A High-Efficiency 16-kb SRAM-Based CIM Architecture With Automatic Write-Back Using 28-nm CMOS Technology

L S S Pavan Kumar Chodisetti
Department of Electrical Engineering
National Sun Yat-sen University
Kaohsiung, Taiwan
pavanece9@gmail.com

Yi-Chun Lin
Department of Electrical Engineering
National Sun Yat-sen University
Kaohsiung, Taiwan
lin.yi7768@gmail.com

Ya-Chuan Chang
Institute of Integrated Circuit Design
National Sun Yat-sen University
Kaohsiung, Taiwan
grf901122@gmail.com

Jeffrey Sean Walling
Bradley Dept. of Electrical
and Computer Eng., Virginia Tech
Blacksburg, VA, USA
jswalling@vt.edu

Yang Yi
Bradley Dept. of Electrical
and Computer Eng., Virginia Tech
Blacksburg, VA, USA
yangyi8@vt.edu

Chua-Chin Wang*
Department of Electrical Engineering
National Sun Yat-sen University
Kaohsiung, Taiwan
ccwang@ee.nsysu.edu.tw

***Abstract*—The growing demand for high-efficiency artificial intelligence (AI) hardware has intensified interest in computing-in-memory (CIM) architectures capable of overcoming the von Neumann bottleneck. This work presents a 16-kb single-ended (S.E.) 6T SRAM–based CIM designed in TSMC 28-nm CMOS (TN28HPCplus) process, featuring with integrated addition and multiplication logic, automatic write-back, and built-in self-test (BIST). The proposed design supports 8-bit input and 8-bit weight operations and achieves significant improvements in energy and area efficiency. Post-layout results demonstrate an energy efficiency of 43.34 TOPS/W, a bitwise energy efficiency of 2774 (TOPS/W)$\times$bit^2, an area efficiency of 148.2 GOPS/mm^2, and a bitwise area efficiency of 9.48 (TOPS/mm^2)$\times$bit^2. These results highlight the potential of the proposed CIM architecture for compact and energy-efficient edge-AI computing.**

***Index Terms*—single-ended SRAM, CIM, 28-nm, area efficiency, automatic write back, TOPS, edge-AI**

I. Introduction

Traditional AI and neural-network hardware architectures are predominantly based on the von Neumann computing model, where memory and computational units such as the arithmetic logic unit (ALU) are physically separated. This separation leads to the well-known von Neumann bottleneck, where frequent data movement between memory and the ALU increases latency and degrades throughput and power efficiency. To address these limitations, researchers have increasingly focused on computing-in-memory (CIM) architectures, e.g., [1], [2]. By performing operations directly within memory arrays, CIM systems significantly reduce data-transfer overhead. In these architectures, data storage, retrieval, and computation are integrated within the same memory structure, supported by peripheral circuits that manage logic operations and coordinate functional transitions. Furthermore, advancements in peripheral circuits, such as I/O buffers, play a crucial role in enhancing the efficiency of CIM systems. For instance, an I/O buffer has been developed using 16-nm FinFET CMOS process, addressing process variations and ensuring reliable high-speed data transfer [3]. Such innovations complement CIM architectures by improving overall throughput and energy efficiency in AI applications.

Among memory technologies used in CIMs, SRAM is commonly favored over DRAM because of its lower access latency and suitability for bitwise logic operations [1], [2]. However, standard SRAM cells consume more power and occupy larger area, motivating alternative designs such as the 4T load-less SRAM [4]. CIM implementations based on conventional 6T SRAM have demonstrated basic logic-in-memory capabilities [5], while single-ended (S.E.) 6T SRAM structures with disturb-free operation have enabled Boolean and arithmetic functions [6]. For AI and convolution-based processing, it is essential to handle both positive and negative data values during parallel operations such as addition and multiplication. Prior CIM work using 40-nm CMOS technology incorporated a ripple-carry adder and multiplier within a disturb-free S.E. 7T 1-kb SRAM array, implemented using a full swing gate-diffusion-input (FS-GDI) technique [7], [8]. Though this approach reduced area and power while preserving full-swing operation, its energy and area efficiencies were limited to 7.66 TOPS/W and 27 GOPS/mm^2, respectively. To improve these metrics, the key circuit blocks of the CIM architecture have been redesigned in TSMC 28-nm CMOS process. The updated design incorporates a 16-kb S.E. SRAM array supporting 8-bit input and 8-bit weight processing, achieving substantial improvements in computational density and power efficiency.

*Corresponding author

979-8-3315-5925-0/26/$31.00 ©2026 IEEE

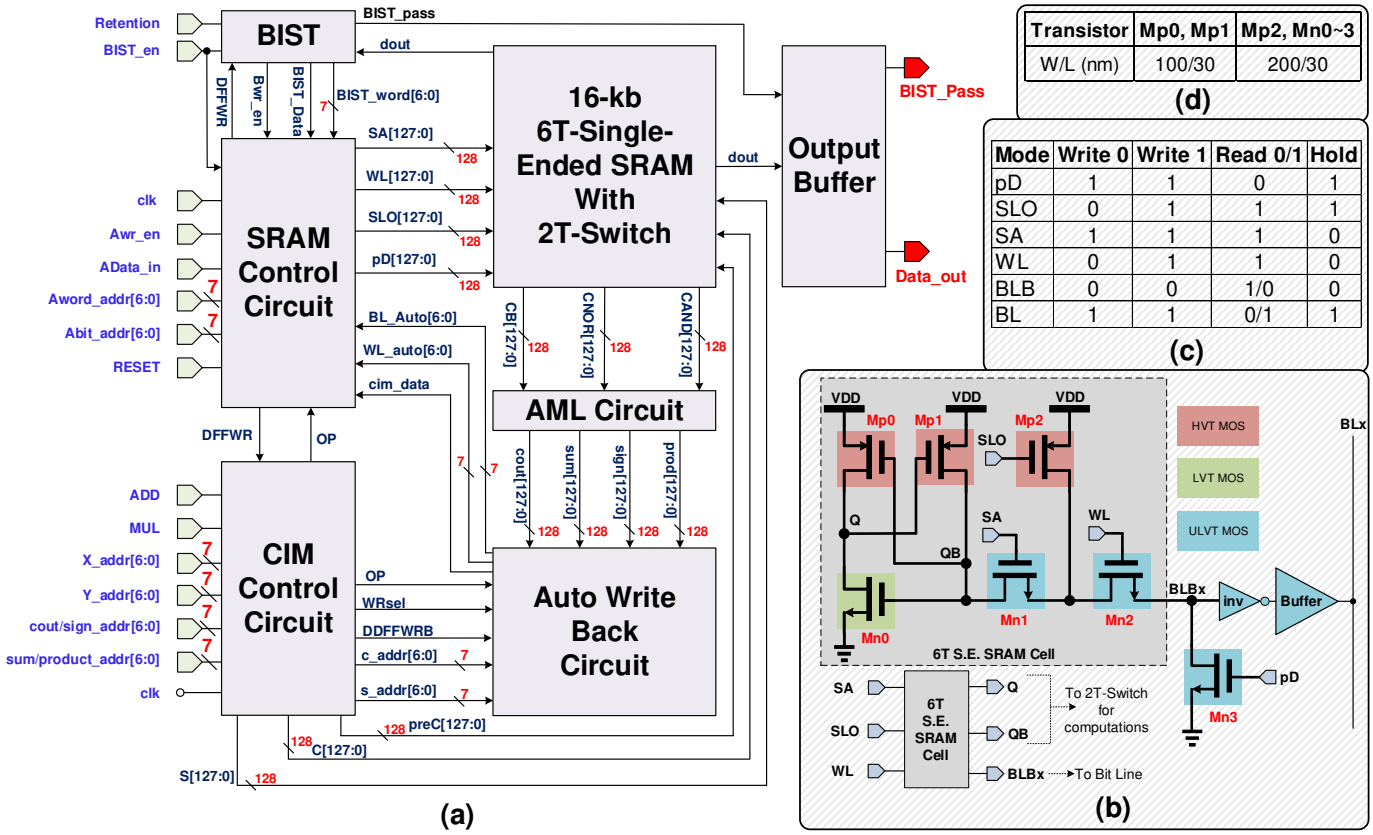

Fig. 1. (a) Architecture of the proposed CIM; (b) 6T S.E. SRAM Cell architecture; (c) Operational modes of the SRAM Cell; (d) W/L values of the SRAM cell

II. Proposed CIM Architecture

Fig. 1(a) illustrates the architecture diagram of the proposed CIM system. Building upon our previous design, the proposed CIM circuit design incorporates a Built-In Self-Test (BIST) circuit, and an Addition & Multiplication Logic Circuit (AML Circuit) [8]. Our novel contributions for this proposed design includes a 16-kb 6T-S.E. SRAM with 2T-Switch, a SRAM Control circuit, a CIM Control circuit and an automatic write-back circuit (Auto Write Back circuit). The Output Buffer follows the design principles of our prior works [9]–[11]. The primary purpose of BIST circuit is to generate random address signals to verify the basic read and write functionality of the memory in the absence of external address and data signals.

The proposed 6T-S.E. SRAM Cell to implement a 128×128 (16-kB) memory array is shown in Fig. 1(b). The corresponding operational modes of the SRAM Cell are presented in Fig. 1(c), and the transistors dimensions are provided in Fig. 1(d). Fig. 2 presents an example of in-memory operations using a 2×2 memory matrix. The user specifies operand locations and desired operations through the CIM Control Circuit. The operation begins by using the PMOS transistors driven by the preC[x] signal to charge the operation lines CB[x], CAND[x], and CNOR[x] to a high potential. Subsequently, different control signals (C[x] and S[x]) are applied to complete logic operations via the 2T-Switches connected to Q and QB points on either side of the memory cells. Based on the data stored in the memory, three primary operation results are generated: CB, CAND (Logical AND operation), and CNOR (Logical NOR operation). These results are then processed further through the AML circuit to perform logical operations, generating outputs such as carry (cout), sum (sum), sign (sign), and product (prod). Finally, the computed operation results are written back into the memory for subsequent operations.

A. SRAM Control Circuit

The SRAM Control Circuit in Fig. 1(a) consists of two of two major sub-modules: the SRAM Input Selection Circuit and the SRAM Execution Unit, as shown in Fig. 3 and Fig. 4, respectively. The main role of the SRAM Input Selection Circuit is to select appropriate input signals based on the operation requirements. It then passes the selection and decoding results to the SRAM execution Unit, which executes the necessary operations for writing data into or reading data from the SRAM cells.

B. CIM Control circuit

Fig. 5 shows the architecture of the CIM Control Circuit as referenced in Fig. 1(a), comprising the CIM Timing Control Circuit and the CIM Address Control Circuit. Its main function is to generate the operation control signals and addresses required for the intended operations. The main function of the

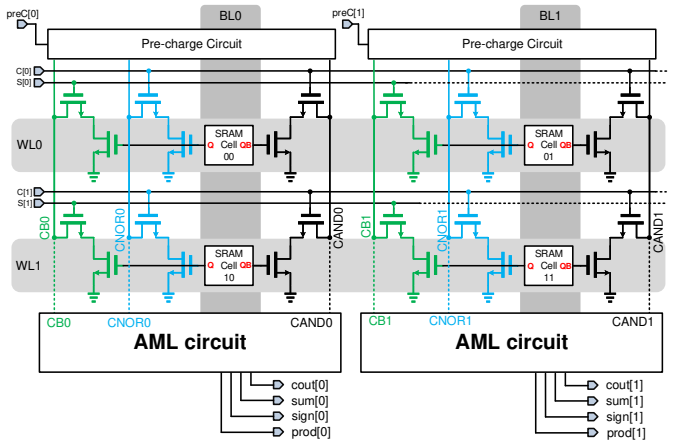

Fig. 2. CIM schematic (2 × 2 illustration)

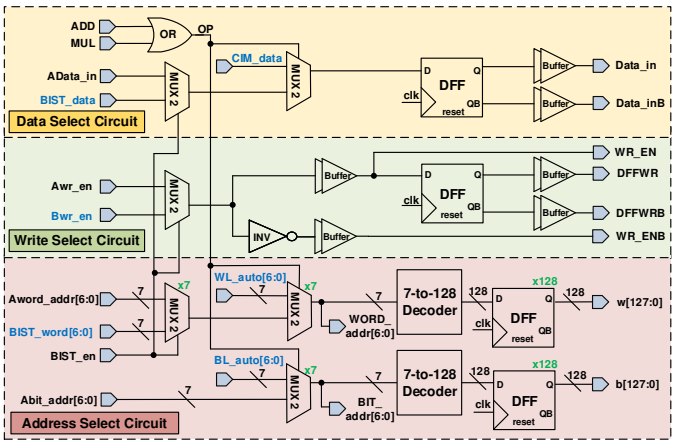

Fig. 3. SRAM Input Selection Circuit

CIM Timing Control Circuit is to perform frequency division and phase adjustment based on the clk and DFFWR signals, and to generate DDFFWRB, CIM_Datasel and WRsel signals. These signals will be provided to the Auto Write Back circuit to ensure its functioning normally.

The CIM Address Control Circuit generates control signals based on user-defined parameters to enable precise in-memory operations. This circuit generates the precharge signal preC[x] and C[x] by using the user-defined operand addresses X_addr[6:0] and Y_addr[6:0], which are provided to the 2T-Switch circuit to implement CNOR[x] and CAND[x] operations. Additionally, this circuit generates the carry bit address selection signal S[x] based on the user-defined addresses cout/sign_addr[6:0].

C. Auto Write Back circuit

The write-back mechanism is essential for arithmetic operations such as addition (ADD) and multiplication (MUL). It ensures that the resulting sum or product is stored at the specified address. Fig. 6 illustrates the architecture of the Auto Write Back circuit, composed of a Bit Address Auto-generate Circuit, a Word Address Auto-generate Circuit and a CIM

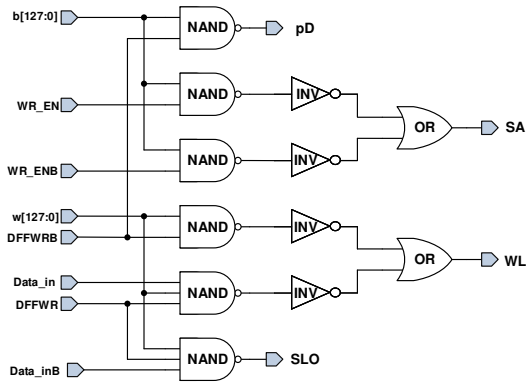

Fig. 4. SRAM Execution Unit Circuit

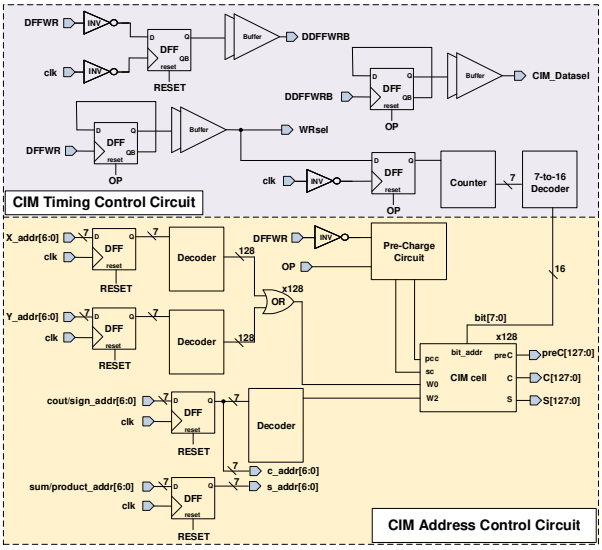

Fig. 5. CIM Control circuit architecture diagram

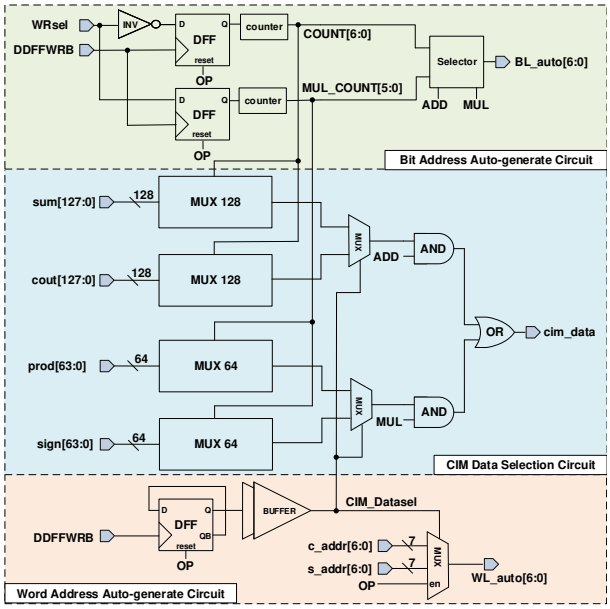

Fig. 6. Auto Write Back circuit architecture diagram

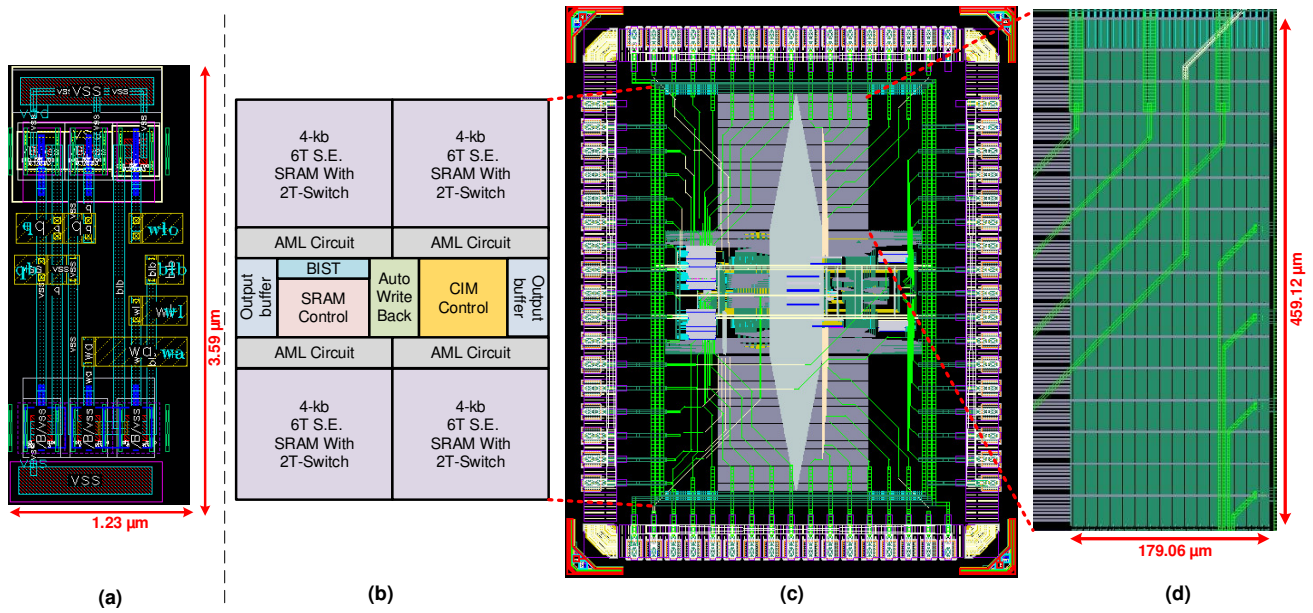

Fig. 7. (a) Layout of the 6T-S.E. SRAM Cell; (b) Floorplan of the CIM; (c) Layout of the CIM; (d) Enlarged view of 4-kb SRAM

Data Selection Circuit. The Bit Address Auto-generate Circuit utilizes the clk signal, WRsel and DDFFWRB, generated by the CIM Control circuit as its input basis. These signals are fed into a counters, which produces outputs COUNT[6:0] and MUL_COUNT[5:0]. Depending on the logic state of the ADD or MUL signal, a selector chooses either COUNT[6:0] or MUL_COUNT[5:0] as the automatically generated bit address (BL_auto[6:0]), thereby enabling efficient automatic control of the bit address. The Word Address Auto-generate Circuit utilizes DDFFWRB, c_addr[6:0] and s_addr[6:0] signals generated by the CIM Control circuit as its input basis. These signals are processed through a multiplexer which alternates the automatic write-back word address (WL_auto[6:0]) under the control of the OP and CIM_Datasel signals, ensuring that the operation results are stored in memory at the specified addresses. The CIM Data Selection Circuit processes the final result of logic operations by utilizing multiplexers to selectively output either the sum and cout from an addition operation or the product and sign from a multiplication operation, based on the CIM_Datasel signals. It integrates these operation results to generate the output data for the in-memory operation (cim_data).

III. Implementation and Verification

The proposed 16-kb SRAM-based CIM was designed using the TSMC 28-nm CMOS Logic (TN28HPCplu) process. Fig. 7(a) illustrates the layout of a 6T single-ended SRAM cell, which occupies an area of 1.23 × 3.59 μm^2. Fig. 7(b) shows the floorplan of the complete CIM architecture, while Fig. 7(c) presents the full-chip layout, with a total chip area size of 1507.6 × 1912.6 μm^2, and a core circuit area of 859.12 × 1435.405 μm^2.

A delay-matched clock-tree layout strategy is employed to minimize timing skew across the memory array. The 16-kb

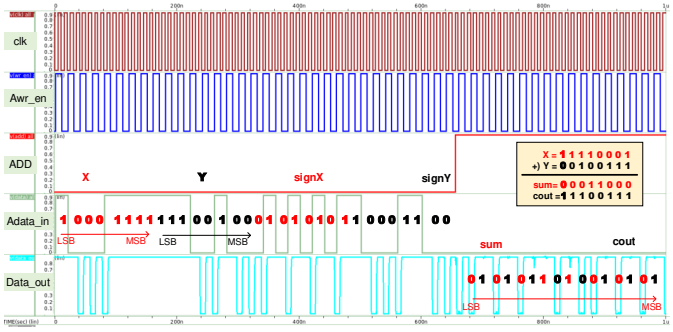

Fig. 8. Post-layout simulation results of the addition operation for user defined inputs

SRAM is partitioned into four 4-kb blocks located near the 4 corners of the chip, while the CIM control and peripheral logic are placed at the center. This arrangement reduces the propagation delay of control signals and ensures proper synchronization during in-memory operations. An enlarged view of a 4-kb SRAM at top-right corner of the chip is shown in Fig. 7(d).

The proposed S.E. SRAM Cell is verified across all PVT corners through post-layout simulations. The minimum Static Noise Margin (SNM) value for the SRAM Cell is 202 mV, observed at the [FS, 0.81V(VDD-10%), 100°C] PVT corner. The worst-case power consumption of the SRAM Cell during read '0', read '1', write '0', write '1' operations is 1.21 μW, 543.5 nW, 16.57 μW, and 22.55 μW respectively.

The proposed 16-kb CIM circuit implements 8-bit input and 8-bit weight multiplication and addition operations in the memory array, and can automatically write back the calculation results. The worst case power consumption of the proposed 16-kb CIM during the computations is 12.653 mW

TABLE I
PERFORMANCE COMPARISON

	[8]	[12]	[13]	Ours
Technology (nm)	40	28	28	28
Supply Voltage (V)	0.9	0.85-1.0	0.9	0.9
SRAM Cell	7T	6T	6T	6T
SRAM Array Size (kb)	1	64	64	16
Weight Bits	4	8	2	4
Input Bits	8	8	8	8
SRAM area (mm^2)	0.2072	NA	1.39	0.328
CIM Core area (mm^2)	0.2208	NA	NA	1.233
Chip area (mm^2)	0.729	NA	NA	2.88
GOPS	5.59	NA	85.3	48.6
Energy efficiency (TOPS/W)	7.66	7.6	42.1	**43.34**
Bitwise energy efficiency (TOPS/W)×bit^2	122.6	486.4	673.6	**2774**
Area efficiency (GOPS/mm^2)	27	NA	61.337	**148.2**
Bitwise area efficiency (TOPS/mm^2)×bit^2	0.432	NA	0.981	**9.48**

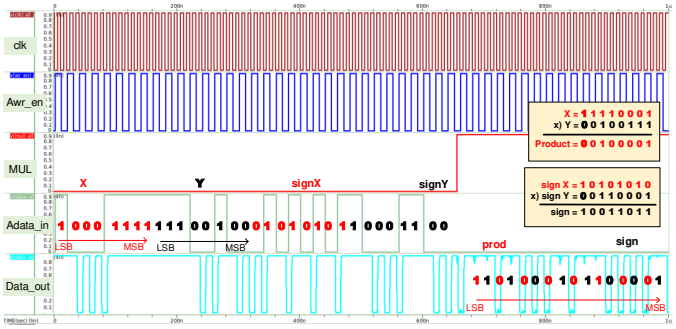

Fig. 9. Post-layout simulation results of the multiplication operation for user defined inputs

(128×128 SRAM Array: 7 mW, Output buffer: 4.531 mW, Remaining core: 1.122 mW), when the voltage supply (VDD) is 0.9 V and the system clock is set to 100 MHz. The presented CIM is incapable of executing multiplication and addition in parallel. The post-layout simulation results of the complete CIM circuit for addition and multiplication, performed under simulation conditions of a system voltage VDD = 0.9 V, the corner of the TT process and a temperature of 25 °C, are shown in Fig. 8 and Fig. 9, respectively. Table I presents a comparison of recent SRAM-based CIM designs with our proposed work.

IV. CONCLUSION

This work presents a 16-kb single-ended SRAM-based CIM implemented in TSMC 28-nm CMOS, integrating arithmetic logic, automatic write-back, and BIST to enable efficient in-memory addition and multiplication. By redesigning the memory cell, peripheral circuits, and CIM control architecture, the proposed system achieves significantly enhanced performance compared with prior SRAM-based CIM designs. Post-layout simulation results demonstrate an an energy efficiency of 43.34 TOPS/W, a bitwise energy efficiency of 2774 (TOPS/W)×bit^2, an area efficiency of 148.2 GOPS/mm^2, and a bitwise area efficiency of 9.48 (TOPS/mm^2)×bit^2. These results indicate that the proposed CIM architecture offers a compact, high-throughput, and energy-efficient solution suitable for next-generation edge-AI and embedded computing applications.

ACKNOWLEDGMENT

The successful completion of this research was supported by the academic resources and research infrastructure provided by the Taiwan Semiconductor Research Institute, National Institutes of Applied Research. We hereby express our sincere gratitude. This work was partially funded by National Science and Technology Council (NSTC), Taiwan, under grant NSTC 112-2923-E-006-003-MY3 and NSTC 114-2923-E-110-001-.

REFERENCES

[1] Q. Dong, S. Jeloka, M. Saligane, Y. Kim, M. Kawaminami, and A. Harada, "A 4 + 2T SRAM for searching and in-memory computing with 0.3-V VDDmin," *IEEE J. of Solid-State Circuits*, vol. 53, no. 4, pp. 1006-1015, April 2018.

[2] A. Agrawal, A. Jaiswal, C. Lee, and K. Roy, "X-SRAM: enabling in-memory boolean computations in CMOS static random access Memories," *IEEE Trans. on Circuits and Syst. I: Regular Papers*, vol. 65, no. 12, pp. 4219-4232, Dec. 2018.

[3] C.-C. Wang, L. S. S. P. K. Chodisetti, J.-Y. Ke, C.-Y. Lo, T.-J. Lee, and L. K. S. Tolentino, "A 6-Gbps 16-nm FinFET CMOS I/O buffer with variation insensitivity ensured by genetic algorithm," *IEEE Trans. on Circuits and Syst. I: Regular Papers,* vol. 71, no. 11, pp. 4961-4972, Nov. 2024.

[4] C.-C. Wang, Y.-L. Tseng, H.-Y. Leo, and R. Hu, "A 4-kB 500-MHz 4-T CMOS SRAM using low-V/sub THN/ bitline drivers and high-V/sub THP/ latches," *IEEE Trans. on Very Large Scale Integration (VLSI) Syst.* vol. 12, no. 9, pp. 901-909, Sept. 2004.

[5] N. S. Dhakad, E. Chittora, G. Raut, V. Sharma, and S. K. Vishvakarma, "In-memory computing with 6T SRAM for multi-operator logic design," *Circuits, Syst., and Signal Process.*, vol. 43, pp. 646–660, 2024.

[6] C.-C. Wang, N. Sulistiyanto, T.-Y. Tsai, and Y.-H. Chen, "Multifunctional in-memory computation architecture using single-ended disturb-free 6T SRAM," *Advances in Electronics Eng.*, pp. 49–57, 2020.

[7] C.-C. Wang, C.-Y. Huang, and C.-H. Yeh, "SRAM-based computation in memory architecture to realize single command of add-multiply operation and multifunction," in *Proc. 2021 IEEE Int. Symp. on Circuits and Syst. (ISCAS)*, Korea, 2021, pp. 1-4.

[8] C.-C. Wang, L. K. S. Tolentino, C.-Y. Huang, and C.-H. Yeh, "A 40-nm CMOS multifunctional computing-in-memory (CIM) using single-ended disturb-free 7T 1-Kb SRAM," *IEEE Trans. on Very Large Scale Integration (VLSI) Syst.*, vol. 29, no. 12, pp. 2172-2185, Dec. 2021.

[9] C.-C. Wang, L. S. S. P. K. Chodisetti, B.-H. Liao, P. Vellanki, and T.-J. Lee, "A 1–6.5 Gbps dual-loop CDR design with coarse-fine Tuning VCO and modified DQFD," *Microelectronics Journal*, vol. 151, pp. 106355, Sep. 2024.

[10] C.-C. Wang, L. S. S. P. K. Chodisetti, D. S. Kamarajugadda, O. L. J. A. Jose, and P. Vellanki, "A 15.13 mW 3.2 GHz 8-bit carry look-ahead adder using single-phase all-N-transistor logic," *Integration*, vol. 98, pp. 102234, Sep. 2024.

[11] L. S. S. P. K. Chodisetti, W.-C. Cheng, T.-J. Lee, and C.-C. Wang, "A dual-loop clock and data recovery system with HLD for extended lock-in range demand," in *Proc. 2025 Int. Conf. on Electronics, AI and Computing (EAIC)*, India, 2025, pp. 1-4.

[12] J.-W. Su, X. Si, Y.-C. Chou, T.-W. Chang, W.-H. Huang, Y.-N. Tu, R. Liu, P.-J. Lu, T.-W. Liu, J.-H. Wang, Z. Zhang, H. Jiang, S. Huang, C.-C. Lo, R.-S. Liu, C.-C. Hsieh, K.-T. Tang, S.-S. Sheu, S.-H. Li, H.-Y. Lee, S.-C. Chang, S. Yu, and M.-F. Chang "15.2 A 28nm 64Kb inference-training two-way transpose multibit 6T SRAM compute-in-memory macro for AI edge chips," in *Proc. 2020 IEEE Int. Solid-State Circuits Conf. - (ISSCC)*, USA, 2020, pp. 240-242.

[13] N. Pan, X. Cui, X. Qiao, K. Xiao, Q. Guo, and Y. Wang, "A 28nm 64Kb SRAM based inference-training tri-mode computing-in-memory macro," in *Proc. 2022 IEEE Int. Symp. on Circuits and Syst. (ISCAS)*, USA, 2022, pp. 2561-2565.