Intrinsically Secure Non-Volatile Memory Using ReRAM Devices

JUNJUN HUAN1, (Member, IEEE), NICHOLAS OLEXA2, BRET HOCHMAN2, SWARUP BHUNIA1, (Senior Member, IEEE), RASHMI JHA2, (Member, IEEE), AND SOUMYAJIT MANDAL3, (Senior Member, IEEE)

1Department of Electrical and Computer Engineering, University of Florida, Gainesville, FL 32606, USA
2Department of Electrical Engineering and Computer Science, University of Cincinnati, Cincinnati, OH 45221, USA
3Instrumentation Division, Brookhaven National Laboratory, Upton, NY 11973, USA

Corresponding author: Junjun Huan (junjun.huan@ufl.edu)

This work was supported in part by the National Science Foundation (NSF) Center for Hardware and Embedded System Security and Trust (CHEST)/Industry-University Cooperative Research Centers Program (IUCRC) at the University of Cincinnati.

ABSTRACT The paper describes a device-level encryption approach for implementing intrinsically secure non-volatile memory (NVM) using resistive RAM (ReRAM). Data are encoded in the ReRAM filament morphology, making it robust to both electrical and optical probing methods. The encoded resistance states are randomized to maximize the entropy of the ReRAM resistance distribution, thus providing robustness to reverse engineering (RE) attacks. Simulations of data encryption and decryption using experimental data from Ru(BE)/ALD-HfO2(MO)/Zr(W)TE ReRAM devices reveals an uncorrected bit error rate (BER) < 0.02 and a maximum key entropy of ≈ 17.3 bits per device. A compensation procedure is also developed for maintaining BER in the presence of temperature changes.

INDEX TERMS ReRAM, non-volatile memory (NVM), hardware security, encryption.

I. INTRODUCTION

Emerging non-volatile memories (NVMs) promise a fundamental advancement in computing by allowing memory and storage systems to be combined. Integration of NVM within CMOS dies thus enables monolithic computer architectures that realize the processor and high-capacity memory/storage system on a single die [1], [2]. Promising NVM technologies include filamentary and non-filamentary resistive RAM (ReRAM) [3], [4], [5], spin transfer torque magnetic RAM (STT-MRAM) [6], and phase change memory (PCM) [7]. Some attractive features of ReRAM include: (i) logic-compatible voltages and currents, (ii) CMOS Back End of Line (BEOL) process-flow compatibility, (iii) multiple resistance states, (iv) non-volatility of states, and (v) high scalability. A large applications space for these devices has accordingly emerged, ranging from high-density storage class memory [8], [9] to multiply and accumulate (MAC) units for neuromorphic computing [10], [11]. ReRAM devices can also be used to realize various security primitives for such architectures due to their reconfigurability and inter-device variability [12], including physical unclonable functions (PUFs) for device authentication [13], [14], true-random number generators (TRNGs) [15], reconfigurable vias enabling split manufacturing [16], switch boxes for non-volatile FPGAs [17], and programmable “memory fingerprints” for provable destruction of digital keys after use [18].

On-chip ReRAM modules enable a variety of emerging applications include key storage for logic locking, hyper-parameter and weight storage for neuromorphic computing, and user profile and key storage for IoT sensor nodes. With the widespread deployment of ReRAM-based NVM in such application domains, several security threats including reverse engineering (RE), piracy, and tampering have emerged as major concerns at different stages of the product life-cycle. Sensitive data stored within such NVM modules has traditionally been protected by reading and writing only encrypted data blocks, with the necessary encryption and decryption operations being performed by an embedded cryptographic processor [19], [20]. However, these operations tend to add significant power, area, and cost overhead, which makes them unsuitable for ultra-low-power applications such as IoT nodes. Additionally, they are prone to data loss since
the key can often be revealed using techniques such as side-channel attacks [21], [22]. Thus, there is ongoing interest in lightweight cryptographic algorithms to replace well-known encryption methods such as AES [23]. The authors in [24] compare various lightweight cryptographic algorithms and their software and FPGA-based hardware implementations, with Throughput/Area (TP/A) and energy/bit chosen as metrics to quantify suitability for IoT applications. Notably, AES implementations on FPGA provide the best energy/bit, but relatively poor TP/A. Here we propose a complementary approach in which the NVM exploits intrinsic device properties to self-encrypt all data prior to storage. The additional security provided by such self-encryption allows the cryptographic processor to use simpler algorithms, thus improving system-level energy efficiency. The device properties of ReRAM result in further improvements in system-level metrics. Firstly, ReRAM has been proven to be extremely scalable: its layout footprint is $4F^2$ where $F$ is the minimum feature size [25], [26], and 3D integration with access diodes (resulting in 1DIR architectures) can lead to further scalability by reducing the footprint to $4F^2/L$ where $L$ is the number of layers [27]. Secondly, ReRAM switching energy and read energy per bit can be in the fJ/bit range ($\sim 0.1$ fJ, respectively). As a result, ReRAM-based self-encrypted NVM is promising for simultaneously achieving low TP/A and energy/bit.

The rest of the paper is organized as follows. Section II describes the properties of ReRAM devices that enable self-encrypted NVM. The proposed self-encryption approach is described and analyzed in Section III, while experimental results are presented in Section IV. Finally, Section V summarizes our contributions and concludes the paper.

II. DEVICE PROPERTIES

ReRAM uses emerging two-terminal devices that can be fabricated with high densities and can assume a range of resistance states based on either fast (10-100 ns) or low-voltage ($< \pm 1$ V) pulse programming, making them attractive for scalable non-volatile data storage and neuromorphic computing [28], [29]. The basic device structure is composed of a very thin metal-oxide (MO) layer placed between a top electrode (TE) and bottom electrode (BE). On application of a high electric field, the oxide breaks down, causing the formation of a conducting filament (CF) as shown in Fig. 1(a) [30]. For simplicity, the CF is generally modeled as a cylinder of diameter $< 50$ nm stretching from BE towards TE.

As an example, we studied custom filamentary ReRAM devices fabricated using a Ru(BE)/ALD-HfO$_2$ (MO)/Zr/W(TE) stack [31]. Filamentary devices were chosen since they are more studied and their models are better understood [32], [33], [34]. Filamentary ReRAM devices also show better endurance and retention values over non-filamentary ReRAM devices [29]. The filament geometry can be controlled by compliance current and reset voltage [28], resulting in a range of resistance states as shown in Fig. 1(b) for a typical device: Widening the filament-electrode gap increases the resistance, while narrowing the gap (or widening the filament) lowers it. The processes used to reach these states are referred to as Set, Reset, and Forming. Forming is the initial operation required for soft breakdown of the MO, shifting oxygen ions toward a ‘storage layer’ at the TE/MO interface and creating a metal-rich filament in the MO that stretches from BE to TE (see Fig. 1(a)). Reset operations reverse this process to a significant degree, i.e., increase the resistance by retracting the filament away from TE. Set operations reduce the resistance, similar to Forming, but by a smaller amount that is determined by the compliance current (CC) used during the operation. In Fig. 1(b), the low-resistance ‘Set’ state ($\sim 0.9$ k$\Omega$) was achieved through Forming, followed by Reset and finally Set. From here, a wide range of resistance states ($2.5$–$250$ k$\Omega$) was achieved via Reset operations at various voltages (in this case, from 1 V to 2.4 V).

ReRAM-based NVM also has attractive properties for self-encryption [35]. The history-dependent voltage response of these devices allows information to be hidden in the filament geometry and later detected by inverting the state to its starting condition. Due to this unique history-dependent voltage response, device resistances can be repeatedly shifted up/down by self-limited Set and Reset operations. In this approach, the device is first electroformed with high CC, then Reset to the desired state, Set again with low CC, and finally Reset back to a desired state. Thereafter, the Set state can be successfully retrieved by careful application of appropriate set voltage without any fixed CC. Instead, the read current is governed by prior history of the Set state, which makes it self-limiting if a precise Set voltage range is used. Thus, unlike CC-based Set operations, self-limiting ones (achieved using prior Set conditions) simply shift the device state based on its current state and the voltage used, as shown in Fig. 1(b).

In order to be self-protected, the device must also have i) maskable states from which data cannot be read without altering the device state; and ii) a final revealed state (reached by applying a key) that allows data to be read. To achieve self-encryption, an unformed ReRAM device is first put into a range of “actual” states using Forming, Reset, and Set operations, as shown in Fig. 1(c); the CC used during these steps controls the width of the filament, and thus the resistance. Next, a state-dependent Reset operation, denoted by $P$, forces each device into the same high-resistance “masked” state. Reverse engineering of the masked data is difficult as filament geometry changes rely on atomic-scale rearrangements. For example, we expect the data to be robust to state of the art optical attacks [36], [37], which have proven detrimental for many CMOS-based memory technologies [38]. Finally, the correct key (another programming function $P^{-1}$) can modify the filament geometry in the masked state to reveal the data, as shown in Fig. 1(c). In this example, one can see that the two devices are first Set to states of 1 k$\Omega$ and 500 $\Omega$ with respective CC values, then Reset to the same state (5 k$\Omega$), which is referred to as the masked state. Observe that the original data is still stored in filament morphology even though an attacker would electrically read the same masked state for both devices. The actual state can be revealed when an appropriate Set voltage is used (without CC), as the filament...
would track the original filament morphology governed by the history-dependent phenomenon.

A. INITIAL MASKING EXPERIMENTS

Preliminary experimental results from our custom ReRAM devices demonstrate reliable data masking and de-masking. Figs. 2(a)-(b) show an example. The figures show resistance values of ReRAM versus read voltage. Binary data is stored in states 500 $\Omega$ in (a) and 1 k$\Omega$ in (b) (‘1’ and ‘0’, respectively); both of these are ‘Set’ states (labeled with green markers) achieved by using two different CC. The error bar shows measurement across different devices on the same wafer. Next, these states (and hence the stored data) were masked to the same state of $\sim$7 k$\Omega$ using two different reset voltages for (a) and (b), shown by ‘Reset’ in orange markers. Note that both devices will read the same resistance in this “masked” state, thus making 1 and 0 indiscernible. Next, to reveal the state, a valid user applies a precise Set voltage (originally used to set the devices). Using this technique, the masked state can be successfully revealed, as apparent from the purple markers for “state-reveal” in both cases. Note that CC was not necessary to reveal the state. This is because a precise set voltage leads the filament to reform in its original morphology, thus, resulting in history-dependent retrieval of the state when proper set voltage is applied. These observations were recorded from ReRAM devices fabricated using a Ru(BE)/ALD-HfO$_2$ (MO)/Zr/W(TE) stack, as in our earlier work [31].

B. ATTACK COMPLEXITY ANALYSIS

While promising, the data masking and de-masking capabilities shown in Fig. 2 are not sufficient for secure memory. There are two main issues. The first is that it is not possible to completely hide state information since a device will respond to an input voltage based on its history. Given a set of devices Formed to one of $n$ states (to store data) and then Reset to mask the chosen state, anyone with hardware access can probe the devices with constant stress signals and measure the difference in responses among the devices to map them into bins, regardless of the key normally used to reveal the hidden states (i.e., data). In this case, guessing the relationship between masked and hidden states only requires $O(n!)$ operations, which is not large since the maximum value of $n$ for reliable data encoding is limited by device-device variability due to i) random filament morphology during write, and ii) random telegraph noise (RTS). Thus, given enough statistical information (i.e., from at least $n!$ devices), it is relatively easy for an attacker with physical access to the memory to decode the stored data via electrical probing. One can of course encode the data prior to storage to ensure that the decoded data is not plaintext but ciphertext; however, in this case the ReRAM memory provides little additional security compared to any other storage technology. The second problem is that the de-masking function (i.e., key) is data dependent, as shown in Fig. 2(b). Since using such a ‘key’ requires knowledge of the data, it is not a true key. In effect, we have simply shifted the secure storage problem from one hardware device (our ReRAM) to another.

A partial solution to these issues may be obtained as follows. Instead of using the key to map the masked (secondary) device states back to the $n$ original (primary) states, we can map them to a larger set of $m$ tertiary states ($m > n$), as shown in Fig. 2(c). In this approach, we recover the data by probing the masked device with $n$ different read voltages $V_k$; the latter constitutes the key. Due to the history-dependent nature of the device response, the final resistance (i.e., the tertiary state) can take one of $m = n^2$ different values. These values can then be mapped to the stored data via a simple look-up table, as shown in the figure. This approach allows the data and key to be decoupled, as desired. However, an attacker can still electrically probe the masked states of a large set of devices with a fixed read voltage to obtain $n$ different resistance values that reveal the stored data (assumed to be ciphertext, i.e., encoded by the key prior to storage). In the next section, we describe an ReRAM data encoding approach that addresses these weaknesses.
III. PROPOSED APPROACH

A. THREAT MODEL

We assume that the attacker i) has physical access to the proposed NVM (which may be used, e.g., to store secure data or keys in an IoT device), and ii) can employ both electrical and optical probing methods to reverse engineer the NVM with the intent of stealing and/or tampering with the data. Available methods include i) multi-channel mixed-signal automatic testing equipment for integrated circuits, ii) nanoprobes for device-level electrical characterization, iii) nanoscale imaging methods (X-ray, scanning electron microscopy, and light microscopy), and iv) thermal cycling systems.

B. REQUIREMENTS FOR SECURE DATA STORAGE

Secure data storage relies on using a complex key comprised of a string of bits to convert the plain text message into some unreadable format. The plain text can then be retrieved by applying the reverse algorithm on the data with the same or ‘symmetric’ key. Other uses of cryptography are in secure data transfer over insecure connections, for which asymmetric or public-private key cryptography (e.g., an algorithm such as RSA) is commonly used. Here we focus on symmetric key encryption/decryption, which is often implemented with the AES (Advanced Encryption Standard) algorithm with a key length of 128, 192, or 256 bits [39].

A major property of AES or any symmetric encryption algorithm is the use of a large (typically, at least 128-bit) key for converting data to the ciphertext, and then using the same key for decryption. Guessing the key and running the decryption process would take an infeasible number of guesses, and therefore time. The key itself should also be completely uncorrelated with the message being encrypted. Thus, the central issue in symmetric encryption is to create a suitable data manipulation function defined by the key. An ideal function should be complex enough mathematically to be infeasible to invert without the key, yet easy to encrypt and decrypt when the key is available. High time complexity for key guessing, ability of users to generate keys, and the use of uncoupled message/key pairs are other requirements for ReRAM-based intrinsically secure data storage.

C. DATA STORAGE PROCEDURE

Here we propose a solution to the previously-described security problems with ReRAM-based self-encrypted NVM. For clarity, we need to clearly distinguish between the stored bit \( b_i \) and the resistance of the device \( R_j \). The goal is to store a limited number of bits per device - low enough to ensure reliable recovery of the bit. However, we should also be able to program many resistance states by varying the applied Forming voltage, which is indeed possible with our ReRAM devices (see Fig. 1(b)). Given this capability, the basic concept for secure storage is to Form the device to an (ideally) random resistance state based on the key value, as shown in Fig. 3(a), instead of one of a small set of resistance states as before. The figure shows that stored data (here represented by a 2-bit value \( b_i \)) is combined with the key bits \( k_i \) to determine the range of resistance values \( s_j \) after Forming; in particular \( s_i = (b_i \oplus k_i) \) where ‘\( \oplus \)’ denotes bitwise-XOR. However, the actual Formed value within this range (denoted by \( R_j \)) is randomized by combining \( s_j \) with the (scaled) output of a true random number generator (TRNG). As a result, an attacker cannot map the devices into resistance bins to guess the relationship between masked and hidden states, which greatly improves data security.

In general, we can define a suitable mapping (encoding) function \( R_j = f(b_i, k_i) \) between the two domains (bits and resistances), where \( k_i \) is the key value. For example, a mapping function for 1-bit data and key variables can of the form \( R_j(s_i) = R_0 P(m_i, \sigma) \) where \( s_i = (b_i \oplus k_i) \), \( R_0 \) is a scaling constant, and \( P(m_i, \sigma) \) denotes a sample taken from one of two probability distribution functions (PDFs) defined by the TRNG; these PDFs have different mean values of \( (m_1, m_2) \), but the same standard deviation of \( \sigma \) for simplicity. Such a function can be created by i) computing the bitwise XOR of \( b_i \) and \( k_i \); ii) using the result \( s_i \) to determine the optimal values of \( (m_1, m_2) \) and \( \sigma \) (via a lookup table); iii) using the TRNG to generate a sample from a suitable PDF; iv) scaling the TRNG output using \( m_i \) and \( \sigma \) to generate \( R_j \) for the particular device in question; and v) using another lookup table \( g(R_j) \) to convert \( R_j \) to the applied programming voltage \( V_j \) for the device. This process is simple to implement in hardware: it only requires an XOR gate, a TRNG, and a DAC, all of...
which can be shared across multiple devices (e.g., one per column).

One of the key implementation issues is the optimal selection of parameters (mean and standard deviation) for the individual PDFs. By optimal, we mean values that make the stored resistances $R_j$ hard to reverse-engineer while also minimizing the probability of making bit errors during decoding. The combined PDF for $R_j$ is $\sum_{i=1}^{N} p_i(m_i, \sigma_i)$, where $N$ is the number of encoded states (such that each device stores $\log_2(N)$ bits), and $p_i$ is the probability of each state (for random data, equal to $1/N$). We typically want to select values of $m_i$ and $\sigma_i$ which ensure that i) the combined PDF for $R_j$ across the array approaches a featureless (i.e., uniform) distribution over a specified range to minimize information leakage to attackers; and ii) the individual distributions $P(m_i, \sigma_i)$ have minimal overlap (since such overlaps cause decoding errors).

The proposed mapping process can only be defined with a certain precision because of inevitable device-device mismatch, finite precision of the write voltages, etc. Thus, in reality the mapping process can be better modeled as $R_j = f(b_i, k_i) + R_n$, where $R_n$ is a "noise" variable that is ideally uncorrelated with the data. Fig. 3(b) shows an example of typical individual and combined distributions for $N = 4$. The TRNG is assumed to generate bits with a uniform PDF, but the corresponding PDFs for the individual resistance states $R_j$ are somewhat broadened by the mismatch and other error sources $R_n$, which are assumed to have a Gaussian distribution. Nevertheless, the combined PDF is nearly uniform and featureless, as desired. Note that the additive noise model used here may not always be valid; for example, correlated and multiplicative noise terms may be present in the ReRAM array. However, it should still be possible to approximate a uniform PDF by optimizing the choices of $m_i$ and $\sigma_i$.

D. DATA RECOVERY PROCEDURE

In order to recover the stored data from the measured resistance $R_j$, we apply a different de-mapping (decoding function), i.e., $b_i = h(R_j, k_i)$ where $k_i$ is the key value, as before. This function can also be implemented using analogous hardware. For example, in the 1-bit case, we can use the mapping $h(d_i, 0) = 0$, $h(d_i, 1) = 1$, $h(d_i, 0) = 1$, and $h(d_i, 1) = 0$ where $d_i = (R_j > R_{th})$ and $R_{th}$ is a global threshold value that may be considered part of the key and can also be adapted with time (e.g., to compensate for temperature variations). Such a mapping function can be implemented simply by using a comparator (to evaluate $d_i$), and an XOR gate, as shown in Fig. 3(c). Note that the output bit is generated with a finite bit error rate (BER) due to the non-zero probability that device mismatch or readout errors cause overlap between the stored states, as shown in Fig. 3(b). Error control coding (ECC) bits can be added to the stored data to detect and correct these errors, analogous to that used in other commercial memory technologies such as optical discs [40] and NOR/NAND flash [41]; methods for implementing ECC are discussed later.

E. INCREASING SECURITY TO REVERSE ENGINEERING ATTACKS

The self-encryption procedure shown in Fig. 3 can be further optimized to ensure greater resilience to reverse engineering attacks. One approach is to increase the complexity of the mapping function $R_j = f(b_i, k_i)$. In our earlier example (shown in Fig. 3(b), for simplicity we assumed an ordered mapping between the encoded data $s_i$ and the corresponding PDFs (resistance ranges). However, it is possible to further increase the entropy of the combined PDF (assuming the "noise" variable $R_n$ is small enough) by using a randomized mapping function with multi-segment PDFs ranges, as shown in Fig. 4. In this example, each 2-bit data value ($s_i = [0..3]$) corresponds to a color-coded PDF with two disjoint segments, with each segment having a random mean and variance; note that these global variables can be considered to be part of the key. The resulting combined PDF has a nearly ideal distribution, as shown on the figure.

Implementing such generalized mapping functions requires slightly more complicated hardware. The encoding circuit now requires multiple uniformly-distributed random variables to define the resistance of each device, so additional TRNGs will be required. Similarly, the comparator in the decoding circuit (see Fig. 3(c)) has to be replaced by a moderate-resolution analog-to-digital converter (ADC).
F. KEY COMPLEXITY ANALYSIS

Complexity and security of the key used by our self-encryption procedure is evaluated based on a few theoretical assumptions. The key consists of two parts: the symmetric key $k_i$ used to recover each hidden data bit and the mean and variance values of all PSD regions (segmented by the $R_{th}$ thresholds) used for random selection of the reset voltage. Selecting those mean and variance values can be reduced to selecting threshold positions from a pool of all accessible resistance states. The number of possibilities can be computed as $C(m - 1, n)$ where $m$ is the number of distinct resistance states and $n$ is the number of $R_{th}$ thresholds. Since all the resulting regions are encoded to $\log_2(N)$-bit data for each device with no repetition occurring between two adjacent regions, the possibilities for the data can be counted based on the condition that $N$ possibilities are available for the first region and $N - 1$ possibilities for all the remaining regions. Attackers thus need an average of $N \times (N - 1)^{m}$ attempts to correctly guess the mapped data for a total of $n$ regions.

We assume that $n$ PSD thresholds $R_{th}$ are used to divide the ReRAM resistance range into $n + 1$ regions of $N$-bit states with no identical states for adjacent regions. To get the correct $i$-bit symmetric key $k_i$ and recover a total of $i$-bits of encrypted data stored within an entire memory block, an attacker will require an average of $G(n)$ guesses, where $G_n$ is given by

$$G(n) = N \times (N - 1)^{n} \times C(m - 1, n) \times 2^i.$$  

(1)

Here $n$ is the number of thresholds (i.e., $R_{th}$ values); $N$ is the number of encoded states given that $\log_2(N)$-bit data are stored on each device; $m$ is the number of distinct resistance states, and $i$ is the number of input data bits to be encoded. Here, $C(m - 1, n)$ is the number of ways to choose $n$ thresholds from the resistance states $m$. The equation shows that the security level of ReRAM-based secure data storage increases when either the number of thresholds or the data size increases. Finally, the complexity of the symmetric key can be exponentially improved by distributing data bits across multiple ReRAM devices. For example, suppose that each data bit defines the parity of a random $r$-bit number. When such numbers are stored in the NVM instead of the actual data bits, the average attack complexity increases to $[G(n)]^r$. This makes the encrypted data much more robust to attacks, albeit at the cost of lower storage density.

G. INTEGRATED ERROR CORRECTION METHODS

In this section, we address the issue of bit errors due to overlapping PDFs in the masked state. Assuming Gaussian PDFs, the resulting BER is proportional to $\text{erfc}(\Delta R/R_n)$ where $\Delta R$ is the distance between the mean values of adjacent PDFs and $R_n$ is their standard deviation. Erasure coding can be used to detect and correct these inevitable errors. Erasure coding is a data protection method in which data is encoded into separate data and parity blocks that are then stored across locations or storage nodes. It is compute intensive but has much lower storage overhead than replication-based data protection methods such as RAID or mirroring [42]. Reed-Solomon (RS) codes are the most popular erasure codes; they use matrix multiplications in a Galois Field ($GF(2^w)$ for $w$-bit words) for encoding and decoding [43]. The large computational overhead of these matrix operations can be reduced by using Cauchy-type RS (CRS) codes, which use 1-bit words to replace integer multiplications with simple bitwise XOR operations in $GF(2)$ [44]. CRS erasure coding can be efficiently implemented within ReRAM cross-bar arrays by exploiting their ability to perform in-memory matrix-vector multiply and add operations [45].

A $(k, m)$ erasure code encodes data blocks ($k$ words long) into longer blocks ($k + m$ words long) by adding $m$ parity words, which allows up to $m$ word errors to be detected, and up to $\lfloor t/2\rfloor$ word errors to be corrected. Assuming $w$ bits per word, the encoding process can be written as the matrix-vector multiplication $c = G^T d$ where $d$ is the $kw$-bit original data vector, $c$ is the $(k+m)w$-bit encoded data vector, and $G^T$ is the $(k+m)w \times kw$-bit generator matrix. The first $kw$ rows of $G^T$ are an identity matrix (such that the original data bits remain unchanged), so only the last $mw$ rows (denoted by the sub-matrix $BG^T$) are needed during encoding. In the case of CRS codes, $BG^T$ is a binary matrix, as shown in Fig. 5. Similarly, the decoding operation can be written as $s = H^T c$, where $s$ is the original data.
where $H$ is the $(k+m)w \times mw$-bit parity matrix (which can be easily derived from $BG^T$) and $s$ is the syndrome vector that encodes the locations of word errors.

The two main operations required for CRS matrix-vector multiplications (MVMs) are dot products and summations. Implementing the summations in $GF(2)$ using ReRAM is the main problem; simply adding up device currents (as in a traditional MVM unit) does not work. Here we propose a time-multiplexed approach to solve this problem, as shown in Fig. 6 for the encoding step. In our method, a single bit of the input data vector $d$ is fed into the ReRAM array on each clock cycle, with the other bits held at zero. The array computes the dot products, after which the current flowing in each column is sensed, compared with a threshold, and XOR-ed with the output from the previous cycle to compute the $GF(2)$ summation. The required circuitry is simple, as shown in the figure, and can be used for both encoding and decoding; however, $mw$ clock cycles are now required to compute the output vector, which results in reduced speed compared to a fully-parallel MVM.

$H$. TEMPERATURE-DEPENDENT ERROR CORRECTION

In this section, the effect of temperature variations on memory decryption performance is evaluated. Here, the High Resistance State (HRS) device model from [46] is used to predict the compliance current density and ReRAM resistance states at different reset voltages and device temperatures. This model indicates that density of the current flowing through the filament in the HRS state can be defined as a function of device temperature as follows [46]:

$$J = n_{trap} \times q \times \mu \times E_i \times e^{(-\phi_B + \frac{qV_r}{kT})},$$  \hspace{1cm} (2)

where $n_{trap} = 112 \text{ cm}^{-3}$ is the local trap density; $q = 1.6 \times 10^{-19} \text{ C}$ is the elementary electric charge; $\mu = 10 \text{cm}^2/\text{V} \cdot \text{S}$ is the electron mobility; $\phi_B$, the barrier height of the defects through the re-oxidized $\text{HfO}_2$ in HRS, is $0.259 \text{ eV}$; $\epsilon_i = 1.77 \times 10^{-12} \text{ F/cm}$ is the permittivity of the material; $k$ is Boltzmann’s constant; $T$ is the device temperature in Kelvin; and $E_i$ can be expressed as:

$$E_i = \frac{V_{\text{read}}}{I_{\text{reox}}}. \hspace{1cm} (3)$$

Here, $V_{\text{read}} = 0.2 \text{ V}$ is the voltage applied when reading the resistance value from the ReRAM device and $I_{\text{reox}}$ is the thickness of the re-oxidized filament. The value of $I_{\text{reox}}$ at different reset voltages can be approximated by a linear function of the reset voltage. For the devices studied, the function $I_{\text{reox}} = -0.33V_{\text{reset}} + 0.42$ (shown in red) fits the measured data at reset voltages from $-1.5 \text{ V}$ to $4 \text{ V}$, as shown in Fig. 8(a) [46]. The current that flows through a filament with a cross-sectional area $A$ is then calculated as $I = J \times A$, and the temperature-dependent resistance states are computed based on Ohm’s law, i.e., $R = V_{\text{read}}/I$.

This ReRAM device model is used to estimate the readout resistance at various reset voltages and device temperatures. The resistance state is expressed as the ratio of resistance at a certain temperature to a reference temperature of $25^\circ\text{C}$, the temperature at which memories are assumed to be encoded. Since temperature variations can induce memory decryption errors, one potential method to compensate for the adverse effect is to use an on-die temperature sensor to keep track of the device temperature when the NVM is being read. The readout resistance values can then be corrected by applying the inverse of the estimated resistance ratio to the decryption algorithm used by the NVM at the measured temperature.

IV. EXPERIMENTAL RESULTS

A. DATA ENCODING AND DECODING

The self-encryption approach described in the previous sections was verified using experimental data from the Ru(BE)/ALD-$\text{HfO}_2$ (MO)/Zr/W(TE) ReRAM devices described in Section II. Fig. 7(a) shows photographs of the fabricated ReRAM devices, each of which has dimensions of $50 \mu\text{m} \times 50 \mu\text{m}$. Simulations of device-level encoding and decoding based on the experimental data were performed using a high-level language (Python). Our main goals were to evaluate i) the entropy achieved after data encryption, and ii) the BER of the recovered data.
The original experimental results were acquired by resetting a total of 15 custom ReRAM devices to different resistance states using 14 different voltage levels. All devices were initially electroformed using a voltage sweep from 0 to 2.5 V with a compliance of 60 µA. Thereafter, the first Reset was performed by sweeping from 0 to −2.4 V. Next, Set was performed at 1000 µA. Finally, multiple resistance states were obtained by resetting with different voltages, as shown in the legend of Fig. 7(b). This figure shows histograms of the measured resistance for each reset voltage. Each histogram was fitted to a Gaussian distribution at different reset voltages, and alternating distributions were removed to reduce inter-state overlap. Thus, only 6 voltage-resistance mapping distributions were ultimately used for key design and encryption. Uncertainty in the mapping distributions (for example, due to resistance measurement errors or mismatches between the 15 ReRAM devices that were studied) was modeled using a Monte Carlo method, i.e., repeating each data encoding experiment 50 times subject to random changes in the mean and standard deviation of each distribution. Specifically, each mean value was varied within a range of −5 kΩ to 5 kΩ, while each standard deviation was scaled by a factor ranging from 0.8× to 1.2×.

Each distribution was mapped to a n-bit binary number based on the preset thresholds \(R_{th}\). Groups of \(n\) bits were then encoded by randomly sampling one resistance state from the distribution corresponding to that \(n\)-bit number. One pattern of \(R_{th}\) values is shown in Fig. 7(c); in this case 4 thresholds are used to segment the available resistance range into 5 bins of alternating binary numbers for encoding single bits (i.e., \(N = 2\) states). In an ideal case, thresholds are placed only at the intersections of adjacent distributions so as to minimize BER after using the chosen reset voltage to program the devices to the desired masked resistance states. Successful encoding of the input data results in a distribution of readout resistance states. The encryption was assumed to be implemented at room temperature, i.e., at 25°C. These states were analyzed for a varying number of thresholds by using histograms. As an example, Fig. 7(d) shows the distribution of the resistance states when 3 \(R_{th}\) thresholds are applied. Visually, this distribution is close to an ideal uniform distribution with no obvious patterns that can be manipulated by attackers during RE attacks.

Quantitatively, the security of the encoded resistance states can be evaluated by using the entropy of the distribution. For this purpose, we calculated the entropy of the simulated distribution for a varying number of thresholds (i.e., \(R_{th}\) values) and compared the result to the entropy of an ideal (i.e., uniform) distribution. The entropy \(H(X)\) of a discrete random variable \(X\) with outcomes of \(x_1, x_2, \ldots, x_n\) which occur with probabilities of \(P(x_1), P(x_2), \ldots, P(x_n)\) is defined as

\[
H(X) = - \sum_{i=1}^{n} P(x_i) \log_b P(x_i) .
\]

Assuming that \(P(x_n)\) is calculated based on the probability density functions that fit the encoded set of resistance states, the measured entropy (including error bars) of the resistance states stored on the devices is shown in Fig. 7(c) as a function of the number of thresholds. Here, the error rates are evaluated based on 50 encryption trials with different random generator seeds. The same method of quantifying simulation errors is used for all the simulation procedures that follow. The entropy of the resistance states is approximately 91% − 97% of the entropy of a uniform distribution. These results indicate that the complexity of guessing the masked...
resistance states is close to the maximum possible value (i.e., the number of guesses required to decode data from a uniform distribution). In addition, we calculated the BER of data decryption (including error bars) for a varying number of thresholds (i.e., \( R_{th} \) values) to evaluate the accuracy of our data encryption method. Bit errors occur when ReRAM devices are reset to unexpected resistance states due to overlaps between distributions. The results in Fig. 7(f) demonstrate that the accuracy of data recovery is over 94.4% ± 1.8% (i.e., BER < 5.6% ± 1.8%) when less than 6 thresholds are configured for data encryption. Thus, a modest amount of error correction is sufficient for reliable data storage when 1-5 threshold values are used.

B. KEY COMPLEXITY

By using the simulation parameters of our model, we can also calculate how many guesses are required, on average, for attackers to hack the key information. The experimental data in Fig. 7(b) reveals that the resistance states have an average standard deviation (e.g., due to device mismatch) of \( \sigma_R \approx 4.42 \, \Omega \), resulting in a total of \( m \approx 34 \) distinct states over the available range from 1.4-150 kΩ. Thus, a total of \( C(33, 4) = 4.092 \times 10^4 \) threshold positions are available within each device to store 1 bit data (\( N = 2 \)) using \( n = 4 \) thresholds. Thus, the estimated number of key guesses \( G(n) \) for each device (\( i = 1 \)) is approximately \( 1.6368 \times 10^5 \), which corresponds to a key entropy of 17.3 bits. This term is the unique contribution of the proposed device-level encryption method. In addition, key entropy also increases with the data block size \( i \), as seen from eqn. (1). As a result, using larger blocks will further increase \( G(n) \) and thus make reverse engineering of the encoded data practically infeasible.

C. EFFECTS OF TEMPERATURE CHANGES

The temperature-dependent HRS ReRAM model was also employed to predict the joint effect of device temperature and reset voltage on resistance states read from the ReRAM devices at 0.2 V. The temperature values were swept from \(-5^\circ \)C to \( 70^\circ \)C and the applied reset voltage applied was varied from \(-1.1 \) V to \(-2.4 \) V. A 2-D heat map of the ratio of resistance at various temperature to that at \( 25^\circ \)C for a reset voltage range of \(-1.1 \) V to \(-2.4 \) V is shown in Fig. 8(b). The results indicate that a ReRAM device behaves similarly to a semiconductor in the HRS state, i.e., the readout resistance decreases monotonically as the temperature increases. For our devices, the predicted resistance states at different temperatures vary over a relatively modest range with reference to the resistance at \( 25^\circ \)C (from 0.85 \( \times \) to 1.15 \( \times \)). The same memory encryption procedure as explained before was then used for simulation.

For clarity, we denote the ambient temperature during memory read and write operations as \( T_{\text{read}} \) and \( T_{\text{write}} \), respectively. The ReRAM device temperature, denoted by \( T_d \), remains approximately equal to \( T_{\text{read}} \) during the read process since the latter dissipates negligible power within the device. However, it transiently exceeds \( T_{\text{write}} \) during the write process since reprogramming the resistance states dissipates a significant amount of power, leading to self-heating [47].

The distributions of the readout resistance states for different values of \( T_{\text{read}} \) were analyzed through histograms for a varying number of thresholds (i.e., \( R_{th} \) values). Figs. 8(c)-(d) demonstrate the distributions of the resistance states at the minimum and maximum read temperatures (\(-5^\circ \)C and \( 70^\circ \)C, respectively), when 3 thresholds are applied for both cases. It is interesting to compare these resistance state distributions to the distribution when no temperature changes are
considered (an example of which is shown Fig. 7(c)). Changes in $T_{\text{read}}$ have little effect on the overall shape of the distribution; both distributions in Figs. 8(c)-(d) remain close to the ideal (i.e., uniform) distribution. Specifically, their entropy is approximately $89.7 \pm 3\%$ to $95.3 \pm 1.4\%$ of the uniform distribution for different $R_{\text{th}}$, which is similar to the entropy at the reference temperature. However, due to the change in absolute resistance values with temperature, the BER of data decryption at both temperature extrema is significantly increased, as shown in the blue curves of Figs. 8(e)-(f). Fortunately, adjusting the $R_{\text{th}}$ values with temperature can effectively restore BER to its original values, as shown in the red curves. These results suggest that temperature compensation is necessary to obtain acceptable BER values when the temperature varies between ReRAM read and write operations (i.e., $T_{\text{read}}$ differs from $T_{\text{write}}$).

To further evaluate the effects of ReRAM self-heating on memory encryption, we acquired new voltage-resistance mapping distributions averaged over different values of device temperature, $T_d$, by integrating the device temperature coefficient from $-5^\circ\text{C}$ to $70^\circ\text{C}$ over the distributions in Fig. 7(b). Specifically, the data generated for each temperature value in this range are combined for each reset voltage and plotted using new sets of histograms. The newly generated histograms are again fitted by Gaussian distributions with alternating distributions removed to minimize inter-state overlaps. Since resistance data at all the write temperature conditions are now fitted by Gaussian distributions for each reset voltage, only 5 distributions remain, as shown in Fig. 9(a). The distributions of the readout resistance are then analyzed through histograms for a varying number of thresholds (i.e., $R_{\text{th}}$ values). Figs. 9(b)-(c) illustrate the histograms of the resistance states at $T_{\text{read}} = -5^\circ\text{C}$ with and without read temperature compensation. Even though the histograms in Figs. 9(b)-(c) seem to have less random distributions as compared with those in Figs. 8(c)-(d), they are still very close to uniformly distributed resistance states in terms of entropy. Specifically, they exhibit relative entropy values of $87\% \pm 2\%$ (considering the error bars) compared to a uniform distribution for a varying number of thresholds, as demonstrated in Fig. 9(d). As in Fig. 8 (which did not consider the effect of $T_d$ variations during write operations), the BER for data decryption at both read temperature extrema ($T_{\text{read}} = -5^\circ\text{C}$ and $70^\circ\text{C}$) increases significantly when the device temperature coefficient is considered, as shown in the blue curves of Figs. 9(e)-(f). However, $R_{\text{th}}$ values can still be adjusted based on the observed value of $T_{\text{read}}$ to effectively reduce BER to values close to those obtained without $T_{\text{read}}$ changes, as shown in the red curves. These results suggest that the BER obtained after $T_{\text{read}}$ compensation is low enough for high-quality data encryption even self-heating effects are considered.

V. CONCLUSION

We have described a self-encryption technique that is designed to provide low-overhead data protection for emerging ReRAM-based NVM technologies. The proposed approach exploits the intrinsic device physics of ReRAM devices to hide the stored data, making it well-suited for energy-constrained platforms such as IoT devices. Specifically, it relies on storing data in the filament morphology of ReRAM, which is difficult to reverse engineer as the changes take place at the atomic level. Simulations using experimental data from Ru(BE)/ALD-HfO$_2$ (MO)/Zr/W(TE)
ReRAM devices shows that device-level encryption can provide significant key entropy (up to \( \approx 17.3 \) bits). In addition, temperature compensation enables low-BER data decoding over a broad temperature range (\(-5^\circ C\) to \(70^\circ C\) in our simulations) even when self-heating effects during ReRAM write operations are considered. Future work will focus on i) integrating the ReRAM devices with access transistors to realize complete NVM blocks; and ii) developing low-power peripheral circuitry to perform read/write operations that implement the proposed encryption/decryption procedure.

**ACKNOWLEDGMENT**

Dr. Rashmi Jha would like to acknowledge CHEST/UUCRC funded by the National Science Foundation at the University of Cincinnati for research support.

**REFERENCES**

[1] M. Jagasivamani, C. Walden, D. Singh, L. Kang, S. Li, M. Asnaashari, S. Dubois, B. Jacob, and D. Yeung, “Memory-systems challenges in realizing monolithic computers,” in *Proc. Int. Symp. Memory Syst.*, Oct. 2018, pp. 98–104.

[2] M. Jagasivamani, C. Walden, D. Singh, L. Kang, S. Li, M. Asnaashari, S. Dubois, B. Jacob, and D. Yeung, “Analyzing the monolithic integration of a ReRAM-based main memory into a CPU’s die,” *IEEE Micro*, vol. 39, no. 6, pp. 64–72, Nov. 2019.

[3] H.-S. P. Wong, H.-Y. Lee, S. Yu, Y.-S. Chen, Y. Wu, P.-S. Chen, B. Lee, F. T. Chen, and M.-J. Tsai, “Metal–oxide RRAM,” *IEEE Trans.*, vol. 100, no. 6, pp. 1951–1970, Jun. 2012.

[4] Z. Gao, Y. Wang, Z. Lv, P. Xie, Z.-X. Xu, M. Luo, Y. Zhang, K. Zhou, G. Zhang, G. Duan, Y. Zhou, and S.-T. Han, “Ferroelectric coupling for dual-mode non-filamentary memristors,” *Appl. Phys. Rev.*, vol. 9, no. 2, Jun. 2022, Art. no. 021417.

[5] Z. Lv, Y. Wang, J. Chen, J. Wang, Y. Zhou, and S.-T. Han, “Semiconductor quantum dots for memories and neuromorphic computing systems,” *Chem. Rev.*, vol. 120, no. 9, pp. 3941–4006, May 2020.

[6] D. Apalkov, D. Apalkov, A. Khvatskyi, S. Watts, V. Xintis, X. Tang, D. Lottis, K. Moon, X. Luo, E. Chen, A. Ong, and A. Driskill-Smith, “Spin-transfer torque magnetic random access memory (STT-MRAM),” *J. Emerg. Technol. Comput. Syst.*, vol. 9, no. 2, pp. 1–35, May 2013.

[7] H.-S. P. Wong, S. Raoux, S. Kim, J. Liang, P. Reifenberg, B. Rajendran, M. Ashgehi, and K. E. Goodson, “Reflection coefficient of HfO\(_2\)/TiO\(_2\) device modeling,” in *IEEE Electron Device Lett.*, vol. 38, no. 2, pp. 168–171, Feb. 2017.

[8] R. Jha, S. Mitra, H.-S. P. Wong, and S. Wang, “Holding stationary hardware Trojans in their tracks,” *IEEE Spectr.*, vol. 20, p. 2015, Jan. 2015.

[9] Y.-C. Chen, W. Wang, H. Li, and W. Zhang, “Non-volatile 3D stacking RRAM-based FPGA,” in *Proc. 22nd Int. Conf. Field Program. Log. App. (FPL)*, Aug. 2012, pp. 367–372.

[10] H. Jiang, C. Li, R. Zhang, P. Yan, P. Lin, Y. Li, J. Yang, D. Holcomb, and Q. Xiao, “A provable key destruction scheme based on memristor crossbar arrays,” *Nature Electron.*, vol. 1, pp. 548–554, Oct. 2018.

[11] Y. Zhang, L. Xu, Q. Dong, J. Wang, D. Blaauw, and D. Sylvester, “Reconfigurable cryptography in hardware: A feasibility analysis,” *IEEE Int. Solid-State Circuits*, vol. 53, no. 4, pp. 995–1005, Apr. 2018.

[12] D. Liu, C. Zhang, H. Lin, Y. Chen, and M. Zhang, “A resource-efficient and side-channel secure hardware implementation of ring-LWE cryptographic processor,” *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 66, no. 4, pp. 1474–1487, Apr. 2018.

[13] A. A. Panmu, K.-S. Chong, W.-G. Ho, and B.-H. Gwee, “Interception of side channel attack on AES-128 wireless communications for IoT applications,” in *Proc. IEEE Asia Pacific Conf. Circuits Syst.* (APCCAS), Oct. 2016, pp. 650–653.

[14] B. Long, Y. Li, and R. Jha, “Switching characteristics of Ru/HfO\(_2\)/TiO\(_2\)/Ru RRAM devices for digital and analog nonvolatile memory applications,” *IEEE Electron Device Lett.*, vol. 33, no. 5, pp. 706–708, May 2012.

[15] J. Deogirikar and A. Vidhate, “Security attacks in IoT: A survey,” in *Proc. Int. Conf. (I-SMAC)*, Feb. 2017, pp. 32–37.

[16] S. Singh, P. K. Sharma, S. Y. Moon, and J. H. Park, “Advanced lightweight encryption algorithms for IoT devices: Survey, challenges and solutions,” *J. Ambient Intell. Humanized Comput.*, pp. 1–18, May 2017.

[17] W. Diehl, F. Farahmand, P. Yalla, J.-P. Kaps, and K. Gaj, “Comparison of hardware and software implementations of selected lightweight block ciphers,” in *Proc. 27th Int. Conf. Field Program. Log. App. (FPL)*, Sep. 2017, pp. 1–4.

[18] H. D. Lee, S. G. Kim, K. Cho, H. Hwang, H. Choi, J. Lee, S. H. Lee, J. Lee, S. O. Chung, and Y. S. Kim, “Integration of 4F2 selectorless crossbar array 2Mb ReRAM based on transition metal oxides for high density memory applications,” in *Proc. Symp. VLSI Tech.* (VLSIT), Jun. 2012, pp. 151–152.

[19] Z. Fang, X. Wang, X. Li, Z. Chen, A. Kamath, G. Lo, and D. Kwong, “Fully CMOS compatible 1T1R integration of vertical nanopillar GAA transistor and oxide-based RRAM cell for high-density nonvolatile memory application,” *IEEE Trans. Electron Devices*, vol. 60, no. 3, pp. 1108–1113, Mar. 2013.

[20] S. Cortese, “Selector devices/architectures for ReRAM crossbar arrays,” Ph.D. dissertation, Dept. Electron. Comput. Sci., Univ. Southampton, Southampton, U.K., 2017.

[21] B. Long, Y. Li, and R. Jha, “Switching characteristics of Ru/HfO\(_2\)/TiO\(_2\)/Ru RRAM devices for digital and analog nonvolatile memory applications,” *IEEE Electron Device Lett.*, vol. 33, no. 5, pp. 706–708, May 2012.

[22] W. Chen, W. Lu, B. Long, Y. Li, D. Gilmer, G. Bersuker, S. Bhunia, and R. Jha, “Switching characteristics of W/Zr/HfO\(_2\)/TiN ReRAM devices for multi-level cell non-volatile memory applications,” *Semicond. Sci. Technol.*, vol. 30, no. 7, p. 075002, 2015.

[23] S. Cortese, “Selector devices/architectures for ReRAM crossbar arrays,” Ph.D. dissertation, Dept. Electron. Comput. Sci., Univ. Southampton, Southampton, U.K., 2017.

[24] W. Sun, B. Gao, M. Chi, Q. Xia, J. J. Yang, H. Qian, and H. Wu, “Understanding memristive switching via in situ characterization and device modeling,” *Nature Commun.*, vol. 10, no. 1, pp. 1–13, Dec. 2019.

[25] T. H. Nguyen, A. Barua, T. Bailey, A. Rush, P. Kosel, K. Leedy, and R. Jha, “Reflection coefficient of HfO\(_2\)-based RRAM in different resistance states,” *Appl. Phys. Lett.*, vol. 113, no. 19, Nov. 2018, Art. no. 192101.

[26] B. Govoreanu, G. S. Kar, Y. Y. Chen, V. Parashiv, S. Kubicek, A. Fantini, L. Goux, G. Clima, R. Degrave, and N. Jossart, “10×10H/HfO\(_2\) crossbar resistive RRAM with excellent performance, reliability and low-energy operation,” in *IEDM Tech. Dig.*, 2011, pp. 31.6.1–31.6.4.

[27] C. Y. Chen, A. Fantini, R. Degrave, A. Redolfi, G. Groeseneken, L. Goux, and G. S. Kar, “Statistical investigation of the impact of program history and oxide-metal interface on OxRRAM retention,” in *IEDM Tech. Dig.*, Dec. 2016, pp. 4–6.

[28] B. Hajri, H. Azira, M. M. Mansour, and A. Chehab, “RRAM device models: A comparative analysis with experimental validation,” *IEEE Access*, vol. 7, pp. 168963–168980, 2019.

[29] N. Olexa, R. Jha, S. Mandal, and S. Bhunia, “ReRAM-based intrinsically secure memory: A feasibility analysis,” in *Proc. IEEE Nat. Aersnop. Electron. Conf. (NAECON)*, Jul. 2019, pp. 218–225.

[30] N. Vashishtha, M. T. Rahman, O. P. Dizon-Paradis, and N. Asadizanjani, “Is backside the new backdoor in modern SoCs?” in *Proc. IEEE Int. Test Conf. (ITC)*, Nov. 2019, pp. 1–10.
[37] L. Lavdas, M. T. Rahman, M. Tehranipoor, and N. Asadizanjani, “On optical attacks making logic obfuscation fragile,” in Proc. IEEE Int. Test Conf. Asia (ITC-Asia), Sep. 2020, pp. 71–76.

[38] H. Shen, N. Asadizanjani, M. Tehranipoor, and D. Forte, “Nanopyramid: An optical scrambler against backside probing attacks,” in Proc. Int. Symp. Test. Failur Anal., Nov. 2018, p. 280.

[39] H. B. Westlund, “NIST reports measurable success of advanced encryption standard,” J. Res. Nat. Inst. Standards Technol., vol. 107, no. 3, p. 307, 2002.

[40] G. Kong and S. Choi, “Performance evaluation of the Reed–Solomon and low density parity check codes for Blu-ray disk channels,” Jpn. J. Appl. Phys., vol. 49, Aug. 2010, Art. no. 08KB02.

[41] F. Sala, K. A. S. Immink, and L. Dolecek, “Error control schemes for modern flash memories: Solutions for flash deficiencies,” IEEE Consum. Electron. Mag., vol. 4, no. 1, pp. 66–73, Jan. 2015.

[42] H. Weatherspoon and J. D. Kubiatowicz, “Erasure coding vs. Replication: A quantitative comparison,” in Proc. Int. Workshop Peer-Peer Syst. Berlin, Germany: Springer, 2002, pp. 328–337.

[43] S. B. Wicker and V. K. Bhargava, Reed–Solomon Codes and Their Applications, Hoboken, NJ, USA: Wiley, 1999.

[44] L. X. James S. plank, “Optimizing Cauchy Reed–Solomon codes for fault-tolerant network storage applications,” in Proc. 5th IEEE Int. Symp. Netw. Comput. Appl. (NCA), 2006, pp. 173–180.

[45] L. Han, S. Tan, B. Xiao, C. Ma, and Z. Shao, “Optimizing Cauchy Reed–Solomon coding via ReRAM crossbars in SSD-based RAID systems,” in Proc. IEEE Non-Volatile Memory Syst. Appl. Symp. (NVMAS), Aug. 2019, pp. 1–6.

[46] B. Long, Y. Li, S. Mandal, R. Jha, and K. Leedy, “Switching dynamics and charge transport studies of resistive random access memory devices,” Appl. Phys. Lett., vol. 101, no. 11, Sep. 2012, Art. no. 113503.

[47] R. Picos, M. M. Al Chawa, C. De Benito, S. G. Stavrinides, and L. O. Chua, “Using self-heating resistors as a case study for memristor compact modeling,” IEEE J. Electron Devices Soc., vol. 10, pp. 466–473, 2022.

JUNJUN HUAN (Member, IEEE) received the B.S. degree in electronic information engineering and the M.S. degree in electrical engineering from the University of Dayton, Dayton, OH, USA. He is currently pursuing the Ph.D. degree in electrical and computer engineering with the University of Florida, Gainesville, FL, USA. His current research interests include electronics for ultrasound imaging and image-guided treatment, wearable medical sensors, and hardware security.

NICHOLAS OLEXA received the M.S. degree in electrical engineering from the University of Cincinnati, in 2020, with a thesis on ReRAM device operation through a physics-based modeling approach. He is currently an Electrical Engineering Graduate with the University of Cincinnati with a focus in electronics and device physics. He is also with KBR, Columbus, OH, USA, working on communication systems in the GPS domain.

BRETT HOCHMAN received the B.S. degree in electrical engineering from the University of Cincinnati, where he is currently pursuing the M.S. degree in electrical engineering. His research interests include device physics and novel device fabrication. He has nearly ten years’ experience working in the software industry on process automation.

SWARUP BHUNIA (Senior Member, IEEE) received the B.E. degree (Hons.) from Jadavpur University, Kolkata, India, the M.Tech. degree from the Indian Institute of Technology (IIT) Kharagpur, and the Ph.D. degree from Purdue University, IN, USA. He is currently a Professor and a Semmoto Endowed Chair with the University of Florida, Gainesville, FL, USA. Earlier, he was appointed as the T. and A. Schroeder Associate Professor of electrical engineering and computer science with Case Western Reserve University, Cleveland, OH, USA. He has over ten years of research and development experience with over 200 publications in peer-reviewed journals and premier conferences. His research interests include hardware security and trust, adaptive nanocomputing, and novel test methodologies. He received the IBM Faculty Award (2013), the National Science Foundation CAREER Award (2011), the Semiconductor Research Corporation Inventor Recognition Award (2009), and the SRC Technical Excellence Award as a Team Member (2005), and several best paper awards/nominations. He has been serving as an Associate Editor of IEEE TRANSACTIONS ON COMPUTER-AIDED DESIGN OF INTEGRATED CIRCUITS AND SYSTEMS, IEEE TRANSACTIONS ON MULTI-SCALE COMPUTING SYSTEMS, ACM Journal on Emerging Technologies in Computing Systems; served as a Guest Editor for IEEE Design & Test of Computers (2010 and 2013) and IEEE JOURNAL ON EMERGING AND SELECTED TOPICS IN CIRCUITS AND SYSTEMS (2014). He has served in the organizing and program committee for many IEEE/ACM conferences.

RASHMI JHA (Member, IEEE) received the B.Tech. degree in electrical engineering from the Indian Institute of Technology (IIT) Kharagpur, India, in 2000, and the M.S. and Ph.D. degrees in electrical engineering from North Carolina State University, in 2003 and 2006, respectively. She is currently a Professor with the Department of Electrical and Computer Engineering, University of Cincinnati. She has more than 18 years of experience in the areas of advanced CMOS and beyond-CMOS logic and memory devices. She has been granted 13 U.S. patents and has authored/coauthored several publications. Her current research interests include the areas of resistive random access memory devices, ferroelectric field effect transistors, CMOS integration, synaptic memories and neuromorphic computing, and hardware trust and security. She was a recipient of the CAREER Award from the National Science Foundation (NSF), in 2013.

SOUMYAJIT MANDAL (Senior Member, IEEE) received the B.Tech. degree from the Indian Institute of Technology (IIT) Kharagpur, India, in 2002, and the S.M. and Ph.D. degrees in electrical engineering from North Carolina State University, in 2003 and 2006, respectively. He has over ten years of research and development experience with over 200 publications in peer-reviewed journals and conferences and has been awarded 26 patents. His research interests include analog and biological computation, magnetic resonance sensors, low-power analog and RF circuits, and precision instrumentation for various biomedical and sensor interface applications. He was a recipient of the President of India Gold Medal in 2002, the MIT Microsystems Technology Laboratories (MTL) Doctoral Dissertation Award in 2009, the T. Keith Glennan Fellowship in 2016, and the IIT Kharagpur Young Alumni Achiever Award in 2018.

**

104588

VOLUME 10, 2022

IEEE Access

J. Huan et al.: Intrinsically Secure NVM Using ReRAM Devices