IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received May 4, 2021, accepted May 6, 2021, date of publication May 12, 2021, date of current version May 19, 2021.

Digital Object Identifier 10.1109/ACCESS.2021.3079425

10T SRAM Computing-in-Memory Macros for Binary and Multibit MAC Operation of DNN Edge Processors

VAN TRUONG NGUYEN[1], JIE-SEOK KIM[2], AND JONG-WOOK LEE[1,2] (Senior Member, IEEE)

[1]Information and Communication System-on-chip (SoC) Research Center, Department of Electronics and Information Convergence Engineering, Kyung Hee University, Yongin 17104, South Korea
[2]Department of Electronics, School of Electronics and Information, Kyung Hee University, Yongin 17104, South Korea

Corresponding author: Jong-Wook Lee (jwlee@khu.ac.kr)

This work was supported in part by the Basic Science Research Program through the National Research Foundation of Korea under Grant 2021R1A2B5B01001475, and in part by National Research and Development Program through the National Research Foundation of Korea funded by the Ministry of Science and ICT under Grant 2020M3H2A1076786.

ABSTRACT Computing-in-memory (CIM) is a promising approach to reduce latency and improve the energy efficiency of the multiply-and-accumulate (MAC) operation under a memory wall constraint for artificial intelligence (AI) edge processors. This paper proposes an approach focusing on scalable CIM designs using a new ten-transistor (10T) static random access memory (SRAM) bit-cell. Using the proposed 10T SRAM bit-cell, we present two SRAM-based CIM (SRAM-CIM) macros supporting multibit and binary MAC operations. The first design achieves fully parallel computing and high throughput using 32 parallel binary MAC operations. Advanced circuit techniques such as an input-dependent dynamic reference generator and an input-boosted sense amplifier are presented. Fabricated in 28 nm CMOS process, this design achieves 409.6 GOPS throughput, 1001.7 TOPS/W energy efficiency, and a 169.9 TOPS/mm^2 throughput area efficiency. The proposed approach effectively solves previous problems such as writing disturb, throughput, and the power consumption of an analog to digital converter (ADC). The second design supports multibit MAC operation (4-b weight, 4-b input, and 8-b output) to increase the inference accuracy. We propose an architecture that divides 4-b weight and 4-b input multiplication to four 2-b multiplication in parallel, which increases the signal margin by 16× compared to conventional 4-b multiplication. Besides, the capacitive digital-to-analog converter (CDAC) area issue is effectively addressed using the intrinsic bit-line capacitance existing in the SRAM-CIM architecture. The proposed approach of realizing four 2-b parallel multiplication using the CDAC is successfully demonstrated with a modified LeNet-5 neural network. These results demonstrate that the proposed 10T bit-cell is promising for realizing robust and scalable SRAM-CIM designs, which is essential for realizing fully parallel edge computing.

INDEX TERMS Computing-in-memory, static random access memory, deep neural network, machine learning, edge processor.

I. INTRODUCTION

Deep neural networks (DNNs) have achieved breakthroughs in a wide variety of artificial intelligence (AI) and machine learning (ML) applications, including image classification [1], speech recognition [2], and facial recognition [3], [4]. Their application has penetrated other disciplines, such as biology, materials science, and physics [5]–[7]. The DNN promises significant benefits for "Internet of Things" (IoT) devices at the edge of the network or edge computing. There are many advantages of edge computing, such as privacy, security, efficiency, and scalability. However, bringing computing to the edge has some specific requirements; the DNN processors running the computing algorithms should be energy and area efficient to make them practical for battery-constrained IoT devices [8].

The DNN contains multiple layers of convolutional neural networks (CNNs) and fully connected networks (FCNs).

The associate editor coordinating the review of this manuscript and approving it for publication was Kathiravan Srinivasan.

In these networks, the computation workload is dominated by multiply and accumulate (MAC) operations. Approaches are proposed to address the high computation and storage burden by reducing weights and the precision of activations. However, the all-digital implementation of CNNs [9], [10] have shown that energy consumption and delays are dominated by the frequent movement of inputs, weights, and intermediate data between the processor and memory. This problem is referred to as the von Neumann bottleneck or memory wall [11]. Several innovative approaches have been presented to address this issue [12]–[15].

Recently, approaches based on computing-in-memory (CIM) have been reported to solve the memory wall problem [16]–[33]. The SRAM-based CIM (SRAM-CIM) structure allowing for the MAC operation within the memory provides two main benefits. First, storing a large amount of intermediate data is mitigated because the weights are not read from memory for every MAC operation. Second, the inherent parallel computing based on the memory structure improves energy efficiency and increases overall throughput.

Recent SRAM-CIM works implemented in silicon demonstrated strong potential for improved efficiency and throughput. Table 1 shows a summary of various SRAM-CIM bit-cell types. The work [16] implements CIM using a six-transistor (6T) bit-cell in a 130-nm CMOS and demonstrates a 113× improved energy efficiency over the conventional digital approach. A deep in-memory architecture (DIMA) [17] reports an architecture supporting 8-b weights and activation. The work named CONV-SRAM [18] uses analog domain computations row-wise using 16 local blocks simultaneously, and the analog-to-digital converters (ADCs) convert back MAC results to the digital value. This design reports 98% accuracy on the modified National Institute of Standards and Technology (MNIST) dataset with 40.3 TOPS/W energy efficiency. However, the previous approaches show design challenges and tradeoffs. Because the work [16] uses a pulse-amplitude modulated (PAM) word line (WL), the PAM driver's nonlinearity limits the inference accuracy to 90%. Because works [16], [17] are based on a 6T bit-cell, they can be susceptible to write disturb. The work [18] uses digital-to-analog converters (DACs) for pulse-width modulation, which requires many synchronous global timing signals for the DAC. Additionally, this work uses an integrating ADC that needs multiple cycles to complete the conversion. The work [19] uses twin-8T SRAM with different transistor sizing to store multibit weights. This design supports multiple inputs (1-b, 2-b, 4-b), multiple weights (1-b, 2-b, 5-b), and 7-b outputs. This work achieves up to 72 TOPS/W energy efficiency. The ADC weight processor on each column uses capacitors to combine the signals of multiple row bit-lines (RBLs) and generate a reference voltage, which reduces the array efficiency to 31.5%.

TABLE 1. Various SRAM bit-cell types.

	[16]	[17]	[19]	[20]	[22]	[23]	This work
Bit-cell	6T	6T	Twin 8T	10T	12T	8T1C	10T
Tech.	130 nm	65 nm	55 nm	45 nm	65 nm	65 nm	28 nm
Added circuit	No	No	One read port	Two read ports	Pull-up/down circuit	Two transistors one cap.	Two read ports
Write disturb	Yes	Yes	No	No	No	No	No
Area efficiency	High	High	Med.	Med.	Low	Low	Med.
TOPS /mm^2	-	-	557-1776	-	403	671.5	170
TOPS/W	11.5	1.94	18.4-72	19.9	5.5	20.2	1002

Some previous CIM designs aimed to binarize the weights and activations to reduce the complexity, power consumption, and hardware costs. This approach is suitable for applications requiring moderate accuracy. The work [21] uses a split word-line for compact 6T bit-cell SRAM to support binary MAC operation. This work performs 4096 operations per cycle and achieves a throughput of 278.2 GOPS; the 6T bit-cell is susceptible to the write disturb. This problem can be addressed by adding more transistors or capacitors into the bit-cell, such as Xcel-RAM [20], XNOR-SRAM [22], and C3SRAM [23]. The Xcel-RAM approach [20] uses 10T bit-cell and performs binary MAC operation between the weight and input that are stored in two different rows. Therefore, it requires two write operations in SRAM mode before the MAC operation. Besides, this structure processes a maximum of one binary MAC operation per section; this work achieves a relatively low throughput of 8.5 GOPS. The XNOR-SRAM approach [22] uses a 12T bit-cell and achieves 614 GOPS throughput. The C3SRAM [23] uses an 8T bit-cell with one capacitor and achieves a relatively high 1638 GOPS throughput. Because these works use flash ADC with multiple comparators and voltage references, they increase power consumption. Besides, capacitors are used inside the bit-cell, which increases the bit-cell area by 27% [23].

In this paper, we propose SRAM-CIM designs addressing the issues of the previous works. The proposed approach focuses on scalable CIM designs using a new 10T bit-cell, robust to the write disturb. We present two SRAM-CIM designs. The first design achieves fully parallel computing and high throughput using 32 parallel binary (1-b weight, 1-b input, and 1-b output) MAC operations. A dynamic reference generator and a sense amplifier (SA) with an input-boosting technique improve the accuracy. Implemented in 28 nm CMOS process, this design achieves 409.6 GOPS throughput, 1001.7 TOPS/W energy efficiency, and 169.9 TOPS/mm^2 throughput-area efficiency. The second design supports multibit (4-b weight, 4-b input, and 8-b output) MAC operation to increase the inference accuracy. We propose an architecture that divides 4-b weight and 4-b input multiplication to four 2-b multiplication in parallel. This approach increases the signal margin by 16× compared to conventional 4-b multiplication. We propose an area-efficient approach for realizing the capacitive DAC in a successive approximation register (SAR) ADC using intrinsic bit-line capacitances of the SRAM-CIM macro.

II. 10T SRAM-CIM FOR BINARY MAC

This section presents a 10T SRAM-CIM unit macro for binary MAC operation realized in the 28 nm CMOS process. Using the process scaling and full parallel computing supported by advanced circuit techniques, the proposed approach achieves significant improvement in energy efficiency (1001.7 TOPS/W) and throughput density (409.6 GOPS/Kb).

A. 10T BIT-CELL FOR BINARY MAC

Fig. 1 shows the schematic and layout of the proposed 10T SRAM bit-cell. The 10T cell consists of a standard 6T SRAM cell for the storage unit and four transistors (M0 - M3) for decoupled read ports. Two read ports of the RBL and RBLX are controlled by differential read word-lines (RWL and RWLX). Compared to [20], our approach is simple with the read ports connected to the ground; the bit-cell structure is suitable for scaling to a large macro size. The area ratio compared with the 6T bit-cell is 1.79. In the SRAM mode, the read and write operations are performed using a write word-line (WWL) and a bit-line pair (BL and BLX), similar to the conventional 6T SRAM bit-cell. In CIM mode, the RWL and RWLX in each row represent the activation input IN[i]. If an input value is IN[i] = '+1', the corresponding RWL[i] is asserted as '1' and RWLX[i] is asserted as '0', and vice versa for IN[i] = '−1'. When the bit-cell stores the weight '+1', the storage nodes have a logic level of Q = 1 and QX = 0, and vice versa for '−1'. Table 2 summarizes the input-weight-product (IWP) function of the 10T bit-cell. The read current I_{RC} on RBL and RBLX (See Fig. 6) represents the IWP between IN[i] and W[i]. If the IWP result is '+1', I_{RC} is generated in either RBL or RBLX. There is no I_{RC} for IWP = '−1'.

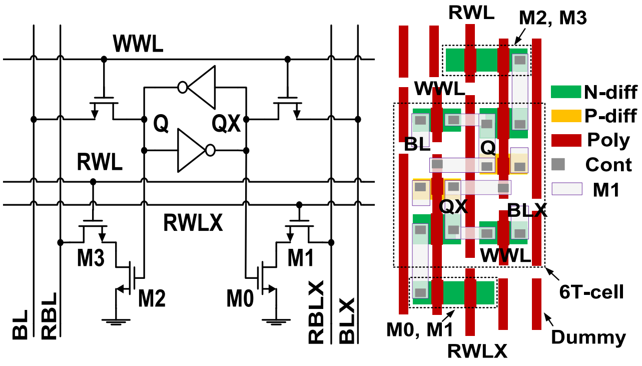

FIGURE 1. Schematic and layout of the proposed 10T SRAM bit-cell.

TABLE 2. Truth table of the proposed 10T SRAM bit-cell.

Input			Weight			IWP		
Value	RWL	RWLX	Value	Q	QX	Value	RBL	RBLX
-1	0	1	-1	0	1	1	0	I_{RC}
-1	0	1	1	1	0	-1	0	0
1	1	0	-1	0	1	-1	0	0
1	1	0	1	1	0	1	I_{RC}	0

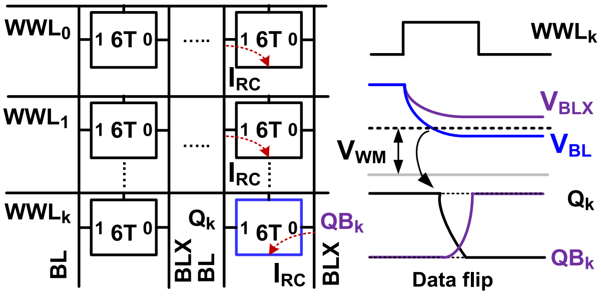

FIGURE 2. Write disturb issue in the conventional 6T bit-cell.

Fig. 2 shows the SRAM-CIM based on 6T bit-cell [18]. Foundry 6T-SRAM has the advantage of a compact layout; however, write disturb can occur [17], [21]. During the CIM operation, multiple WWLs are activated for the same bit-line. If the bit-line voltage (V_{BL} or V_{BLX}) of this column decreases below V_{WM}, there might be a pseudo-write operation in one of the accessed bit-cells, which overwrites the stored data. The write margin V_{WM} is defined as the bit-line value at the point where the stored data can flip, and V_{BL} should be limited during the CIM operation. Previously, techniques such as WL under-drive [16] and BL diode clamping [21] are used to mitigate the write disturb. The under-drive, limiting $V_{WL} < 0.4$ V, slows down the pass transistor operation and eventually impacts speed. Limiting the BL voltage can impact the signal-to-noise ratio by reducing the range for various MAC values [31].

Fig. 3 compares the signal margin V_{SM} of the proposed 10T and conventional 6T bit-cells. In the 6T cell, the signal margin is calculated using the V_{DD}, write margin V_{WM}, and the number of activated WWLs (or word-lines). The voltage of various MAC results on the BL is set from V_{WM} to V_{DD} to prevent the write disturb scenario. Monte Carlo simulations considering global and local variations are used to obtain V_{WM}. And the mean and the standard deviations are 362 and 19 mV. Using V_{WM} for the 6T cell, the signal margin is obtained as $V_{SM} = (V_{DD} - V_{WM})/(\text{number of activated WWL})$ [19], [34]. In the case of 10T bit-cell, V_{SM} is the smallest

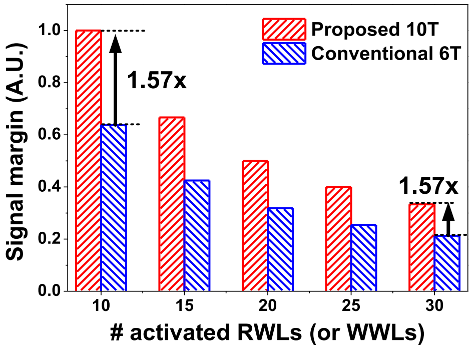

FIGURE 3. Comparison of the signal margin of the proposed 10T bit-cell and conventional 6T bit-cell.

difference between the two switching voltages on the RBL after multiplication. The MAC result on the RBL can vary from 0 to V_{DD} without write disturb [19]. Then, V_{SM} depends on the number of activated RWLs as $V_{SM} = V_{DD}/(\text{number of activated RWLs})$. The result shows that the proposed 10T bit-cell achieves a 1.57× higher signal margin than that of the conventional 6T bit-cell for the same number of activated word-lines.

Because DNNs are usually error-tolerant, it may be worth exploiting the advantage of 6T SRAM bit-cell for a compact layout. Work [16] presents an extensive study on handling errors of analog in-memory computing using 6T bit-cell. A strong classifier is built using the results from weak classifiers for boosting and weighted voting. The result shows that the extended AdaBoost can overcome the error, even from circuit non-idealities; however, the accuracy is achieved after the increased iterations. Because this approach uses a multitude (> 40) weak classifiers with an additional voting circuit, it can eventually consume more power, which is indicated by the moderate energy efficiency of 11.5 TOPS/W [16] and 1.94 11.5 TOPS/W [17]. We note that the proposed 10T bit-cell with the decoupled ports is robust to write disturb. This approach allows the enhanced signal margin and achieves an excellent energy efficiency of 170 TOPS/W (See Table 1).

B. MACRO ARCHITECTURE

Fig. 4 shows the block diagram of the proposed 10T SRAM-CIM unit-macro for binary MAC operation. The proposed unit-macro includes 32 rows × 32 columns of 10T bit-cells, a reference (REF) array block, a read word line (RWL) driver, a CIM input-output (CIM_IO) block, and a CIM control block (CIM_CTL). The RWL driver consists of 32 decoders and driver cells. The CIM_IO includes 32 evaluation (EVAL) cells, 32 sense amplifiers (SAs), a V_{REF} generator (VREF_GEN) cell, and a sense enable generator (SE_GEN) cell. The REF array block uses three columns of 10T bit-cells for generating the reference voltage V_{REF} and the sense enable signal SEN. All the I_{RC} of the RBL and RBLX in the same column are summed in the EVAL cell. The EVAL cell converts the summed current to an analog voltage. The SA cell compares this analog voltage with V_{REF} to produce a binary output.

The proposed 10T SRAM-CIM macro supports two modes: SRAM and CIM modes. The SRAM mode performs the write operation to store the trained weights into the array using the read/write control (RW_CTL) and read-write IO (RW_IO) blocks. A single active WWL accesses one row via the RW_IO block. The CIM mode is used for binary MAC operations. The proposed unit-macro supports 32 binary MAC operations in parallel for 32 activation inputs (IN) and 32 weights (W). The weights (W) are stored into a number m ($m = 0$ to 31) of 10T bit-cells in the same column. For MAC operation, multiple rows are activated simultaneously, and each IN[n] is fed through an RWL driver to activate the RWL_n and RWLX_n pair ($n = 0$ to 31). By activating multiple RWL pairs, the current sum of the n bit-cells is sensed at RBL and RBLX to determine the IWP = IN × W.

Fig. 5 shows the mapping of the inputs and weight for the proposed SRAM-CIM macro. Multiple weight channels are stored in multiple columns of the array. Each column of the 10T bit-cell is correlated with one weight channel. The inputs corresponding to the weight are applied to multiple rows by activating each row's RWL pair. The IWP is processed in each 10T bit-cell, and currents are accumulated in each column to generate the MAC output OUT[0:31]. The proposed unit macro supports a maximum of 32 MAC operations. High throughput is achieved by computing all

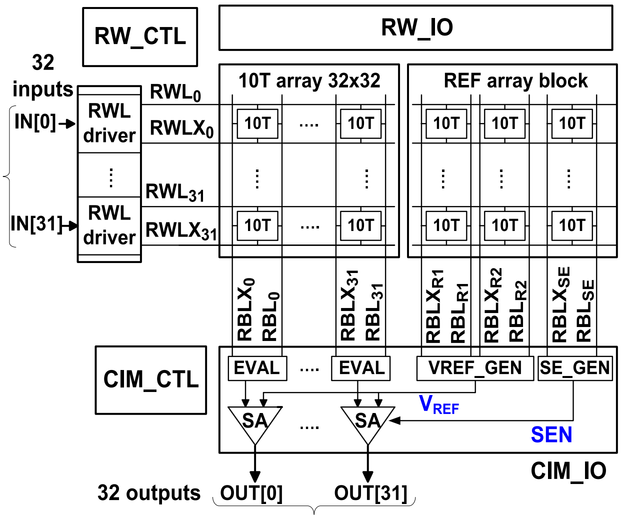

FIGURE 4. Overall architecture of the proposed SRAM-CIM for binary MAC operation.

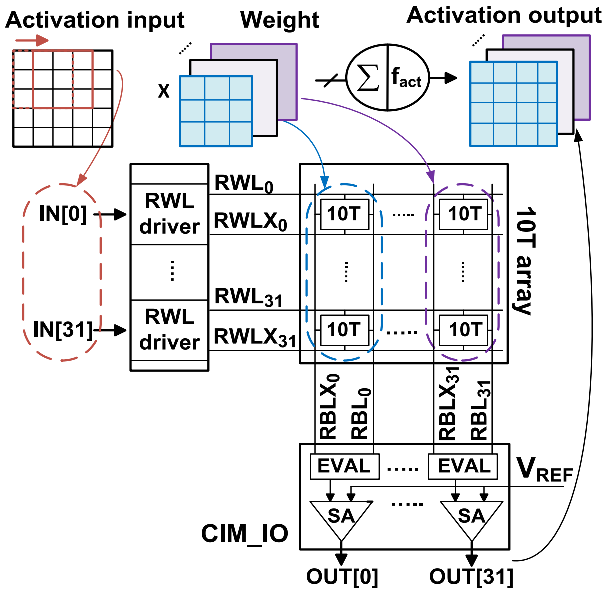

FIGURE 5. Mapping of activation inputs and weight using the proposed SRAM-CIM macro.

columns in parallel, and the proposed approach achieves a maximum of 1024 MAC operations per cycle.

C. DYNAMIC VREF GENERATOR AND EVALUATION TIME TRACKING

Fig. 6 shows the circuit-level binary MAC operation with related waveforms. The IWP of each 10T bit-cell results in I_{RC}, with the currents summed in the EVAL cell. All of the I_{RC} from the RBL and RBLX are summed by turning on the MP1 using EVAL_EN, which closes the current path. The RBL voltage $V_{RBL,IN}$ depends on the summed current by the voltage divider action provided by MP1. The $V_{RBL,IN}$ is compared with V_{REF} to generate the MAC output OUT[0:31]. The EVAL_EN signal is turned on in a short time ($\sim$ 2 nsec) during the evaluation period.

Fig. 7 shows the schematic of the REF array block. It consists of three reference columns (RC1,2,3) of the 10T bit-cell and a REFWL header. Using the REF array block, VREF_GEN provides dynamic V_{REF} generation, and SE_GEN provides evaluation-time tracking for the SA. The REFWL header block sets all the REFWLs to '1' (all REFWLXs to '0') using the tie-cell. The tie-cell contains four transistors generating tie-high (TH) = V_{DD} and tie-low (TL) = gnd signals.

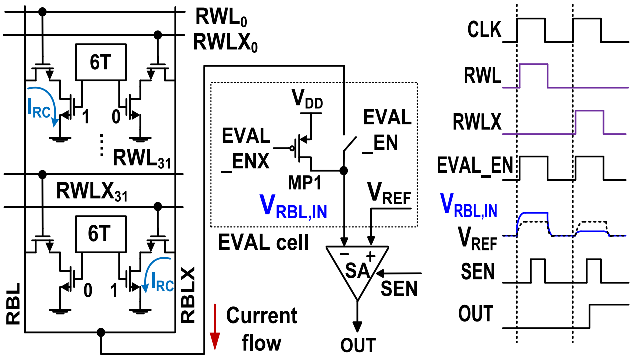

FIGURE 6. Binary MAC operation in each column with related waveforms.

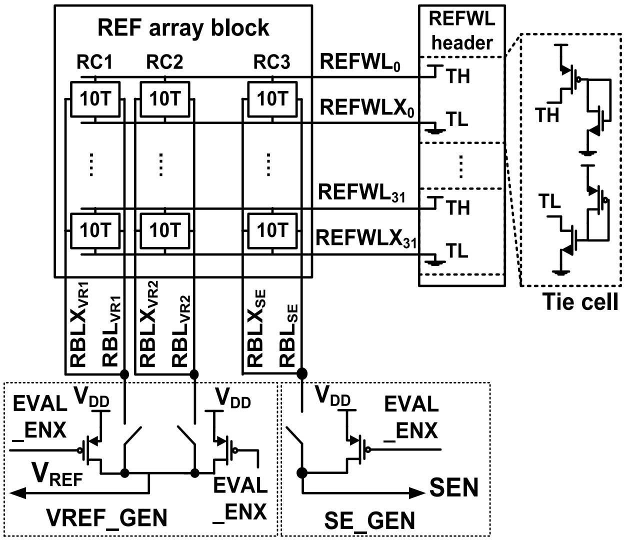

FIGURE 7. Schematic of the reference array block, dynamic V_{REF} generator, and sense enable signal generator.

Two reference columns, RC1,2 are used to generate V_{REF}, depending on the number of activation inputs. During the binary MAC operation, OUT is determined by the number N_{IWP+1} of IWP = '+ 1' and N_{IWP-1} of IWP = '− 1' bit-cells in the same column. When $N_{IWP+1} > N_{IWP-1}$, the MAC output is 1, and vice versa. Therefore, the N_{IWP+1} of the reference columns RC1,2 is set to half of the inputs so that the VREF_GEN cell generates appropriate V_{REF}. The number of bit-cells in RC1,2 storing data '1' and '0' depends on n. If n inputs is an even number, the n/2 bit-cells of RC1 and (n-2)/2 bit-cells of RC2 store data '1' and the remaining bit-cells store data '0'. If n is an odd number, the (n + 1)/2 bit-cells of RC1 and (n-1)/2 bit-cells of RC2 store data '1', and the remaining bit-cells store data '0'.

One reference column of RC3 is used as a replica of the loading in RBL/RBLX. Using the replica, the SE_GEN produces the SEN signal by tracking the worst-case loading of the column line. When EVAL_ENX is asserted, the RBLs/RBLXs are pre-charged to a voltage level corresponding to multiplying results on the columns, and the maximum output is V_{DD}; all bit-cells on the RC3 column are set to '0'. Then, RBL_{SE} and $RBLX_{SE}$ are charged from 0 to V_{DD} when EVAL_EN is asserted to start the evaluation phase (See Fig. 6). When generating the SEN signal for the SA, this setting represents the worst-case rise time in RBL_{SE}/$RBLX_{SE}$. This approach ensures the proper evaluation timing tracking for the SA to evaluate the final output.

Fig. 8 shows the simulated RBL/RBLX voltage as a function of the number of activated inputs. The result includes the RBL/RBLX voltages for the OUT = 0 and 1 states. Here, V_{OUT0} is the lowest $V_{RBL,IN}$ value when the MAC output is 0, and the V_{OUT1} is the highest $V_{RBL,IN}$ when the output is 1. We note that the generated V_{REF} closely tracks both V_{OUT0} and V_{OUT1} for a different number of activated inputs, indicating the effectiveness of the dynamic V_{REF} generator.

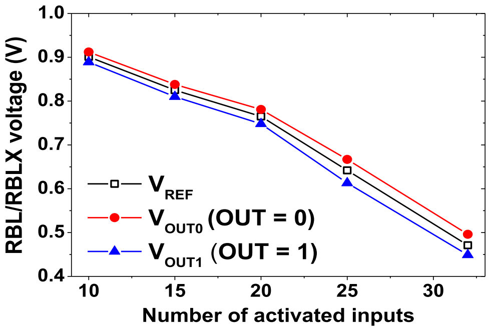

FIGURE 8. A plot of V_{REF} and RBL/RBLX levels as a function of the number of activated inputs.

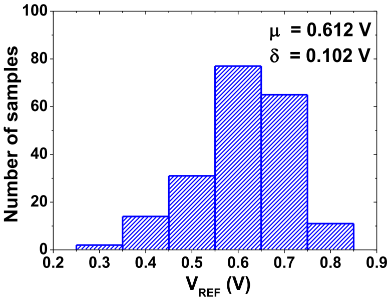

FIGURE 9. Statistical distribution of V_{REF} for the case of 25 activated inputs obtained using 200 Monte Carlo simulations, with V_{DD} = 1 V.

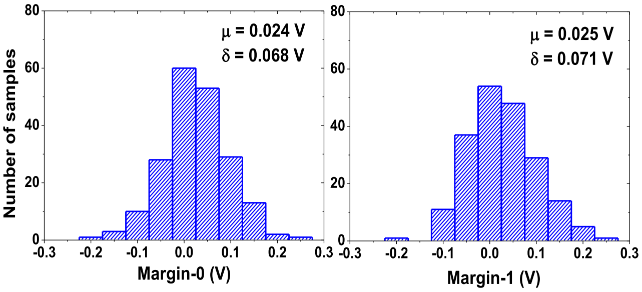

FIGURE 10. Statistical distribution of the margin.

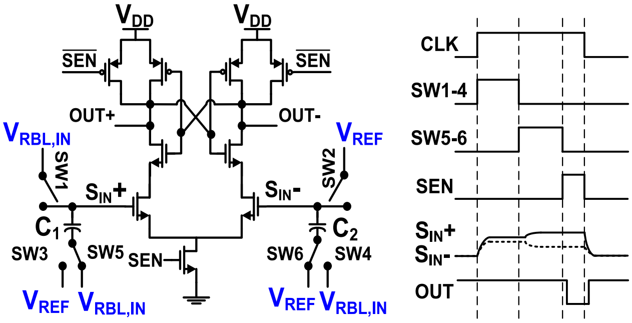

FIGURE 11. Schematic of the proposed SA with associated waveforms.

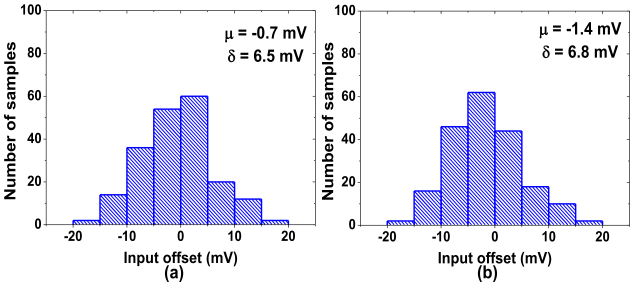

FIGURE 12. Statistical distribution of the Input offset of the proposed SA with (a) V_{CM} = 0.6V, (b) V_{CM} = 0.7 V.

Fig. 9 shows the statistical distribution of V_{REF} for 25 activated inputs. The mean value is 0.612 V with a standard deviation of 0.102 V (~16%). Fig. 10 shows the distribution of the Margin-0 (the difference between V_{OUT0} and V_{REF}) and Margin-1 (the difference between V_{REF} and the V_{OUT1}) for the case of 25 activated inputs. When we consider the mean value, the results show that both Margin-0 (24 mV) and Margin-1 (25 mV) are greater than the offset spread (~6 mV) of the SA. The results indicate that the proposed VREF_GEN produces appropriate V_{REF} in the middle of V_{OUT0} and V_{OUT1} by closely tracking the output of the computing columns.

The proposed structure is scalable to support a large capacity SRAM-CIM. When the number of columns increases, we can reuse the periphery circuits without modification, which keeps the signal margin unchanged. When the number of rows increases, the proposed VREF_GEN generates appropriate V_{REF} by tracking the number of rows in the computing columns. Because the number of rows affects the signal margin, the upper limit for the scaling can be set by considering the tradeoff between the throughput and the signal margin.

D. SENSE AMPLIFIER USING INPUT BOOSTING

Fig. 11 shows the schematic of the SA. The SA is based on a Strong-Arm latch comparator and the input boosting circuit (IBC). The IBC realized using two capacitors and six switches, similar to the one used in ternary content-addressable memory (TCAM) [35]; the SA's input is determined by the MAC operation in the SRAM-CIM while it is determined by the number of the mismatched cells in the TCAM.

The SA operates in three phases. In the first phase, SW1-4 are on, and SW5-6 are off, which sets $V_{RBL,IN}$ connected to the positive input $S_{IN}+$ (V_{REF} connected to S_{IN}-). The bottom node of C_1 and C_2 are connected to V_{REF} and $V_{RBL,IN}$, respectively. In the second phase, SW1-4 turns off, and SW5-6 turns on, which sets the $S_{IN}+$ and S_{IN}- nodes floating. The bottom of C_1 changes from V_{REF} to $V_{RBL,IN}$ to generate a coupling voltage across C_1 such that $S_{IN}+$ is increased to ($2V_{RBL,IN} - V_{REF}$). Similarly, S_{IN}- changes to ($2V_{REF} - V_{RBL,IN}$). Then, the input difference of the SA is boosted to $3(V_{RBL,IN}- V_{REF})$. In the third phase, the SEN signal is enabled to generate the output. The IBC allows tolerating three times smaller input offset than conventional SA. Fig. 12 shows the distribution of the input offset voltages of the proposed SA using the common-mode (CM) voltage V_{CM} = 0.6 V and 0.7 V. The 200 Monte Carlo simulations are performed at V_{DD} = 1 V and room temperature. In both cases, the sum of the mean and the standard deviation is less than 8.5 mV.

Fig. 13 shows the comparison of the input offset (or the smallest differential input) as a function of V_{CM}. The result shows that the one sigma input offset of the proposed SA is 2.5× smaller than that of a conventional latch-type SA. Fig. 14 shows a comparison of the sensing delay between proposed and conventional SA as a function of V_{CM}. Using the IBC, the proposed SA operates with reduced delay across a wide range of V_{CM}. The sensing delay of the proposed SA is 1.64× smaller than that of conventional SA at V_{CM} = 0.4 V.

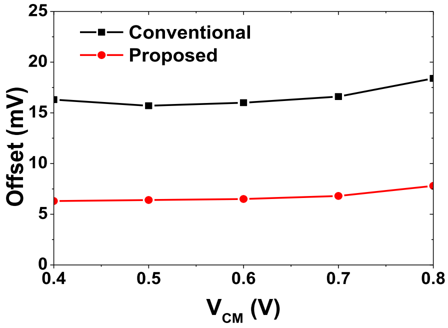

FIGURE 13. Comparison of the input offset of the proposed and conventional SA.

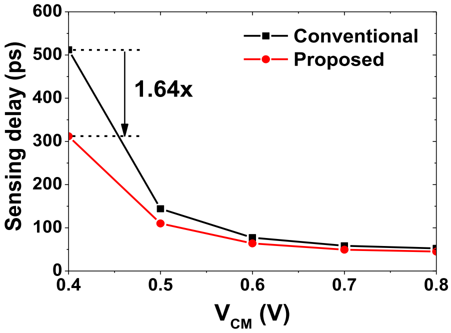

FIGURE 14. Comparison of the sensing delay as a function of the input common-mode voltage.

E. MEASUREMENT RESULTS

The SRAM-CIM unit-macro with a 32×32 array is fabricated using a 28 nm CMOS process. Fig. 15 shows the microphotograph with a core area of 37 × 64 μm^2. The area breakdown shows that the array efficiency is 34.2%, which can be further increased using a larger macro size. The power breakdown shows that the CIM_IO consumes 63.6% of the power with the current switching of EVAL, VREF_GEN, SE_GEN cells. The remaining blocks dissipate power during the input and control signal switching. Table 3 shows the summary of the proposed SRAM-CIM unit-macro.

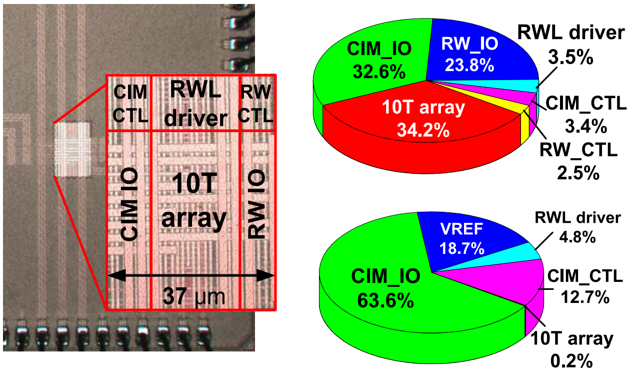

FIGURE 15. Microphotograph of the fabricated 10T SRAM-CIM unit macro. The area and power breakdowns are also shown.

TABLE 3. Chip summary of the SRAM-CIM for binary MAC.

Technology	28 nm CMOS
Unit-macro size	1 Kb
Bit-cell type	10T
Bit-cell size	0.39 µm × 1.56 µm
Input, weight, output (bit)	1, 1, 1
Cycle time (ns)	5
Maximum operation per cycle	2048

Fig. 16 shows the test setup for the SRAM-CIM unit macro. The computer sends the input images to the field-programmable gate array (FPGA), and the FPGA performs inference tasks with the SRAM-CIM. The inference result is collected through the USB port, and the FPGA interfaces with the SRAM-CIM by general-purpose input/output (GPIO) and level-shifters. A modified LeNet-5 neural network model [36] is implemented for inference of the MNIST dataset. The network contains two convolution layers (CONV1, CONV3), two max-pooling layers (POOL2, POOL4), and three fully connected layers (FL5, FL6, FL7). There are additional implemented layers for batch normalization and nonlinear binary activation. Table 4 shows the parameter mapping of CONV1 and FL7 layers for the SRAM-CIM. The weight size of the CONV1 is 5 × 5, with one input channel and eight output channels. Each of the eight output channels is mapped to one array column. In every cycle, 25 (=5 × 5) activation inputs (IN[i]) are sent through the buffer to 25 rows of the SRAM-CIM array, and eight activation outputs are computed in parallel. Therefore, the SRAM-CIM processes 25 × 8×2 operations per clock cycle. For the FL7 layer, 32 output channels are mapped to 32 columns, and

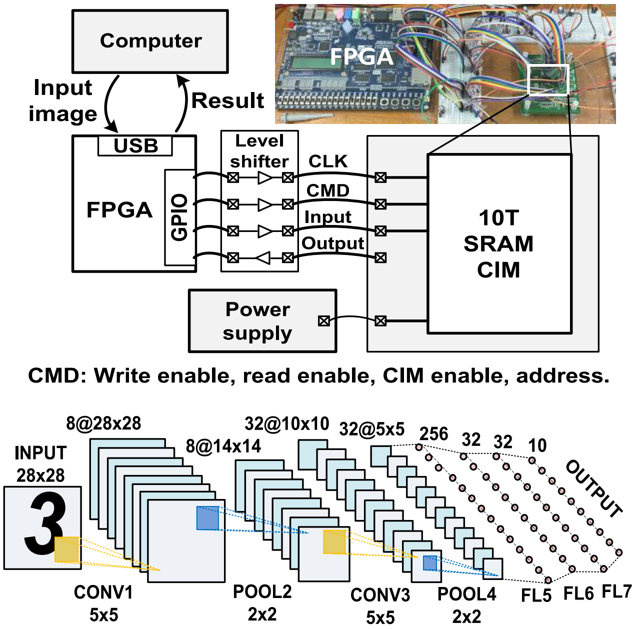

FIGURE 16. Test setup for the 10T SRAM-CIM unit macro. The modified LeNet-5 structure is also shown.

TABLE 4. Parameter mapping of the layer for the SRAM-CIM.

Parameters	CONV1	FL7
Weight size	1× 5 × 5 × 8*	32 × 32**
# Channels	8	32
# Array rows	25	32
# Array columns	8	32
# Operation/cycle***	25 × 8 × 2	32 × 32 × 2

* CONV weight size = (input channel × height × width × output channel).
** FL weight size = (input channel × output channel).
*** 1 MAC = 2OPs (1 multiply + 1 add).

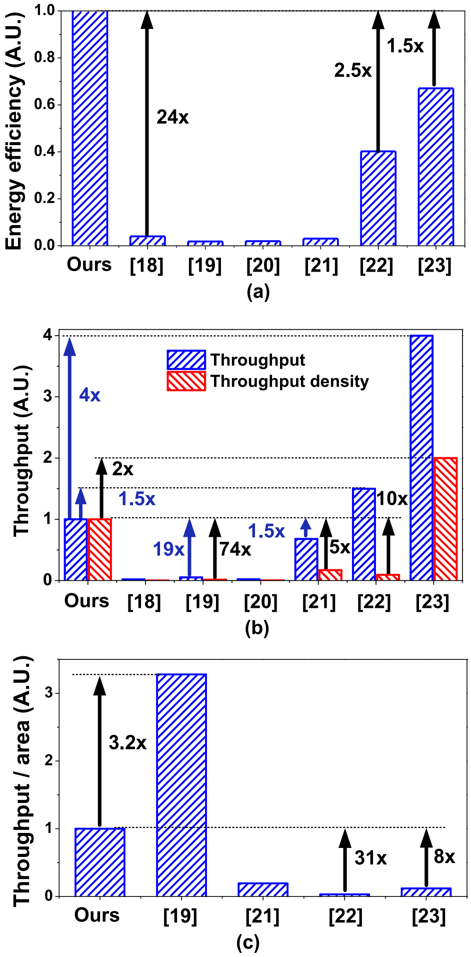

FIGURE 17. Comparison of (a) energy efficiency, (b) throughput, (c) area efficiency with other works.

the 32 activation inputs are sent to the array every cycle to process the computation, resulting in 32 × 32×2 operations per cycle. The remaining layers (CONV2, FL5,6) can be implemented by additionally loading to the proposed SRAM-CIM one after the other; all the layers in the network need to be accelerated to avoid the execution bottleneck. The control, nonlinear layers (pooling layers, activation, and batch-norm layer), and channel accumulation operations are implemented in the FPGA. The data are transferred back and forth between the PC and the test chip. An inference accuracy of 96.5% is achieved using the implementation. The accuracy loss compared to the simulation is within 2%. If the network size exceeds the fabricated SRAM-CIM's capacity or the layer size is larger than 32 × 32, multiple re-loading of inputs and weights are needed to emulate the scenario of using multiple macros [28]. In this case, limiting the output to 1-bit can lead to a relatively large loss of accuracy [21].

Fig. 17(a) shows the energy efficiency comparison. Our work achieves excellent energy efficiency compared to other works. For example, our work achieves 24×, 2.5×, and 1.5× higher energy efficiencies than [18], [22], and [23], respectively. Fig. 17(b) shows the throughput compared with other works. Our work achieves a relatively high throughput compared to [18]–[21], for example, more than 19× higher than those of [18]–[20]. When we consider throughput per macro size (GOPS/Kb) or throughput density, ours is higher than other studies, except [23]. Our work achieves 74× and 5× improved throughput density than [19] and [21], respectively. Compared to work [22], our work achieves 10× higher throughput density and 1.5× lower throughput. We note that [22] and [23] use additional transistors (12T) and a capacitor (8T1C) for the bit-cell; our work achieves 31× and 8× higher throughput per area (or area efficiency in TOPS/mm^2), respectively, as shown in Fig. 17(c). Additionally, [22] uses flash ADC with large power consumption and requires more than ten external voltage references. On the other hand, work [19] achieves 3.2× higher area efficiency than ours; this is achieved using about a four-times larger array size than ours.

Table 5 shows a comparison of the proposed architecture with previous SRAM-CIM works. Compared to [18], [19], which uses multibit MAC operation, the inference accuracy for the MNIST dataset is 2-3% lower; however, the throughput and energy efficiency are 6× and 14× higher, respectively. Our work achieves a similar inference accuracy of 96.5% with the work [21] using 1-b precision; our work achieves an improvement of 1.5× in throughput, 18× in energy efficiency, and 5× in area efficiency compared to [21]. Compared to Xcell-RAM [20], which supports up to 5-b output precision, our work achieves 48× and 8× higher throughput and operations per cycle, respectively. In summary, our work achieves the highest energy efficiency among other works. Throughput per area is higher than other works, except [19]. Throughput per macro size or throughput density (GOPS/Kb) is higher than other works, except [23].

To investigate whether the benefits of our work is derived from the process advancement (28 nm) or the circuit techniques, we use a technology scaling factor S_{tech} [37], which is 2.32, 1.96, and 1.61 for 65 nm, 55 nm, and 45 nm CMOS; we assume that other designs are also implemented in 28 nm technology and calculate the performance metric as follows: throughput $\propto (S_{\text{tech}})$, energy efficiency $\propto (S_{\text{tech}})^2$, and throughput/area $\propto (S_{\text{tech}})^2$. Compared to [18], [19], the energy efficiency of our work is still 4.6× and 3.6× higher, respectively. Compared to [20], our work achieves 30× and 19× higher throughput and energy efficiency, respectively.

TABLE 5. Comparison with similar studies.

	[18]	[19]	[20]	[21]	[22]	[23]	This work
Technology	65 nm	55 nm	45 nm	65 nm	65 nm	65 nm	28 nm
Macro size	16 Kb	3.84 Kb	8 Kb	4 Kb	16 Kb	2 Kb	1 Kb
Bit-cell	10T	T8T	10T	Split-WL 6T	12T	8T1C	10T
Input (bit)	6	1/2/4	1	1	1	1	1
Weight (bit)	1	2/5	1	1	1	1	1
Output (bit)	6	3/5/7	5	1	3.5	5	1
Operating clock	5 MHz	98-312 MHz	22.2 MHz	-	-	50 MHz	200 MHz
Operation /cycle	1600	216/ 512	256	4096	128	32768	2048
Accuracy (MNIST)	98.3%	99.5%	-	96.5%	98.8%	98.3%	96.5%**
Throughput (GOPS)	8 (18.5)*	21.2-67.5 (41-132)*	8.5 (13.7)*	278 (645)*	614 (1425)*	1638 (3803)*	409.6
Throughput density (GOPS/Kb)	0.5 (1.2)*	5.5-17.5 (11-34)*	1.1 (1.8)*	69.5 (161)*	38.4 (89)*	819 (1901)*	409.6
Energy eff. (TOPS/W)	40.3 (217)*	18.4-72 (71-277)*	19.9 (51.4)*	55.8 (301)*	403 (2172)*	671.5 (3619)*	1002
Throughput/area (TOPS/mm^2)	-	1776 (6852)*	-	33.1 (178)*	5.5 (29.6)*	20.2 (109)*	170

* value when technology scaling factor is used.
** result when CONV1 and FL7 layers are implemented in the SRAM-CIM.

Compared to [21], our work achieves an improvement of 2.5× in throughput density and 3.3× in energy efficiency. Compared to [22], our design obtains 4.6× improved throughput density and 2× lower energy efficiency. Our design has lower throughput and energy efficiency than [23]; however, ours achieves a 1.6× gain in throughput/area.

III. 10T SRAM-CIM FOR MULTIBIT MAC

This section presents a 10T SRAM-CIM unit macro for multibit MAC operation. The result provides proof-of-concept validation of two parallel MAC operations in a 180 nm CMOS process.

A. 10T BIT-CELL FOR MULTIBIT MAC

Fig. 18 shows the schematic and layout of the 10T SRAM bit-cell. The two identical cells of M10T and L10T form a pair. The M10T and L10T share the same RBL_M and RBL_L, but use separate RWL_M (for M10T) and RWL_L (for L10T). Each 10T-SRAM cell consists of a 6T-cell as the storage unit and four transistors (N0-N3) for the decoupled read port. The transistors N0-N1 and N2-N3 form two differential read ports on RBL_M and RBL_L, respectively. The studies [19], [20] use additional read ports, as does ours; the connection and the usage of them are different from ours. We note two advantages of the proposed 10T bit-cell: 1) it increases the swing on the RBL_M and RBL_L without the write disturb, which can occur in the CIM based on 6T bit-cells, 2) the 10T pair cell processes two IWPs in parallel on the two bit-lines of RBL_M and RBL_L. The area ratio of the 10T pair compared with two 6T bit-cells is 1.52.

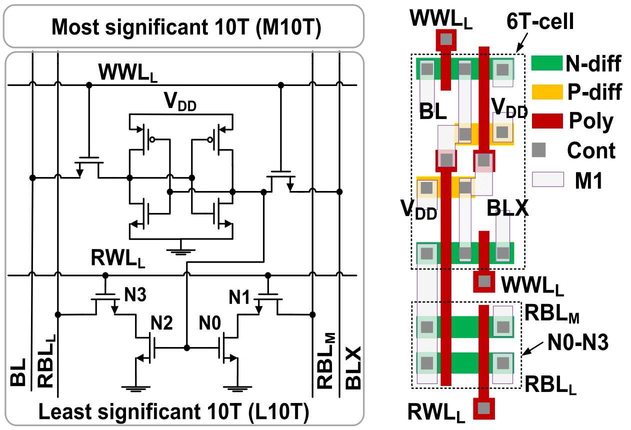

FIGURE 18. Schematic and layout (L10T) of the 10T SRAM bit-cell for multibit MAC.

The SRAM-CIM unit-macro supports two modes: SRAM and CIM modes. In the SRAM mode, the 10T cell works in a functionally similar manner to the standard 6T cell using the bit-line pair (BL, BLX) and a write word-line pair (WWL_M for M10T and WWL_L for L10T). The stored weights are accessed using a standard read/write peripheral circuit via a single active WWL. Two cycles are needed to access the M10T and L10T. Each 4-b kernel is stored in four array rows (See Fig. 22), which needs four cycles to write one kernel into the array. In each cycle, the write operation starts by applying the kernel to the RW_IO block. Then, they are converted to the bit-line level through the write driver. The WWL on the selected row turns on to write the weight into the bit-cell. In the CIM mode, RBL_M and RBL_L are pre-charged to an analog level corresponding to the digital inputs IN[3:2] and IN[1:0], respectively. Then, RWL_M and RWL_L are asserted by the different numbers of the pulse. One pulse for RWL_L and two pulses for RWL_M represent multibit weights. The generated current on each read bit-line (RBL_M and RBL_L) is proportional to the 2-bit weight value (W[1:0]). The most-significant-bit (MSB) and least-significant-bit (LSB) of W[1:0] are stored in M10T and L10T, respectively. Then, the output voltage on each RBL_M and RBL_L represents the IWP between the 2-b input and 2-b weight as $IWP_{RBLM} = IN[3{:}2] \times W[1{:}0]$ and $IWP_{RBLL} = IN[1{:}0] \times W[1{:}0]$.

B. MACRO ARCHITECTURE

Fig. 19(a) shows the block diagram of the proposed SRAM-CIM unit-macro. It consists of 128 rows × 16 columns of 10T SRAM bit-cells separated into two MAC blocks (MACB# 0 and MACB# 1) to support two parallel multibit MAC operations. The unit-macro includes 16 column-wise digital-analog converters (DACs) in the CIM_IO block, an RWL pulse generator, two ADC blocks (ADCBs), and weight combined blocks (WCBs). There are some periphery blocks to support the SRAM mode, such as a row decoder, RWL driver, and RW_IO block. The RW_IO includes sense amplifiers and

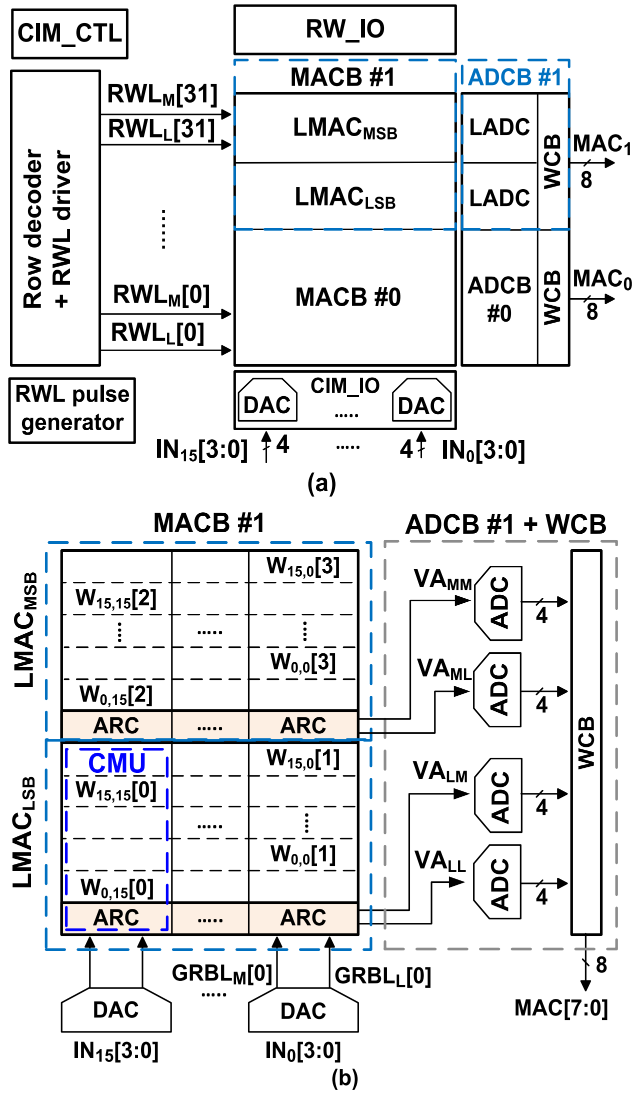

FIGURE 19. Block diagram of the proposed SRAM-CIM unit-macro for multibit MAC operation, (b) schematic of the MACB, ADCB, and WCB.

write drivers for the read and write operations. Fig. 19(b) shows a detailed schematic of the MACB. It is divided into two local MAC blocks (LMAC) storing 4-b weights $W_{k,j}[3:0]$ ($W_{k,j}[3:2]$ stored in $LMAC_{MSB}$ and $W_{k,j}[1:0]$ stored in $LMAC_{LSB}$). Each LMAC consists of 16 column-wise multiplication units (CMUs). Each CMU has sixteen 10T pair bit-cells with one accumulation and reference cell (ARC). The ADCB includes four ADCs divided into two local ADC blocks (LADC), which simultaneously convert four analog values VAs to four 4-b digital values. The four VAs include VA_{LL}, VA_{LM}, VA_{ML}, and VA_{MM}. The WCB consists of two 6-b and one 8-b ripple carry adders, and it combines four 4-b units of data from ADCs and generates an 8-b output.

In CIM mode, the MAC operation consists of four phases.

1) The first phase: sixteen 4-b inputs (IN<15:0>[3:0]) are fed into 16 column-wise DACs, which convert the 4-b digital code to the analog voltages on two global read bit-lines ($GRBL_M$ and $GRBL_L$). The analog levels on $GRBL_M$ and $GRBL_L$ represent 2-b of the MSB (IN[3:2]) and 2-b of the LSB (IN[1:0]), respectively. The $GRBL_M$ and $GRBL_L$ are shared by two RBL_M and RBL_L of all the LMAC arrays.

2) The second phase: each CMU computes two IWPs and generates voltages on RBL_M and RBL_L. Two outputs from the sixteen CMUs are accumulated in the ARC blocks, which generate two VAs in each LMAC block.

3) The third phase: VAs are converted to 4-b data by the ADCB. The conventional method converts the 4-b input to an analog voltage and then performs multiplication with 4-b weights stored in the array. Instead, our approach first divides it into four 2-b multiplications. The ADCB converts four multiplied outputs to four 4-b outputs. Then, the WCB generates the final 8-b output by combining four 4-b elements of data. The four VAs of kth kernel operation represent the product as

$$\begin{aligned} VA_{LL} &= \sum_{i=0}^{15}(IN_i[1:0]W_{k,i}[1:0]),\\ VA_{LM} &= \sum_{i=0}^{15}(IN_i[3:2]W_{k,i}[1:0]),\\ VA_{ML} &= \sum_{i=0}^{15}(IN_i[0:1]W_{k,i}[3:2]),\\ VA_{MM} &= \sum_{i=0}^{15}(IN_i[3:2]W_{k,i}[3:2]). \end{aligned} \tag{1}$$

This operation is performed in parallel for all LMAC blocks, converting the analog outputs VAs (from ARC blocks) to the corresponding four 4-b outputs.

4) The fourth phase: the WCB combines four 4-b elements of data from ADCB and generates the 8-b output as

$$\begin{aligned} &MAC[7:0]\\ &= MAC_{LL}[3:0] + MAC_{LM}[3:0]\times 2^2\\ &\quad + MAC_{ML}[3:0]\times 2^2 + MAC_{MM}[3:0]\times 2^4\\ &= \sum_{i=0}^{15}(IN_i[1:0]W_{k,i}[1:0]) + \sum_{i=0}^{15}(IN_i[3:2]W_{k,i}[1:0])2^2\\ &\quad + \sum_{i=0}^{15}(IN_i[1:0]W_{k,i}[3:2])2^2\\ &\quad + \sum_{i=0}^{15}(IN_i[3:2]\ W_{k,i}[3:2])2^4\\ &= \sum_{i=0}^{15}(IN_i[3:0]W_{k,i}[3:0]). \end{aligned} \tag{2}$$

C. CURRENT MODE DAC

Fig. 20 shows the DAC in the CIM_IO block, with the related waveforms. The CIM_IO includes sixteen current-steering DACs. Each DAC consists of two groups of binary-weighted current sources using a cascode PMOS stack biased in the

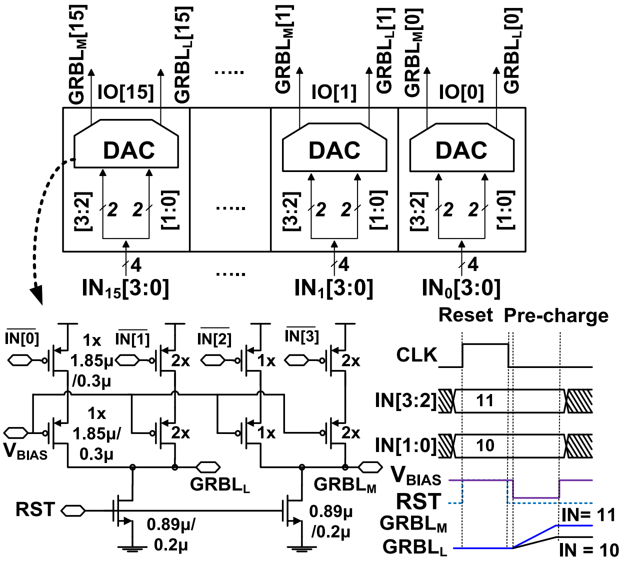

FIGURE 20. Schematic of the CIM_IO block and current-steering DAC, with related waveforms.

saturation region. The V_{BIAS} controls the pre-charged pulse width, and the RST signal resets the previous state of the $\mathrm{GRBL_M}$ and $\mathrm{GRBL_L}$. During the reset phase, the 4-b inputs ($\mathrm{IN_i}$[3:0]) of each column are split into two groups $\mathrm{IN_i}$[3:2] and $\mathrm{IN_i}$[1:0] and fed into the DAC. After the pre-charge phase, the DAC converts $\mathrm{IN_i}$[3:2] and $\mathrm{IN_i}$[1:0] values to the analog voltage of $\mathrm{GRBL_M}$ and $\mathrm{GRBL_L}$, respectively.

D. MULTIBIT PARALLEL COMPUTING SCHEME

Fig. 21 shows the multibit parallel computing scheme. The waveform for multiplication and accumulation in an LMAC block is also shown. The proposed approach supports four multiplications between the 2-b input and 2-b weight stored in the array. The pre-charging signal (PRE) in the ARC block is turned on during the first phase (DAC phase). Then, the $\mathrm{RBL_M}$ and $\mathrm{RBL_L}$ are pre-charged to the voltage levels corresponding to $\mathrm{GRBL_M}$ and $\mathrm{GRBL_L}$, respectively, representing 4-b input values. The second phase starts by asserting one pulse for the $\mathrm{RWL_L}$ and two pulses for the $\mathrm{RWL_M}$. The number of RWL pulses realizes the multibit weights [26]. These pulses activated on the same pair row in the $\mathrm{LMAC_{MSB}}$ and $\mathrm{LMAC_{LSB}}$ blocks are generated from the RWL pulse generator in the CIM_CTL. The input address ADR[3:0] for the selected pair row is driven to the ADR latch and the pre-decoder circuit. They are fully decoded by a row decoder in each paired row, which generates the WL_EN signal. After multiplying the 2-b input and 2-b weight on each $\mathrm{RBL_M}$ and $\mathrm{RBL_L}$, the average signal (AVG) is turned on to accumulate all the $\mathrm{RBL_M}$ and $\mathrm{RBL_L}$ values on 16 columns, which generates the four VA values.

Fig. 22 shows the data mapping of the kernels and feature inputs into the MACB. The proposed SRAM-CIM stores multiple 4-b kernels ($\mathrm{W_{k,j}}$[3:0]) in the MACB. Two 10T bit-cells are combined to form a bit-cell pair to store the 2-b weight. Each LMAC contains 16 bit-cell pair rows to store sixteen 2-b kernels. Each 4-b kernel is separated into MSB ($\mathrm{W_{k,j}}$[3:2]) and LSB fragments ($\mathrm{W_{k,j}}$[1:0]), which are mapped to a cell-pair row in the $\mathrm{LMAC_{MSB}}$ and $\mathrm{LMAC_{LSB}}$, respectively. The unit-macro supports a maximum of $k = 16$ kernels, $i = 16$ inputs, and $j = 16$ kernel element values.

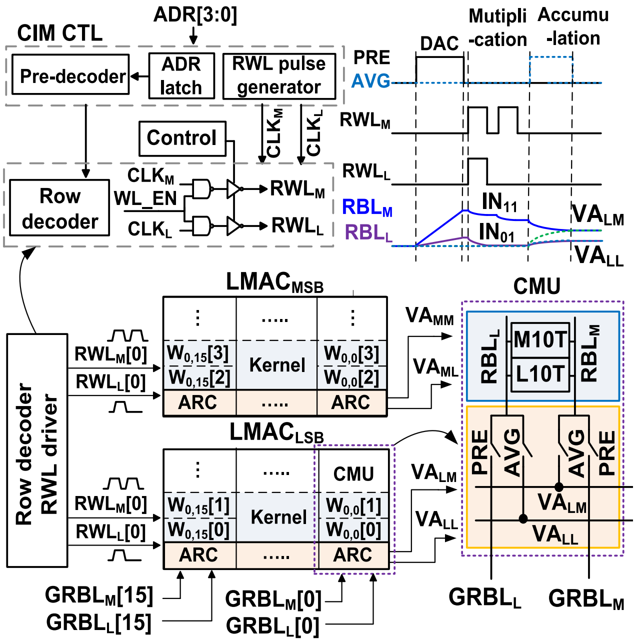

FIGURE 21. Multibit parallel computing scheme with the waveforms of multiplication and accumulation operation in an LMAC.

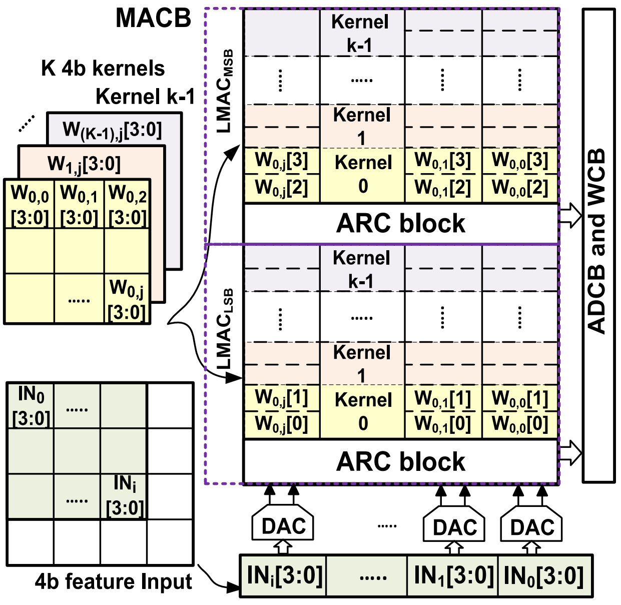

FIGURE 22. Schematic showing the mapping of the feature inputs and kernels into the MACB.

E. PERFORMANCE ANALYSIS

Fig. 23 presents a signal margin comparison between 4-b multiplication and 2-b multiplication. The conventional approach of computing the 4-b input and the 4-b weight by

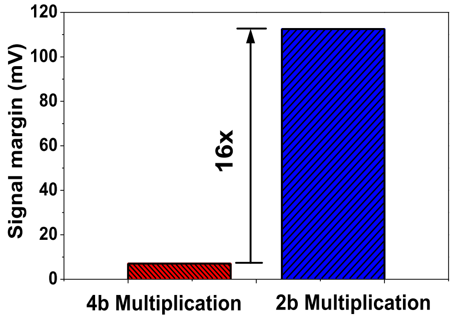

FIGURE 23. Signal margin comparison between the conventional 4-b and proposed 2-b multiplication approaches.

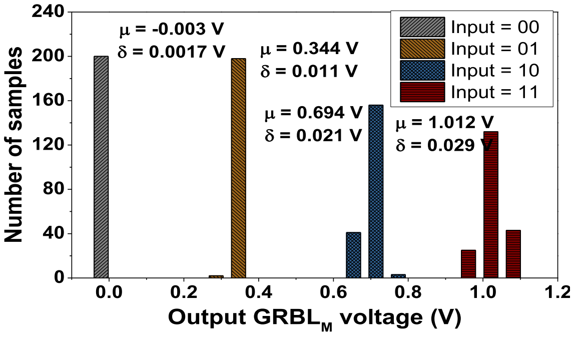

FIGURE 24. Statistical distribution of the DAC outputs obtained using 200 Monte Carlo simulations.

a 4-b multiplication has 256-b levels with a 7 mV unit step (V_{DD} = 1.8 V). The proposed approach divides it into four 2-b multiplications having 16 levels with a 112.5 mV unit step. The simulation result shows that this approach improves the signal margin by 16 times. Fig. 24 shows the statistical distribution of the DAC outputs of GRBL$_M$. The result is similar for GRBL$_L$under the same layout symmetry. The pulse width of the pre-charged signal is controlled by the reference column containing a replica DAC and an array column to achieve the matched loading for GRBL$_M$ and GRBL$_L$. This technique mitigates the effect of the process variations on the PMOS stack and output loading. Simulations show that both channels of the DAC achieve differential nonlinearity (DNL) < 0.5 LSB under three (FF, TT, SS) process corners.

Fig. 25 shows the statistical distribution of the multiplication between the 2-b input and 2-b weight in the MSB channel of the CMU. The result is obtained by feeding the 4-b input into each column. When RWL$_M$ and RWL$_L$ turn on, the multiplied result of W[1:0] and IN[3:2] (IN[1:0]) are generated on the RBL$_M$ (RBL$_L$), which are converted to VAs. The layout-extracted netlist of the MSB channel in the CMU is used for the Monte Carlo simulation. Considering the netlist size, we use 200 simulations; the sample number is sufficient to obtain the mean and standard deviation values. The result is similar for the LSB channel under the layout symmetry.

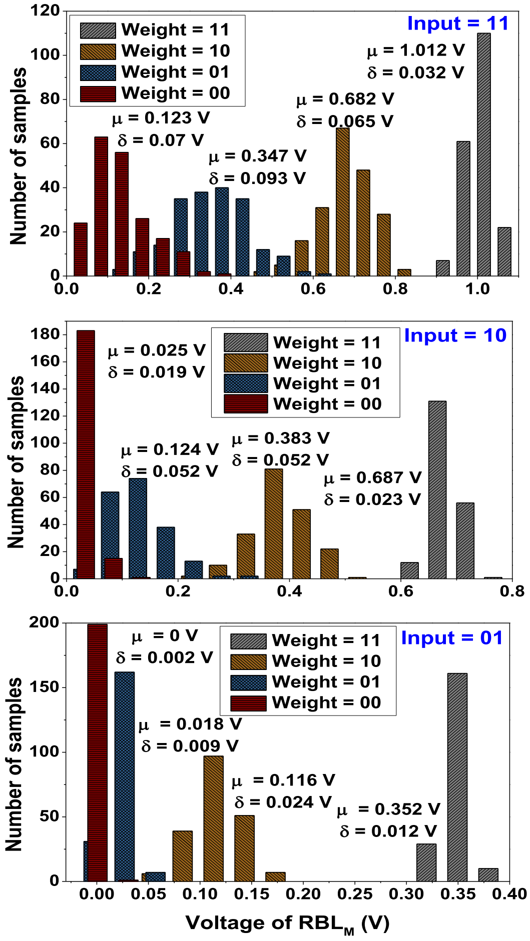

FIGURE 25. Statistical distribution of the multiplication result of the MSB channel of the CMU.

The SAR ADC uses a capacitive DAC for the charge redistribution to generate the reference voltage for the comparator. When the capacitive DAC is realized using extra capacitors, it can cost a large area. In this work, the capacitive DAC is realized using the inherent capacitance of the 16 BL/BLX pairs in the array. The sixteen BLs are divided into five groups to form the binary-weighted capacitors for the DAC. Fig. 26 shows the statistical distribution of the

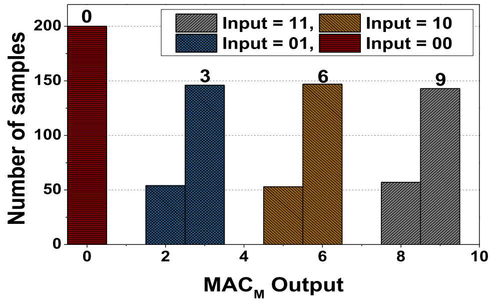

FIGURE 26. Statistical distribution of the ADC output.

ADC output. The ADC operates at 160 MHz. All inputs are fed with the same input code, from 00 to 11, to characterize the ADC, and the weights are written with the same weight of W = 11, and the expected output 4-b precision from the ADC is observed. The result indicates that the capacitive DAC, realized using the intrinsic BL capacitance, achieves a relatively low variation with the symmetry layout.

F. MEASUREMENT RESULTS

Fig. 27(a) shows the microphotograph of the SRAM-CIM unit-macro. The macro size of 128 × 16 is chosen for proof-of-concept validation of two parallel multibit MAC operations. Fig. 27(b) shows the area breakdown of SRAM-CIM unit-macro. The array efficiency of 10T cells is 38.4%, which can be increased by using a larger size SRAM macro. Fig. 27(c) shows the power breakdown, where the ADC and CIM_IO consume 43.1% and 23.8%, respectively. Table 6 shows the summary of the proposed 10T SRAM-CIM unit-macro.

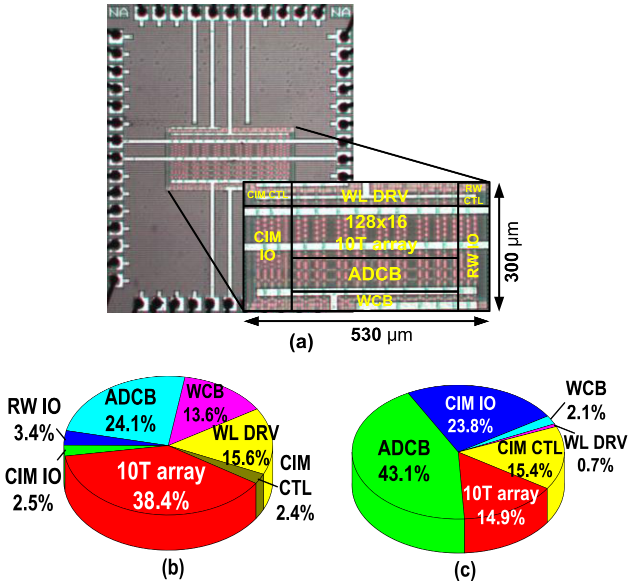

FIGURE 27. (a) Microphotograph of the fabricated SRAM CIM unit-macro, (b) area breakdown, (c) power breakdown of the SRAM-CIM unit-macro.

TABLE 6. Chip summary of the 10T SRAM-CIM for multibit MAC.

Technology	180 nm CMOS
Unit-macro size	2 Kb
Bit-cell type	10T
Bit-cell size	1.48 μm × 7.19 μm
Input, weight, output (bit)	4, 4, 8
Main clock (compute mode)	20 MHz
ADC clock	160 MHz
Max. operation per cycle	64
Power consumption (mW)	4.63

A multibit DNN is implemented using the proposed SRAM-CIM macro and an FPGA. The FPGA interfaces with the SRAM-CIM unit-macro by GPIO and level-shifters. The nonlinear layers (pooling, activation, and batch-norm) are implemented in the FPGA. Test setup and network structure are similar to those used in the binary MAC (See Fig. 16), modified for multibit MAC operation. All weights are trained using 4-b precision. The inputs and outputs are quantized using 4-b and 8-b precision, respectively. Each 4-b weight channel is divided into two groups of W[3:2] and W[1:0], and they are stored in one pair row in $\mathrm{LMAC_{MSB}}$ and $\mathrm{LMAC_{LSB}}$, respectively. The activation inputs corresponding to weight are applied to multiple columns, and they are converted to analog voltage to produce IWP. The CONV1 and FL7 layers are implemented on the SRAM-CIM. Table 7 shows the mapping of CONV1 and FL7 layers for the SRAM-CIM. For the CONV1 layer, the weight size is 3 × 3, with one input channel and eight output channels. Each of the eight output channels is mapped to one pair row in the array. In every cycle, 9 (=3 × 3) activation inputs (IN[i]) are sent to nine columns of 10T SRAM-CIM array, and computation of two 8-b outputs are performed in parallel. Because the unit-macro supports two MAC operations in parallel, this process is repeated four cycles to complete the CONV1 layer, resulting in 9 × 2×2 operations per cycle. For the FL7 layer, sixteen output channels are mapped to 16 pair rows. Sixteen activation inputs are sent to the array every cycle to process two parallel multibit MAC operations, resulting in 16 × 2×2 operations per cycle. An inference accuracy of 97.5% is achieved using the implementation. The accuracy loss compared to the simulation is less than 0.5%.

TABLE 7. Parameter mapping of the layers for the SRAM-CIM.

Parameters	CONV1	FL7
Weight size	1 × 3 × 3 × 8*	32 × 32**
# Channels	8	32
# MACB	2	2
# Pair rows/local array	8	16
# Columns/local array	9	16
# Repeated computations	4	32
# Operation/cycle	9 × 2 × 2***	16 × 2 × 2***

*CONV weight size = (input channel × height × width × output channel).
**FL weight size = (input channel × output channel).
***1 MAC = 2OPs (1 multiply + 1 add).

Recent works show large capacity SRAM-CIM macros, for example, 64 Kb [28], 384 Kb [29], and 4.5 Mb [33], which allows the application to speech recognition and image classification. Works [19] and [23] report the inference results using the CIFAR-10 dataset, which demands a relatively large network with many inputs. The number of input is 64 and 256 for the works [19] and [23]. Our design has 16 inputs which are not enough to process the CIFAR-10 dataset.

Table 8 shows a comparison with the previous SRAM-CIM works supporting multibit MAC operation. Compared to [16], [17], which is based on a 6T bit-cell, our work is robust to the write disturb. Compared to [16], [18], [19], our work supports higher weight (4-b) and output (8-b) precision. The work [17] supports 8-b precision; however, this work achieves a relatively low accuracy of 92%, attributed to the sensitive read current by word-line voltage variation. Compared to [18], which has similar accuracy with ours,

TABLE 8. Comparison with previous SRAM-CIM works.

	[16]	[17]	[18]	[19]	This work
Technology	130 nm	65 nm	65 nm	55 nm	180 nm
Macro size	16 Kb	16 Kb	16 Kb	3.84 Kb	2 Kb
Bit-cell	6T	6T	10T	Twin 8T	10T
Input (bit)	5	8	6	1/2/4	4
Weight (bit)	1	8	1	2/5	4
Output (bit)	1	8	6	3/5/7	8
Main clock	50 MHz	312.5 kHz	5 MHz	98-312 MHz	20 MHz
ADC clock	-	-	250 MHz	-	160 MHz
Operation/cycle	-	256	1600	216/512	64
Accuracy (MNIST)	90%	92%	98.3%	99.5%	97.5%*
Throughput (GOPS)	57.6	10.2	8	21.2-67.5	1.28
Throughput density (GOPS/Kb)	3.6	0.63	0.5	5.5-17.5	0.64

*result when CONV1 and FL7 layers are implemented in the SRAM-CIM.

our work achieves a 28% higher throughput density. The limited throughput of our work is attributed to a relatively small unit-macro size of only 2 Kb. If we assume that the throughput is directly proportional to macro size, our work, scaled to 16 Kb, can achieve higher throughput than [17] and [18]. Besides, our work proposes a new method of realizing the SAR ADC using intrinsic bit-line capacitances. This approach solves the area issue of the conventional SAR ADC [19], the low performance of integrating an ADC [18], and the high power consumption of flash ADC [20].

IV. CONCLUSION

This paper presents two SRAM-CIM unit macros using the new 10T SRAM bit-cell for robust and scalable designs. The first SRAM-CIM macro supports binary MAC operation to achieve a high throughput and energy efficiency. Advanced circuit techniques are presented using an input-dependent dynamic reference generator and an input-boosted sense amplifier. The unit macro, fabricated in a 28-nm CMOS process, achieves 409.6 GOPS throughput and 1001.7 TOPS/W energy efficiency. The second SRAM-CIM macro supports multibit MAC operations to increase the inference accuracy with 4-b weight, 4-b input, and 8-b output precision. A new approach of realizing four 2-b multiplication in parallel increases the signal margin by 16× compared to the conventional approach. We present an area-efficient approach for realizing the capacitive DAC in the SAR ADC using intrinsic bit-line capacitances of the SRAM-CIM macro. The proposed approach of realizing four 2-b parallel multiplication using the capacitive DAC is successfully demonstrated with a modified LeNet-5 neural network. Key contributions of this work can be summarized: 1) this work demonstrates an approach for realizing one of the best energy efficiency (1001.7 TOPS/W) and throughput density (409.6 GOPS/Kb) using the advantage of both process scaling (28 nm) and full parallel computing, 2) we present proof-of-concept validation of realizing two parallel multibit (4-b weight, 4-b input, and 8-b output) MAC operations. The key new finding is that the successful operation of the proposed unit-macro for both multibit and binary MAC operation is demonstrated using the proposed 10T bit-cell. Therefore, the 10T bit-cell is promising for robust and scalable SRAM-CIM designs, which will be useful for realizing full parallel computing of the AI edge processors.

ACKNOWLEDGMENT

The chip fabrication and CAD tools were supported by the IC Design Education Center (IDEC).

REFERENCES

[1] G. Hinton, L. Deng, D. Yu, G. Dahl, A.-R. Mohamed, N. Jaitly, A. Senior, V. Vanhoucke, P. Nguyen, T. Sainath, and B. Kingsbury, "Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups," *IEEE Signal Process. Mag.*, vol. 29, no. 6, pp. 82–97, Nov. 2012.

[2] Y. Taigman, M. Yang, M. Ranzato, and L. Wolf, "DeepFace: Closing the gap to human-level performance in face verification," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2014, pp. 1701–1708.

[3] A. B. Nassif, I. Shahin, I. Attili, M. Azzeh, and K. Shaalan, "Speech recognition using deep neural networks: A systematic review," *IEEE Access*, vol. 7, pp. 19143–19165, Feb. 2019.

[4] R. Qi, R.-S. Jia, Q.-C. Mao, H.-M. Sun, and L.-Q. Zuo, "Face detection method based on cascaded convolutional networks," *IEEE Access*, vol. 7, pp. 110740–110748, Aug. 2019.

[5] Z. Fang and J. Zhan, "Deep physical informed neural networks for metamaterial design," *IEEE Access*, vol. 8, pp. 24506–24513, Feb. 2020.

[6] D. Liu, Y. Tan, E. Khoram, and Z. Yu, "Training deep neural networks for the inverse design of nanophotonic structures," *ACS Photon.*, vol. 5, no. 4, pp. 1365–1369, Feb. 2018.

[7] M. Mahmud, M. S. Kaiser, A. Hussain, and S. Vassanelli, "Applications of deep learning and reinforcement learning to biological data," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 29, no. 6, pp. 2063–2079, Jun. 2018.

[8] N. Whatmough, S. K. Lee, H. Lee, S. Rama, D. Brooks, and G. Y. Wei, "A 28 nm SoC with a 1.2 GHz 568 nJ/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.

[9] B. Moons and M. Verhelst, "An energy-efficient precision-scalable ConvNet processor in 40-nm CMOS," *IEEE J. Solid-State Circuits*, vol. 52, no. 4, pp. 903–914, Apr. 2017.

[10] Y. Li and Y. Du, "A novel software-defined convolutional neural networks accelerator," *IEEE Access*, vol. 7, pp. 177922–177931, Nov. 2019.

[11] Y. H. Chen, T. Krishna, J. S. Emer, and V. Sze, "Eyeriss: An energy-efficient reconfigurable accelerator for deep convolutional neural networks," *IEEE J. Solid-State Circuits*, vol. 52, no. 1, pp. 127–138, Jan. 2017.

[12] W. Choi, K. Choi, and J. Park, "Low cost convolutional neural network accelerator based on bi-directional filtering and bit-width reduction," *IEEE Access*, vol. 6, pp. 14734–14746, Apr. 2018.

[13] M. T. Hailesellasie and S. R. Hasan, "MulNet: A flexible CNN processor with higher resource utilization efficiency for constrained devices," *IEEE Access*, vol. 7, pp. 47509–47524, Apr. 2019.

[14] C. Bao, T. Xie, W. Feng, L. Chang, and C. Yu, "A power-efficient optimizing framework FPGA accelerator based on winograd for YOLO," *IEEE Access*, vol. 8, pp. 94307–94317, May 2020.

[15] S. Lee, S. Joo, H. K. Ahn, and S.-O. Jung, "CNN acceleration with hardware-efficient dataflow for super-resolution," *IEEE Access*, vol. 8, pp. 187754–187765, Oct. 2020.

[16] 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.

[17] 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.

[18] A. Biswas and A. P. Chandrakasan, "CONV-SRAM: An energy-efficient SRAM with in-memory dot-product computation for low-power convolutional neural networks," *IEEE J. Solid-State Circuits*, vol. 54, no. 1, pp. 217–230, Jan. 2019.

[19] X. Si, R. Liu, S. Yu, R.-S. Liu, C.-C. Hsieh, K.-T. Tang, Q. Li, M.-F. Chang, J.-J. Chen, Y.-N. Tu, W.-H. Huang, J.-H. Wang, Y.-C. Chiu, W.-C. Wei, S.-Y. Wu, and X. Sun, "A twin-8T SRAM computation-in-memory unit-macro for multibit CNN-based AI edge processors," *IEEE J. Solid-State Circuits*, vol. 55, no. 1, pp. 189–202, Jan. 2020.

[20] A. Agrawal, A. Jaiswal, D. Roy, B. Han, G. Srinivasan, A. Ankit, and K. Roy, "Xcel-RAM: Accelerating binary neural networks in high-throughput SRAM compute arrays," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 66, no. 8, pp. 3064–3076, Aug. 2019.

[21] X. Si, M.-F. Chang, W.-S. Khwa, J.-J. Chen, J.-F. Li, X. Sun, R. Liu, S. Yu, H. Yamauchi, and Q. Li, "A dual-split 6T SRAM-based computing-in-memory unit-macro with fully parallel product-sum operation for binarized DNN edge processors," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 66, no. 11, pp. 4172–4185, Nov. 2019.

[22] 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.

[23] Z. Jiang, S. Yin, J.-S. Seo, and M. Seok, "C3SRAM: An in-memory-computing SRAM macro based on robust capacitive coupling computing mechanism," *IEEE J. Solid-State Circuits*, vol. 55, no. 7, pp. 1888–1897, Jul. 2020.

[24] K. Roy, I. Chakraborty, M. Ali, A. Ankit, and A. Agrawal, "In-memory computing in emerging memory technologies for machine learning: An overview," in *Proc. 57th ACM/IEEE Design Autom. Conf. (DAC)*, Jul. 2020, pp. 1–6.

[25] J. Yue, X. Feng, Y. He, Y. Huang, and Y. Wang, "A 2.75-to-75.9TOPS/W computing-in-memory NN processor supporting set-associate block-wise zero skipping and ping-pong CIM with simultaneous computation and weight updating," in *IEEE Int. Solid-State Circuits Conf. (ISSCC) Dig. Tech. Papers*, Feb. 2021, pp. 238–239.

[26] Z. Chen, X. Chen, and J. Gu, "A 65 nm 3T dynamic analog RAM-based computing-in-memory macro and CNN accelerator with retention enhancement, adaptive analog sparsity and 44TOPS/W system energy efficiency," in *IEEE Int. Solid-State Circuits Conf. (ISSCC) Dig. Tech. Papers*, Feb. 2021, pp. 240–241.

[27] Q. Dong, M. E. Sinangil, B. Erbagci, D. Sun, W.-S. Khwa, H.-J. Liao, Y. Wang, and J. Chang, "A 351TOPS/W and 372.4GOPS compute-in-memory SRAM macro in 7 nm FinFET CMOS for machine-learning applications," *IEEE Int. Solid-State Circuits Conf. (ISSCC) Dig. Tech. Papers*, Feb. 2020, pp. 242–243.

[28] R. Guo, Y. Liu, S. Zheng, and S. Y. Wu, "A 5.1pJ/Neuron 127.3us/inference RNN-based speech recognition processor using 16 computing-in-memory SRAM macros in 65 nm CMOS," in *Proc. Symp. VLSI Circuits*, Jun. 2019, pp. 120–121.

[29] J.-W. Su, Y. C. Chou, R. Liu, T. W. Liu, and P. J. Lu, "A 28 nm 384kb 6T-SRAM computation-in-memory macro with 8b precision for AI edge chips," in *IEEE Int. Solid-State Circuits Conf. (ISSCC) Dig. Tech. Papers*, Feb. 2021, pp. 250–252.

[30] Y.-C. Chiu, Z. Zhang, J.-J. Chen, X. Si, R. Liu, Y.-N. Tu, J.-W. Su, W.-H. Huang, J.-H. Wang, W.-C. Wei, J.-M. Hung, S.-S. Sheu, S.-H. Li, C.-I. Wu, R.-S. Liu, C.-C. Hsieh, K.-T. Tang, and M.-F. Chang, "A 4-Kb 1-to-8-bit configurable 6T SRAM-based computation-in-memory unit-macro for CNN-based AI edge processors," *IEEE J. Solid-State Circuits*, vol. 55, no. 10, pp. 2790–2801, Oct. 2020.

[31] M. E. Sinangil, B. Erbagci, R. Naous, K. Akarvardar, D. Sun, W.-S. Khwa, H.-J. Liao, Y. Wang, and J. Chang, "A 7-nm compute-in-memory SRAM macro supporting multi-bit input, weight and output and achieving 351 TOPS/W and 372.4 GOPS," *IEEE J. Solid-State Circuits*, vol. 56, no. 1, pp. 188–198, Jan. 2021.

[32] J. Wang, X. Wang, C. Eckert, A. Subramaniyan, R. Das, D. Blaauw, and D. Sylvester, "A 28-nm compute SRAM with bit-serial logic/arithmetic operations for programmable in-memory vector computing," *IEEE J. Solid-State Circuits*, vol. 55, no. 1, pp. 76–86, Jan. 2020.

[33] H. Jia, M. Ozatay, Y. Tang, H. Valavi, R. Pathak, J. Lee, and N. Verma, "A programmable neural-network inference accelerator based on scalable in-memory computing," in *IEEE Int. Solid-State Circuits Conf. (ISSCC) Dig. Tech. Papers*, Feb. 2021, pp. 236–237.

[34] K. Zhang, U. Bhattacharya, Z. Chen, F. Hamzaoglu, D. Murray, N. Vallepalli, Y. Wang, B. Zheng, and M. Bohr, "A 3-GHz 70-mb SRAM in 65-nm CMOS technology with integrated column-based dynamic power supply," *IEEE J. Solid-State Circuits*, vol. 41, no. 1, pp. 146–151, Jan. 2006.

[35] C.-X. Xue, W.-C. Zhao, T.-H. Yang, Y.-J. Chen, H. Yamauchi, and M.-F. Chang, "A 28-nm 320-kb TCAM macro using split-controlled single-load 14T cell and triple-margin voltage sense amplifier," *IEEE J. Solid-State Circuits*, vol. 54, no. 10, pp. 2743–2753, Oct. 2019.

[36] 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.

[37] J. P. Uyemura, *Introduction to VLSI Circuits and Systems*. Hoboken, NJ, USA: Wiley, 2002.

VAN TRUONG NGUYEN was born in Vietnam, in 1991. He received the B.S. degree in electrical and electronics engineering from the Ho Chi Minh City University of Technology, in 2014. He is currently pursuing the M.S. degree with the School of Electronics and Information, Kyung Hee University, South Korea. His research interest includes computation in memory for machine learning.

JIE-SEOK KIM was born in South Korea, in 1995. He received the B.S. degree from the School of Electronics and Information, Kyung Hee University, South Korea, in 2018, where he is currently pursuing the M.S. degree with the School of Electronics and Information. His research interests include deep learning and digital circuits.

JONG-WOOK LEE (Senior Member, IEEE) was born in South Korea, in 1970. He received the B.S. and M.S. degrees in electrical engineering from Seoul National University, Seoul, South Korea, in 1993 and 1997, respectively.

From 1994 to 1996, he served in the military. From 1998 to 2002, he was a Research Assistant with the School of Electrical and Computer Engineering, Purdue University, West Lafayette, IN, USA. From 2003 to 2004, he was a Postdoctoral Research Associate with the University of Illinois at Urbana–Champaign, Champaign, IL, USA. In 2004, he joined the School of Electronics and Information, Kyung Hee University, South Korea. His research interests include low-power sensor IC, wireless power transfer, RFID tag IC, and power management IC design.