IEEE Journal on Exploratory Solid-State Computational Devices and Circuits

Received 7 July 2021; revised 18 August 2021 and 21 September 2021; accepted 5 October 2021.
Date of publication 13 October 2021; date of current version 4 November 2021.

Digital Object Identifier 10.1109/JXCDC.2021.3119489

NeuroSOFM: A Neuromorphic Self-Organizing Feature Map Heterogeneously Integrating RRAM and FeFET

SIDDHARTH BARVE (Student Member, IEEE),
JOSHUA MAYERSKY (Student Member, IEEE), ANDREW J. FORD (Student Member, IEEE),
ALEXANDER JONES (Student Member, IEEE), BAYLEY KING (Student Member, IEEE),
AARON RUEN (Student Member, IEEE), and RASHMI JHA (Member, IEEE)

Department of Electrical Engineering and Computer Science, University of Cincinnati, Cincinnati, OH 45221 USA

CORRESPONDING AUTHOR: S. BARVE (barvesh@mail.uc.edu)

This work was supported in part by the National Science Foundation (REU supplements) under Award CCF-1718428 and Award ECCS-1926465 and in part by the Air Force Research Laboratory and the Southwestern Council for Higher Education under Grant FA8650-19-2-9300.

This article has supplementary downloadable material available at https://doi.org/10.1109/JXCDC.2021.3119489, provided by the authors.

ABSTRACT Many currently available hardware implementations of the unsupervised self-organizing feature map (SOFM) algorithm utilize complementary metal–oxide–semiconductor (CMOS)-only circuits that often compromise key behaviors of the SOFM algorithm due to complexity. We propose a neuromorphic architecture harnessing the unique properties of ferroelectric field-effect transistors (FeFETs) and gated-resistive random access memory (RRAM) for in-memory computing to implement the SOFM algorithm. The FeFET-based synapse, organized in a novel circuit, is able to compute the input-weight Euclidean error in memory via the saturation drain current. The self-decaying states of the gated-RRAM allow for a self-decaying neighborhood and learning rate implementation to allow for convergence and lifelong learning. This novel architecture is able to successfully cluster benchmarks (RGB colors and MNIST handwritten digits) and real-life datasets, such as COVID-19 patient chest X-rays completely unsupervised. The architecture also demonstrates a significant amount of robustness to device variability and damaged neurons. In addition, the proposed architecture is completely parallelized and provides a power-efficient platform for implementing the SOFM algorithm.

INDEX TERMS Ferroelectric field-effect transistor, neuromorphic, resistive random access memory (RRAM), self-organizing feature map (SOFM), unsupervised.

I. INTRODUCTION

THE computational time and power required to implement deep neural networks (DNNs) prevent widespread use in low-power real-time applications, such as wearable devices, sensors, and the Internet of Things (IoT). The problem lies with the high memory bandwidth requirement for DNNs and the Von Neumann bottleneck of current computer architectures [5]. Field-programmable gate arrays (FPGAs) and application-specific integrated circuits (ASICs) have increased memory bandwidth via near- and in-memory computing [5] to accelerate neural network training and inferencing [1]–[3], [8]–[12]. Near-memory computing couples the components that perform computation and memory devices in proximity. In-memory computing houses the memory and computation inside the same unit, resulting in higher memory bandwidth, lower circuit area, and power consumption. Bioinspired neuromorphic architectures are gaining interest for ultralow-power computing [1]–[3], [12]. Although many of the current implementations rely on complementary metal–oxide–semiconductor (CMOS) devices [1], [3], [8]–[11], there has been a significant interest in incorporating emerging analog memory devices for more bioinspired and energy-efficient computation [1], [2], [12]. However, a significant amount of research in developing

ASICs for neural network acceleration has focused on supervised learning, which requires large amounts of labeled data during training. Labeled data are a scarce resource in many applications. This constraint has led to a larger demand for DNNs using unsupervised learning, which requires little or no labeled data [3], [4], [8]–[12]. Kohonen's self-organizing feature map (SOFM) is an example of an unsupervised algorithm that has been investigated for implementation in low-power architectures [6], [7].

In spite of its strengths, there has been limited research in developing neuromorphic hardware for implementing SOFM algorithms [8]–[12]. Many currently available hardware implementations of the SOFM algorithm utilize CMOS-only circuits that are penalized by the limitations of CMOS technology [8]–[11]. For example, CMOS memory devices store a single bit, requiring multiple memory devices per synapse to capture the analog nature of the incoming data. The use of separate memory modules to store weights [8]–[10] results in a lower memory bandwidth and higher energy consumption. Moreover, updating the weights serially [11] further increases computation time, especially for high-dimensional datasets (i.e., the COVID-19 chest X-ray images [14]). The higher computational time and energy requirements prevent real-time application in domains where the rapid energy-efficient computation of results is crucial (e.g., medical diagnosis and autonomous vehicles).

Beyond computational challenges, some of these implementations often compromise key behaviors of the SOFM algorithm. The omission of learning rate decay [8], [9], [11], [12] may prevent the network from properly converging depending on the application and prevents lifelong continuous learning of the SOFM by overtraining the network. In addition, many implementations may rely on external circuitry for core behaviors costing computation time, circuit area, and power consumption [11]. These implementations also require preprocessing of the data for computation (i.e., every input must be normalized [11], [12] or each feature must be trained independently and serially [12]). This preprocessing increases computational time and may require assumptions about the data that are not desired. In addition, current hardware implementations of the SOFM algorithm utilize dot-product similarity instead of pairwise Euclidean distance or error [11], [12]. Pairwise Euclidean distance can capture information on both magnitude and shape of the raw data more accurately than dot-product similarity [13]. Using the Euclidean error as the distance measure allows for the SOFM to cluster the original data with little or no preprocessing and prior assumptions—both of which are beneficial for unsupervised learning.

Interestingly, near- and in-memory computing for DNNs has led to a larger demand for programmable analog memory devices such as resistive random access memory (RRAM) [15] and ferroelectric field-effect transistors (FeFETs) that have been heterogeneously integrated with CMOS neural network circuits [16]. These devices are able to reduce the number of memory devices required by storing nonbinary states.

In this article, we propose a neuromorphic architecture that is able to harness the unique properties of FeFET and gated-RRAM for in-memory computing to implement the SOFM algorithm. With this architecture, we seek to provide a low-power solution for local autonomous clustering of data that can be integrated with sensors and IoT devices. This architecture is referred to as NeuroSOFM. A combination of FeFET and gated-RRAM in conjunction with CMOS circuits allows for the efficient implementation of Euclidean error calculations and learning rate decay that are central to the SOFM algorithm. In addition, using these memory devices, the architecture is completely parallelized and requires little or no preprocessing and external circuitry. The remainder of this article is organized as follows. Section II is an introduction to the SOFM algorithm. Sections III and IV describe the implementation of the FeFET and gated-RRAM memory devices, respectively. Section V details the overall architecture and its behavior. Finally, Section VI illustrates and discusses the results.

II. SELF-ORGANIZING FEATURE MAP

The SOFM is a map of interconnected neurons, which reflects the topology of a learned dataset through competitive learning [6], [7]. The SOFM projects the dataset from a higher dimensional input space into a lower dimensional neuron space, allowing patterns in data to be visualized. Each neuron in the SOFM is comprised of a weight vector of the same dimension as the input space of the dataset, as shown in Fig. 1.

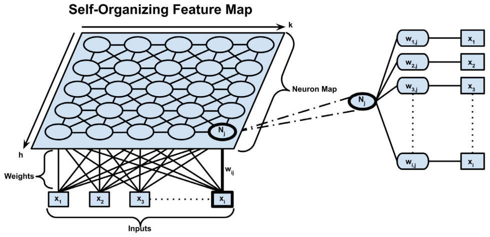


FIGURE 1. **Illustration of SOFM network.**

First, each neuron computes the Euclidean error ($\varepsilon_{ij} = (x_i - w_{ij})^2$) between the each input (x_i) and its corresponding weight (w_{ij}). The total error input-weight Euclidean error for each neuron is then calculated. The neuron with the least total Euclidean error becomes the winning neuron or best matching unit (BMU) the index of which is denoted as j_{BMU}. The BMU is the neuron best representing the input. During learning, every neuron's weight vector moves toward the input vector in relation to a neuron's position in the neighborhood of the BMU. The BMU neighborhood (Λ_j) is often defined using the Gaussian distribution with respect to the Euclidean distance between a neuron and the BMU (e.g., neurons in proximity to BMU experience a larger weight update) and the neighborhood rate (σ). The weight update is scaled by the learning rate (η) and input-weight error. The learning rate and neighborhood undergo exponential decay over time by their corresponding decay time constant (τ) until the network converges. The map captures a topography (clusters) of the data based on feature similarity. The computational simplicity of this algorithm allows for low-power real-time hardware implementation. Although computationally simple, the SOFM is still able to cluster input data completely unsupervised. Our proposed architecture divides the SOFM algorithm into three subcircuits: input-weight Euclidean error computation in the FeFET synapse, BMU selection with CMOS-based modified winner-take-all (MWTA), and self-decaying learning parameter computation with gated-RRAM (neighborhood and learning rate controllers), as shown in Fig. 2. It is apparent that there are many similarities between the software and hardware implementation of the SOFM algorithm. How this was achieved will be detailed in the remainder of this article.

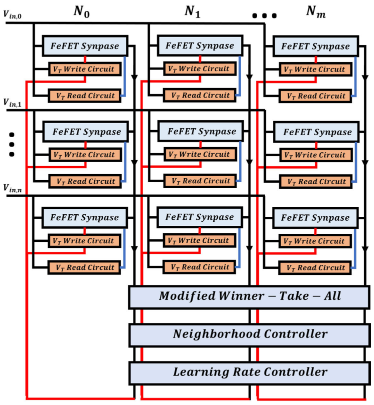


FIGURE 2. Functional block diagram of *n* input by *m* neuron NeuroSOFM architecture.

III. FeFET SYNAPSE IN-MEMORY EUCLIDEAN ERROR COMPUTATION

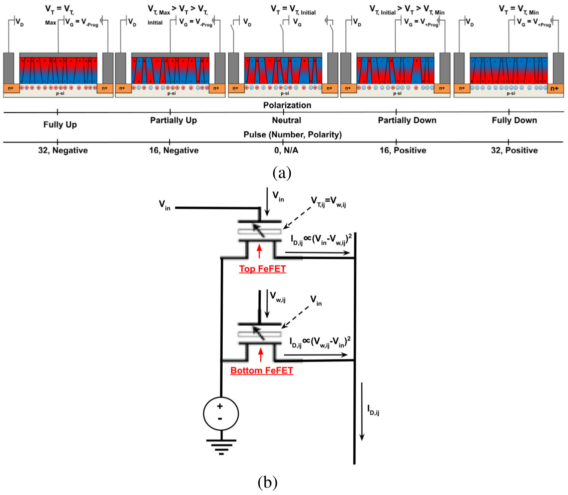
In this section, we demonstrate that nonvolatile multidomain FeFET devices offer an excellent opportunity to implement in-memory computation of the input-weight Euclidean error.

FeFET is a metal–oxide–semiconductor field-effect transistor (MOSFET) device with ferroelectric material integrated in the gate-stack. Ferroelectricity is a permanent-induced polarization in the material due to an externally applied electric field. The magnitude of electric field that causes a change in polarization in the material is known as the coercive field E_c. This permanent remanent polarization, P_r, can arise due to a switchable deformation of a polar ion in the unit cell. Multiple unit cells oriented in the same direction form a ferroelectric domain. The number, orientation, and size of these ferroelectric domains can vary by method of deposition. The individual polarization orientations of each domain may be summed together to form a net polarization of the entire ferroelectric material [17]. By manipulating the polarization in ferroelectric material, the FeFET device can be programmed to manifest multiple threshold voltage (V_T) states, as shown in Fig. 3.

This tunable nature of V_T of FeFETs allows for the weights of the SOFM to be programmed as specific V_T states in the synapses of this architecture. The multistate memory allows for a single FeFET device to be used to store the weight in place of multiple nonvolatile CMOS. The saturation drain current (I_{DS}) of an n-channel FET is given by

$$I_{\mathrm{DS}} = \frac{\mathrm{W}C'_{\mathrm{ox}}\mu}{2L}\left(V_{\mathrm{GS}} - V_T\right)^2, \quad |V_{\mathrm{DS}}| \geq |V_{\mathrm{GS}} - V_T| \quad (1)$$

the width (W), length (L), effective mobility carrier (μ), and oxide capacitance per unit area (C'_{ox}) (F/cm^2) of the FET. In this equation, one can observe that I_{DS} is directly proportional to the Euclidean distance between the applied gate voltage (V_{GS}) and V_T of the FeFET. We capitalize on this intrinsic device characteristics and propose a FeFET-based synapse for in-memory computation of the input-weight Euclidean error via I_{DS}, as shown in Fig. 3(b). When the input ($V_{\mathrm{in},i}$) is applied as the gate bias (V_{GS}) and the weight ($V_{w,ij}$) is programmed as V_T for the top FeFET, the top FeFET produces I_{DS} proportional to the input-weight Euclidean error when $V_{\mathrm{in},i} > V_{w,ij}$. However, when $V_{\mathrm{in},i} < V_{w,ij}$, then $V_{\mathrm{GS}} < V_T$; therefore, the FeFET is no longer conducting ($I_{\mathrm{DS}} \approx 0$) and no longer represents the Euclidean error. To account for this asymmetry in the Euclidean error computation, we added another n-channel FeFET (bottom FeFET). The bottom FeFET's V_T is programmed to $V_{\mathrm{in},i}$, while V_T of the top FeFET (weight) is applied as V_{GS}. Therefore, when $V_{\mathrm{in},i} < V_{w,ij}$, the bottom FeFET produces I_{DS} proportional to the Euclidean error. The behavior of this FeFET synapse is modeled using the following equation:



$V_T = V_{T,\mathrm{Max}}$
$V_{T,\mathrm{Max}} > V_T > V_{T,\mathrm{Initial}}$
$V_T = V_{T,\mathrm{Initial}}$
$V_{T,\mathrm{Initial}} > V_T > V_{T,\mathrm{Min}}$
$V_T = V_{T,\mathrm{Min}}$
V_D
$V_G = V_{-\mathrm{Prog}}$
V_G
$V_G = V_{+\mathrm{Prog}}$
n+
p-si
Polarization
Fully Up
Partially Up
Neutral
Partially Down
Fully Down
Pulse (Number, Polarity)
32, Negative
16, Negative
0, N/A
16, Positive
32, Positive
V_{in}
$V_{T,ij}=V_{w,ij}$
$I_{D,ij} \propto (V_{in}-V_{w,ij})^2$
Top FeFET
$V_{w,ij}$
$I_{D,ij} \propto (V_{w,ij}-V_{in})^2$
Bottom FeFET
$I_{D,ij}$

(a)

(b)

FIGURE 3. (a) Ferroelectric domain switching and threshold voltage as a function of programming pulses. In the presence of $V_{\mathrm{GS}} \geq V_{\mathrm{Programming\ Voltage}}$, the polarization of the domains the ferroelectric material aligns to oppose the electric field produced by the gate bias. The aligning of the domains results in a saturation of minority charge carriers ($V_T > V_{T,\mathrm{Initial}}$) or majority charge carriers ($V_T < V_{T,\mathrm{Initial}}$) in the channel between the diffusion wells, which changes the threshold voltage of the FET. (b) Circuit diagram of FeFET pairwise Euclidean error calculation.

$$\begin{aligned}
&\text{if } V_{\mathrm{in},i} > V_{w,ij}\\
&\text{Top FeFET: } I_{D,ij} \propto \left(V_{\mathrm{in},i} - V_{w,ij}\right)^2\\
&\text{if } V_{\mathrm{in},i} < V_{w,ij}\\
&\text{Bottom FeFET: } I_{D,ij} \propto \left(V_{w,ij} - V_{\mathrm{in},i}\right)^2. \qquad (2)
\end{aligned}$$

To the best of our knowledge, this work is the first to report in-memory computing of Euclidean error in FeFET devices utilizing I_{DS} to compute a cost function, or Euclidean error, between the input and the weight. Cost function computation is an integral part of many supervised networks as well. In-memory computation of Euclidean error via FeFETs may allow for low-power real-time hardware implementation of conventional backpropagation techniques.

A. READING AND WRITING V_T OF FeFET DEVICE

External circuitry would be required to read V_T of the top FeFET and program the V_T states of the FeFETs, as shown in Fig. 2.

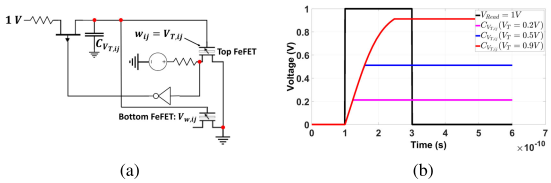


FIGURE 4. (a) Circuit diagram of the proposed V_T read circuit. (b) LTspice simulation of the V_T read circuit.

We emulated a read sweep commonly done on FeFET devices by applying V_{GS} via a charging capacitor for the V_T read circuit, as shown in Fig. 4. A constant input of 1 V at the input of the synapse charges the $C_{V_{T,ij}}$ capacitor until the potential stored across the capacitor exceeds V_T of the top FeFET. Once the potential across the capacitor exceeds V_T of the top FeFET, the operational amplifier-based comparator applies a logic high bias on the pMOS transistor, so it is no longer conducting. This causes $C_{V_{T,ij}}$ to become a floating capacitor allowing it to retain the potential (approximately equal to the weight or V_T of Top FeFET), as shown in Fig. 4(b). This stored potential across the capacitor is applied as V_{GS} to the bottom FeFET. The average total power consumption of a single V_T read circuit, as measured in LTspice (45-nm transistor model at 1 V), was approximately 0.81 μW per read of a single synapse. The estimated latency of the V_T read circuit would be a maximum of 0.15 ns per synapse using a 1-fF capacitor and the transistor specifications of the FeFET (see Table 1 in the Supplementary Material).

The V_T write circuit was modeled based on the axon-hillock circuit, as shown in Fig. 5(a). This modified axon-hillock circuit used an additional nMOS transistor to allow for $I_{D,ij}$ to be scaled by the output of the neighborhood and learning rate controllers ($V_{P,j} = \Lambda_j \eta$), as shown in Fig. 5(a). Therefore, if either $I_{D,ij} = 0$ (no Euclidean error) or $V_{P,ij} = 0$ (neuron too far from BMU or learning rate too small), no programming pulses were produced by V_T write circuit resulting in no weight change, as shown in Fig. 5(b). The output pulses of the V_T write circuit were applied to the gate of the top FeFET (if bottom FeFET produced nonzero I_{DS}), while the source and drain were grounded to increase the weight and decrease error. We use a programming voltage greater than 1 V (read voltage) to ensure that the FeFET error computation does not cause drift in V_T of the FeFET devices. If the top FeFET produced nonzero I_{DS}, the output pulses were applied to the source or drain of the top FeFET, while the gate was grounded to increase the weight and decrease error. The average total power consumption of a single V_T write circuit measured in LTspice was approximately 69 μW per synapse.

These V_T read and write circuits can be implemented per synapse, per neuron, or globally. Implementing the read and write circuits at each synapse results in the fastest computation, latency of architecture is equivalent to latency of a single synapse, due to parallelization at the cost of the scaled area and power consumption. Considering an ultralow-power budget of approximately 0.1 mW, we are able to implement 100 V_T read circuits (one per neuron) and a single global V_T write circuit. Depending on different powers, performances, and area budgets, a combination of serial and parallel implementations of these V_T read and write circuits can be explored.

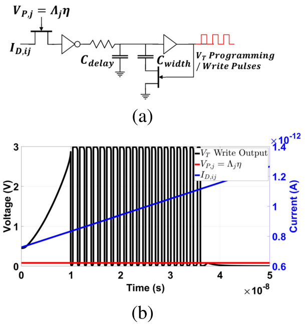


FIGURE 5. (a) Circuit diagram of the proposed V_T write circuit. (b) LTspice simulation of the V_T write circuit demonstrating modulation of pulse frequency proportional to the FeFET I_{DS} (Euclidean error).

IV. MODIFIED WINNER-TAKE-ALL

To produce clusters, the SOFM algorithm must identify a BMU. However, first, the error from each synapse of a neuron must be accumulated to compute the total error. We use a capacitor at each neuron to accumulate error, where the charging rate of the capacitor would be proportional to the summation of $I_{D,ij}$ of each FeFET synapse in the corresponding neuron. The BMU in our architecture will be the neuron with the slowest charging capacitor or least total synaptic I_{DS}.

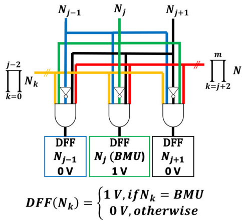
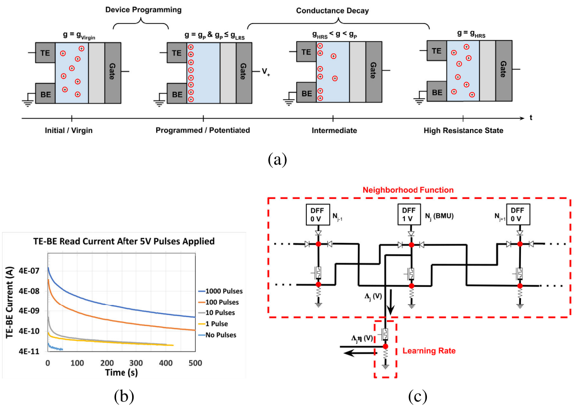


FIGURE 6. Circuit diagram of the MWTA.

We developed an MWTA subcircuit to select the BMU. We decided to use a CMOS-based approach for the MWTA using digital logic and gates to allow the architecture to be more modular, as shown in Fig. 6. This will result in all of the non-BMU neurons to charge to the logic high of the CMOS-based MWTA before the BMU. The delay between the BMU charging to logic high and the non-BMU neurons is used to identify the BMU. The MWTA inverts the neuron corresponding to the MWTA output and and's the inverted neuron with all the other noninverted neurons to identify whether it is the last to charge to logic high (BMU). Once a BMU is identified, the MWTA produces a pulse until the BMU charges to logic high. Therefore, a D flip-flop (DFF) is used to store the MWTA decision (the pulse can behave as



(a)

(b)

(c)

FIGURE 7. (a) Diagram of different biasing states of a Gated-RRAM device. (b) Measured state decay of RRAM devices. (c) Circuit diagram of self-decaying learning parameter implementation.

a rising edge clock signal to store logic high). The MWTA output is logic high for the BMU and logic low for non-BMU neurons, as shown in Fig. 6. The latency of the BMU selection is dependent on current integrating capacitor and on the error difference between the BMU and the second BMU since the second closest BMU is the last to reach high voltage before the BMU can be selected.

V. GATED-RRAM-BASED SELF-DECAYING LEARNING PARAMETERS

In this section, we demonstrate that gated-RRAM devices with self-decaying states offer an excellent opportunity to implement the in-memory computation of the decaying learning parameters (neighborhood function and learning rate) in the SOFM algorithm.

Gated-RRAM is a type of gated-memristive device that has been recently reported [18]. As shown in Fig. 7, gated-RRAM devices consist of a top electrode, a bottom electrode, a channel oxide containing oxygen vacancies (V_o^{2+}) through which current can flow between the top and bottom electrodes, and a gate terminal capacitively coupled to the channel oxide through an insulating layer. When a positive bias is applied on the gate with respect to the top and bottom electrodes, V_o^{2+} in the channel oxide drift toward the top and bottom electrode interface of the channel oxide increasing the conductance of the device measured between the top and bottom electrodes. This results in the low-resistance state (LRS) of the device, which saturates over time with continued application of gate bias as the concentration of V_o^{2+} reaches its maximum value. Once the gate bias is removed, V_o^{2+} tends to diffuse back toward the channel oxide leading to a decay in the conductance over time causing the device to approach its high-resistance state (HRS) as shown in Fig. 7(a) and the measured data in Fig. 7(b). We modeled this decay using exponential decay modulated by a decay time constant hyperparameter. The gate terminal also allows simultaneous read and write feature, which eases the peripheral circuit requirement for programming these devices.

The rate of decay of the gated-RRAM conductance can be tuned by applying a subprogramming voltage bias on the gate of the device. This bias is insufficient to potentiate or depress the device; however, the bias either accelerates the diffusion of the defects moving toward the gate (negative bias) or prevents diffusion of the defects (positive bias), as shown in Fig. 18(a). A continuous subprogramming voltage positive bias may increase the conductance state at which the device settles. The device settling to a higher conductance state is useful in cases when the application requires the neurons to retain more plasticity than the HRS offers. However, this positive bias can be pulsed or removed over a period to allow for the device to return to its original HRS.

The intrinsic passive decay of the gated-RRAM and its tunable nature is important for use for the short-term plasticity of the neurons in our proposed SOFM architecture. In the SOFM algorithm, the plasticity of the network decays over time as both the learning rate and neighborhood rate decay. We implemented the neighborhood function using chained identical voltage dividers, consisting of a gated-RRAM device ($R_{\sigma,\mathrm{RRAM}}$) and fixed value resistor (R_σ), for each neuron shown in Fig. 7(c). Therefore, as the conductance of the gated-RRAM decays passively, the output of each voltage divider decays accordingly. The voltage divider output decay results in the narrowing neighborhood function shown in Fig. 18(b). The chained voltage dividers [Fig. 7(c)] allow for the attenuation of the neighborhood output as a function of the Manhattan distance from the BMU [$d_1(j, j_{\mathrm{BMU}})$] as modeled in the following equation:

$$\Lambda_j = \frac{R_{\mathrm{sigma}}}{R_{\mathrm{sigma}} + R_{\mathrm{sigma,RRAM}}}^{d_1(j,j_{\mathrm{BMU}})} V_{\mathrm{DD}} - V_\gamma \left(\sum_{n=0}^{d_1(j,j_{\mathrm{BMU}})} \frac{R_{\mathrm{sigma}}}{R_{\mathrm{sigma}} + R_{\mathrm{sigma,RRAM}}}^{n} \right) \tag{3}$$

where V_{DD} is the output of the DFF of the BMU from the MWTA and V_γ is the forward voltage of the diodes. The neighborhood function output feeds into the input of an additional gated-RRAM voltage divider, which implements the learning rate resulting in (4). The learning rate decays as well due to the passive decay of the gated-RRAM device. This passive in-memory computation results in a nearly constant 0.1 mW per input of power consumption throughout the training of the architecture

$$\Lambda_j \eta = \Lambda_j \frac{R_\eta}{R_\eta + R_{\eta,\mathrm{RRAM}}}. \tag{4}$$

External circuitry can be implemented for tuning the decay rate if required by applying a continuous or pulsed bias to the gate of the gated-RRAM device. The gated-RRAM device may be potentiated at any time, increasing the plasticity of the network. Increasing the plasticity of the network can help it learn completely new environments, help it evolve with the dynamics of the dataset, or compensate for any damage accrued. In addition, the decay rate hyperparameter in the SOFM can be implemented in device characteristics of gated-RRAM.

VI. NeuroSOFM ARCHITECTURE

The NeuroSOFM architecture (Fig. 8) consists of a crossbar of the dual-FeFET synapses computing the Euclidean error [Fig. 3(b)], feeding into the CMOS-based MWTA for BMU

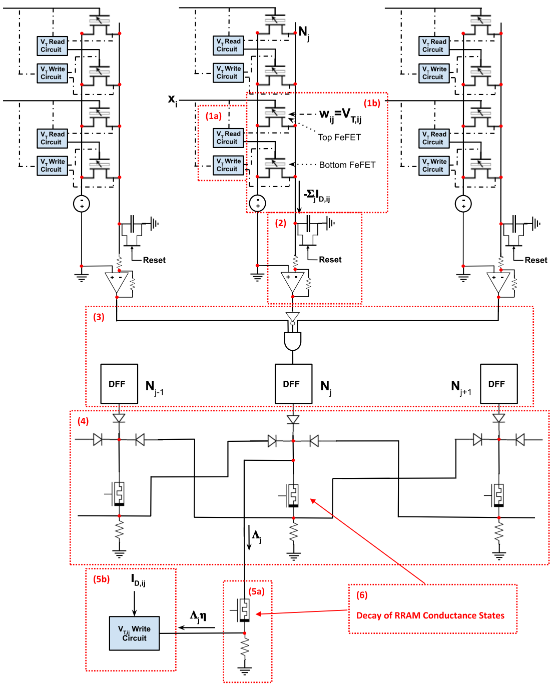


FIGURE 8. Circuit diagram of NeuroSOFM architecture. (1a) V_T read and write circuits. (1b) Dual-FeFET synapse. (2) Integration of error. (3) MWTA. (4) Neighborhood controller. (5a) Learning rate controller. (5b) Feedback to V_T write circuit of each synapse.

selection (Fig. 6), and finally the gated-RRAM network to compute the self-decaying learning parameters [Fig. 18(b)].

First, V_T of the top FeFET in every FeFET synapse is initialized with a random value. Prior to applying an input, V_T of the top FeFET of each synapse is read and stored by the V_T reading circuit to apply as V_{GS} for the bottom FeFET. In addition, at this time, V_T of the bottom FeFET is programmed to its lowest V_T state to allow for V_T to be programmed to $V_{\mathrm{in},i}$ to be shown to the network. After the V_T states of the FeFET synapse have been prepared, the input can be applied to the network. At this time, V_{in} is applied as V_{GS} to the bottom FeFET, and the bottom FeFET is programmed via the V_T write circuit until it produces a nonzero I_{DS} (its $V_T = V_{\mathrm{in},i}$). V_{in} is applied as V_{GS} to the top FeFET and the stored voltage read from V_T of the top FeFET is applied as V_{GS} to the bottom FeFET. This FeFET configuration results in a single FeFET per synapse producing a nonzero I_{DS}, which captures the Euclidean error between the input and the weight (2). Since only a single FeFET produces a significant current per synapse, only a single FeFET device per synapse contributes to the power consumption of the circuit. Each FeFET synapse is able to compute the Euclidean error simultaneously. In addition, each synapse can have its own circuitry for reading and writing V_T. Therefore, the FeFET synapses are completely parallelized and would result in constant time complexity as the size of the network is scaled.

The total current in the crossbar is proportional to the total pairwise Euclidean error of the neurons as seen in subcircuit 2 in Fig. 8. An integrate-and-fire circuit produces a logic high once the capacitor is charged to a sufficient voltage. The rate of charging of the capacitor in the integrate-and-fire circuit is dependent on the total current along the neuron branch. The MWTA circuit then selects the BMU by finding the neuron to last reach logic high. The output of the MWTA is stored in a DFF shown in subcircuit 3 in Fig. 8 (detailed in Fig. 6).

The DFF of the BMU has a logic high, 1 V, while the DFFs of the remaining neurons hold 0 V. This logic high from the winning neuron propagates through the neighborhood function circuit, as shown in subcircuit 4 in Fig. 8 [detailed in Fig. 7(c)]. Each neuron has a corresponding voltage divider in the neighborhood function circuit. The diodes at the input of the voltage divider ensure that the highest voltage inputted into the voltage divider is passed through the divider. The diodes ensure that only the feedback from the shortest distance is considered in the neighborhood output. The output of the neighborhood controller from each neuron feeds into an additional independent voltage divider with gated-RRAM, which emulates the learning rate that decays over time.

The output of the final voltage divider is a combination of the neighborhood function and learning rate (5). A nonzero I_{DS} in the top FeFET results in positive programming pulses to the top FeFET to depress the weight or reduce V_T. A nonzero I_{DS} in the bottom FeFET results in negative programming pulses to the top FeFET to potentiate the weight or increase V_T. The number of pulses produced by the programming circuit is proportional to the product of the final learning parameter output of the corresponding neuron and the I_{DS} Euclidean error of the corresponding synapse. This weight update is modeled as follows:

$$\Delta V_{w,ij} \propto \Lambda_j \eta \, (-1)^{(V_{\mathrm{in},i} < V_{w,ij})} \, I_{D,ij} \tag{5}$$

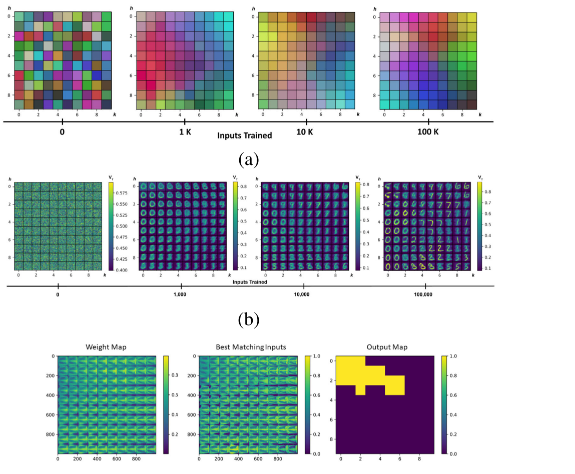
where $\Delta V_{w,ij}$ is the change in V_T of the top FeFET.

The nature of the gated-RRAM allows for the decay rate of the neighborhood rate and learning rate to be tuned for each application [Fig. 18(b)]. The combination of FeFET devices for Euclidean error computation and gated-RRAM for learning parameter implementation results in an architecture requiring little or no external circuitry.

VII. RESULTS AND DISCUSSION

After simulating each circuit module in LTspice, the proposed NeuroSOFM architecture was modeled and simulated at a higher level using Python 3.8 with both the measured and simulated V_T states for the FeFET devices. The functions that were used to model every step of the algorithm are detailed in Sections III–VI. A V_T lookup table was extracted from a FeFET model and was used for the V_T states of the devices. The gated-RRAM decay model was validated on a gated synaptic device (GSD) model developed to emulate the behavior of various fabricated GSD devices.

The proposed architecture was trained and tested on benchmark datasets: RGB color (10 000 randomly generated RGB colors) and MNIST handwritten digits (5000 images sized 28 × 28) datasets. The proposed architecture was also trained and tested on a dataset of chest X-rays (148 images compressed to 100 × 100) of healthy subjects and the subjects after being diagnosed with COVID-19, to show more practical application-oriented results. In all experiments, we tested a 10 × 10 neuron SOFM, the number of synapses or weights differed between datasets. The weights were randomly initialized ([0,1]) for experiment. Topographical error, the average



(a)

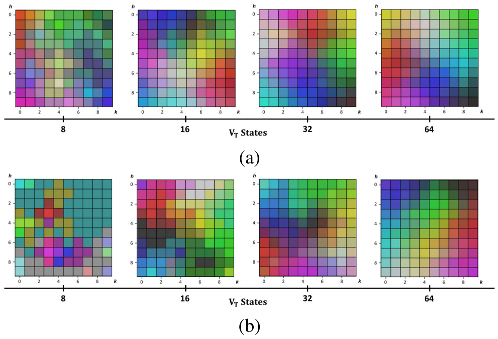
(b)

(c)

FIGURE 9. Weight map evolution of 10 × 10 NeuroSOFM trained on (a) RGB colors. (b) MNIST handwritten digits. (c) COVID-19 chest X-ray images. The color bar illustrates the V_T range of the FeFET devices.

distance between the first BMU and the second BMU (neuron with second least Euclidean error), was to measure the ability of the self-organized map to preserve the topology of the data [20]. Quantization error, the average Euclidean error between an input and the weight vector of its BMU, was used to measure the ability of the neurons in the self-organized map to represent the individual inputs [7]. We show the evolution of the weight maps over the training period for all three datasets in Fig. 9(a)–(c). We tested the networks robustness to device variability in the FeFET devices Gaussian distribution to offset the V_T states. We also tested the robustness of the architecture by simulating neurons failing and by removing neurons from the network prior to training.

The network was able to completely learn the RGB dataset and converge after 50 000 inputs with a topographical error of 1.50 and a quantization error of 0.22 [Fig. 9(a)]. Visually, we can observe the development of various colors including both brighter and darker colors, which have not been shown in the existing networks using the dot product. The network was able to completely learn the MNIST dataset and converge after 50 000 inputs with a topographic error of 1.67 and a quantization error of 136.00 [Fig. 9(b)]. It is expected that topographical and quantization error for more complex datasets (higher dimensions) is larger. Visually, we can observe the development of the various digits in the weight map with a level of expected noise, as shown in Fig. 9(b). The network was able to completely learn the COVID-19 chest X-ray dataset and converge after 592 inputs with a topographical error of 1.52 and a quantization error of 1029.97. The output map illustrates that the SOFM successfully separated clusters of healthy and COVID-19 diagnosed chest X-rays while retaining significant detail (observable rib and lung features) in Fig. 9(c). Discrete states in the FeFET devices and the projection of a large dataset on few neurons will result in a nonzero quantization error. However, a low topographic and quantization error for the COVID-19 results and intrinsic explainability of SOFMs [21] shows the practicality of the network in real-world applications. In addition, the quality preservation in all tests shows that the decaying plasticity of the network can prevent overtraining, while the network is still functional.



(a)

(b)

FIGURE 10. Weight map of NeuroSOFM with FeFET devices with 8, 16, 32, and 64 nonvolatile V_T states (a) from the measured set V_T states and (b) from the simulated set V_T states.

We tested to see how the number of nonvolatile V_T states in the FeFET devices affect learning in the SOFM architecture. We observed that the architecture was able to cluster to a degree with as little as eight states for the measured V_T states, as shown in Fig. 10(a). However, having 32 nonvolatile V_T states [17] yielded higher quality clustering. 64 nonvolatile states had little or no effect. The optimal number of nonvolatile states required may vary between applications. Comparing the measured V_T results with the simulated V_T results in Fig. 10(b), we observe that at a higher number of V_T states (≥16 states), both the V_T state distributions were able to cluster. These results demonstrate intrinsic robustness to different distributions of V_T states of the FeFET device, which can result from varying fabrication methods or programming schemes [17]. However, we do note that a more uniform distribution of states, such as those in the measured states, resulted in higher quality maps even as the number of states was reduced to 8. The distribution of the simulated V_T states did not allow for proper clustering at eight nonvolatile states. For the neuromorphic architectures based on emerging devices, device-level failure and variability are important concerns. Variability in the FeFET devices would be especially detrimental since the algorithm relies on the initial Euclidean error computation. We observed that the SOFM was highly robust even to an offset distribution with a standard deviation exceeding the average difference between V_T states shown in Fig. 11(a). Visually, the clustering was not as continuous and tight as the offset variance was increased. Our architecture was also highly robust to large amounts of the neurons, up to 50%, failing or being removed from the network, as shown in Fig. 11(b). We also observed that if the network failed in the middle of the training process, the remaining neurons would adapt and compensate for the failed neurons by reorganizing [Fig. 11(c)]. The results in Fig. 11(c) illustrate that the network is able to learn unseen and unlearned inputs given sufficient plasticity (yellow portion in the halved network). Integrating gated-RRAM allows for easy tuning of the plasticity. The relearning of lost information due to broken neurons does not require additional external circuitry and is able to be completely internalized.

We recorded the average error for each FeFET synapse, the average number of programming pulses to the synapses,

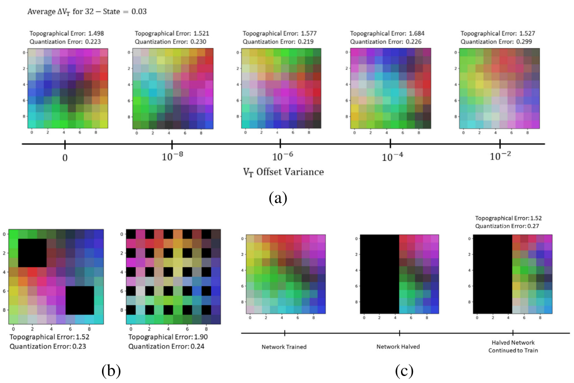


FIGURE 11. (a) Weight map of NeuroSOFM, using 32-state FeFET devices, with V_T offset variance from 0 to 0.01, (b) with neurons broken or removed at initialization, and (c) with neurons broken or removed in the middle of training.

and the minimum error difference between the BMU and second BMU to estimate the power consumption and latency of the architecture. The average input-weight error for FeFET synapses was approximately 0.0005 resulting in an average power consumption of 70 nW per synapse for Euclidean error computation. Although the power consumption increases in proportion to the number of synapses per neuron, for higher dimensional data such as the COVID-19 dataset, the time period for Euclidean error computation (latency of BMU selection) also decreases. The minimum error difference, between the BMU and second BMU, for the RGB dataset (three attributes) resulted in a maximum latency of 0.28 μs, while the minimum error difference for the COVID-19 dataset (10 000 attributes) resulted in a maximum latency of 0.1 ps. The power consumption of the entire passive RRAM-based learning parameter controller measured to be approximately 0.1 mW. This demonstrates the ultralow power consumption due to in-memory computation of the FeFET synapse and RRAM-based learning parameter controller.

In conclusion, we have proposed a neuromorphic SOFM architecture, based on emerging FeFET and gated-RRAM memory devices, that is able to learn simple benchmark datasets such as RGB colors to more complex application-specific datasets such as the COVID-19 chest X-rays. By utilizing the underlying device physics of the device technologies, the architecture utilizes very little power and requires little or no external circuitry making it completely autonomous. The low power and autonomity of this architecture allow it to be easily interfaced with sensors and IoT devices for real-time clustering of the data and environment. The interaction between neighboring neurons allows for the network to handle severe damage to itself and still operate at a functional level. This type of durability is fantastic for systems that are exposed to high-risk environments such as space (due to radiation exposure) [22] by not requiring monitoring and repairing systems. It also allows the architecture to utilize synaptic devices with lower durability if damaged neurons can be detected and shut off improving manufacturing scalability. Our architecture has the ability to learn lifelong due to its self-decaying learning parameter controller. Lifelong learning allows for the architecture to adapt to the dynamics of the environment without external interference (e.g., retraining), as shown in the results in Fig. 11(a)–(c). The ability to adapt to the environment is especially useful for transfer learning and in applications such as navigation. The architecture is completely unsupervised, meaning that it requires no labeled data. In the future, we would like to further examine the ability of the SOFM algorithm and architecture to interact with other neural networks such as recurrent networks like the attractor network for fully unsupervised association between signals or inputs [12]. This would allow for networks to utilize the topography and features captured by the SOFM for classification or signal correlation/association.

ACKNOWLEDGMENT

The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of Southwestern Council for Higher Education and the Air Force Research Laboratory (AFRL) or the U.S. Government. The authors would like to thank Dr. Len Orlando, Dr. Shanee Pacley, and Dr. Amber Reed at the Air Force Research Laboratory (AFRL) as well as many members of the MIND lab for useful discussions and feedback.

REFERENCES

[1] S. Ambrogio *et al.*, "Equivalent-accuracy accelerated neural-network training using analogue memory," *Nature*, vol. 558, no. 7708, pp. 60–67, 2018, doi: 10.1038/s41586-018-0180-5.

[2] T. Gokmen and Y. Vlasov, "Acceleration of deep neural network training with resistive cross-point devices: Design considerations," *Frontiers Neurosci.*, vol. 10, p. 333, Jul. 2016, doi: 10.3389/fnins.2016.00333.

[3] M. Davies *et al.*, "Loihi: A neuromorphic manycore processor with on-chip learning," *IEEE Micro*, vol. 38, no. 1, pp. 82–99, Jan. 2018, doi: 10.1109/MM.2018.112130359.

[4] H. W. Lee, N. Kim, and J. H. Lee, "Deep neural network self-training based on unsupervised learning and dropout," *Int. J. Fuzzy Logic Intell. Syst.*, vol. 17, no. 1, pp. 1–9, 2017, doi: 10.5391/IJFIS.2017.17.1.1.

[5] G. Singh *et al.*, "A review of near-memory computing architectures: Opportunities and challenges," in *Proc. 21st Euromicro Conf. Digit. Syst. Design (DSD)*, Prague, Czech Republic, Aug. 2018, pp. 608–617, doi: 10.1109/DSD.2018.00106.

[6] T. Kohonen, "Self-organized formation of topologically correct feature maps," *Biol. Cybern.*, vol. 43, no. 1, pp. 59–69, 1982, doi: 10.1007/BF00337288.

[7] T. Kohonen, M. R. Schroeder, and T. S. Huang, *Self-Organizing Maps*, 3rd ed. Berlin, Germany: Springer-Verlag, 2001.

[8] M. Abadi, S. Jovanovic, K. B. Khalifa, S. Weber, and M. H. Bedoui, "A scalable and adaptable hardware NoC-based self organizing map," *Microprocessors Microsyst.*, vol. 57, pp. 1–14, Mar. 2018, doi: 10.1016/j.micpro.2017.12.007.

[9] H. Hikawa, "FPGA implementation of self organizing map with digital phase locked loops," *Neural Netw.*, vol. 18, nos. 5–6, pp. 514–522, 2005, doi: 10.1016/j.neunet.2005.06.012.

[10] S. T. Brassai, "FPGA based hardware implementation of a self-organizing map," in *Proc. IEEE 18th Int. Conf. Intell. Eng. Syst. INES*, Tihany, Hungary, Jul. 2014, pp. 101–104, doi: 10.1109/INES.2014.6909349.

[11] J. R. Mann and S. Gilbert, "An analog self-organizing neural network chip," in *Proc. NIPS*, 1988, pp. 739–747.

[12] M. Pedró, J. Martín-Martínez, M. Maestro-Izquierdo, R. Rodríguez, and M. Nafría, "Self-organizing neural networks based on OxRAM devices under a fully unsupervised training scheme," *Materials*, vol. 12, no. 21, p. 3482, Oct. 2019, doi: 10.3390/ma12213482.

[13] K. L. Elmore and M. B. Richman, "Euclidean distance as a similarity metric for principal component analysis," *Monthly Weather Rev.*, vol. 129, no. 3, pp. 540–549, 2001, doi: 10.1175/1520-0493(2001)129<0540:EDAASM>2.0.CO;2.

[14] J. P. Cohen, P. Morrison, and L. Dao, "COVID-19 imagedata collection," Mar. 2020, *arXiv:2003.11597*. [Online]. Available: https://arxiv.org/abs/2003.11597 and https://github.com/ieee8023/covid-chestxray-dataset

[15] T. J. Bailey, A. J. Ford, S. Barve, J. Wells, and R. Jha, "Development of a short-term to long-term supervised spiking neural network processor," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 28, no. 11, pp. 2410–2423, Nov. 2020, doi: 10.1109/TVLSI.2020.3013810.

[16] S. Dutta, C. Schafer, J. Gomez, K. Ni, S. Joshi, and S. Datta, "Supervised learning in all FeFET-based spiking neural network: Opportunities and challenges," *Frontiers Neurosci.*, vol. 14, p. 634, Jun. 2020, doi: 10.3389/fnins.2020.00634.

[17] M. Jerry *et al.*, "Ferroelectric FET analog synapse for acceleration of deep neural network training," in *IEDM Tech. Dig.*, Dec. 2017, pp. 6.2.1–6.2.4.

[18] E. Herrmann, A. Rush, T. Bailey, and R. Jha, "Gate controlled three-terminal metal oxide memristor," *IEEE Electron Device Lett.*, vol. 39, no. 4, pp. 500–503, Apr. 2018, doi: 10.1109/LED.2018.2806188.

[19] A. Jones and R. Jha, "A compact gated-synapse model for neuromorphic circuits," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 40, no. 9, pp. 1887–1895, Sep. 2021, doi: 10.1109/TCAD.2020.3028534.

[20] K. Kiviluoto, "Topology preservation in self-organizing maps," in *Proc. Int. Conf. Neural Netw.*, Washington, DC, USA, 1996, pp. 294–299, doi: 10.1109/ICNN.1996.548907.

[21] B. King, S. Barve, A. Ford, and R. Jha, "Unsupervised clustering of COVID-19 chest X-ray images with a self-organizing feature map," in *Proc. IEEE 63rd Int. Midwest Symp. Circuits Syst. (MWSCAS)*, Springfield, MA, USA, Aug. 2020, pp. 395–398, doi: 10.1109/MWSCAS48704.2020.9184493.

[22] V. V Belyakov, A. I Chumakov, A. Y Nikiforov, V. S Pershenkov, P. K Skorobogatov, and A. V. Sogoyan, "IC's radiation effects modeling and estimation," *Microelectron. Rel.*, vol. 40, no. 12, pp. 1997–2018, 2000, doi: 10.1016/S0026-2714(00)00021-4.

[23] T. K. Song, "Landau-Khalatnikov simulations for ferroelectric switching in ferroelectric random access memory application," *J. Korean Phys. Soc.*, vol. 46, no. 1, pp. 5–9, 2005.

[24] B. V. Zeghbroeck, *Principles of Electronic Devices*. Boulder, CO, USA: Univ. Colorodo, 2011.

[25] Y. Gagou *et al.*, "Intrinsic dead layer effects in relaxed epitaxial BaTiO$_3$ thin filmgrown by pulsed laser deposition," *Mater. Des.*, vol. 122, pp. 157–163, May 2017.

[26] R. Xu *et al.*, "Kinetic control of tunable multi-state switching in ferroelectric thin films," *Nature Commun.*, vol. 10, no. 1, pp. 1–10, 2019.

[27] D. Zhao, I. Katsouras, K. Asadi, W. A. Groen, P. W. M. Blom, and D. M. de Leeuw, "Retention of intermediate polarization states in ferroelectric materials enabling memories for multi-bit data storage," *Appl. Phys. Lett.*, vol. 108, no. 23, 2016, Art. no. 232907.

[28] J. Mayersky, A. Hilton, S. Pacley, and R. Jha, "Investigation and characterization of the annealing effects on the ferroelectric behavior of PLD BaTiO$_3$," *MRS Commun.*, vol. 11, pp. 288–294, Mar. 2021.

SIDDHARTH BARVE (Student Member, IEEE) received the B.S. degree in electrical engineering from the University of Cincinnati, Cincinnati, OH, USA, in 2021, where he is currently pursuing the Ph.D. degree in electrical engineering.

He works as an Graduate Research Assistant at the University of Cincinnati, where he focuses his research on neuromorphic computing and emerging memory devices.

JOSHUA MAYERSKY (Student Member, IEEE) received the B.S. and M.S. degrees in electrical engineering from the University of Cincinnati, Cincinnati, OH, USA, in 2016 and 2018, respectively, where he is currently pursuing the Ph.D. degree in electrical engineering.

He works as a Research Assistant of microelectronics and integrated systems with the NeuroCentric Devices (MIND) Laboratory, where he focuses his research on ferroelectric materials for neuromorphic computing and process trust and assurance.

ANDREW J. FORD (Student Member, IEEE) received the B.S. and M.S. degrees in electrical engineering from the University of Cincinnati, Cincinnati, OH, USA, in 2019 and 2020, respectively.

He worked as a Graduate Research Assistant at the University of Cincinnati, where he researches neuromorphic computing, emerging memory devices, and applications of electroencephalography data.

ALEXANDER JONES (Student Member, IEEE) received the B.S. degree in computer engineering and the Master of Science degree in computer engineering from the University of Cincinnati, Cincinnati, OH, USA, in 2015 and 2016, respectively, and the Ph.D. degree in computer engineering from the University of Cincinnati, in 2020. The focus of his research is the design and simulation of neuromorphic architectures that use novel memory technologies.

BAYLEY KING (Student Member, IEEE) received the B.S. degree in electrical engineering from the University of Cincinnati, Cincinnati, OH, USA, in 2019, where he is currently pursuing the Ph.D. degree in computer engineering.

He works as a Teaching Assistant and a Graduate Assistant with the Department of Electrical Engineering and Computer Science, where he focuses his research on complex systems and hardware security.

AARON RUEN (Student Member, IEEE) received the B.S. degree in electrical engineering from the University of Cincinnati, Cincinnati, OH, USA, in 2021, where he is currently pursuing the Ph.D. degree in computer engineering.

He works as a Graduate Assistant with the Department of Electrical Engineering and Computer Science, where he focuses his research on resistive random access memory and hardware security.

RASHMI JHA (Member, IEEE) received the B.Tech. degree in electrical engineering from IIT Kharagpur, Kharagpur, India, in 2000, and the M.S. and Ph.D. degrees in electrical engineering from North Carolina State University, Raleigh, NC, USA, in 2003 and 2006, respectively.

She is currently a Professor with the Department of Electrical Engineering and Computer Science, University of Cincinnati, Cincinnati, OH, USA.