Received 31 August 2024; revised 30 November 2024; accepted 11 December 2024.
Date of publication 16 December 2024; date of current version 3 January 2025.

Digital Object Identifier 10.1109/JXCDC.2024.3518633

E-MAC: Enhanced In-SRAM MAC Accuracy via Digital-to-Time Modulation

SAEED SEYEDFARAJI[1] (Member, IEEE), SALAR SHAKIBHAMEDAN[1], AMIRE SEYEDFARAJI[2], BASET MESGARI[1], NIMA TAHERINEJAD[1,3] (Member, IEEE), AXEL JANTSCH[1] (Fellow, IEEE), and SEMEEN REHMAN[4]

[1]Technische Universität Wien (TU Wien), 1040 Vienna, Austria
[2]Department of Electrical Engineering, Faculty of Engineering, Alzahra University, Tehran 1993893973, Iran
[3]Institute of Computer Engineering (ZITI), Heidelberg University, 69120 Heidelberg, Germany
[4]Institute of Parallel Computing Systems, Computer Science, University of Amsterdam, 1098 XH Amsterdam, The Netherlands
CORRESPONDING AUTHOR: S. SEYEDFARAJI (saeed.seyedfaraji@tuwien.ac.at)

This work was supported in part by Vienna University of Technology (TU Wien) Bibliothek through the Open Access Funding Program and in part by European Union's Horizon 2020 Research and Innovation program through the Marie Skłodowska Curie under Agreement 956090 (APROPOS: Approximate Computing for Power and Energy Optimisation).

ABSTRACT In this article, we introduce a novel technique called E-multiplication and accumulation (MAC) (EMAC), aimed at enhancing energy efficiency, reducing latency, and improving the accuracy of analog-based in-static random access memory (SRAM) MAC accelerators. Our approach involves a digital-to-time word-line (WL) modulation technique that encodes the WL voltage while preserving the necessary linear voltage drop for precise computations. This eliminates the need for an additional digital-to-analog converter (DAC) in the design. Furthermore, the SRAM-based logical weight encoding scheme we present reduces the reliance on capacitance-based techniques, which typically introduce area overhead in the circuit. This approach ensures consistent voltage drops for all equivalent cases [i.e., $(a \times b) = (b \times a)$], addressing a persistent issue in existing state-of-the-art methods. Compared with state-of-the-art analog-based in-SRAM techniques, our E-MAC approach demonstrates significant energy savings (1.89×) and improved accuracy (73.25%) per MAC computation from a 1-V power supply, while achieving a 11.84× energy efficiency improvement over baseline digital approaches. Our application analysis shows a marginal overall reduction in accuracy, i.e., a 0.1% and 0.17% reduction for LeNet5-based CNN and VGG16, respectively, when trained on the MNIST and ImageNet datasets.

INDEX TERMS 6T-static random access memory (SRAM), convolutional neural network (CNN), image classification, processing in memory (PIM).

I. INTRODUCTION

WITH the advent of neural network (NN) and deep neural network (DNN), the prospect of processing data comes with new challenges due to the large data movement between the processing and memory units. Studies indicate that data movement is the primary source of energy consumption in today's systems [1], [2], [3]. Among various data-intensive applications, NN has multiple load and store operations and associates energy overhead, which becomes more critical when running on devices with a limited power budget, such as embedded systems.

In recent times, processing in memory (PIM) accelerators received lots of attention for designing and implementing the multiplication and accumulation (MAC) operations to improve the energy efficiency of convolutional neural networks (CNNs) [3], [4], [5]. To reduce the load and store operations and associated energy/power overheads, many PIM approaches enable MAC computations inside the memory by exploiting its analog properties.

Before implementing PIM techniques, it is imperative to first identify the characteristics of these various options. Among various memory technologies used for PIM, static random access memory (SRAM) offers high subbank divisibility (i.e., provides parallel computing), ease of access, and low fabrication cost [4]. Different in-SRAM-based PIM techniques, both in analog and digital domains, have been proposed to accelerate MAC operation [4], [5], [6], [7], [8], [9], [10], [11], [12], [13], [14], [15], [16], [17]. Digital methods exploit additional logic to operate on the peripheral circuitry of memories (i.e., analog-to-digital converter

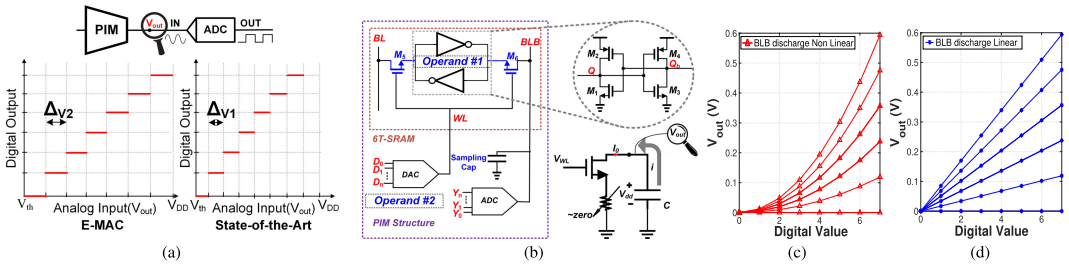


FIGURE 1. (a) Conceptual ADC transfer function for E-MAC versus state-of-the-art techniques. (b) 6T-SRAM memory structure and equal large-signal model during DAC operation. State-of-the-art BLB discharge behavior including (c) (red) nonlinear [4], [5], [7], [8] and (d) (blue) linear technique [6].

(ADC), digital-to-analog converter (DAC), additional logic, etc.) [9], [10], [11], [13]. Analog-based strategies often exploit the analog behavior of the memory, with little to no circuit overhead [4], [6], [7], [18].

A. PROBLEM MOTIVATION AND RESEARCH CHALLENGE

So far, the above state-of-the-art analog- and digital-based MAC accelerator techniques suffer from two fundamental drawbacks.

1) *Nonlinearity:* The current state-of-the-art PIM techniques [4], [5], [7], [8], [19] face a significant error source due to the quadratic current–voltage relationship of a MOSFET. As described in [4] and [6], the discharge behavior does not follow a linear trend with respect to the word-line (WL) voltage. Most proposed setups use a DAC to generate a linear WL voltage, but this approach leads to nonlinear outcomes during the multiplication process. Although state-of-the-art techniques [4], [6], [7] have shown improved accuracy at the circuit level, their design complexity increases due to the incorporation of extra circuitry. Consequently, these methods experience up to a 2× rise in power and energy consumption, which is a concern, particularly in scenarios involving complex applications such as NN.
2) *Nonideality:* When the multiplicands are generated via the WL [4], [5], [6], [7], [8], while V_{GS} is slightly higher than V_{th}, a small discharge occurs due to the nonzero source-drain current of a MOSFET. This phenomenon can significantly affect the result if it happens at the most significant bits (MSBs), as the contribution to the total discharge increases exponentially with the bit weight. As a consequence, this circuit nonideality can lead to different output analog voltages for $a \times b$ compared with $b \times a$. In such cases, although the mathematical results of the two multiplications are identical, the voltage drops will differ. Such discrepancy is not only unfavorable but also reduces the available voltage swing for the total number of multiplications, ultimately decreasing the accuracy of the computation. As depicted in Fig. 1(a), by leveraging the E-MAC technique, we can achieve a larger voltage margin between consecutive voltage drops ($\Delta_{V1} \leq \Delta_{V2}$), representing different multiplication results. This ultimately translates to higher output accuracy. In numerous circuit-level PIM developed thus far, a topology yielding equal outcomes for $a \times b$ and $b \times a$—equivalent voltage drops—has been absent. However, the proposed structure accomplishes this without incurring additional overhead or circuit complexities.

B. OUR NOVEL CONTRIBUTION

To tackle the outlined research challenges, we present a novel E-MAC technique with the following key innovations.

1) We introduce a constant-amplitude, digital-to-time PIM-based MAC accelerator that eliminates the need for a DAC. Using an analog time-based charge sampling technique directly on the bit-line-bar (BLB) of the SRAM, our design enhances MAC operations for CNN (see Section III). Integrating DAC functionality within the SRAM cell significantly reduces area and energy overhead, resulting in better efficiency and performance compared with state-of-the-art methods.
2) Our architecture generates logical bit values using memory cells, in contrast to state-of-the-art techniques that rely on capacitance-based weight generation or multiamplitude voltage methods [4], [6]. Capacitance-based designs increase circuit area and exhibit nonideal behaviors, while multiamplitude voltage approaches lead to nonlinear BLB voltage drops (see Section II-B).
3) To address nonidealities, we use differential sampling techniques on the BLB, ensuring consistent discharge voltages for operations such as $a \times b$ and $b \times a$. This reduces inaccuracies, improves ADC performance, and minimizes the impact of circuit and quantization noise. By eliminating the need for higher voltage levels, our approach enhances overall computational accuracy (see Section III-A).

The rest of this article is organized as follows. In Section II, we discuss the concept of in-SRAM MAC operation and state-of-the-art analog-based PIM techniques, elaborating on memory behavior with a simplified equivalent circuit in computation mode. Section III introduces our novel mathematical model for MAC operation and the proposed circuit based on a large-signal model. Section IV covers data-aware design and implementation concepts, along with application analysis results. Finally, Section V presents the conclusion.

II. CONCEPT OF IN-SRAM MULTIPLICATION AND ACCUMULATION

Fig. 1(b) shows the fundamental building block of a 6T-SRAM. It includes six transistors, with four transistors dedicated to storing data by the formation of two back-to-back inverters (i.e., M_1–M_4), while the remaining two transistors (i.e., M_5 and M_6) are connected as access transistors. To read from the cell, both bit-line (BL) and BLB are

initially floating high, i.e., the gate voltage of the access transistors (M_5, M_6) is discharged to the ground (WL is almost zero) assuming that Q is equal to zero and Q_b is subsequently V_{dd}. BL and BLB must be charged back to V_{dd}, and the WL is enabled by applying V_{dd} voltage to the gates of M_5, and M_6. In this case, the gate–source voltage of M_6 is almost zero (WL = V_{dd} and $Q_b = V_{dd}$), resulting in no significant current, hence no change in BLB and Q_b. However, BL is pulled down through M_5 and M_1, indicating that a stored zero ($Q = 0$) is being read from the cell. Similarly, when V_{dd} is stored in the cell, BL remains unchanged, but BLB must be discharged to zero.

For the write operation, the charging directions of BL and BLB must be different. For instance, to write V_{dd} into the cell, BL should be precharged to V_{dd} while BLB is pulled down to zero, and then V_{WL} is increased.

A. MAC USING THE SRAM READ AND WRITE OPERATION

A comprehensive analysis of the read and write operations within SRAM is elaborated in [4] and [6]. Analog-based PIM techniques exploit the conventional read operation of the memory element to perform the MAC operation. First, one operand is written to the memory element [Operand #1 in Fig. 1(b)]. The second operand is then passed as a coded voltage (proportional to the digital value) to the memory element's WL via a DAC (i.e., $D_0, D_1, \ldots, D_n$). The voltage drop on the BLB due to WL activation is sampled by capacitance and later interpreted as a digital output of the multiplication using an ADC [i.e., $Y_0, Y_1, \ldots, Y_n$ in Fig. 1(b)]. This concept can be used for MAC operations, providing results for $Y = D \times X$ and $Y = D + X$. Here, X denotes the first operand that is prewritten into the 6T-SRAM cell using a conventional write operation [i.e., Operand #1 in Fig. 1(b) or $[Q_n \ldots Q_1]$ in Fig. 3(a)].

The proposed architecture inherently supports both multiplication and addition operations. For multiplication, the circuit handles multiple distinct voltage levels and is more susceptible to errors due to nonlinearity and nonideality issues. Our technique demonstrates robust performance under these conditions by mitigating nonlinearity and ensuring consistent voltage headroom. Given that addition requires fewer distinct voltage levels than multiplication, the same in-SRAM circuit can perform addition reliably, leveraging the available voltage headroom without significant circuit modifications. Specifically, in addition mode, the ADC interprets the accumulated voltage on the BL to produce the sum. The integration of the accumulation process within the same in-SRAM architecture ensures efficient addition. This capability highlights the versatility and robustness of our design, as a circuit that performs well in the more demanding multiplication operation will excel in the simpler addition operation.

B. NONLINEAR VOLTAGE DROPS DURING MAC OPERATION

Considering the SRAM cell shown in Fig. 1(b), during an MAC operation, transistors M_2 and M_3 are in the deep triode region. At the same time, M_1 and M_4 are cut off. Therefore, Fig. 1(b) could be simplified using Kirchhoff's voltage law and Kirchhoff's current law, and V_{out} and I_0 is obtained as follows:

$$\frac{I_0}{C}t = V_{dd} - V_{out} \tag{1}$$

$$I_0 = \frac{\beta}{2}(V_{WL} - V_{th})^2 \tag{2}$$

$$\beta = \mu_n C_{ox}\left(\frac{W}{L}\right). \tag{3}$$

V_{out} is the output voltage of the computation, V_{dd} is the supply voltage equal to 1 V in the 65-nm CMOS technology, I_0 is the current passing through the transistors, C is the constant parasitic capacitance of the BL or BLB, and t denotes the time. V_{WL} is the voltage of the WL and V_{th} is the threshold voltage of the transistor. β is defined as (3), where μ_n refers to the electron mobility of the nMOS, and C_{ox} is the oxide capacitance per unit area. For simplicity, we neglect the effect of channel length modulation.

Since C is constant and relates to the design's physical properties, the only two free parameters to control the output voltage are t and I_0 [see (1)]. State-of-the-art approaches such as [4], [5], [6], [7], [8] have used I_0 as the control signal. However, the quadratic factor in I_0 [see (2)] introduces nonlinear behavior in V_{out}, as illustrated in Fig. 1(c). This nonlinearity challenges digital output interpretation, leading to increased errors and reduced ADC accuracy. Overlapping V_{out} values for different multiplications causes incorrect interpretations or requires high-precision ADCs, increasing system complexity and causing area and energy overhead.

Previous studies [4], [5], [7], [8] have addressed the issue of nonlinearity in in-memory computation. State-of-the-art techniques [4], [5], [7], [8] suffer from nonlinear output behavior, particularly in scenarios involving lower value multiplications [see Fig. 1(c)]. In such cases, the output voltages are closely spaced, making it difficult for the ADC to differentiate between values. This results in significant accuracy degradation, especially in applications requiring precise computations.

The technique proposed in [6] attempts to mitigate this nonlinearity by introducing additional circuitry, such as a root function circuit, which linearizes the voltage trend [see Fig. 1(d)]. While this approach improves accuracy by reducing the nonlinear effects, it increases the design complexity, area, and energy overhead. The root function circuit adds significant power consumption and area requirements compared with E-MAC, which achieves linearity without such additional hardware. These tradeoffs highlight the need for solutions that balance accuracy, energy efficiency, and hardware complexity, which our proposed approach addresses in Sections III-A–III-C.

In this article, we use the active time duration of a single WL [i.e., the parameter t in (1)] as a control signal, keeping the current source (I_0) constant. A replicated instance of the linear current generates distinct values. As a result, E-MAC improves the accuracy of the MAC operation compared with the state-of-the-art [4], [6], while also reducing energy consumption and circuit design complexity.

To summarize, compared with the state of the art, this article presents a novel technique (i.e., E-MAC) that differs significantly in several ways.

1) State-of-the-art techniques [4], [8], [20], [21], [22] use separate WL for each bit, requiring four control signals for a 4-bit multiplication to create logical weights. In contrast, our E-MAC approach uses a single WL and accumulative memory cells for a 4-bit MAC operation. This reduces error probability and circuit complexity, as only a single WL control timing is necessary, and generates weights using simple, accurately managed switches. This method consumes less energy compared with controlling four individual WLs.
2) State-of-the-art approaches [20], [21], [22] use an approximated Taylor series for the voltage–current equation. Our E-MAC technique solves a nonlinear differential equation [see (4)–(7)] to ensure output voltage accuracy.
3) We present a novel approach to multiplication that reduces design cost and avoids the nonidealistic overload of expensive DAC. While state-of-the-art techniques rely on DAC to interpret multiplication output, we use a shared ideal memory cell. This achieves our objectives while maintaining accuracy and efficiency.

III. OUR NOVEL E-MAC TECHNIQUE, AND CONCEPT OVERVIEW

In this section, we introduce and present an analytical model that analyzes circuit behavior, facilitating efficient circuit design. We then propose our time-based solution to address the issue of obtaining identical results despite the diverse voltage drops present in state-of-the-art approaches.

A. WL's TIME MODULATION MODELING AND CONSTRAINTS

In our approach to achieve meaningful interpolation between the digital input and the analog voltage, we first need to create a coding scheme to map the 3-bit operand into a 7-bit operand according to the 4-2-1 logical weighting scheme stored in the memory column (First Operand → Coded Operand). Using the already existing memory cells without the need for additional circuitry, our approach avoids imposing any area overhead on the circuit. Regarding the additional write required for this process, the power gained through analog multiplication compensates for the added energy consumption.

Second, the other operand needs to be coded into a time interval (using a programmable frequency divider [23]) to keep the WL activated (see Table 1). For this coding, we will not activate the WL for an operand of "0" (i.e., the result will always be zero), and therefore, no discharge will occur in this case, resulting in an output of zero (in the case of multiplication). Fig. 2 displays the design structure of the proposed methodology.

Note that you can create the pulses mentioned above, equal to Δt_i, using a frequency functional synthesizer alongside the system's existing clock. It is important to emphasize that although the PWM technique used here may appear commonplace and obsolete, as it has been proposed in other computing systems, our approach takes a unique perspective. Our primary objective is to preserve the linearity of BL drops. As discussed earlier, any method, including those proposed in state-of-the-art references [4], [5], [6], [7], [8], which attempts to encode data in the amplitude of the WL, will inevitably result in a quadratic response in the output, leading to nonlinearity.

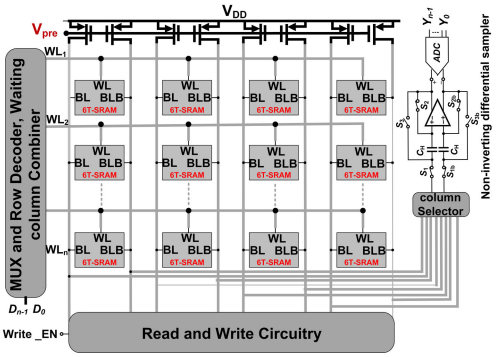


FIGURE 2. Architecture of the proposed 6T-SRAM PIM.

TABLE 1. Encoding scheme of operands in an MAC process.

First Operand	Coded operand	Second Operand	$\Delta t_{2'd}$ (ns)
000	0000000	000	0
001	0000001	001	$(1/7)T_{max}$
010	0000110	010	$(2/7)T_{max}$
011	0000111	011	$(3/7)T_{max}$
100	1111000	100	$(4/7)T_{max}$
101	1111001	101	$(5/7)T_{max}$
110	1111110	110	$(6/7)T_{max}$
111	1111111	111	T_{max}

To clarify the analog coding for the second nonzero operand (as indicated in Table 1), we refer to Figs. 1(b) and 2, and the conditions outlined in (1) and (2), and the requirement for the nMOS transistor to remain in the active (saturation) region [6]. The analog behavior of the circuit relies on maintaining transistor $M5$ in saturation to ensure accurate voltage sampling during the discharge process. By accounting for the provided supply power and operational constraints, the maximum allowable sampling time can be calculated.

Equation (2) is valid only while $M5$ remains in the saturation region. However, as the BLB discharges, $M5$ transitions to the triode (linear) region, slowing the BLB discharge rate and introducing inaccuracies in the computed result. To mitigate this, it is critical to sample the BLB voltage before $M5$ enters the triode region. This requires precise control of the WL voltage (V_{WL}) pulsewidth, denoted as T_{Max}. Controlling T_{Max} ensures that sampling occurs while $M5$ remains in saturation, preventing systematic errors due to delayed discharge.

The maximum pulsewidth (T_{Max}) is derived based on the saturation condition of the nMOS transistor. Using the parameters from (4) to (6), T_{Max} is calculated to ensure compliance with the saturation condition and to maintain the accuracy of the discharging behavior described in (2). This approach ensures robust and error-free operation across varying conditions, addressing potential variability introduced as capacitor C discharges and I_0 changes over time

$$V_{\mathrm{out}} \leq \alpha V_{\mathrm{dd}} - V_{\mathrm{th}} \tag{4}$$

$$T_{\mathrm{max}} = \frac{[(1-\alpha)V_{\mathrm{dd}} - V_{\mathrm{th}}] \times C}{\frac{\beta}{2} \times [(1-\alpha)V_{\mathrm{dd}} - V_{\mathrm{th}})]^2} \tag{5}$$

$$\Delta t_i = \left(\frac{T_{\max}}{2^N - 1}\right) \times A_{(10)} \tag{6}$$

$$V_{\text{out}} = V_{\text{dd}} - \frac{\sum_{i=0}^{N-1} I_i}{C} \Delta t_i. \tag{7}$$

Hence to calculate each sampling time for other operands, the time coding will follow (5). Δt_i is the required time interval interpolated with digital input (0 up to 7 in Table 1). $T_{\max}$ is the maximum required time that the transistor M_6 remains active and it belongs to 7×7 multiplication, $A_{(10)}$ is the E-MAC equivalent of the input, and N is the bit-width of the input. The behavior of the proposed architecture is modeled as Fig. 3(a). The MSB, including four memory cells, is modeled into a transistor with a size of $4(W/L)$, and in this case, the LSB fits into $1(W/L)$ transistor. The second operand will be passed to the gate of lower transistors, and it would be either "0"(V) or V_{dd}. Therefore, by exploiting KVL for V_{out} and KCL on the branches, we calculate as (7). Consider that α is a scaling factor, typically less than 1, applied to the gates of the transistors associated with the WL. The selection of α depends on the desired rise and fall times of the WL voltages. Lowering the voltage at the gates results in faster rise and fall times.

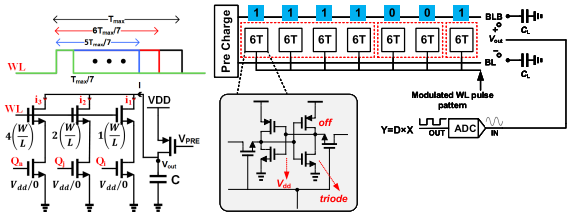


FIGURE 3. (a) Large-signal model of circuits during MAC operations and (b) encoding scheme for (5 × 4).

Example Scenario: To better illustrate our methodology, let us consider the following example. Suppose we need to multiply 5×4 in our application. First, the first operand (**5**) is encoded using seven memory elements, as shown in Table 1. The resulting binary code, **"1111001,"** is then written into seven memory cells, highlighted in blue [see Fig. 3(b)].

Next, the second operand (4) is applied via the shared WL of these memory cells, represented by the time duration of the WL signal. Throughout this process, the amplitude of the WL signal remains constant, ensuring consistency and accuracy in the operation.

As depicted in Fig. 3(a), after the precharge process, the WL signal remains active for $(4/7) \times T_{\max}$, causing the capacitance C_L discharge via weighted (4-2-1) equal current sources. In this example, the middle two cells are passing zero ampere current. Finally, the output can be sampled once the WL signal is deactivated.

B. PROBLEM OF IDENTICAL RESULTS WITH DIVERSE VOLTAGE DROPS

The utilization of analog in-SRAM multiplication introduces an inherent issue due to the method in which the multiplication process is executed. As mentioned before, one of the operands is stored inside the cell while the other is provided as an analog input to the WL. The analog input can be in the form of voltage amplitudes, as seen in previous studies, or as a time notion, as presented in this article. When the WL is activated, there is a decline in the charge retained in either the BL or BLB capacitance. Later, this decline is retrieved and interpreted as the multiplication outcome by configuring an ADC at the output. This way, one operand is in the digital domain, while the other behaves as an analog signal. Consequently, it is possible to encounter a phenomenon where different voltage drops lead to nonidentical outputs. To clarify, in the case of multiplying two numbers, A and B, where A and B range from 0 to 3 (i.e., up to 3-bit-width), there are only seven feasible individual results.

The multiplication of 4-bit numbers often results in patterns with a limited number of distinct products. These similarities are notable across various combinations, with only a few unique products. However, current analog in-SRAM multiplication techniques, as used in state-of-the-art methods, have not addressed this issue. A specific example is when multiplying 1×7 or 7×1, where unequal voltage drops occur over the BL/BLB despite both the operations yielding the same product of 7. This inconsistency leads to additional circuit complexity and cost. To address this, we applied the differential voltage sampling (DVS) technique, which allowed us to use the extra voltage margin to improve multiplication accuracy, increase bit width, or reduce energy consumption. In contrast to the existing methods, which typically sample the voltage drops on BL or BLB to calculate the result, our approach ensures that these voltages can be represented more efficiently, as described by the corresponding equation

$$V_{\text{BL/BLB}} = V_{\text{dd}} - \frac{\sum_{i=0}^{N-1} I_i}{C} \Delta t_i. \tag{8}$$

Equation (8) is inherently nonlinear due to the existing V_{dd} factor which varies by time while sampling. By leveraging DVS, the nonlinear term V_{dd} will be eliminated (since both BL and BLB have one V_{dd} factor in common), and therefore, we arrive at the following equation:

$$|V_{\text{BL/BLB}}(\text{DVS})| = \frac{\sum_{i=0}^{N-1} I_i}{C} \Delta t_i. \tag{9}$$

Using this differential technique, we have successfully eliminated the nonlinearity issue in the output voltage. This problem has been a challenge in state-of-the-art approaches. It is crucial to emphasize that achieving linearity is essential for ensuring equal voltage drops and identical results. In this approach, for example, the output voltage for both (4×5) and (5×4) computations is equal. In the first scenario, the first operand (i.e., 4) is equal to $4.I_0$ multiplied by $(5/7) \cdot T_{\max}$ which is equal to $(20/7) \cdot I_0 \cdot T_{\max}$ which is exactly equal to the second scenario for 5×4.

C. ANALYTICAL MODEL AND E-MAC CIRCUIT IMPLEMENTATION AND RESULTS

We developed our analytical model in MATLAB R2022a. The circuit architecture of our proposed 6T-SRAM PIM (Fig. 2) is implemented in a 65-nm CMOS technology with a supply voltage of 1 V. The circuit parameters, such as the size of transistors, BLB capacitance, and the $T_{\max}$, are optimized using Cadence SPECTRE transient simulation according to (5)–(7) ($V_{\text{th}} = 250$ mV and $V_{\text{DD}} = 1$ V).

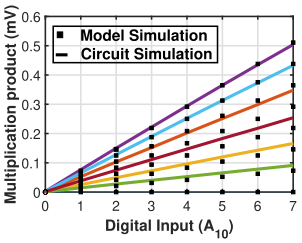


FIGURE 4. Circuit results verify the accuracy of the proposed analytical model.

Our optimization objective is to minimize the power and area of our circuit. Fig. 4 displays the overlapping plot of our model and the circuit simulation outcomes. As shown, our model accurately predicted the behavior of the final circuit. The slight deviation in specific numerical values is due to parasitic capacitance and the physical parameters of the model, which were also considered in the circuit simulation. Nevertheless, this does not impact the model's reliability since it follows the same linear trend for the multiplication result. Moreover, when the multiplication result is the same, the discharge of the BL is also similar, as evident from Fig. 4. The incorporation of the mathematical model alongside the circuit simulation validates the effectiveness of our approach in preserving the necessary linearity of voltage drop.

Considering two signed integer numbers in the range of [−7:+7] for multiplication, we performed a 1000-point Monte Carlo (MC) simulation to depict the effect of threshold voltage, gate oxide thickness, and various mobility of transistors on the accuracy of the result in Fig. 5(b). We achieved a 24.95-mV standard deviation from the accurate result for the worst case multiplication (7×7), which outperforms the state-of-the-art accuracy by 73%. Fig. 5(c) depicts the result for worst cases (7×7). Table 2 demonstrates the comprehensive analysis of the proposed architecture in circuit metrics compared with state-of-the-art approaches [4], [5], [6], [7], [8]. As can be seen, E-MAC outperforms the state-of-the-art and achieves 47.31% energy improvement compared with the best result, and in terms of accuracy, we improved by 73.25%. We should elaborate that there exist other techniques within the domain of in-SRAM, as detailed in [14]. These alternatives may operate either in the digital domain or as near-memory core accelerators. However, we have chosen the aforementioned techniques to compare due to their alignment with the proposed abstraction level. It is important to note that digital or near-memory approaches would not perform as effectively as pure analog in-SRAM techniques [14], [24] in this context, rendering a comparison meaningless.

TABLE 2. Comprehensive comparison to state-of-the-art in-SRAM analog-based MAC accelerators.

	E-MAC	[4]	[5]	[6]	[7]	[8]
Technology (nm)	65	65	65	65	180	65
Supply Voltage (V)	1	1.2	1.2	1	1.8	1
Bits-width	4	5	5	4	5	8
MAC Energy (pJ)	0.147	0.279	3.5	0.523	1.167	1.3
Accuracy $(std.dev)$	0.023	0.6	/	0.086	/	/
CLK Freq. (MHz)	103.1	100	2.5	200	/	60-125

In our study, we have rigorously addressed the impact of parasitic capacitance through three comprehensive approaches. First, we defined parameter C to encapsulate the cumulative effect of all the parasitic capacitors from the node to the ground, as detailed in Figs. 1(b) and 3(a). This parameter includes contributions from the drain–bulk, drain–source, and drain–gate capacitances of all connected transistors, along with a column capacitor influencing the discharge behavior of each column. The significance of parameter C in circuit behavior is thoroughly analyzed in (1) and (5). Second, the 6T-SRAM cell layout, which occupies $5.07 \times 4.89\mu$ m in the 65-nm CMOS technology, presented in Fig. 5(a), accounts for the parasitic capacitances of the transistors and interconnects between them. The hierarchical layout approach ensures accurate postlayout simulations, including peripheral circuitry, with the results depicted in Figs. 4 and 5(b) and (c). Finally, our proposed structure underwent postlayout circuit simulation using the TSMC 65-nm CMOS design process, incorporating precise parasitic modeling via BSIM 4 and validated through MC simulations.

While our current study focuses on the 65-nm technology node, we acknowledge the potential significance of parasitic capacitance in more advanced nodes such as 28 and 14 nm. The study [6] demonstrated that despite pronounced deep-submicrometer effects in smaller channel lengths, the discharge behavior of the BLB port remains unaffected, indicating robustness in our design. Furthermore, the shift from quadratic to linear current behavior in more advanced technologies is anticipated to enhance the accuracy of our design, as detailed in our previous findings.

IV. APPLICATION EVALUATION

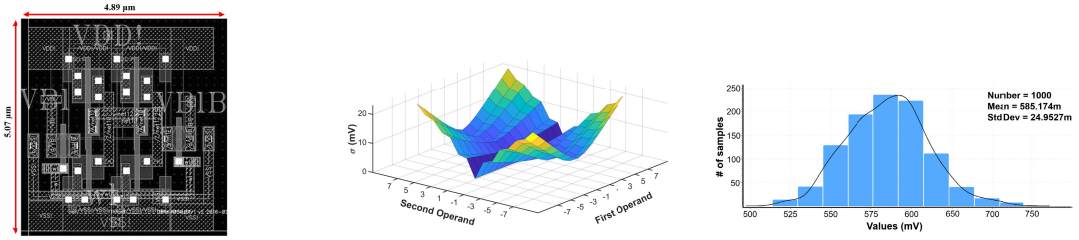
The proposed in-memory multiplier technique offers significant potential for energy-efficient DNN inference on resource-constrained devices.

To evaluate the effectiveness of the proposed technique, we have considered CNN due to their significant energy consumption and data intensity requirements. This article uses a Lenet5-Inspired CNN [25], [26] and the VGG16 [27] architecture to conduct an image classification task on the MNIST [28] and ImageNet [29]. The Lenet5-Inspired CNN consists of eight layers, including five convolutional layers and three fully connected (FC) layers, with the last layer considered the output layer indicating the class of the input image.

Most operations in DNN are multiplications that have a higher energy consumption compared with other arithmetic operations [30]. Therefore, techniques such as the in-memory multiplier, which can reduce the energy consumption of multiplication operations, are crucial for improving the energy efficiency of DNN. The 4-bit signed quantization [31], [32], [33] is a specific type of quantization used in DNN. Since 4-bit signed quantization uses only eight absolute values, it can represent only eight distinct values. This significantly reduces the memory and computational requirements of DNN, making them more efficient and easier to deploy on hardware with limited resources. However, the reduction in precision can also lead to quantization errors and decreased model accuracy.

A. CNN-BASED IMAGE CLASSIFICATION (MNIST)

Our experiments use the MNIST dataset [28] for training and inference (testing). The training was conducted using ADAM



(a) (b) (c)

FIGURE 5. **(a) Area of the proposed in-SRAM multiplier, (b) σ of multiplication results, and (c) MC simulation for 1000 points for worst (7 × 7) multiplications.**

TABLE 3. **Performance results for float quantized, and proposed model for LeNet5-Inspired CNN.**

Model	Accuracy	Precision	Recall	F1-Score
Original (float32)	99.29	99.28	99.28	99.28
Quantized (int4)	99.01	99.01	99.02	99.01
In-Memory (int4)	98.91	98.91	98.90	98.90

TABLE 4. **Energy performance comparison to baseline and related works.**

	Energy (nJ/inference) (MNIST)	Relatively Energy Gain (MNIST)	Energy (nJ/inference) (ImageNet)	Relatively Energy Gain (ImageNet)
von Neumann system	3866.78	1	16.77	1
[4]	994.85	1.89	-	-
[6]	1864.90	3.55	-	-
E-MAC	524.17	7.37	2.27	7.37

TABLE 5. **Comparison to related works, tested with the MNIST.**

Methods	Input/Weight Precision	Algorithm	Baseline Accuracy	Accuracy Degradation
[4]	5/5	LeNet-5	**99.20%**	0.15%
[36]	6/4	LeNet-5	99.7%	1.30%
[37]	-/4	MLP	98.27%	0.11%
[38]	ternary	MLP	98.77%	0.12%
[39]	(16 or 32)/(8 or 16)	MLP	98.36%	0.14%
[40]	16/4	LeNet-5	98%	1%
[41]	14/(4 or 5)	SNN	90%	-
E-MAC	**3/4**	CNN	99.01%	**0.10%**

optimizer [34], 2048 as the batch size, and for 2000 epochs via categorical cross-entropy as the loss function. Afterward, the trained CNN was quantized to the int-4 (−7 to 7) by considering the maximum absolute value of each layer in the trained network. In the following, all the multiplication operations, replaced by the proposed in-memory multiplier and so the (in-memory) trained quantized CNN (4-bit weights and activations), reached 98.91% accuracy of the MNIST test data. The confusion matrix and performance criteria of the trained quantized (int-4) CNN are depicted in Fig. 6(d) and Table 3, respectively.

As demonstrated, the accelerator's output is an 8-bit output that has been interpreted from an analog voltage of the BLB. The voltage of the BLB could be from 0.4 to 1 V. This voltage interval is discretized into 256 different levels, and each level is dedicated to a number in the range from −128 to 127 interval using an ADC. As mentioned earlier, the weights of the CNN have been quantized to int-4 representation −7 to 7 (symmetric), while the MNIST dataset is in the uint-8 format; its number interval is 0 to 255. Therefore, it is not aligned with the int-4 representation. Consequently, a mapping is done to the uint-8 to the (0–7) interval to be aligned with the int-4 representation.

From the t-distributed stochastic neighbor embedding (t-SNE) plot [35], the probability distribution, discrimination, and distance between the data points can be studied. It is shown that the quantization effect on the MNIST dataset is negligible. The distributions of the MNIST dataset in int-8 and int-4 representations are demonstrated in Fig. 6(a) and 6(b), respectively.

As demonstrated in Fig. 6(a), our quantization method has not affected the MNIST data distribution significantly. In the inference phase, as mentioned earlier, all the multiplication operations are carried out using the proposed in-SRAM multipliers for achieving energy efficiency with negligible performance degradation. As shown by the criteria (see Table 3), performance of CNN almost remains the same with negligible accuracy degradation. In addition to overall performance that is demonstrated in Table 3 and Fig. 6(d).

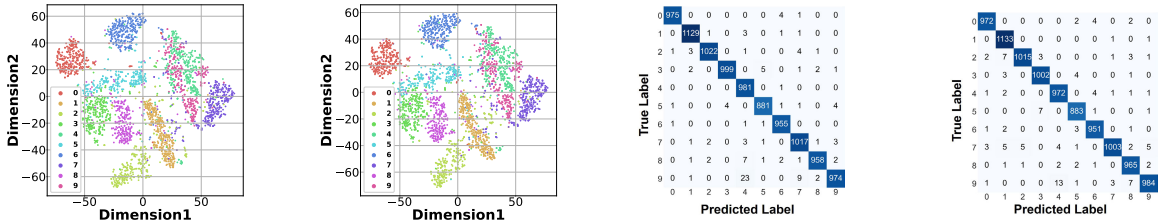
As can be seen in Fig. 6(c), and (d), although the overall accuracy has a negligible drop compared with the baseline architecture in 4-bit data representation, the classification accuracy of classes "4," "7," and "9" has been improved. This happened because of the pattern and spatial structure and also the numbers that have been used to represent these classes.

As the next step, the performance of the proposed CNN is evaluated by considering the energy aspect. We compared our proposed in-SRAM multiplier with a conventional von Neumann system [36] and state-of-the-art in-memory multiplier accelerators. To have a fair comparison, we consider all other experimental setups and circuit parameters equal to [4] and [6]. We have reported the energy gain and energy per inference in Table 4.

For a comprehensive study, we also compare our model performance with the other in-SRAM multipliers with analog computation nature which are implemented for the MNIST dataset. As shown in Table 5, we obtain minimal accuracy degradation which means that E-MAC in-SRAM multiplier minimally impacts the performance of the CNN while providing energy efficiency up to 15.53×. In addition, we gained the least accuracy degradation compared with the state-of-the-art (see Table 5).

B. VGG16 (IMAGENET)

In this case, we used the ImageNet dataset for inference (testing) purposes in our experiments. We evaluated VGG16 using the proposed in-memory multiplier to assess its effect on a



(a) (b) (c) (d)

FIGURE 6. t-SNE visualization of the MNIST dataset in (a) 8-bit and (b) 3-bit representation. Confusion matrix for the accelerated trained quantized CNN in (c) 8-bit and (d) 4-bit number representations.

TABLE 6. Performance results for float, quantized, and proposed model for VGG16.

Model	Top-1 Accuracy	Top-5 Accuracy	Geomean Of Top-1 & Top-5 Accuracies
Original (float32)	71.30%	90.10%	80.15%
Quantized (int4)	70.29%	89.24%	79.20%
In-Memory Multiplier	70.12%	89.18%	79.08%

higher level. To evaluate our classifier, we use the following performance metrics (top-1 accuracy and top-5 accuracy) of the original and quantized models.

A modified TF-Lite [42] quantization specification has been used to quantize the trained VGG16 weights from float-32 to int-4. We then replaced all the multiplications in our model with the proposed in-memory multiplier. As demonstrated in Table 6, due to quantization, top-1 accuracy and top-5 accuracy have been degraded by only 1.01% and 0.86%, respectively.

However, using the in-memory multiplier resulted in up to 0.17% and 0.06% degradation in top-1 accuracy and top-5 accuracy, respectively, compared with our quantized baseline model, due to the compactional error of the aforementioned in-memory multiplier. Nonetheless, our proposed in-memory multiplier can be used meaningfully in VGG16 due to the negligible change in performance criteria.

Our results show that we can reach 7.40× energy gain by sacrificing 1.18% and 0.92% of top-1 accuracy and top-5 accuracy, respectively. These results show that in scenarios where we face power constraints (such as embedded systems and edge devices) or demands for real-time inference, we can address these priorities by tolerating a negligible accuracy drop. To have criteria that contain both resource requirements (energy) and performance (top-1 accuracy and top-5 accuracy), we can multiply the resource (energy) gains in Table 4 with the performance values in Table 6. This allows us to better evaluate and select suitable designs when both performance and resources have a similar importance (weight) in the system. The combined gains are determined through the multiplication of the energy gains derived from Table 4 with the corresponding performance measurements (namely, top-1 and top-5 accuracies) presented in Table 6. Based on the results of our experiments, we observed that VGG16, trained on the ImageNet and using our proposed in-SRAM multiplier, provides high accuracy with energy efficiency.

In summary, we used MNIST for training and inference of our proposed LeNet5-inspired CNN. Afterward, we quantized the trained CNN to int-4 representation, reaching 99.01% accuracy on the quantized MNIST test data. The quantized CNN demonstrated negligible impact on the distribution of the MNIST, and the performance of the CNN almost remained the same, with a little degradation. The proposed in-SRAM multipliers used for carrying out all multiplication operations in the inference phase achieved energy efficiency, with negligible performance degradation.

In comparison to a conventional von Neumann system and state-of-the-art in-memory multiplier accelerators, our proposed in-SRAM multiplier showed better energy efficiency and accuracy degradation. The E-MAC achieved the minimum effect on the performance of the CNN while providing energy efficiency of up to 7.37×. In addition, we evaluated the E-MAC on VGG16 which is pretrained on ImageNet. Our proposed method and reported results lay a strong foundation for further research in the area of image classification task.

Overall, using the E-MAC for implementing image classification algorithms provides an efficient and effective solution and demonstrates high accuracy with energy efficiency. The results of our experiments open new avenues for future research in the field, with potential applications in a wide range of fields, including healthcare, autonomous systems, robotics, etc.

V. CONCLUSION

This article introduced a novel in-SRAM MAC accelerator, E-MAC, which uses a time-modulated technique to regulate the WL voltage, thus eliminating the need for a DAC. To assess the effectiveness of this approach, we evaluated its performance at both the circuit and application levels. E-MAC was designed to meet the requirements of two CNN (LeNet-5 inspired and VGG16) and tested on the MNIST and ImageNet datasets. The architecture is optimized for minimal energy consumption while maintaining high inference accuracy. Implemented in a 65-nm CMOS process, E-MAC demonstrated a 1.89× improvement in energy efficiency and a 73.25% increase in accuracy per MAC over previous designs. Application-level evaluation showed minimal impact on inference accuracy, with reductions of only 0.1% for LeNet-5 and 0.17% for VGG16, while achieving a substantial energy saving of 7.37×.

REFERENCES

[1] A. Boroumand et al., "Google workloads for consumer devices: Mitigating data movement bottlenecks," in *Proc. 23rd Int. Conf. Archit. Support Program. Lang. Operating Syst.*, 2018, pp. 316–331.

[2] W. Wulf and S. McKee, "Hitting the memory wall: Implications of the obvious," *Sigarch Comput. Archit. News*, vol. 23, no. 1, pp. 20–24, 1995.

[3] J. Zhang, Z. Wang, and N. Verma, "In-memory computation of a machine-learning classifier in a standard 6T SRAM array," *IEEE J. Solid-State Circuits*, vol. 52, no. 4, pp. 915–924, Apr. 2017.

[4] M. Ali, A. Jaiswal, S. Kodge, A. Agrawal, I. Chakraborty, and K. Roy, "IMAC: In-memory multi-bit multiplication and accumulation in 6T SRAM array," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 67, no. 8, pp. 2521–2531, Aug. 2020.

[5] K. Sanni, T. Figliolia, G. Tognetti, P. Pouliquen, and A. Andreou, "A charge-based architecture for energy-efficient vector-vector multiplication in 65nm CMOS," in *Proc. IEEE Int. Symp. Circuits Syst. (ISCAS)*, May 2018, pp. 1–5.

[6] S. Seyedfaraji, B. Mesgari, and S. Rehman, "AID: Accuracy improvement of analog discharge-based in-SRAM multiplication accelerator," in *Proc. Design, Autom. Test Eur. Conf. Exhib. (DATE)*, Mar. 2022, pp. 873–878.

[7] T. Chen, J. Botimer, T. Chou, and Z. Zhang, "An SRAM-based accelerator for solving partial differential equations," in *Proc. IEEE Custom Integr. Circuits Conf. (CICC)*, Apr. 2019, pp. 1–4.

[8] M. Gong, N. Cao, M. Chang, and A. Raychowdhury, "A 65nm thermometer-encoded time/charge-based compute-in-memory neural network accelerator at 0.735pJ/MAC and 0.41pJ/update," *IEEE Trans. Circuits Syst. II, Exp. Briefs*, vol. 68, no. 4, pp. 1408–1412, Apr. 2020.

[9] M. Zhou, Y. Guo, W. Xu, B. Li, K. W. Eliceiri, and T. Rosing, "MAT: Processing in-memory acceleration for long-sequence attention," in *Proc. 58th ACM/IEEE Design Autom. Conf. (DAC)*, Dec. 2021, pp. 25–30.

[10] L. He, C. Liu, Y. Wang, S. Liang, H. Li, and X. Li, "GCiM: A near-data processing accelerator for graph construction," in *Proc. 58th ACM/IEEE Design Autom. Conf. (DAC)*, Dec. 2021, pp. 205–210.

[11] F. Zhang, S. Angizi, N. A. Fahmi, W. Zhang, and D. Fan, "PIM-quantifier: A processing-in-memory platform for mRNA quantification," in *Proc. 58th ACM/IEEE Design Autom. Conf. (DAC)*, Dec. 2021, pp. 43–48.

[12] S. Seyedfaraji, B. Mesgari, and S. Rehman, "SMART: Investigating the impact of threshold voltage suppression in an in-SRAM multiplication/accumulation accelerator for accuracy improvement in 65 nm CMOS technology," in *Proc. 25th Euromicro Conf. Digit. Syst. Design (DSD)*, Aug. 2022, pp. 821–826.

[13] K. Soundrapandiyan, S. K. Vishvakarma, and B. S. Reniwal, "Enabling energy-efficient in-memory computing with robust assist-based reconfigurable sense amplifier in SRAM array," *IEEE J. Emerg. Sel. Topics Circuits Syst.*, vol. 13, no. 1, pp. 445–455, Mar. 2023.

[14] C.-J. Jhang, C.-X. Xue, J.-M. Hung, F.-C. Chang, and M.-F. Chang, "Challenges and trends of SRAM-based computing-in-memory for AI edge devices," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 68, no. 5, pp. 1773–1786, May 2021.

[15] I. A. Papistas et al., "A 22 nm, 1540 TOP/s/W, 12.1 TOP/s/mm2 in-memory analog matrix-vector-multiplier for DNN acceleration," in *Proc. IEEE Custom Integr. Circuits Conf. (CICC)*, Apr. 2021, pp. 1–2.

[16] Q. Dong et al., "15.3 A 351TOPS/W and 372.4 GOPS compute-in-memory SRAM macro in 7nm FinFET CMOS for machine-learning applications," in *IEEE Int. Solid-State Circuits Conf. (ISSCC) Dig. Tech. Papers*, Feb. 2020, pp. 242–244.

[17] I. Chakraborty, D. Roy, and K. Roy, "Technology aware training in memristive neuromorphic systems for nonideal synaptic crossbars," *IEEE Trans. Emerg. Topics Comput. Intell.*, vol. 2, no. 5, pp. 335–344, Oct. 2018.

[18] A. Biswas and A. P. Chandrakasan, "Conv-RAM: An energy-efficient SRAM with embedded convolution computation for low-power CNN-based machine learning applications," in *IEEE Int. Solid-State Circuits Conf. (ISSCC) Dig. Tech. Papers*, Feb. 2018, pp. 488–490.

[19] M. Kang, S. K. Gonugondla, A. Patil, and N. R. Shanbhag, "A multi-functional in-memory inference processor using a standard 6T SRAM array," *IEEE J. Solid-State Circuits*, vol. 53, no. 2, pp. 642–655, Feb. 2018.

[20] M. Kang, S. K. Gonugondla, and N. R. Shanbhag, "A 19.4 nJ/decision 364K decisions/s in-memory random forest classifier in 6T SRAM array," in *Proc. 43rd IEEE Eur. Solid State Circuits Conf. (ESSCIRC)*, Sep. 2017, pp. 263–266.

[21] S. K. Gonugondla, M. Kang, and N. Shanbhag, "A 42pJ/decision 3.12 TOPS/W robust in-memory machine learning classifier with on-chip training," in *IEEE Int. Solid-State Circuits Conf. (ISSCC) Dig. Tech. Papers*, Feb. 2018, pp. 490–492.

[22] M. Kang, M.-S. Keel, N. R. Shanbhag, S. Eilert, and K. Curewitz, "An energy-efficient VLSI architecture for pattern recognition via deep embedding of computation in SRAM," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, May 2014, pp. 8326–8330.

[23] C. S. Vaucher, I. Ferencic, M. Locher, S. Sedvallson, U. Voegeli, and Z. Wang, "A family of low-power truly modular programmable dividers in standard 0.35-μm CMOS technology," *IEEE J. Solid-State Circuits*, vol. 35, no. 7, pp. 1039–1045, Jul. 2000.

[24] J.-S. Seo et al., "Digital versus analog artificial intelligence accelerators: Advances, trends, and emerging designs," *IEEE Solid-State Circuits Mag.*, vol. 14, no. 3, pp. 65–79, Aug. 2022.

[25] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proc. IEEE*, vol. 86, no. 11, pp. 2278–2324, Nov. 1998.

[26] S. Shakibhamedan, N. Amirafshar, A. S. Baroughi, H. S. Shahhoseini, and N. TaheriNejad, "ACE-CNN: Approximate carry disregard multipliers for energy-efficient CNN-based image classification," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 71, no. 5, pp. 2280–2293, May 2024.

[27] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," 2014, *arXiv:1409.1556*.

[28] L. Deng, "The MNIST database of handwritten digit images for machine learning research," *IEEE Signal Process. Mag.*, vol. 29, no. 6, pp. 141–142, Nov. 2012.

[29] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "ImageNet: A large-scale hierarchical image database," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2009, pp. 248–255.

[30] Y. Chen, Y. Xie, L. Song, F. Chen, and T. Tang, "A survey of accelerator architectures for deep neural networks," *Engineering*, vol. 6, no. 3, pp. 264–274, Mar. 2020.

[31] X. Zhao, Y. Wang, X. Cai, C. Liu, and L. Zhang, "Linear symmetric quantization of neural networks for low-precision integer hardware," in *Proc. Int. Conf. Learn. Represent.*, Apr. 2020, pp. 1–16.

[32] S. K. Lee et al., "A 7-nm four-core mixed-precision AI chip with 26.2-TFLOPS hybrid-FP8 training, 104.9-TOPS INT4 inference, and workload-aware throttling," *IEEE J. Solid-State Circuits*, vol. 57, no. 1, pp. 182–197, Jan. 2022.

[33] X. Sun et al., "Ultra-low precision 4-bit training of deep neural networks," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 33, 2020, pp. 1796–1807.

[34] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," 2014, *arXiv:1412.6980*.

[35] L. van der Maaten and G. Hinton, "Viualizing data using t-SNE," *J. Mach. Learn. Res.*, vol. 9, pp. 2579–2605, Nov. 2008.

[36] M. Kang, S. Lim, S. Gonugondla, and N. R. Shanbhag, "An in-memory VLSI architecture for convolutional neural networks," *IEEE J. Emerg. Sel. Topics Circuits Syst.*, vol. 8, no. 3, pp. 494–505, Sep. 2018.

[37] A. Jaiswal, I. Chakraborty, A. Agrawal, and K. Roy, "8T SRAM cell as a multibit dot-product engine for beyond von Neumann computing," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 27, no. 11, pp. 2556–2567, Nov. 2019.

[38] S. Yin, Z. Jiang, J.-S. Seo, and M. Seok, "XNOR-SRAM: In-memory computing SRAM macro for binary/ternary deep neural networks," *IEEE J. Solid-State Circuits*, vol. 55, no. 6, pp. 1733–1743, Jun. 2020.

[39] P. N. Whatmough, S. K. Lee, H. Lee, S. Rama, D. Brooks, and G.-Y. Wei, "14.3 a 28nm SoC with a 1.2GHz 568nJ/prediction sparse deep-neural-network engine with >0.1 timing error rate tolerance for IoT applications," in *IEEE Int. Solid-State Circuits Conf. (ISSCC) Dig. Tech. Papers*, Feb. 2017, pp. 242–243.

[40] B. Moons and M. Verhelst, "A 0.3–2.6 TOPS/W precision-scalable processor for real-time large-scale ConvNets," in *Proc. IEEE Symp. VLSI Circuits (VLSI-Circuits)*, Jun. 2016, pp. 1–2.

[41] J. K. Kim, P. Knag, T. Chen, and Z. Zhang, "A 640M pixel/s 3.65 mW sparse event-driven neuromorphic object recognition processor with on-chip learning," in *Proc. Symp. VLSI Circuits (VLSI Circuits)*, Jun. 2015, pp. C50–C51.

[42] M. Abadi. (2015). *TensorFlow: Large-scale Machine Learning on Heterogeneous Systems*. [Online]. Available: Software available from tensorflow.org