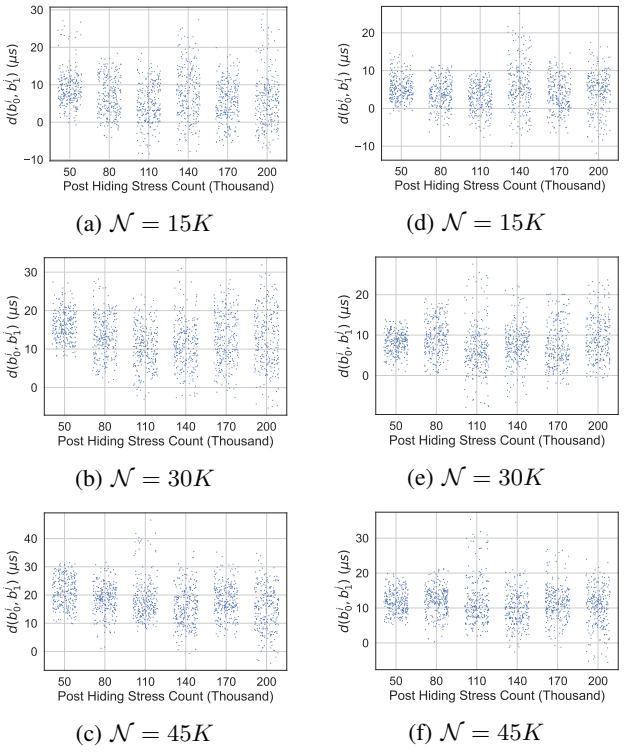


(a) $\mathcal{N} = 15K$ (d) $\mathcal{N} = 15K$

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

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

Fig. 11: Robustness analysis (a)–(c) before (d)–(f) after high-temperature baking ($80°C$) with- $t_{Set,256}$.



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

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

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

Fig. 12: Robustness analysis (a)–(c) before (d)–(f) after high-temperature baking ($80°C$) with- $t_{Reset,256}$.

[9]ReRAMs used in this experiment are rated to operate up to $85°C$.

VII. Security Analysis

While the previous section discusses how the *write* time is manipulated to store hidden information reliably, this section discusses whether an attacker gains access to the memory containing hidden information and reveals it.

A. 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 fresh and stressed memory cells; therefore, even the correct threshold value cannot distinguish the inaccurate group's fresh and stressed cells.

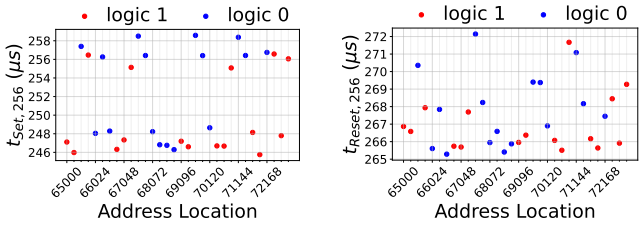
Fig. 13 illustrates the retrieved data with an incorrect initial address where the minimum ($\mathcal{N} = 15K$) stress level is used to hide information. The red and blue dots represent the imprinted logic 1s and 0s, respectively. In this experiment, we selected the initial address in three different 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 overlaps 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. 13 is constructed with the ($\mathcal{N} = 15K$) stress count, a similar characteristic is valid for a higher stress count (i.e., up to $45K$). Fig. 13 shows that there is no clear separation between fresh and stressed memory cells for *case 3* and we observed similar results for the other two cases as well. Hence, it is difficult for attackers to retrieve hidden data without the correct initial address because the value of hidden bits can not be retrieved through thresholding, and each group may have fresh and stressed memory cells.

B. Data hide with enhanced security

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 hide through replication and random rotation to strengthen security. To this end, we hide 1st copy of 32-bit scramble hidden data in 32 consecutive addresses, then the 2nd copy in another 32 consecutive addresses, and so on. Fig. 14 demonstrates this method 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 ($\mathcal{K}$). The displacement should be uniform within the possible range to maximize security (in this case, $7 \leq \mathcal{K} \leq 0$). Here, the $\mathcal{K}$ for each replica can be considered as the key. Note that, instead of a circular shift, we can also use the stream cipher algorithm to randomize the pattern [26]. While retrieving the hidden information, the key must reverse the rotation 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.



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

Fig. 13: Retrieved data with an incorrect initial address at ($15K$) stressing used to hide information with- (a) $t_{Set,256}$ (b) $t_{Reset,256}$.

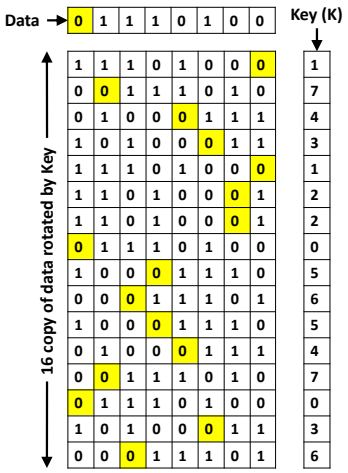
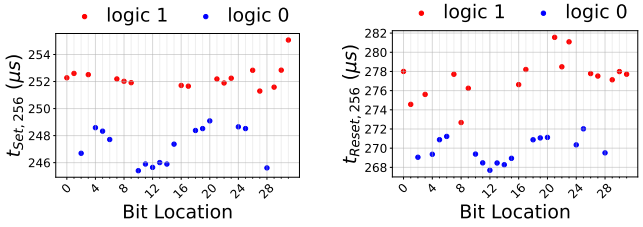


Fig. 14: Enhancing security through replication & left circular rotation. Rotation starts with yellow bits.

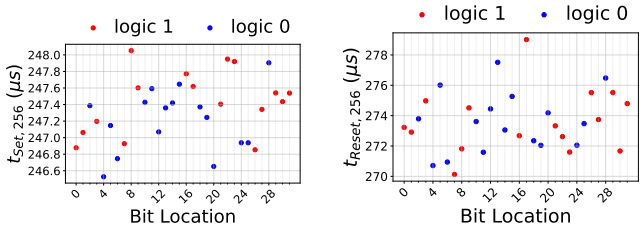


(a) $\mathcal{N} = 15K$ (b) $\mathcal{N} = 15K$

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

Fig. 15 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 the fresh and stressed memory cells. On the other hand, Fig. 16 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 will likely have fresh and stressed memory cells.



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

Fig. 16: Retrieved data with an incorrect key with $\mathcal{N} = 15K$ stressing used to hide information with- (a) $t_{Set,256}$ (b) $t_{Reset,256}$.

Given an unlimited amount of time, an attacker may extract the hidden information by distinguishing the stress and fresh cells by observing the *write (set/reset)* time of the memory cells. However, non-volatile memories usually have millions of cells, and getting timing information for individual cells is very time-consuming and expensive. Moreover, the *hiding key* (Fig. 2) gives another layer of security and is also kept in secure memory and not available to attackers. Furthermore, different ReRAM cells age at different rates since the data written into the memory chip are random. Therefore, with the presence of used cells, an attacker would experience more false positives and negatives than a fresh ReRAM because some cells are less stressed than others and randomly distributed. Finally, since stressed and fresh cells provide different timing information, we can add random delays at the hardware or software level to provide another layer of security. This random timing noise can easily hide the timing information of stressed and fresh cells. The random delay block will be active during the regular operation and inactive while we retrieve the hidden data. We assume that the attacker has no control over this random delay block and requires administrative privilege to deactivate the random delay block during the hidden data recovery mode. If the key/administrative privilege is uncompromised, the attacker cannot tamper with the proposed security scheme, keeping the *coincidence probability* low [29].

C. Potential Application Scenarios

Our proposed scheme is not a replacement for traditional encryption but rather more comparable to physical steganography in digital information, i.e., adding new information on top of existing information without any trace (e.g., without requiring additional storage space). Likewise, in steganography, data hiding should not be used as an alternative to data encryption techniques. However, data hiding can still be useful in several applications, such as making robust watermarks, second-layer protection over encrypted data, secure and covert data storage, data integrity, covert military/police communication, and digital rights management to protect intellectual property and data from tampering. For example, one can hide sensitive secret information in ReRAM with higher confidence; others cannot retrieve the information even when the device is lost or stolen. Therefore, data hiding offers an extra protection layer over conventional encryption by thwarting an adversary from reading/copying the ciphertext. Moreover, our proposed technique can be extremely useful in a ransomware attack because the attackers cannot easily modify the hidden data.

VIII. Discussion

Overall, we draw the following 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 $\sim 0.12 bit/s$ and $\sim 3.26 Kbits/s$, 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 temperature variations.
6) Our proposed technique can be incorporated with a secret key to enhance the security of the hidden information.
7) Modification of the hidden data with our proposed technique will require repetitive *set/reset* operations and can be easily detected by the system by monitoring the number of *set/reset* operations in a particular address.

Note that, repeated stressing on ReRAM gradually introduces irreversible crystal defects (dielectric breakdown), resulting in more charge carriers in the oxide layer than in the fresh condition. Therefore, the HRS of ReRAM reduces over time. In the future, we will investigate if any form of physical attack (such as SEM-PVC [30]) can be used to determine the change in ReRAM physical properties.

IX. Conclusion

This paper demonstrated a cost-effective technique to hide information using the memory cells' *set/reset* time of commercially available ReRAMs. *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 survive successfully even after thousands of memory operations. Our proposed technique utilizes repeated switching operations to change the physical properties of the memory cells. Without the knowledge of the hiding key, analog physical characteristics measurement will not help reveal the hidden information. Additionally, our proposed technique is robust against temperature variation and requires no hardware modifications.

ACKNOWLEDGMENTS

This work was supported by the National Science Foundation under Grant Number DGE-2114200.

REFERENCES

[1] M. Rostami, F. Koushanfar, J. Rajendran, and R. Karri, "Hardware security: Threat models and metrics," in *2013 IEEE/ACM International Conference on Computer-Aided Design*, 2013, pp. 819–823.
[2] S. Rai *et al.*, "Security promises and vulnerabilities in emerging reconfigurable nanotechnology-based circuits," *IEEE Transactions on Emerging Topics in Computing*, vol. 10, no. 2, pp. 763–778, 2022.
[3] J. Rajendran *et al.*, "Nano meets security: Exploring nanoelectronic devices for security applications," *Proceedings of the IEEE*, vol. 103, no. 5, pp. 829–849, 2015.
[4] M. I. Rashid *et al.*, "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, 2021.
[5] F. Ferdaus, B. M. S. Bahar Talukder, M. Sadi, and M. T. Rahman, "True random number generation using latency variations of commercial mram chips," in *2021 22nd International Symposium on Quality Electronic Design*, 2021, pp. 510–515.
[6] "Crossbar reram technology," https://www.crossbar-inc.com/technology/reram-advantages/, (Accessed: 8 May 2022).
[7] Fujitsu Semiconductor Memory Solution, "Reram (resistive random access memory)," 2019, (Accessed: 15 Oct 2021). [Online]. Available: https://www.fujitsu.com/jp/group/fsm/en/products/reram/
[8] F. Ferdaus, B. M. S. Bahar Talukder, and M. T. Rahman, "Watermarked reram: A technique to prevent counterfeit memory chips," in *Great Lakes Symposium on VLSI*, ser. GLSVLSI '22, 2022.
[9] M. Arita *et al.*, "Switching operation and degradation of resistive random access memory composed of tungsten oxide and copper investigated using in-situ tem," in *ICCD*, vol. 5, 2015.
[10] M. Mao *et al.*, "Optimizing latency, energy, and reliability of 1t1r reram through appropriate voltage settings," in *ICCD*, 2015, pp. 359–366.
[11] B. Govoreanu *et al.*, "Complementary role of field and temperature in triggering on/off switching mechanisms in Hf/HfO_2 rram cells," *IEEE transactions on electron devices*, vol. 60, no. 8, pp. 2471–2478, 2013.
[12] A. Cheddad, J. Condell, K. Curran, and P. Mc Kevitt, "Digital image steganography: Survey and analysis of current methods," *Signal Processing*, vol. 90, no. 3, pp. 727–752, 2010.
[13] Y. Wang *et al.*, "Hiding information in flash memory," in *2013 IEEE Symposium on Security and Privacy*, 2013, pp. 271–285.
[14] Petitcolas *et al.*, "Information hiding-a survey," *Proceedings of the IEEE*, vol. 87, no. 7, pp. 1062–1078, 1999.
[15] N. Provos and P. Honeyman, "Hide and seek: an introduction to steganography," *IEEE Security Privacy*, vol. 1, no. 3, pp. 32–44, 2003.
[16] Y. Pang *et al.*, "A rram-based data hiding technique utilizing the impact of form condition on set performance," in *IEEE International Memory Workshop*, 2020, pp. 1–4.
[17] I. Sutherland, G. Davies, and A. Blyth, "Malware and steganography in hard disk firmware," *Journal in computer virology*, vol. 7, no. 3, pp. 215–219, 2011.
[18] J. Mahmod and M. Hicks, "Invisible bits: Hiding secret messages in sram's analog domain," in *ACM International Conference on Architectural Support for Programming Languages and Operating Systems*, 2022, p. 1086–1098.
[19] A. Srinivasan, J. Wu, P. Santhalingam, and J. Zamanski, "Deaddrop-in-a-flash: Information hiding at ssd nand flash memory physical layer," *SECURWARE*, vol. 79, p. 2014, 2014.
[20] S. Jia *et al.*, "Deftl: Implementing plausibly deniable encryption in flash translation layer," in *ACM SIGSAC Conference on Computer and Communications Security*, 2017, pp. 2217–2229.
[21] P. R. Prucnal *et al.*, "Optical steganography for data hiding in optical networks," in *2009 16th International Conference on Digital Signal Processing*, 2009, pp. 1–6.
[22] K. Szczypiorski and W. Mazurczyk, "Steganography in ieee 802.11 ofdm symbols," *Sec. and Comm. Networks*, vol. 9, no. 2, pp. 118–129, 2016.
[23] A. M. Mehta *et al.*, "Steganography in 802.15.4 wireless communication," in *International Symposium on Advanced Networks and Telecommunication Systems*, 2008, pp. 1–3.
[24] N. F. Johnson and S. Jajodia, "Exploring steganography: Seeing the unseen," *Computer*, vol. 31, no. 2, pp. 26–34, 1998.
[25] Y. Chen, "ReRAM : History, status, and future," *IEEE Transactions on Electron Devices*, vol. 67, no. 4, pp. 1420–1433, 2020.
[26] T. W. Cusick, C. Ding, and A. R. Renvall, *Stream ciphers and number theory*. Gulf Professional Publishing, 2004.
[27] D. Wei *et al.*, "Nrc: A nibble remapping coding strategy for nand flash reliability extension," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 35, no. 11, pp. 1942–1946, 2016.
[28] W. Feng, H. Shima, K. Ohmori, and H. Akinaga, "Investigation of switching mechanism in hfox-reram under low power and conventional operation modes," *Scientific reports*, vol. 6, no. 1, p. 39510, 2016.
[29] F. Koushanfar, I. Hong, and M. Potkonjak, "Behavioral synthesis techniques for intellectual property protection," *ACM Trans. Des. Autom. Electron. Syst.*, vol. 10, no. 3, p. 523–545, jul 2005.
[30] G. T. Becker, F. Regazzoni, C. Paar, and W. P. Burleson, "Stealthy dopant-level hardware trojans," in *Cryptographic Hardware and Embedded Systems - CHES 2013*, G. Bertoni and J.-S. Coron, Eds. Springer Berlin Heidelberg, 2013, pp. 197–214.

Farah Ferdaus (S'20–M'24) is a Postdoctoral Researcher at Argonne National Laboratory. She received her Ph.D. degree in Electrical and Computer Engineering from Florida International University in 2022. Her research interests include AI Accelerators, high-performance, energy-efficient computer architecture, emerging memory technologies, and hardware security.

B. M. S. Bahar Talukder (S'18) received his Ph.D. degree in Electrical and Computer Engineering from Florida International University. He is currently a GPU implementation engineer at Apple. His primary research interests include hardware security, secured computer architecture, machine-learning applications in system security, and emerging memory technologies.

Md Tauhidur Rahman (S'12–M'18-SM'21) is an Assistant Professor in the Department of Electrical and Computer Engineering at Florida International University (FIU). He received his Ph.D. degree in Computer Engineering from the University of Florida in 2017. His current research interests include hardware security and trust, side-channel analysis, memory systems, embedded security, and privacy. He is one of the recipients of 2019 NSF CRII Award.