2024 IEEE Computer Society Annual Symposium on VLSI (ISVLSI) | 979-8-3503-5411-9/24/$31.00 ©2024 IEEE | DOI: 10.1109/ISVLSI61997.2024.00067

Energy-Efficient Near-Sensor Event Detector based on Multilevel Ga_2O_3 RRAM

Mehrdad Morsali[†], Sepehr Tabrizchi[‡], Ravi Teja Velpula[*], Mano Bala Sankar Muthu[*],
Hieu Pham Trung Nguyen[*], Mohsen Imani[§], Arman Roohi[‡], and Shaahin Angizi[†]
[†]Department of Electrical and Computer Engineering, New Jersey Institute of Technology, Newark, NJ, USA
[‡]School of Computing, University of Nebraska–Lincoln, Lincoln, NE, USA
[*]Department of Electrical and Computer Engineering, Texas Tech University, Lubbock, TX, US
[§]Department of Computer Science, University of California Irvine, Irvine, CA, USA
hieu.p.nguyen@ttu.edu, m.imani@uci.edu,aroohi@unl.edu, shaahin.angizi@njit.edu

***Abstract*—In this paper, a cost-effective Near-Sensor Processing (NSP) platform is developed based on an experimentally-measured Ti/TiN/Ga_2O_3/Ti/Pt Resistive Random Access Memory (RRAM) device that facilitates event detection for edge vision sensors without the requirement for power-intensive Analog-to-Digital Converters (ADCs). The platform is supported with a hardware-friendly background comparison technique providing adjustable precision that allows for a dynamic balance between accuracy and efficiency at runtime. Our device-to-architecture simulation results demonstrate that the proposed platform achieves on average 66% and 63% energy saving over STT-MRAM and SOT-MRAM counterparts due to utilizing the ADC-less method.**

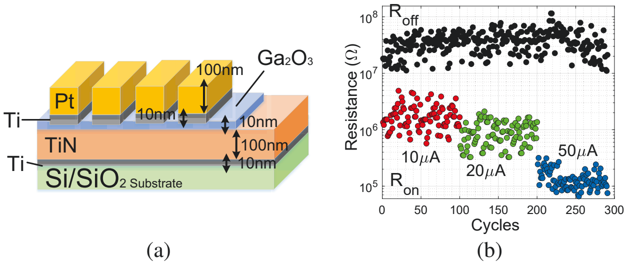

Fig. 1. (a) Schematic diagram and (b) Characteristics of endurance of Ti/TiN/Ga_2O_3/Ti/Pt RRAM devices over 300 operating cycles.

I. Introduction

Over the past decade, there has been extensive development in CMOS imagers featuring on-chip feature extraction and compression with the primary objective of optimizing computing resources and reducing overall power consumption [1]–[5]. At the same time, Non-Volatile Memories (NVMs) have gained significant interest as possible substitutes for conventional volatile memory technologies. This increased interest is due to the unique attributes of NVM, such as non-volatility, robustness, long endurance, extremely low standby power, suitability for intermittent computing, and high integration density [6], [7]. In scenarios such as embedded applications and low-power IoT systems where on-chip cache plays a crucial role, a resilient NVM has the potential to augment memory capacity and enhance overall performance. Magnetic Tunnel Junction (MTJ) devices are among the most common NVMs. Experiments and fabrications of Spin-Transfer Torque (STT) MTJs demonstrate fast magnetic switching in the subnanosecond range. By utilizing Spin-Orbit Torque (SOT), faster switching is also achieved [8]–[12]. Despite their advantages such as long retention times (up to 10 years) and minimal energy consumption for writing data (in the fJ/bit range), these technologies face challenges due to their low ON/OFF ratios (typically below 10), leading to reliability issues associated with the current-driven switching approach. Moreover, MTJs are able to store only one bit per device, which limits the memory capacity, and the three-terminal structure of SOT-MTJ increases the complexity of the memory array and reduces scalability.

Lately, Resistive Switching (RS) devices, specifically Resistive Random-Access Memory (RRAM), are being recognized as one of the most promising emerging memory technologies with implementations on basic binary transition metal oxide materials, such as HfO_2 [13], Al_2O_3 [14], and TiO_2 [15], where carrier transport is predominantly influenced by oxygen vacancies. RRAM offers a much higher ON/OFF ratio than MTJs. Moreover, by presenting multi-level resistance and a two-terminal device structure, they offer higher memory capacity and scalability. In this paper, we present a Near-Sensor Processing (NSP) architecture for event detection applications at the edge that saves the background image in an accurate RRAM array modeled based on our fabricated Ti/TiN/ Ga_2O_3/Ti/Pt [16]. The main contributions of this paper are as follows: (1) We design an NSP architecture with inventive micro-architectural and circuit-level strategies in pixel and RRAM peripherals tailored for energy efficiency and speed up; (2) We propose a versatile hardware-aware method for event detection, tailored to identify events through background variations; and (3) We present an inclusive bottom-up evaluation framework designed to gauge the overall performance of the system.

II. Multilevel RRAM Device

The RRAM is a two-terminal NVM that stores data in different resistive states by creating/rupturing a conductive filament within the metal oxide insulator. At the device level, we designed a Ti/TiN (bottom electrode, BE)/Ga_2O_3/Ti/Pt (top electrode, TE) thin-film RRAM device [16]. The schematic diagram of the Ga_2O_3 based RRAM device is depicted in Fig. 1(a) that is used in a 1T1R cell as the central storage component in the envisioned near-sensor accelerator. During the set phase, by applying a positive voltage to the top electrode, the conductive filament establishes a connection between the top and bottom electrodes, resulting in a Low Resistance State (LRS). Conversely, in the reset phase, a negative voltage is applied to the top electrode and the filament undergoes a breakage, causing an increase in the device's resistance and transitioning it into a High Resistance State (HRS). Figure

2159-3477/24/$31.00 ©2024 IEEE
DOI 10.1109/ISVLSI61997.2024.00067

1(b) reports the endurance test results for 300 cycles at a read voltage of 0.2 V. As the I_{cc} (set current) increases from 10 to 50 μA, a noticeable trend emerges in LRS, indicating the presence of three distinct LRS states. In contrast, the HRS remains nearly constant across all compliance currents. The memory states for presented Ga_2O_3 based RRAM are experimentally measured and represented by the 4 chosen resistance states, i.e., ~140kΩ (I_{cc}=50μA), ~900kΩ (I_{cc}=20μA), ~1.8MΩ (I_{cc}=10μA), and ~38MΩ (R_{off}). The presented multi-level RRAM device offers excellent resistance switching properties and a high R_{off}/R_{on} ratio (up to 10^3).

III. Proposed Near-Sensor Event Detector

Utilizing the multi-level RRAM device, an always-on event detector architecture has been proposed on top of the processing near-sensor scheme. The suggested architecture primarily comprises three main components. A 256×256 pixel array for capturing the frames, a 64×64 RRAM array that saves the background data, and a near sensor comparator component. The peripherals include pixel array and RRAM array write and readout circuits, a demux, and a control logic to control the operation of the arrays. In the presented NSP architecture, the main idea involves storing particular pixels from an input frame as background data on the RRAM array. The pixel selection algorithm will be explained in the following section. After saving background data, the NSP platform operates in event detection mode, utilizing an ADC-less approach to compare input frames with the pre-stored background data. Upon detecting a new object through a mismatch between the input frame and background data, the platform switches to sensing mode, capturing frames in high resolution and transmitting them to any deep-learning accelerator for further processing.

Pixel Array. The proposed NSP platform utilizes a 256×256 pixel array to capture input frames, as depicted in Fig. 2(a). Three-transistor/one-photodiode (PD) pixels form the pixel array. As illustrated in Fig. 2(c), the output of the pixel array is connected to a 1-to-2 demux. During the regular sensing mode, the demux sends the pixel's output signal to the ADC-based high-resolution readout circuitry. Conversely, during the event detection mode, through the demux, the pixel's output is forwarded to the near-sensor comparator part for detecting events in an ADC-less method. Moreover, to store the data of selected pixels of our array as background data, ADCs are utilized but with lower bit-width of 2-bit or 3-bit resolution to reduce the ADC's overall power consumption.

Background RRAM Array. In our NSP platform, a 64×64 RRAM array is considered to store background data. The RRAM array is shown in Fig. 2(b), where 1T1R cells are used to form the array. To store each pixel's data in the array, two RRAM cells are dedicated. As mentioned in the previous section, each RRAM can have four different resistance levels. Three of these levels are considered to be utilized in our design. We have utilized 900kΩ, 1800kΩ, and 38MΩ resistance levels for storage purposes. The reason for using only these three levels is that during reading the RRAMs generated current by an RRAM with 900kΩ resistance is exactly double of generated current of an RRAM with 1800kΩ resistance and in the case of 38MΩ resistance, the RRAM almost passes zero current through it. Later, these current levels by using the presented comparison mechanism, will present a digital representation of the stored background data. The size of the RRAM array is also defined considering the pixel selection algorithm where one pixel out of 9, 25, or 49 pixels will be saved as background. The algorithm has been explained in a more detailed way in the next section. Thus, a 64×64 RRAM array is enough for background storage.

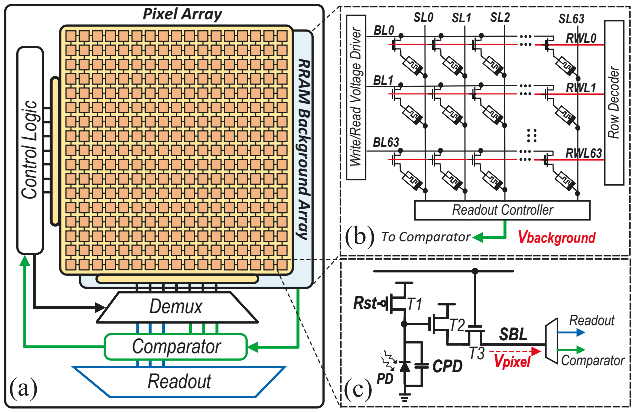

Fig. 2. (a) The proposed NSP platform, (b) MEFET background array, and (c) 3-T pixel structure.

During backup, each pixel's data is coded to two RRAM resistance levels with 2-bit or 3-bit resolution. Theoretically, 9 different combinations of two RRAM's resistance levels can be generated. By using two different reading voltages one of them is doubled the other, some levels overlap and the final generated voltage will be the same. In practice, in our method, seven distinct voltage levels can be generated by two background RRAMs. Thus, pixel data can be mapped to these levels. In the case of 3-bit data precision, we need eight levels. Fig. 3 shows the output voltage of a pixel under different light intensities from the brightest to the darkest. As can be observed in Fig. 3, after a point the output voltage of the pixel doesn't change much when the environment gets darker(right end of the curve), thus we can consider two "110" and "111" data same when we are mapping them to the RRAMs. The voltage levels that pre-stored 3-bit and 2-bit background data generates are also shown in Fig. 3. As an example, Table I explains how we code 3-bit pixel data to the resistance levels of two RRAMs in the background array, and also, generated voltage by the resultant stored data of the background array is illustrated. The background voltages later are used for comparison purposes during event detection mode.

TABLE I
Current Levels for the 3-bit Precision Background Array

Background data	R1(kΩ)	R2(kΩ)	$V_{background}(mV)$
"000"	38000	38000	0
"001"	38000	1800	10
"010"	1800	38000	19
"011"	1800	1800	27
"100"	900	38000	35
"101"	900	1800	45
"110" ("111")	900	900	53

Near-Sensor Comparator. In the event detection mode, the presented NSP platform is responsible for continuously comparing the pre-stored background data with the sensed pixel data to find any mismatches. We present a new comparison method to directly compare the analog signal coming from pixels with digital background data without using power-hungry ADCs. As mentioned before, primarily, the pixel's data can be stored with 2-bit or 3-bit precision in two RRAM devices in the background array. By considering two different RRAM read voltages, according to Table I, seven different voltages can be generated by the stored background data. These voltages are used to be compared with the pixel's output voltage. As shown in Fig. 3, the pixel's output voltage spans from 0 to around 60 mV. This span is divided into seven subranges equal to the number of voltage levels that background data can generate as highlighted with shaded and plain areas in Fig. 3. The proposed near-sensor comparator has been depicted in Fig. 4. For comparison, the voltage generated in the background array ($V_{\text{background}}$) as shown in Fig. 4(a), and the pixel's output voltage (V_{pixel}), are fed to a new circuit designed based on a voltage divider and voltage comparators. As shown in Fig. 4(b), $V_{\text{background}}$ and V_{pixel} are fed to the gates of T1 and T2 transistors, which form a voltage divider. As $V_{background}$ and V_{pixel} are quite small, T1 and T2 work in the subthreshold region. If $V_{background}$ and V_{pixel} are equal, in T1 and T2 operating points, the voltage divider's output voltage (V_{out}), will be almost 100 mV. and if $V_{background}$ and V_{pixel} are not equal, it results in the V_{out} being deviated from 100 mV. By detecting any deviation from the working point (100 mV) by comparators, a mismatch can be detected. As V_{pixel} is continuous analog voltage, $V_{background}$ and V_{pixel} may never be equal exactly even when the pixel data matches with background data. Thus, the V_{out} is fed as input to two voltage comparators whose reference voltages (V_{rf1} and V_{rf2}) define a small range around 100 mV which resembles the ranges we defined in Fig. 3 around any of to $V_{background}$ levels. As shown in Fig. 4(c), If V_{out} is bigger than V_{rf1}, the output of the first comparators (V_{c1}) will be '0' and If V_{out} is smaller than V_{rf2}, the output of the second comparators (V_{c2}) will be '0'. Thus, only when V_{out} is in the range of V_{rf1} to V_{rf1}, which means that the $V_{background}$ and V_{pixel} are almost equal, V_{c1} and V_{c2} will be '0' at the same time, and the precharged match line will remain precharged. Other than that at least one of the V_{c1} or V_{c2} voltages will be '1' leading the match line to discharge. We keep tracking the match line and whenever it is discharged, a mismatch between background data and collected pixel data is detected.

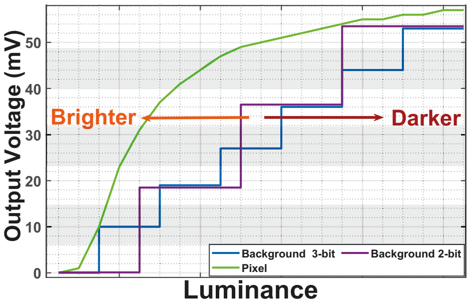

Fig. 3. The pixel's output voltage and generated voltage levels by the different background data.

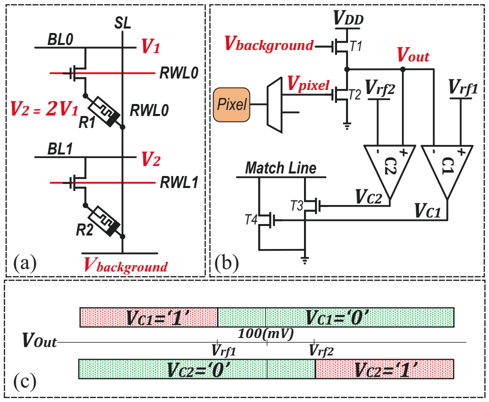

Fig. 4. Near-Sensor comparator unit. (a) Reading background data using two different read voltages, (b) Near-Sensor comparator circuit, (c) Comparators outputs when the different voltage from voltage divider (V_{out}) is fed to them.

IV. Event Detection Mechanism

To enhance energy efficiency, not all pixels are saved as the background; instead, some of them are algorithmically selected and stored in the background array. Then, they are used to be compared with the new pixel data to detect new events. For pixel selection purposes during the backup stage, we employ the concept of pixel boxes. We define a box around a group of pixels and select only one pixel from each box to be stored as the background data. Fig. 5 illustrates the pixel boxing and selection for backup and comparison purposes, where a box has been shown with a red dashed line.

The presented NSP offers configuration flexibility to enable dynamic trade-offs between accuracy and energy efficiency. By adjusting $box\_size \in \{3, 5, 7\}$ and $precision \in \{2, 3\}$, various design configurations can be determined where box_size indicates the dimensions of specified pixel groups, and $precision$ represents the bit-width of selected pixel data. As shown in Fig. 5, one pixel in each box is activated during backup and event detection mode, and the rest of the pixels are in the off state. As activated pixels remain ON during all operating phases, we refer to them as Always-On pixels. In the fabrication process, those pixels can be allocated a distinct VDD rail from other pixels. This adjustment allows designers

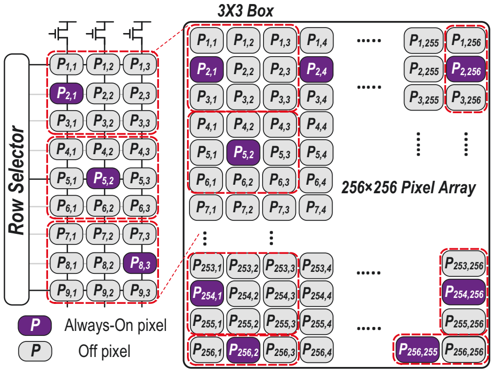

Fig. 5. Pixel selection using boxing method with the box size of 3×3.

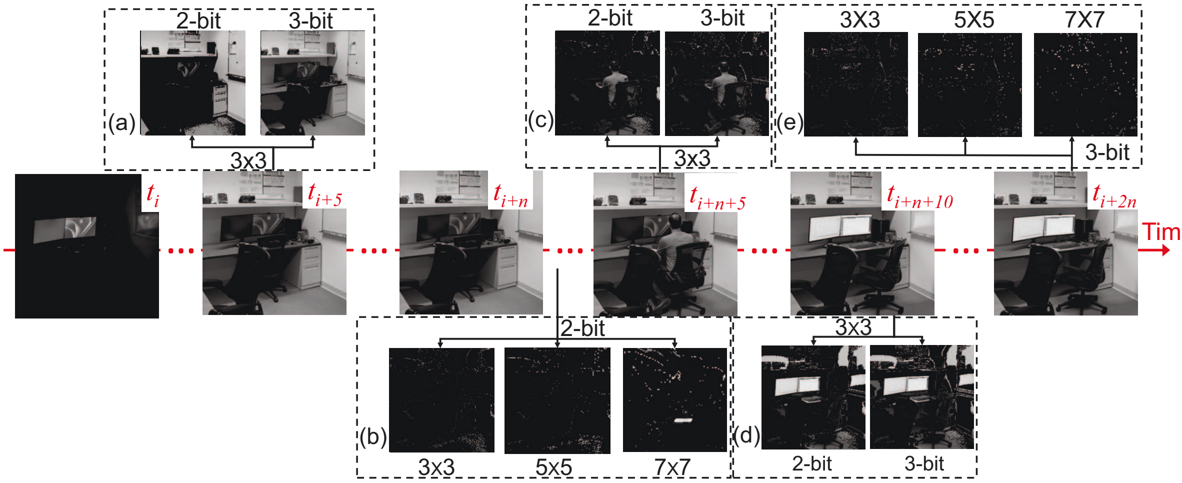

Fig. 6. Object detection timeframes of the NSP platform. In t_i, t_{i+n}, and t_{i+2n} background is updated. (a) to (d) shows the difference between the pre-stored background and captured image at different times under different box_size and $precision$ configurations. In (a), (c), and (d), a new event is detected. In (b) and (e) comparison shows zero difference means no new event.

to completely deactivate pixels other than the Always-On ones during background updating and event-detection modes without a need for a complex controlling mechanism. The horizontal positions of Always-On pixels are located in the central row of each box, while the column number changes. In this boxing method, only one pixel is activated in each column during each cycle. Thus, within every column of boxes, data from multiple boxes can be read and compared according to the box size. As an example, with $box\_size = 3$, the selected pixel of 3 boxes encompassing 9 rows of the pixel array can be read in a single cycle as depicted in Fig. 5. We begin pixel selection by choosing the left pixel of the first box in a column of boxes and for the subsequent boxes, we shift the selected pixel of the box one column to the right. As shown in Fig. 5, $P_{2,1}$, $P_{5,2}$, and, $P_{8,3}$ are chosen pixels and they can be compared in one cycle. To do this, in each cycle, the row selector activates three desired rows containing the selected pixel of the box. Since selected Always-On pixels are allocated distinct VDD rails, the rest of the pixels remain OFF even when their rows are selected by the row selector. Thus, each $n \times n$ pixel box as depicted in Fig. 5, contains one ON pixel and $(n^2 - 1)$ OFF pixels. For $box\_size \in \{5, 7\}$, similar to $box\_size = 3$, we start selection from the central pixel of the leftmost column and end with the rightmost column of the boxes. The near-sensor comparator keeps comparing this data with the background pixel value stored in the RRAM background array.

Algorithm 1 Proposed NSP Event Detection Algorithm

```
 1: Input_1: $box\_size \in \{3, 5, 7\}$ & $precision \in \{2, 3\}$-bit
 2: Input_2: $\text{threshold}_{\text{pixels}}$, $time_\tau$
 3: Output: sensor_mode status
 4: turn_on_list = []
 5: procedure EVENT-DETECTION
 6:     if time $\geqslant time_\tau$:          ▷ Merge steady objects with the background.
 7:         update (background)
 8:     for $i = \lfloor \frac{box\_size}{2} \rfloor + 1$ to 256 with step= $box\_size^2$
 9:         parallel_activate ($\text{row}_{\text{i}}$, $\text{row}_{\text{i}+box\_size}$, $\text{row}_{\text{i}\,+\,(2\times\, box\_size),\,\dots}$)
10:         pixel_values ← read_rows ()                    ▷ $j \in \{1, \dots, 256\}$
11:         changed_array ← parallel_comp (precision, pixel_values,
    old_values)
12:         turn_on_list.push (changed_array)              ▷ $i, j$ are box index.
13:     if (length (turn_on_list)$\geq$ $\text{threshold}_{\text{pixels}}$)
14:         time += 1                          ▷ Use it to update the background.
15:         enable SENSOR MODE
16:     else:
17:         time = 0
18: end procedure
19: procedure SENSOR MODE
20:     result =[]
21:     while (length (turn_on_list) !=0) do
22:         result +=read_box (turn_on_list.pop)
23:     end while
24:     compress_send (result)
25: end procedure
```

Algorithm 1 illustrates the procedure, including the event detection and sensing modes of the presented architecture. The box_size, $precision$, $\text{threshold}_{\text{pixels}}$, and $time_\tau$ values are given to the algorithm as inputs. The `turn_on_list` is utilized as an array to store the number of mismatches between background data and pixel data during event detection mode. If the number of mismatches exceeds the $\text{threshold}_{\text{pixels}}$, it is considered a new event. It should mention this variable reset after each event-detection run. The $time_\tau$ parameter serves as the time threshold for updating the background. Line 9 activates the rows containing the selected pixels, while line 10 reads their values. In line 11, the **`parallel_comp`** function conducts comparisons of new values and old values of 256 pixels based on the $precision$ parameter. All the indexes with change are pushed in the `turn_on_list` in line 12. The length of the `turn_on_list` is evaluated after finishing all rows. If it is bigger than $\text{threshold}_{\text{pixels}}$, the mode transitions to the sensing mode, and the time counter is added by one. This counter keeps track of how many times the NSP platform switches continuously to the sensing mode. Once this counter reaches time_τ, the NSP platform updates the background with the new values (line 7). In the **Sensor Mode**, all pixel values within the boxes defined by the `turn_on_list` are activated. Subsequently, these values are sequentially read on a row-by-row basis. It is important to note that the reading process is strictly row-oriented, which implies that an entire row is read even if only a single box intersects with it. This approach may lead to the generation of redundant data values. To mitigate the issue of transmitting duplicate data, a compression mechanism is applied to the acquired values before their transmission to the cloud. Specifically, line 24 of the algorithm details this compression process, ensuring that only new data are sent. Figure 6 depicts the object detection timeframe and qualitatively compares different scenarios under variations in $box_{s}ize$ and $precision$. At t_i, t_{i+n}, and t_{i+2n}, the background is updated when the light is OFF (t_i), ON (t_{i+n}), and after the position of a chair and the color of monitors are changed (t_{i+2n}). Figures 6(a), (b), (c), (d), and (e) show the differences between the background and the image captured at t_{i+5}, t_{i+n}, t_{i+n+5}, t_{i+n+10}, and t_{i+2n} respectively. Since the background is updated at t_{i+n} and t_{i+2n} and no changes have occurred, zero differences are detected in Figures 6(b) and (e). However, at t_{i+5}, the light is turned ON, at t_{i+n+5} a person has entered the frame, and at t_{i+n+10} the position of a chair is changed and the monitors are turned ON. Thus, Figures 6(a), (c), and (d) show the difference between the stored background and the captured image at those times.

V. Evaluations

Framework. The assessment framework is established using a bottom-up approach as shown in Fig. 7. Initially, at the device level, we utilized our experimental switching data from RRAM and formulated a Verilog-A model for co-simulation with interface CMOS circuits in Cadence Spectre and SPICE. At the circuit level, we commenced by implementing the RRAM/pixel array and associated peripheral circuitry using NCSU 45nm PDK [17] in Cadence, from which we extracted output voltages and currents on SLs. We employ the Synopsys Design Compiler for the creation of the controller, utilizing a standard industry-level 45nm technology. At the architectural level, we have adapted PiPSim [18] as a tool for evaluating in-/near-sensor performance. This tool enables the reporting of array-level read/write energy and latency. Moving to the application level, we have engineered an HW/SW simulator that integrates the proposed event detection method. This simulator utilizes architectural-level data of the RRAM background array and pixel array, facilitating the estimation of system performance.

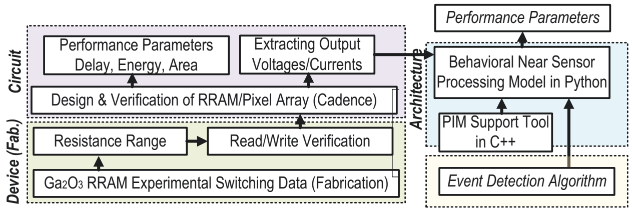
Performance Parameters
Circuit
Performance Parameters Delay, Energy, Area
Extracting Output Voltages/Currents
Architecture
Behavioral Near Sensor Processing Model in Python
Design & Verification of RRAM/Pixel Array (Cadence)
PIM Support Tool in C++
Device (Fab.)
Resistance Range
Read/Write Verification
Ga2O3 RRAM Experimental Switching Data (Fabrication)
Event Detection Algorithm

Fig. 7. Proposed evaluation framework.

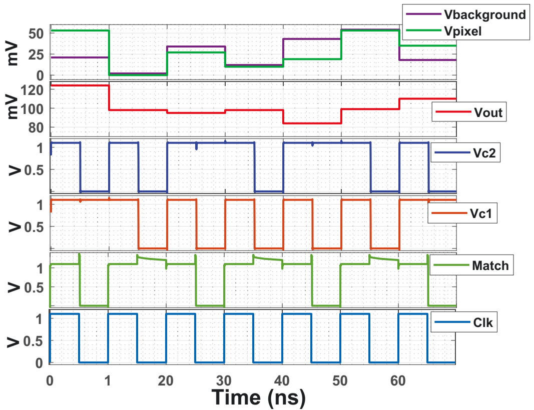

Fig. 8. Transient waveforms of the Near-Sensor Comparator unit.

Functionality Verification. Transient simulations are done to verify the functionality of the presented near-sensor comparator. The waveforms of these simulations have been depicted in Fig. 8. Here, in each clock cycle, different values of $V_{background}$ and V_{pixel} are fed to the near-sensor comparator unit. It is worth mentioning that in these simulations, 3-bit background data has been considered. V_{out} depicts the output of the voltage divider. As shown in Fig. 8, when the values of $V_{\text{background}}$ and V_{pixel} are very close to each other, the output of our voltage divider closely approximates its working point (100 mV). Conversely, when $V_{\text{background}}$ and V_{pixel} are not close, V_{out} deviates from its equilibrium. Voltage comparators are responsible for detecting this deviation. When the outputs of both comparators (V_{c1} and V_{c2}) are '0', the voltage of the matchline remains at '1'; otherwise, it discharges to '0'. The Clk signal synchronizes the operation: during Clk='1', precharge occurs in the matchline and comparators, while during Clk='0', evaluation occurs. According to Fig. 8, during the first, third, fifth, and seventh clock cycles, the Match signal is discharged to '0' because $V_{\text{background}}$ and V_{pixel} are not equal or close enough. However, during the second, fourth, and sixth clock cycles, the Match signal is '1' as $V_{\text{background}}$ and V_{pixel} almost match each other.

Comparative Analysis. Figure 9 illustrates the distribution of energy consumption during event detection mode, which involves background updating and event detection, across three platforms utilizing STT-MRAM, SOT-MRAM, and RRAM as the background array. This experiment explores scenarios involving both 2-bit and 3-bit configurations and considers three different box sizes. In the event detection mode, we presume that 5% of the time is dedicated to updating the background, while the remainder is allocated for identifying mismatches between pixel value and the previously stored background in the three platforms being tested. It's important to note that during the sensing mode, all platforms exclusively handle the same pixel array, resulting in equal energy consumption across all designs. Thus, the energy consumption of the pixel array is not considered in the comparisons.

According to Fig. 9, in background updating, platforms based on the SOT-MRAM and RRAM act better than the STT-MRAM-based one, due to their smaller switching energy. However, due to using an ADC-less comparison method in our platform, the overall energy of the presented RRAM-based design is lower than the other two platforms in all of the different configurations. Based on the results we note that (i) the NSP platform utilizing RRAM consumes approximately 66% and 63% less energy on average compared to designs based on STT-MRAM and SOT-MRAM, respectively. This is primarily attributed to the elimination of energy-intensive ADCs in the proposed near-sensor comparator. (ii) Since the switching energy of the utilized SOT-MTJ is slightly less than

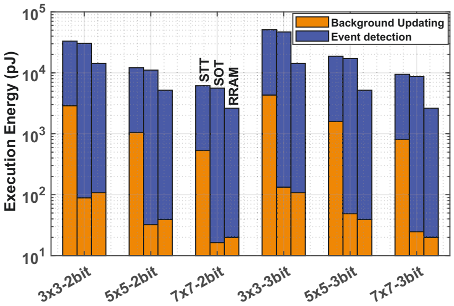

Fig. 9. Breakdown of energy consumption for three under-test platforms. In each bar-group from left to right: STT-MRAM, SOT-MRAM, and RRAM.

the switching energy of RRAM, for background updating the required energy by the SOT-MRAM-based platform is less than our platform in 2-bit configurations. However, in 3-bit configurations, our platform acts better as it only demands two RRAM switching per pixel while the SOT-MRAM-based platform needs three MTJs to be written. (ii) As precision increases (ranging here from 2 to 3 bits), there is a corresponding increase in the energy requirements for the edge device to conduct near-sensor computation. (iii) For a given precision level, employing a larger box size results in greater energy efficiency for the system. Figure 10 illustrates the execution time across six combinations of box sizes and precision levels for three platforms under test. Each bar represents two components: the execution time of event detection and background updating. As it can be observed in Fig. 10, the STT-MRAM-based platform has the largest execution time due to the high switching time of the STT-MTJs. Our presented platform has ∼34× smaller execution time on average compared to the STT-MRAM-based platform. Compared to the SOT-MRAM-based platform, our platform has on average 33% larger execution time, basically because RRAMs have larger switching delay than SOT-MTJs. However, the event detection delay of our platform is ∼3.3× smaller than the SOT-MRAM-based platform on average and as the time we operate for detecting events is much larger than the allocated time for background updating, the larger background update delay of our platform can be neglected.

VI. Conclusion

The paper introduces an energy-efficient near-sensor event detection platform based on a multilevel RRAM device. By presenting a new method of comparison and removing power-hungry ADCs, the presented design offers high energy efficiency suitable for edge devices. Design reconfigurability offers users a trade-off between precision and energy efficiency, allowing them to select the desired configuration according to their application. Our evaluation results show that the proposed platform achieves, on average, 66% and 63% energy savings over STT-MRAM and SOT-MRAM counterparts due to the utilization of the ADC-less method.

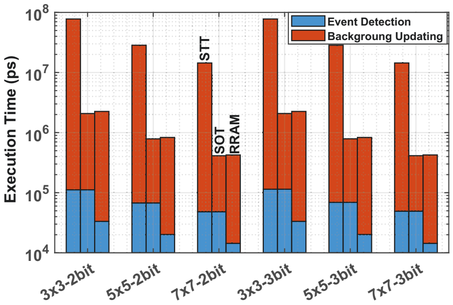

Fig. 10. Breakdown of execution time for three under-test platforms. In each bar group from left to right: STT-MRAM, SOT-MRAM, and RRAM-based designs.

Acknowledgment

This work is supported in part by the National Science Foundation under Grant No. 2228028, 2216772, 2216773, 2247156 and Semiconductor Research Corporation (SRC).

References

[1] M. Morsali *et al.*, "Design and evaluation of a near-sensor magneto-electric fet-based event detector," *IEEE Transactions on Electron Devices*, vol. 70, no. 9, pp. 4822–4828, 2023.

[2] M. Benetti *et al.*, "A low-power vision system with adaptive background subtraction and image segmentation for unusual event detection," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 65, no. 11, pp. 3842–3853, 2018.

[3] S. Tabrizchi *et al.*, "Nese: Near-sensor event-driven scheme for low power energy harvesting sensors," *arXiv preprint arXiv:2302.05431*, 2023.

[4] M. Morsali *et al.*, "Deep mapper: A multi-channel single-cycle near-sensor dnn accelerator," in *2023 IEEE International Conference on Rebooting Computing (ICRC)*, 2023, pp. 1–5.

[5] S. Angizi *et al.*, "Pisa: A non-volatile processing-in-sensor accelerator for imaging systems," *IEEE TETC*, 2023.

[6] A. Sridharan *et al.*, "A 1.23-ghz 16-kb programmable and generic processing-in-sram accelerator in 65nm," in *ESSCIRC 2022-IEEE 48th European Solid State Circuits Conference (ESSCIRC)*. IEEE, 2022, pp. 153–156.

[7] M. T. Nasab *et al.*, "High performance and low power spintronic binarized neural network hardware accelerator," in *2022 30th International Conference on Electrical Engineering (ICEE)*, 2022, pp. 774–778.

[8] X. Fong *et al.*, "Spin-transfer torque devices for logic and memory: Prospects and perspectives," *IEEE TCAD*, vol. 35, no. 1, pp. 1–22, 2015, doi: 10.1109/TCAD.2015.2481793.

[9] K. Ali *et al.*, "Energy- and area-efficient spin–orbit torque nonvolatile flip flop for power gating architecture," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 26, no. 4, pp. 630–638, 2018.

[10] M. Morsali *et al.*, "Nvlcff: An energy-efficient magnetic nonvolatile level converter flip-flop for ultra-low-power design," *Circuits, Systems, and Signal Processing*, vol. 39, no. 6, pp. 2841–2859, 2020.

[11] M. o. Abedin, "Material to system-level benchmarking of CMOS-integrated RRAM with ultra-fast switching for low power on-chip learning," *Scientific Reports*, vol. 13, no. 14963, 2023.

[12] S. Angizi *et al.*, "Cmp-pim: an energy-efficient comparator-based processing-in-memory neural network accelerator," in *DAC*, 2018, pp. 1–6.

[13] K.-L. Lin *et al.*, "Electrode dependence of filament formation in HfO2 resistive-switching memory," *JAP*, vol. 109, 04 2011.

[14] Y. Wu, B. Lee, and H.-S. P. Wong, "Al_2O_3-based rram using atomic layer deposition (ald) with 1-μA reset current," *IEEE Electron Device Letters*, vol. 31, no. 12, pp. 1449–1451, 2010.

[15] K. M. Kim, B. J. Choi, Y. C. Shin, S. Choi, and C. S. Hwang, "Anode-interface localized filamentary mechanism in resistive switching of TiO2 thin films," *Applied Physics Letters*, vol. 91, no. 1, p. 012907, 07 2007.

[16] R. T. Velpula *et al.*, "Low-power multilevel resistive switching in -ga2o3 based rram devices," *Nanotechnology*, vol. 34, no. 7, p. 075201, dec 2022. [Online]. Available: https://dx.doi.org/10.1088/1361-6528/aca418

[17] (2011) Ncsu eda freepdk45. [Online]. Available: http://www.eda.ncsu.edu/wiki/FreePDK45:Contents

[18] A. Roohi *et al.*, "Pipsim: A behavior-level modeling tool for cnn processing-in-pixel accelerators," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 2023.