Journal of Signal Processing Systems _####################_
https://doi.org/10.1007/s11265-026-01998-7

RESEARCH

A 75.47 TOPS/W 4-kb Single-Ended SRAM-Based Compute-in-Memory Architecture Using 16-nm FinFET Technology

L S S Pavan Kumar Chodisetti[1] · Cheng-Yao Lo[1] · Jeffrey Sean Walling[2] · Yang Yi[2] · Leenendra Chowdary Gunnam[3,4] · Chua-Chin Wang[1,3]

Received: 9 February 2026 / Revised: 9 February 2026 / Accepted: 10 March 2026
© The Author(s), under exclusive licence to Springer Science+Business Media, LLC, part of Springer Nature 2026

Abstract
In this paper, a 16-nm 1P9M FinFET technology is used to design a computing-in-memory architecture based on a 4-kb single-ended SRAM. The proposed 6T SRAM cell enables an energy-efficient architecture by eliminating the direct connection between the complementary bit line (BLB) and the storage node Q. An Ultra-low threshold voltage (ULVT) transistors in the 2T switch reduce the requirements for capacitors and current compensation circuitry. Additionally, ULVT transistors are integrated into the control circuits of both the SRAM core array and the CIM core, reducing delay caused by threshold voltage drops in the ultra-low supply voltage environment of the 16-nm technology node. The resulting architecture efficiently supports the core operations for convolutional neural networks (CNNs), including addition, signed multiplication, and logical functions. This work highlights the potential of SRAM-based CIM architectures that deliver an energy efficiency of 75.47 TOPS/W and an area efficiency of 140.25 GOPS/mm^2 at a maximum operating frequency of 500 MHz.

Keywords Computing-in-memory (CIM) · Convolution · FinFET · Single-ended SRAM · Ultra-low threshold voltage (ULVT) transistors

✉ Chua-Chin Wang
ccwang@ee.nsysu.edu.tw

L S S Pavan Kumar Chodisetti
pavanece9@gmail.com

Cheng-Yao Lo
loyao0819@gmail.com

Jeffrey Sean Walling
jswalling@vt.edu

Yang Yi
yangyi8@vt.edu

Leenendra Chowdary Gunnam
leenendra.c@srmap.edu.in

[1] Department of Electrical Engineering, National Sun Yat-sen University, No. 70, Lien-hai Road, Gushan District, Kaohsiung City 80424, Taiwan

[2] Bradley Department of Electrical and Computer Engineering, Virginia Tech, Blacksburg, VA 24061, USA

[3] Institute of Integrated Circuit Design, National Sun Yat-sen University, No. 70, Lien-hai Road, Gushan District, Kaohsiung City 80424, Taiwan

[4] Department of Electronics and Communication Engineering, School of Engineering and Sciences, SRM University-AP, Amaravati 522240, Andhra Pradesh, India

1 Introduction

Rapid growth of data-intensive workloads, particularly conventional AI (artificial intelligence) and machine learning applications, demands high-performance and energy-efficient computing hardware [1]. Conventional von Neumann architectures face the memory wall, driven by frequent data transfers between memory and the processing unit, such as the arithmetic logic unit (ALU), impede performance and reduce efficiency in terms of timing, throughput, and energy usage [2]. A promising approach to address the memory wall bottleneck by using Computing-in-Memory (CIM), computations occur directly within the memory array, thus consequently dispensing with extensive data movement between the processing unit and memory unit [3]. In these systems, data are retrieved, computed, and saved within the same memory structure, supported by specialized circuitry to handle additional operations and enable smooth transitions between different computational tasks [4]. Volatile memories such as SRAM and DRAM, as well as newly developed nonvolatile memories (NVMs) such as ReRAM, PCRAM, and STT-MRAM are examples of memory technologies for

compute-in-memory (CIM) that offer trade-offs in latency, density, durability, and integration complexity [5]. Though NVMs make logic - and compute-in-memory paradigms possible, large-scale adoption is restricted by their low write endurance, device variability, and immature fabrication [6]. Besides, these solustions are not cost-effective. In contrast, SRAM is a feasible and energy-efficient option for CIM in CNN accelerators and other data-intensive applications due to its mature CMOS compatibility, quick access, and reliable endurance [7].

In CIM solutions, SRAMs are often preferred over DRAMs due to their faster access speeds and efficient execution of bit-wise logical operations, which enhance their reliability for AI applications [8, 9]. Using conventional 6T SRAM, a CIM design is introduced to perform basic logic operations [10]. Previously, a SRAM-based CIM was implemented using 40-nm CMOS technology [11], utilized a ripple carry adder (RCA) and multiplier (RCAM) within a disturb-free SE. 6T SRAM with disturb-free functionality has also been introduced to enable Boolean logic and addition operations [12]. For AI and convolution computations, managing both positive and negative values is critical, especially during simultaneous operations such as addition and multiplication. Previously, a SRAM-based 7T 1-kb SRAM, employing the full swing-gate diffusion input (FS-GDI) approach. Though this method reduced the chip area, minimized power consumption and ensured full voltage swing resolution, its energy efficiency and area efficiency were limited to 7.66 TOPS/W and 27 GOPS/mm^2, respectively.

Furthermore, CMOS-based 6T SRAM CIM designs suffer from challenges such as more peripheral overhead, increased read/write unsteadiness, and yield degradation when scaled below 20-nm [13, 14]. While implementing large-scale CIM arrays for convolutional neural networks (CNNs), area inefficiency and control complexity become bottlenecks [15]. In addition, FinFET technology reduces off-state leakage and enhances energy efficiency, while aggressive scaling of nodes results in compact CIM implementation with enhanced electrostatic control and improved drive strength compared to planar CMOS [16, 17]. Developments in peripheral components, such as I/O buffers, play a crucial role in enhancing the efficiency of CIM systems. For example, an I/O buffer has been developed using a 16 nm FinFET CMOS process, addressing process variations and ensuring reliable high-speed data transfer [18]. Previous studies also showed that ultralow threshold voltage (ULVT) transistors significantly improve SRAM functionality under low voltage conditions [19]. For example, a 10-nm FinFET near-threshold SRAM demonstrates improved stability, reduced leakage, and robust operation in sub-threshold voltage [20]. Such innovations complement CIM architectures by improving overall throughput and energy efficiency in AI applications.

High-throughput MAC operations for both integer and floating-point (FP) workloads are made possible by SRAM-based CIMs, which has become a viable option for energy-efficient AI edge processing [21]. While ADC-free and transpose CIM architectures with flexible data-flow control offer unified inference and training within a single macro [22], recent FP-CIM designs support customizable precision such as BF16 and FP32 for low-power neural network training [21]. Additionally, pipelined computation and efficient data conversion greatly increase DCNN workload energy efficiency [23]. Nevertheless, the complexity of FP dataflow adds overheads, and decreasing the cost of exponent normalization and mantissa alignment remains critical for realistic FP-CIM implementations in AI edge devices [24].

In contrast, SRAM-based CIM based designs facing overhead challenges: a 22-nm charge domain FinFET design suffered from scalability and ADC-induced energy overhead which restricts its efficiency and robustness [25], a 6T SRAM CIM macro provides 8-bit precision through charge sharing and bit-line combining schemes but depends on ADCs which yields to area/power penalties [26]; and where as 8T SRAM CIM macros reach 50.1 TOPS/W yet remain suffered from analog hardware bottlenecks [27]. More recently, 7T SRAM CIM designs utilized dynamic matching quantization to avoid PVT variations, but their cell area increased [28]. Though the designs aid CNN workloads with arithmetic and logical operations achieving high energy efficiency and PVT insensitivity, the accuracy is moderate when scaled to advanced 12-nm FinFET technology [29]. Further advancing, a double bit 6T-SRAM floating point CIM has emerged for advanced AI tasks, to handle ShareFloatv2 data formats with enhanced efficiency and accuracy, but suffers from preprocessing overheads [30]. In our proposed design, the 8-bit input and weight processing is supported by a 4-kb SRAM array built on single-ended cells (S.E.), which improves energy efficiency. The architecture maintains functional flexibility for CNN workloads while achieving great energy efficiency by utilizing ULVT-based circuitry along with S.E. cells.

Building on these design innovations, the key contributions of this work are:

1. A single-ended 6T SRAM cell that enables in-memory arithmetic without differential bit lines.
2. A digitally friendly in-array multiplication and addition scheme with automatic write-back.
3. A 2T ULVT-assisted 2T switch that mitigates charge sharing in low-voltage operation.

4. Comprehensive post-layout evaluation under all PVT variations demonstrating competitive energy efficiency and precision-scaled performance.

2 Proposed CIM Architecture

The proposed 4-kb CIM architecture combines integrated computing with a 4-kb 6T single-ended SRAM, enabling arithmetic and logic operations to be carried out inside the memory array. The architecture reduces data flow between memory and external processors by taking advantage of natural bit-line charge sharing and accumulation, which lowers latency and energy consumption.

As shown in Fig. 1, the CIM comprises of SRAM array with In-Memory Computing, CIM Control Circuit, Auto Write-Back Circuit, SRAM Control Circuit, BIST, and Output Buffer. To implement column-parallel logic and arithmetic operations, two operand rows chosen by X_addr[5:0] and Y_addr[5:0] are concurrently engaged throughout CIM operation. While the bit-lines are precharged using prec[63:0], the associated word-lines WL[63:0] are asserted. Charge-based accumulation across all columns is made possible by broadcasting the stored data from the chosen rows onto the shared bit-lines.

The CIM Control Circuit configures the computation by choosing the type of operations, operand addresses (X_addr[5:0], Y_addr[5:0]), and destination addresses for result storage, as well as sum/product_addr[5:0] and carry/sign_addr[5:0]. The Auto Write-Back Circuit captures the computation outputs (sum[63:0], cout[63:0], prod[63:0], and sign[63:0]) and certainly writes them back into the SRAM array over the control signals WL_auto[5:0] and BL_auto[5:0], permitting in-place and iterative computation without external data movement. The SRAM Control Circuit controls both standard memory operations and CIM-specific functions, such as write enable (Awr_en, DFWR), bit-line precharge (prec[63:0]), word-line control (WL[63:0]), and address decoding (Aword_addr[5:0], Abit_addr[5:0]). The BIST module facilitates functional testing and retention verification with BIST_en and Retention signals, while the Output Buffer sends read data (Data_out) and test status (BIST_Pass) to external interfaces. The Output Buffer follows the design principles of our prior works to ensure reliable signal readout [31, 32].

Figure 2 illustrates the schematic diagram of operations in memory, using a 4×4 memory matrix as an example, integrates an SRAM array with In-Memory Computing consists of a 2-Transistor Switch Computing Module (2T-SCM) and a logic-and-arithmetic core (LAC) to enable computation directly within the memory array.

During each operation, the bit-lines are first precharged and one or more word-lines are activated, allowing the stored data (Qx(y) and Qbx(y) (x =0,1,2,3 and y= 0,1,2,3)) from selected rows to be broadcast to 2T-SCM. 2T-SCM computes CBx,CNORx, and CANDx based on the C(i) and S(i) values (i=0,1,2,3) and then the results are further processed in LAC . The LAC generates the final logic outputs (Cix, XORx, CXNORx, and CNANDx) and the arithmetic outputs (Sum, Cout, and Product). The architecture offers four parallel columns of operations per access through this closely connected memory–compute flow, greatly reducing data travel while increasing computing throughput and energy efficiency.

2.1 6T Single-ended SRAM Cell

As shown in Fig. 3, the 6T single-ended SRAM cell proposed in this study consists of six transistors: **Mp0**, **Mp1**, **Mp2**, **Mn0**, **Mn1**, and **Mn2**. Unlike traditional 6T-SRAM designs that use a cross-coupled inverter structure, this design utilizes a single inverter (composed of **Mp0** and **Mn0**) and

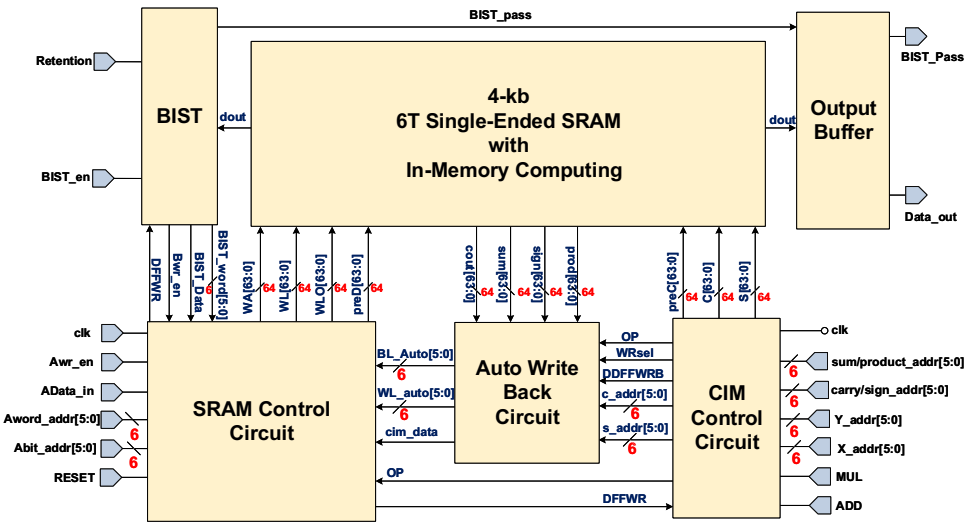


Figure 1 Proposed block diagram of the 4-kb CIM architecture.

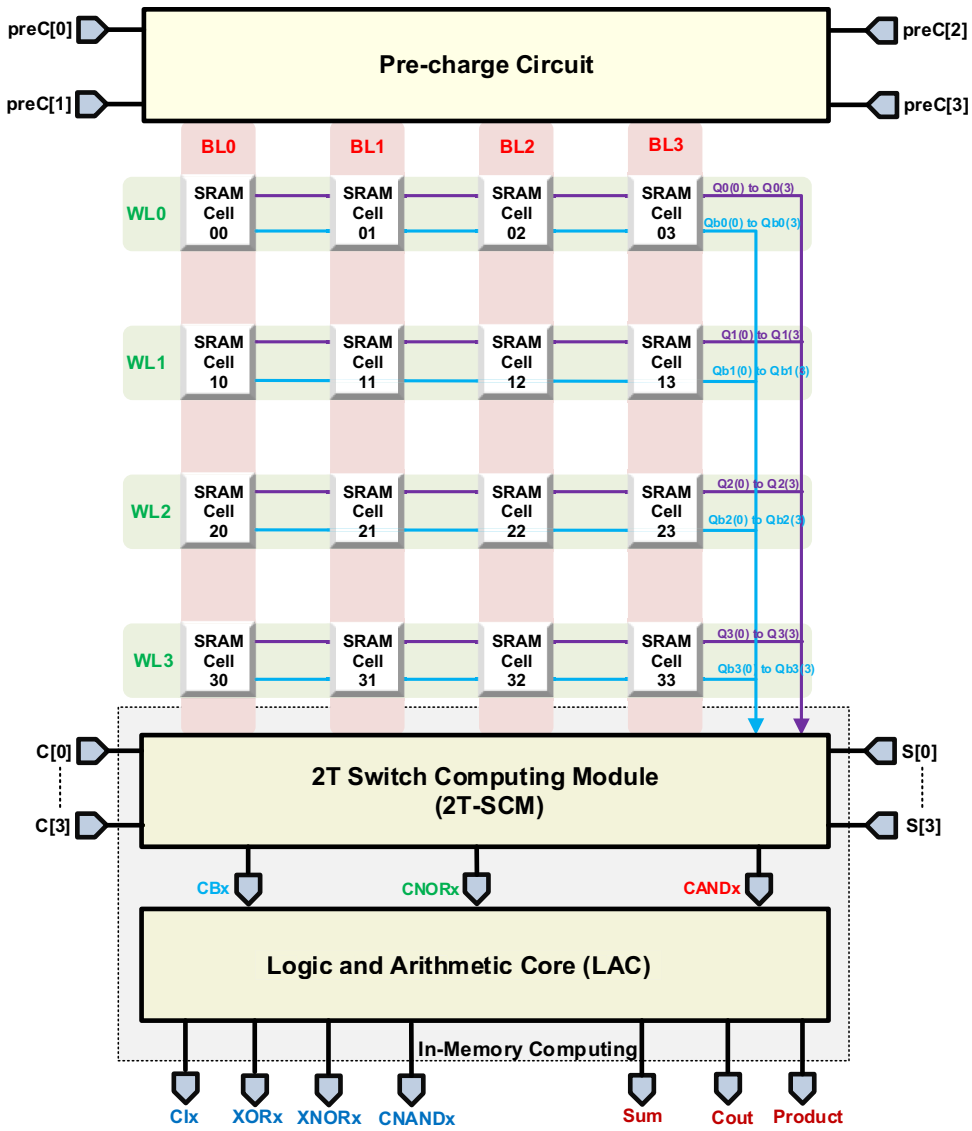


Figure 2 CIM schematic (4 × 4 illustration).

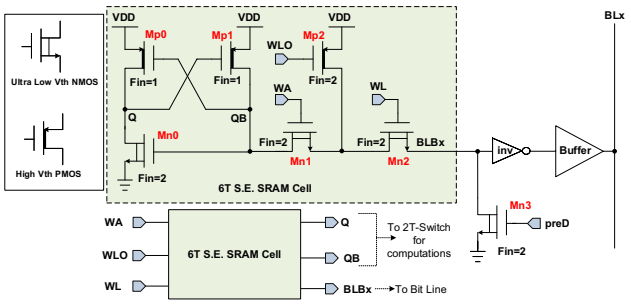


Figure 3 6T single-ended SRAM circuit architecture diagram.

an independent PMOS transistor (**Mp1**) as the main data storage element. To reduce the impact of leakage current on the stored values, **Mp0** and **Mp1** employ high critical voltage PMOS components, while **Mn1** uses extremely low critical voltage NMOS components to enhance the circuit's operating speed.

Additionally, SRAM reading and writing are controlled through the transistors **Mn1** and **Mn2**, in combination with **Mp2** and the precharge control transistor **Mn3**, which is shared by a column of SRAM cells. To reduce the layout area when selecting the critical voltage, **Mp2**, **Mn1**, and **Mn2** are designed to share the same N-type well, allowing them to use the same well voltage. To further enhance speed, **Mn1** and **Mn2** are implemented with components that feature an extremely low threshold voltage. The proposed single-ended architecture incorporates more transistors to enable computation in contrast to a conventional 6T SRAM; nevertheless, the sharing of sensing inverters and buffers at the column level reduces the effective area overhead.

Next, we will provide a detailed explanation of how the SRAM architecture implements the data writing, reading, and storage functions with the corresponding timing diagram in Fig. 4.

1. **Data Hold:** Data Hold relies primarily on three transistors: **Mp0**, **Mn0**, and **Mp1**. When the QB point has a low potential (representing data 0), **Mp0** is turned on to charge the Q point to a high potential (data 1), while **Mp1** is turned off. When the QB point has a high potential (representing data 1), **Mn0** is turned on, discharging the Q point to a low potential (data 0), while **Mp1** is turned on to maintain the QB point at a high potential. This process ensures that the stored data remains intact until it is modified.
2. **Write 0:** To write a data state 0, firstly set the WLO input to low, which turns on **Mp2**, and simultaneously set the write access (WA) input voltage to high to turn on **Mn1**. This causes the QB point to be charged to a high potential, and **Mn0** discharges the Q point to a low potential, effectively writing data 0 into the memory.
3. **Write 1:** To write data state 1, firstly set the preD input to high, turning on **Mn3**, and simultaneously set the write access (WA) terminal and the row select (WL) terminal to high, turning on **Mn1** and **Mn2**. This results in the QB point being discharged to a low potential, after which **Mp0** is turned on to charge the Q point to a high potential, thus completing the writing of data 1.
4. **Data Read:** To read data state from the SRAM, the write access (WA) terminal and the row select (WL) terminal are set to high to turn on **Mn1** and **Mn2**. **Mp2** and **Mn3** are turned off to allow the BLBx voltage to approach the QB point. Then, the inverter (inv) and Buffer ensure that the voltage on the BL line closely matches the voltage at the Q point. This process successfully reads the stored data value from the Q point.

2.2 2T Switch Computing Module (2T-SCM)

As shown in Fig. 5, the two-transistor (2T) switch computing module (SCM) eliminates the requirement for explicit logic gates by enabling logic evaluation through conditional charge transfer. The 2T switch consists of an access transistor driven by the select signal S[x] (or C[x]) and a data-dependent evaluation transistor constructed using ultra-low threshold voltage (ULVT) transistors U_{T1} and U_{T2}. The charge held at the internal SRAM storage node Q[x] or its complement Qb[x] determines the conductivity of the evaluation transistor. Q[x] or its complement Qb[x], where the logical state is encoded by the existence or lack of charge.

The precharge, evaluation, and write-back phases of the 2T-SCM are depicted in the timing diagram as shown in Fig. 5, along with how control signals (preC[x], C[x] and S[x]) coordinate the conditional charge transfer and output updates. The preC[x] signal is activated (low potential), charging the CB, CNOR and CAND lines to a high potential. The preC[x] signal is deactivated (high potential) to stop the charging process, the access transistor allows communication between a precharged compute node (CB or CNOR or CAND) and the storage node when the select signal is asserted. Charge redistribution happens during evaluation if the stored charge at the storage node biases the U_{T2} into

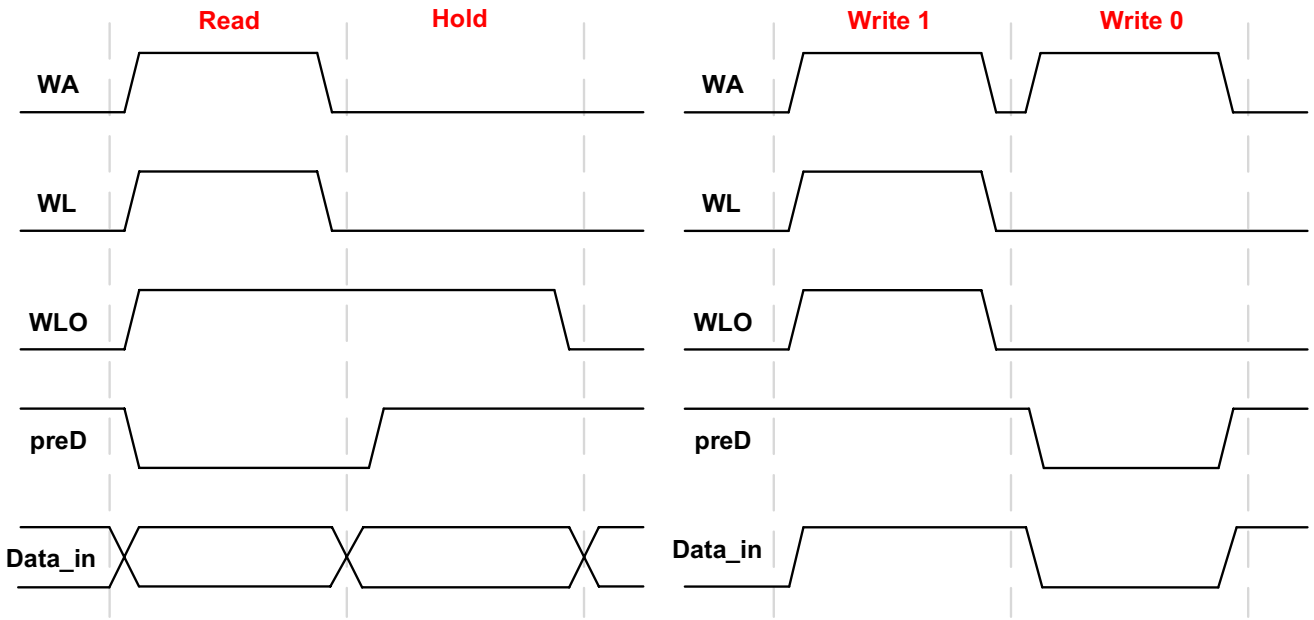


Figure 4 SRAM status control signal timing diagram.

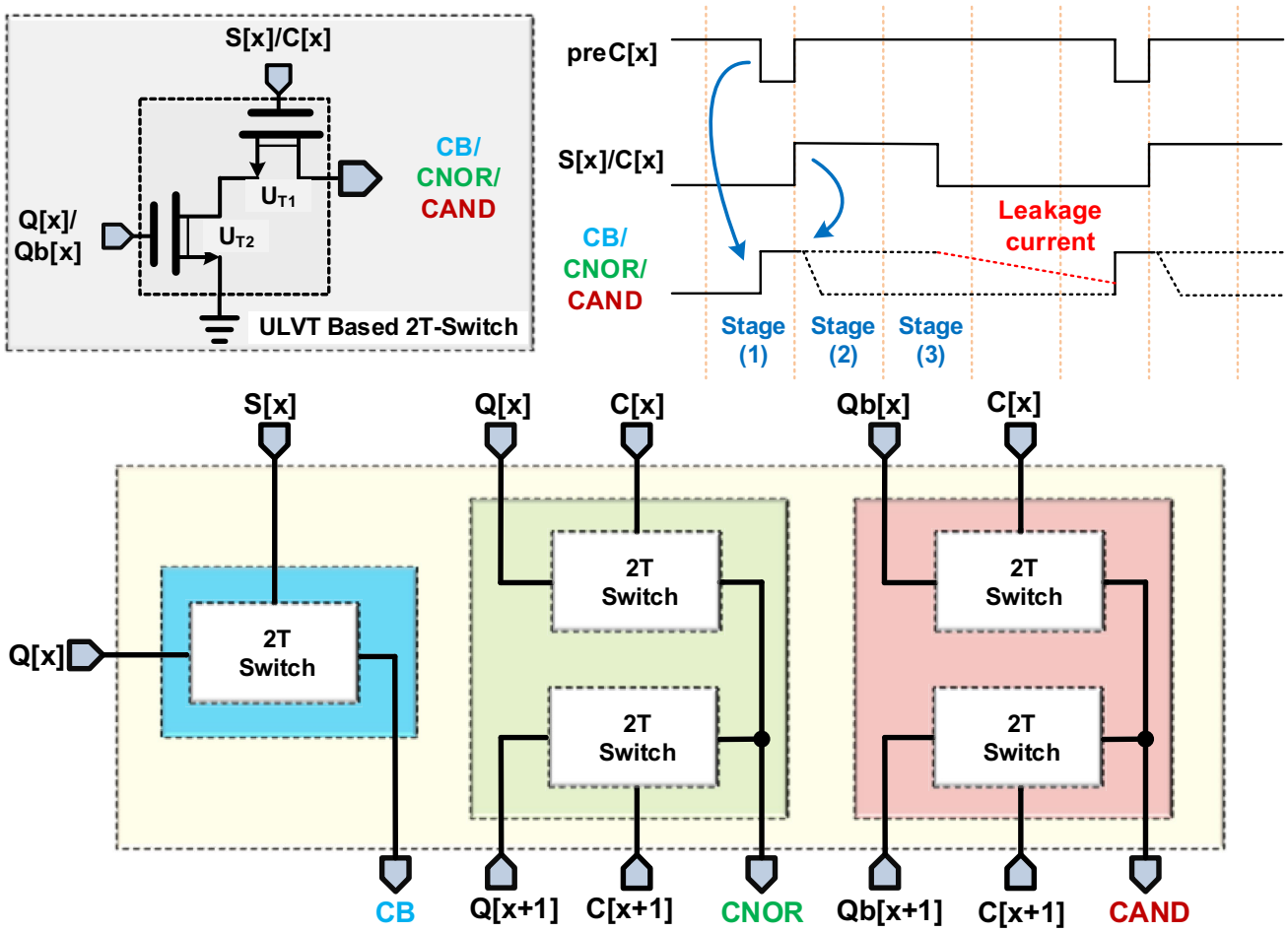


Figure 5 2T Switch computing module.

conduction, causing a voltage drop at the computation node. The ULVT improves evaluation sensitivity and robustness because of its lower threshold voltage, which guarantees dependable conduction even in low-voltage or weak-charge situations. The computation node keeps its precharged voltage if the evaluation transistor stays non-conductive. As a result, logic processing is implemented as a conditional discharge mechanism that is dependent on the SRAM cell's stored state. The various output states on the three computation lines for different logic setups are reported in Table 1.

2.3 Logic and Arithmetic Core (LAC)

The Logic and Arithmetic Core (LAC) expands the computation in-memory bitwise for arithmetic and logical operations as shown in Fig. 2. It eliminates frequent memory-to-processor data transfers by directly using computation signals produced by the 2T-SCM to logic outputs using Hybrid Logic Circuit and generate multi-bit arithmetic results using sum and carry generation circuits as shown in Fig. 6.

Table 1 Status of internal nodes and output lines for 2T-SCM operations.

Input			State		Output		
S[x+1]	C[x+1]	C[x]	Q[x]	Qb[x]	CB	CNOR	CAND
–	1	1	0	0	–	1	0
–	1	1	0	1	–	0	0
–	1	1	1	0	–	0	0
–	1	1	1	1	–	0	1
1	–	–	Q[x+1] = 0		1	–	–
1	–	–	Qb[x+1] = 1		0	–	–

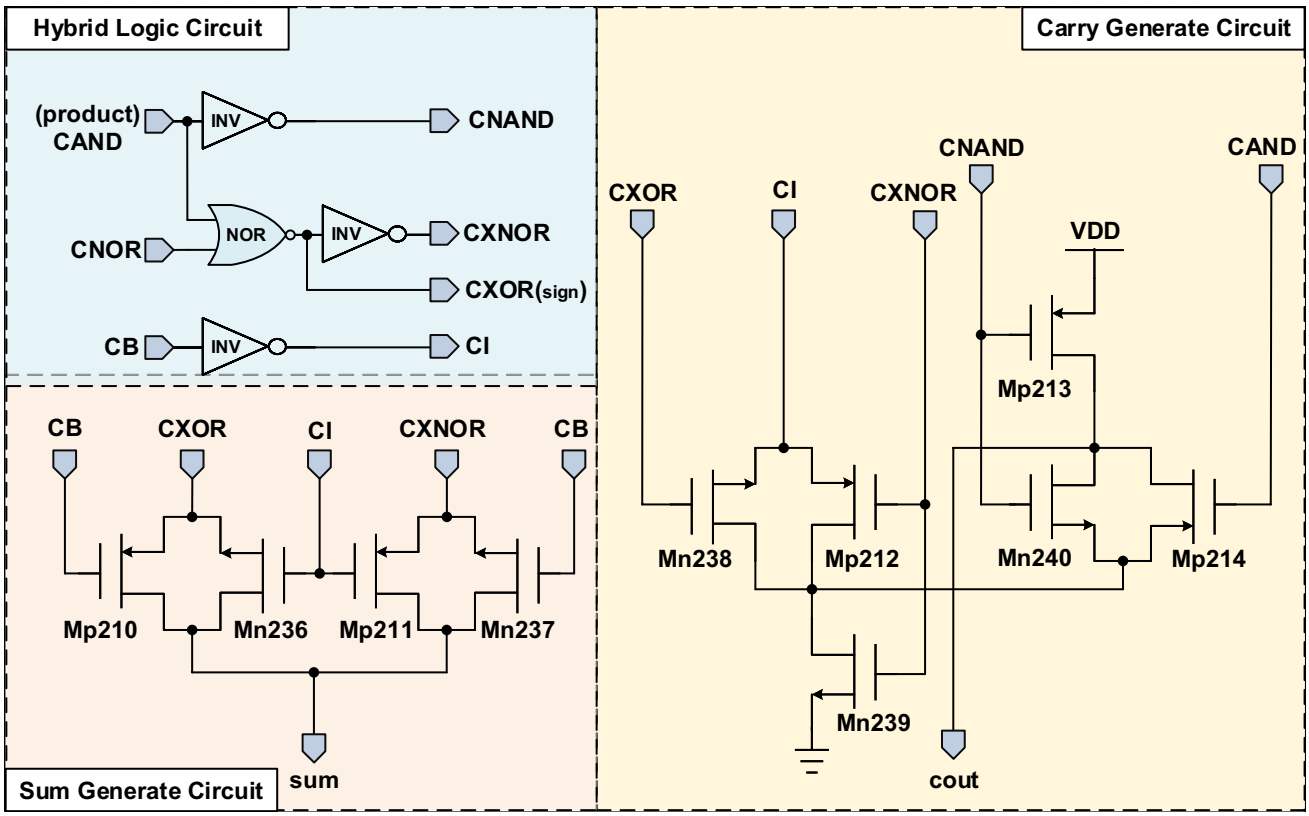


Figure 6 Logic and arithmetic core [11].

2.3.1 Logical Operations in the LAC

The bitwise relationship between operand bits from the 2T-SCM is encoded using CAND, CNOR, and the auxiliary signal CB. As seen in Fig. 6, the Hybrid Logic Circuit in the LAC uses these signals to resolve final logic outputs. XOR is produced by inverting CNOR, NAND by inverting CAND, and CXNOR is derived directly from CNOR. This makes it possible to perform bitwise XNOR, XOR, and NAND operations over several columns entirely in memory without moving any data.

2.3.2 Arithmetic Operations in the LAC

Addition As illustrated in Fig. 6, bit-level addition is carried out by cascading in-memory logic across columns. LAC signals are used to calculate the sum and carry (cout):

$$\text{sum} = \text{CXOR} \oplus \text{CI} \tag{1}$$

$$\text{cout} = \text{CAND} + (\text{CI} \cdot \text{CXOR}) \tag{2}$$

where CNOR and CB are the sources of CXOR, CAND, and CI. Carry propagation is managed by CAND, CNAND, and CNOR, whereas the sum-generation path conditionally chooses CXOR or CXNOR based on CI. Multi-bit addition moves from LSB to MSB while results are locally stored in SRAM cells.

Multiplication As seen in Fig. 7, multiplication is done totally in-memory using a shift-and-accumulate method. The multiplicand i is programmed onto WL0, with the sign bit in the MSB column and the magnitude bits stored from BL0 (LSB) to higher bitlines. To find the final product sign, the sign bits of both operands are sent to the sign-evaluation circuitry after the multiplier j is similarly loaded into WL1.

The 2T-SCM as illustrated in Fig. 5, evaluates the bitwise AND between the multiplicand and the current multiplier bit to produce partial products directly on the CAND lines for each active multiplier bit and the resultant partial products available on word line WL2.

$$\text{Partial Product}_{i,j} = \text{CAND}_{i,j} \tag{3}$$

The left-shift needed for multiplication is naturally implemented by storing each partial product in WL2, aligned to columns that correspond to the combined importance of the multiplicand and multiplier bits. Complementary CNAND lines are used to generate carry signals, which minimize

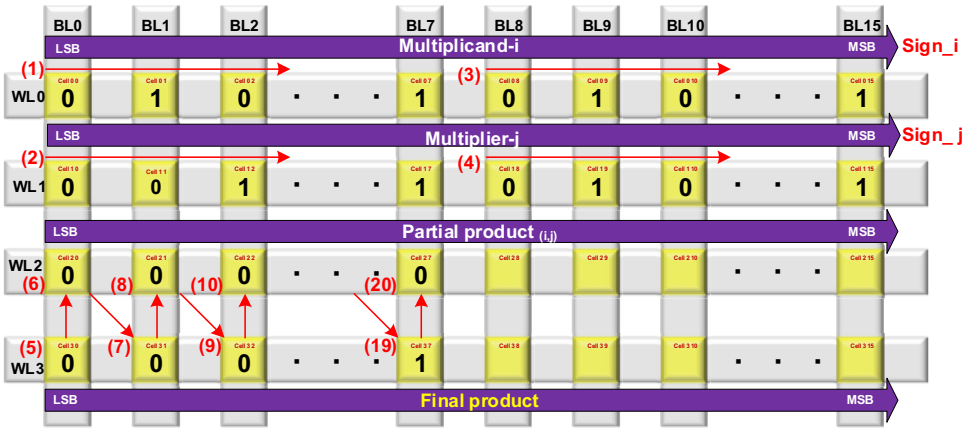


Figure 7 Data flow for the multiplication operation.

needless computation by remaining inactive when the multiplier bit is 0.

The starting value of the accumulation row (WL3) is set to zero. The LAC is used to add partial products from WL2 to WL3. As shown in the Fig. 6, CXOR/CXNOR logic is used to determine sums, while CNAND/CI is used to propagate carry. As highlighted in the Fig. 7:

1. Labels (5) and (6) indicate the generation of partial products on WL2 for the first active multiplier bit.
2. Labels (7) and (8) show the initialization of the accumulation row (WL3) to zero.
3. Labels (9) and (10) illustrate the addition of partial products from WL2 into WL3, including horizontal carry propagation and sum resolution.

This shift-and-accumulate procedure is repeated for every multiplier bit, until the final product is fully stored in WL3, accumulation occurs, with each new partial product being aligned to higher-order bitlines based on its relevance. For signed multiplication, the product sign is determined as:

$$\text{Sign_i} \oplus \text{Sign_j} = 1 \tag{4}$$

A single CI is applied during the accumulation of an inverted row to complete two's-complement correction, guaranteeing accurate sign extension, after certain partial products are conditionally inverted using the CB signal.

The final signed product, which consists of sum bits from BL0 to BL15 plus the evaluated sign, is fully stored in the memory array after all partial products have been accumulated, with the most significant carry being kept in the MSB column are avaible on word line WL3 as shown in Fig. 7. All in-memory multiplication is done locally, without transferring any data between the memory and any processor.

2.4 Auto Write Back circuit

By guaranteeing that the results of in-memory operations, including addition (ADD) and multiplication (MUL), are accurately recorded within the SRAM array, the Auto Write Back circuit is a crucial part that supports the Logic and Arithmetic Core (LAC). Its architecture is shown in Fig. 8, composed of three primary blocks: CIM Data Selection Circuit, Word Address Auto-generate Circuit and Bit Address Auto-generate Circuit. For the LAC results, the Bit Address Auto-generate Circuit offers automatic column selection. It uses WRsel, and DDFFWRB signals from CIM Control circuit as inputs to a counter, generating 6-bit outputs COUNT[5:0] and MUL_COUNT[5:0]. Depending on whether an addition or multiplication operation is carried out, a selector either selects COUNT[5:0] or MUL_COUNT[5:0], resulting in BL_auto[5:0], the automatic bit-line address for write-back.

The Word Address Auto-generate Circuit handles row-wise addressing for LAC results. It employs a multiplexer controlled by OP and CIM_Datasel to choose the proper word-line address WL_auto[5:0] after receiving DDFFWRB, c_addr[5:0], and s_addr[5:0] from CIM Control circuit. This ensures that each LAC operation result is written to the right memory row.

Lastly, the outputs of LAC activities are integrated by CIM Data Selection Circuit. Based on CIM_Data signals, it employs multiplexers to send either the product and sign from multiplication or the sum and carry-out from addition. The selected output creates CIM_Data, which is subsequently rewritten into the memory array, completing the in-memory calculation cycle and enabling seamless functioning of the LAC without external data movement.

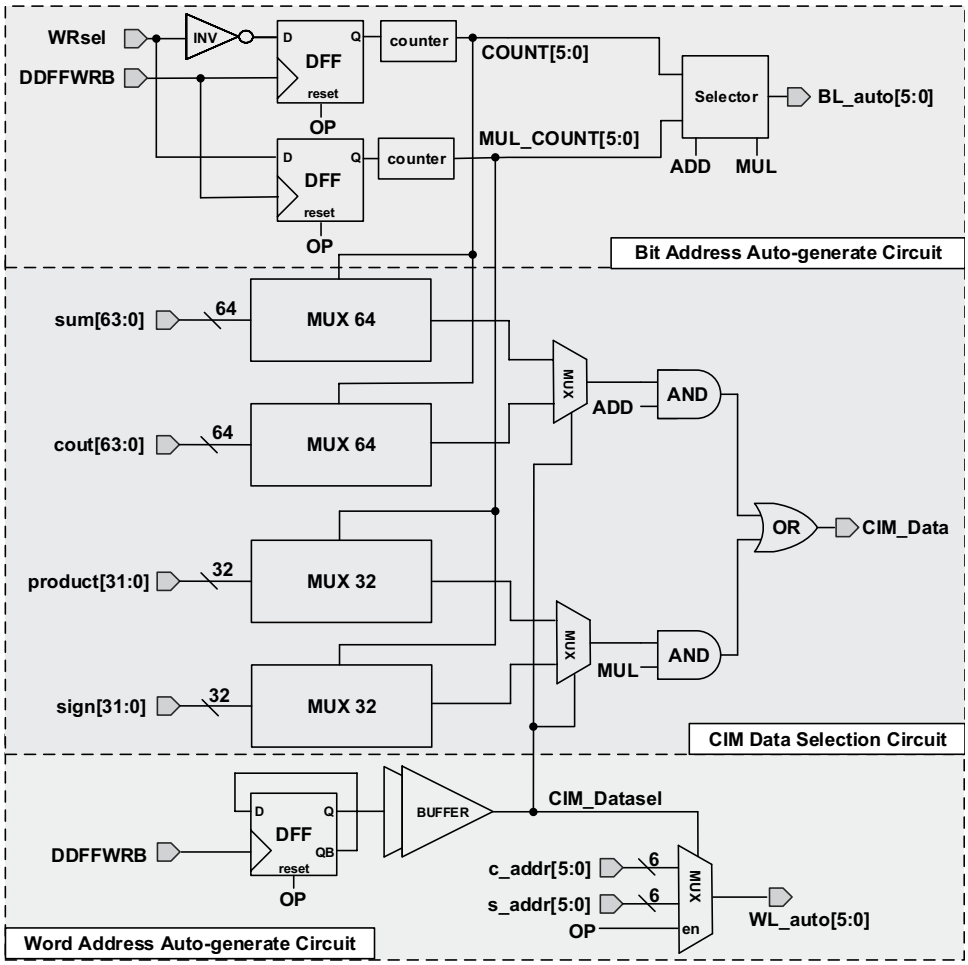


Figure 8 Auto Write Back circuit architecture diagram.

2.5 CIM Control circuit

Figure 9 shows the architecture of CIM Control Circuit as referenced in Fig. 1, comprising of CIM Timing Control Circuit and 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 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 circuit operation timing diagram is shown in Fig. 10, which details the time relationship and control logic of each signal.

The CIM Address Control Circuit generates control signals based on user-defined parameters to enable precise in-memory operations. It circuit generates the precharge signal pre C[x] and C[x] by using the user-defined operand addresses X_addr[5:0] and Y_addr[5:0], which are provided to the 2T-SCM to carry out CNOR and CAND operations.

Additionally, this circuit generates the carry bit address selection signal S[x] based on the user-defined address cout/sign_addr[5:0]. The operation timing diagram of this circuit is shown in Fig. 11. When the ADD or MUL signal is activated during the read phase, the precharge circuit is triggered, followed by the activation of the carry bit address selection signal S[x] and the operation bit address selection signal C[x]. During the subsequent write phase, the result is accurately written back to memory through the Auto Write Back circuit in coordination with the SRAM control circuit.

2.6 SRAM Control Circuit

The SRAM Control Circuit in Fig. 1 consists of two primary sub-circuits: SRAM Input Selection Circuit and SRAM

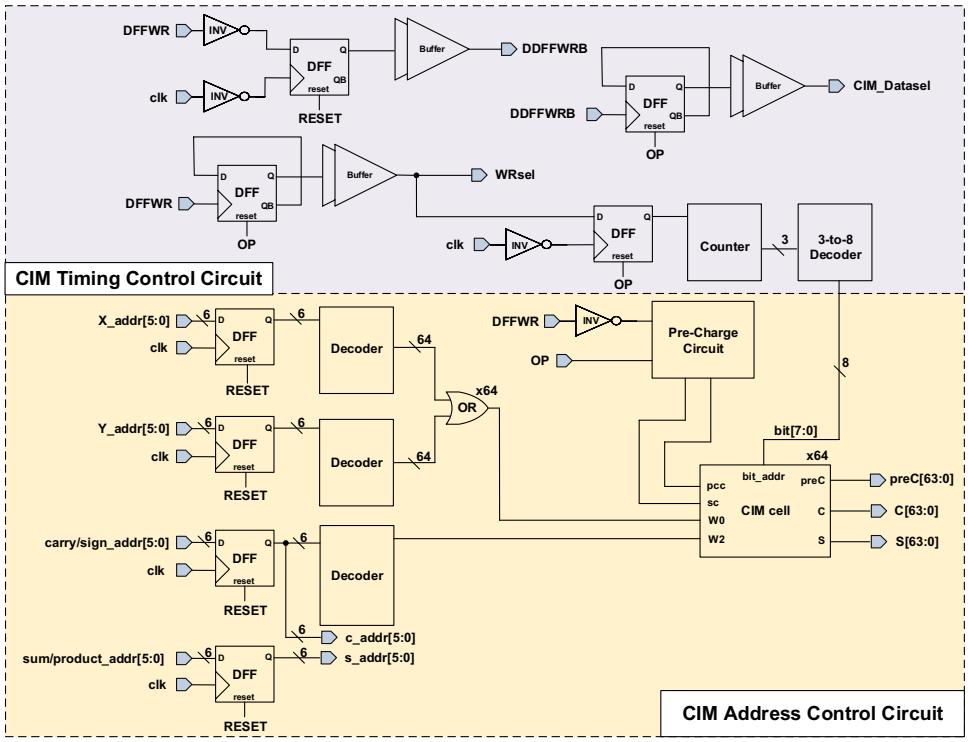


Figure 9 CIM Control circuit architecture diagram.

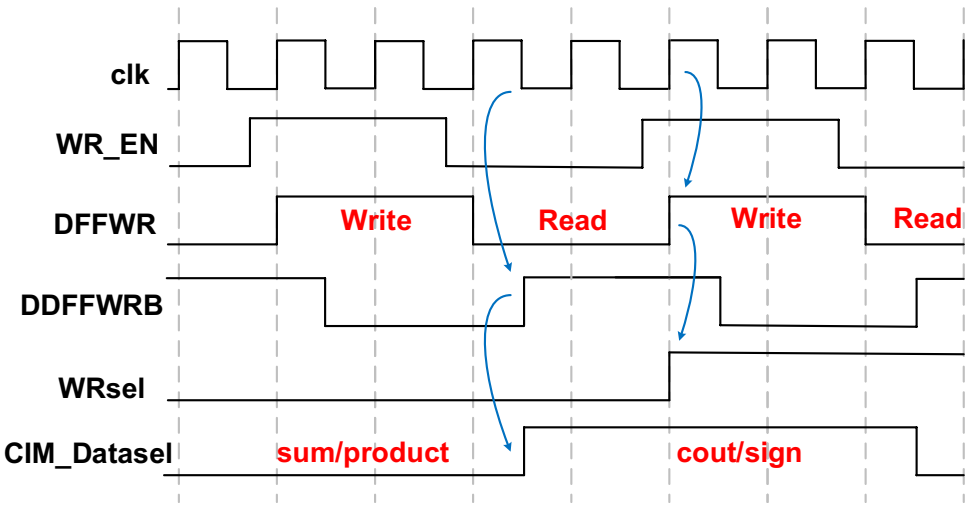


Figure 10 Timing diagram of CIM Timing Control Circuit.

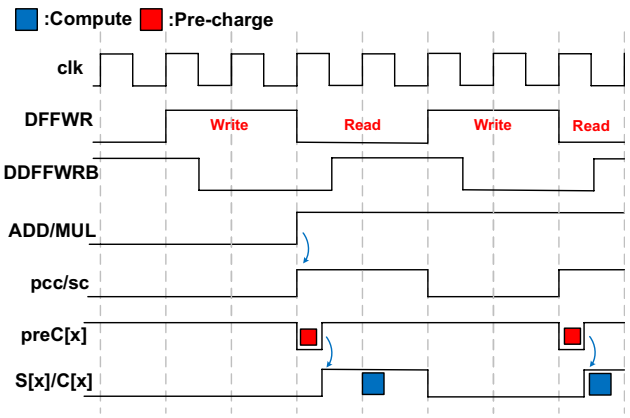


Figure 11 Timing diagram of CIM Address Control Circuit.

Execution Unit, as shown in Figs. 12 and 13, 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.

2.6.1 SRAM Input Selection Circuit

As shown in Fig. 12, the SRAM Input Selection Circuit is primarily composed of three parts: Data Select Circuit, Write Select Circuit, and Address Select Circuit.

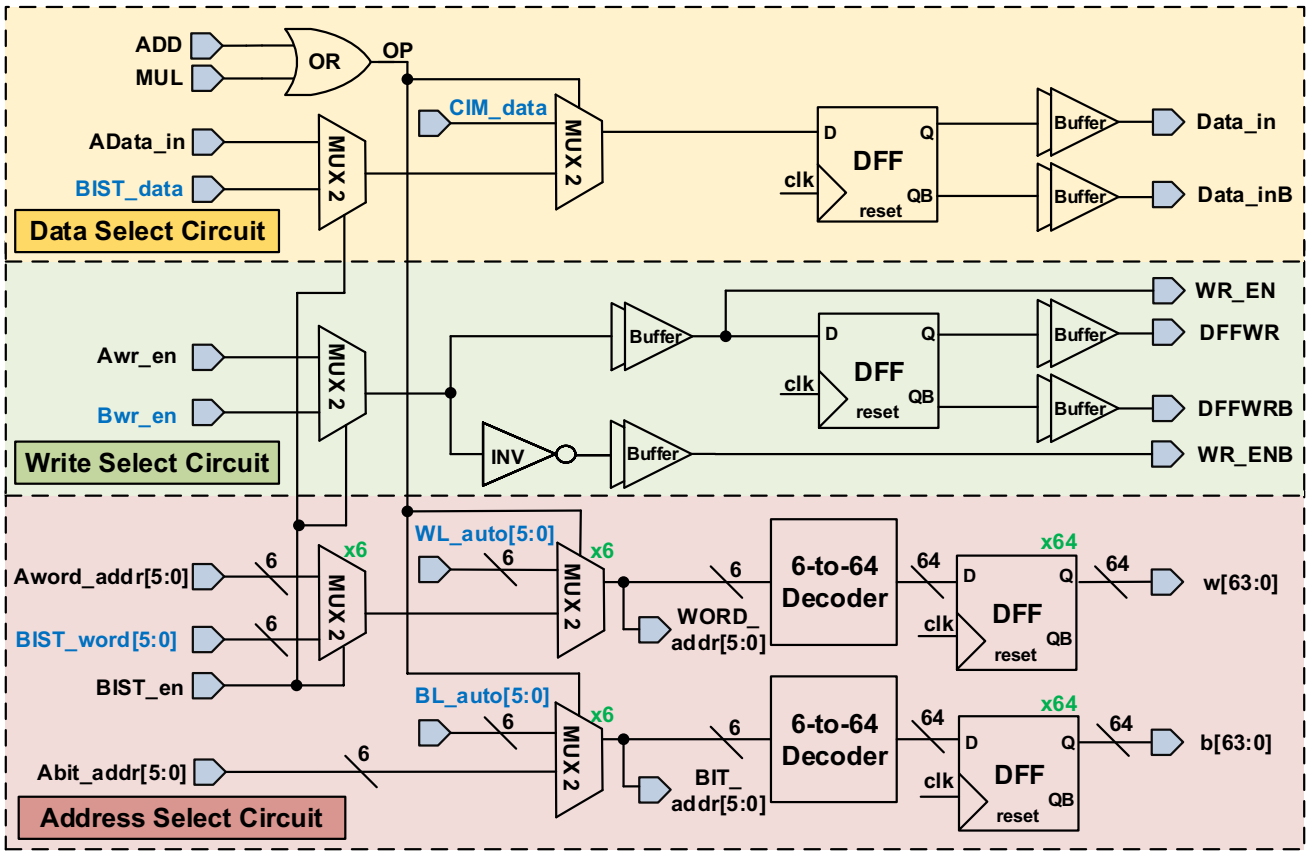


Figure 12 SRAM input selection circuit.

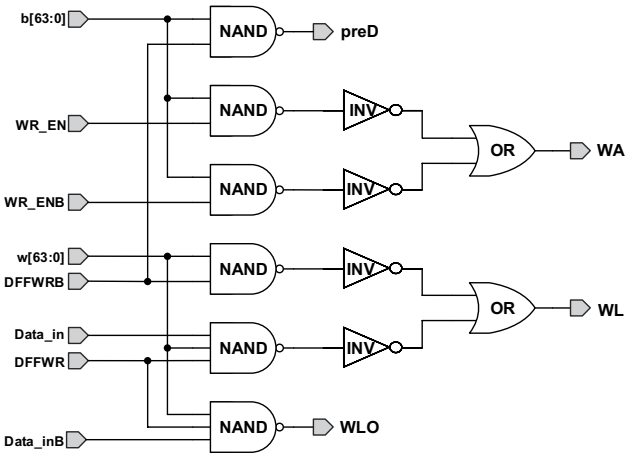


Figure 13 SRAM execution unit circuit.

Firstly, the Data Select Circuit is primarily responsible for selecting the source of input data for the SRAM. The sources of data include:

1. AData_in: The externally supplied signal used as Data X, Data Y, Sign X, and Sign Y.
2. BIST_data: The data generated by the BIST Circuit.
3. CIM_data: The result of in-memory operations such as Sum, Carry, Product and Sign. It is judged by the different MUXs.

2.6.2 SRAM Execution Unit

The schematic diagram of SRAM Execution Unit is shown in Fig. 13. The primary function of this unit is to generate the necessary control signals, such as WL, WLO, WA, and preD, which are essential for executing the read, write, hold operations on an SRAM cell, as detailed in Section 2.1. These control signals for a single SRAM cell are generated based on the bit address (b[63:0]) and word address (w[63:0]) signals, which are produced by the SRAM Input Selection Circuit.

3 Implementation and Verification

As illustrated in Fig. 14, the proposed 4-kb CIM based on single-ended SRAM cells is designed utilizing the TSMC 16-nm CMOS Logic FinFET (TN16FFC 1P9M) process. With a core circuit area of $520.4 \times 643.3\ \mu m^2$, the overall area of the chip is $908.1 \times 985.9\ \mu m^2$. The single-ended

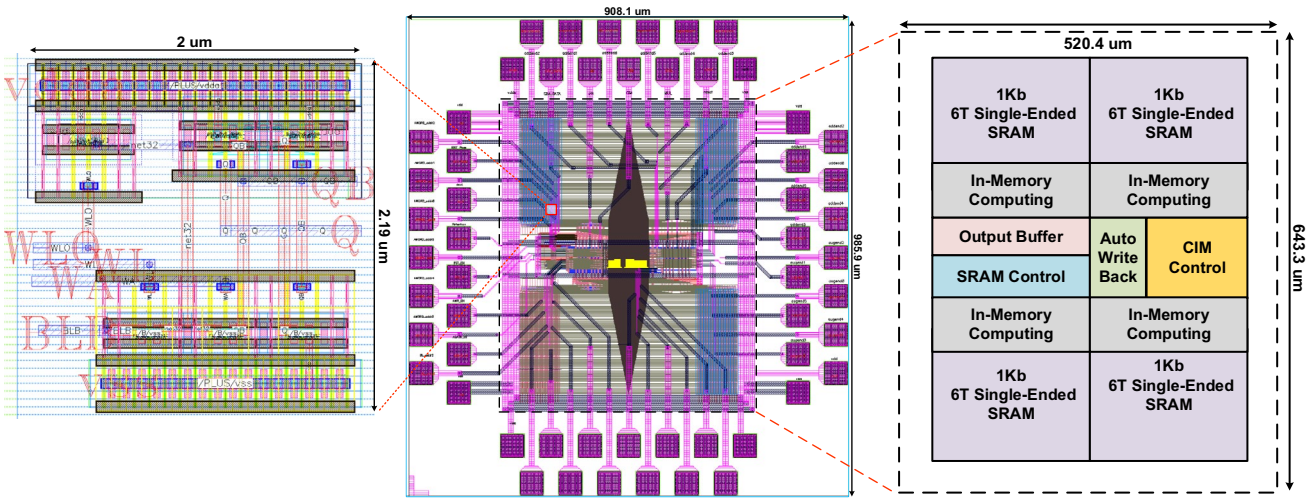


Figure 14 Proposed 4-kb CIM layout and floor plan.

6T SRAM cell's architecture takes up $2 \times 2.19\ \mu m^2$. The proposed 4-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 power consumption of the proposed 4-kb CIM is 3.69 mW, when the voltage supply (VDD) is 0.8 V and the system clock is set to 500 MHz.

In the layout design, the Clock Tree Layout method based on Delay Matching is used. The 4-kb SRAM is equally divided into four banks, each distributed at a corner of the chip, while the control circuit is positioned at the center. This strategic layout minimizes the delay difference in transmitting control signals to the memory array, ensuring the logic circuit operates correctly and maintaining synchronization during operations.

3.1 SRAM Cell Simulation Results

The simulation results of the SRAM under read and write operations in all PVT corners are shown in Fig. 15, including voltages of 0.72 V, 0.8 V, and 0.88 V; process corners of SS, SF, TT, FS, and FF; and temperatures of 0 °C, 25 °C, and 80 °C. An essential metric to assess the stability of SRAM cells is the Static Noise Margin (SNM). The "butterfly diagram" obtained by the Voltage Transfer Characteristics (VTC) of two inverters is typically used to determine SNM. The SNM value is the largest square side length that fits inside the curve.

However, the SRAM cell in this design uses a single-sided inverter with a PMOS as the storage structure, making it impossible to directly draw a butterfly diagram to estimate

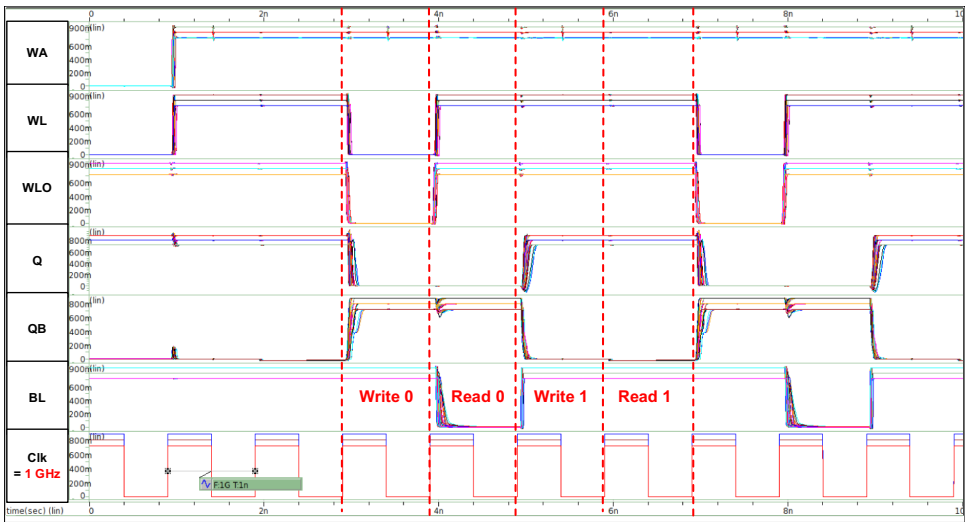


Figure 15 All-PVT-corner simulations of SRAM cell read and write functions.

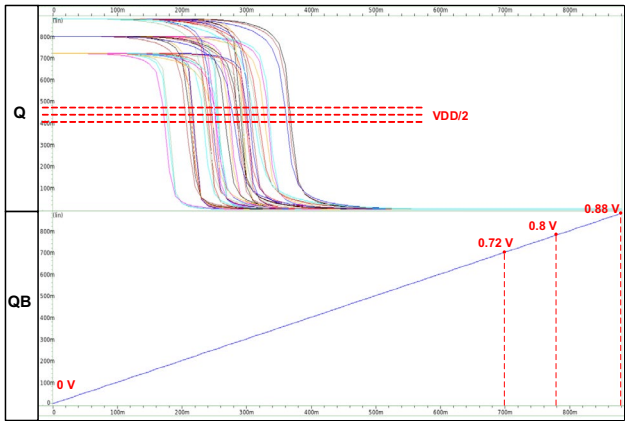


Figure 16 SNM simulations for the 6T single-ended SRAM cell.

the SNM value. Therefore, we use the VTC of the inverter as an alternative evaluation criterion for noise immunity. The simulation method involves scanning the voltage of the QB node from 0 volts to VDD and observing when the Q node's voltage conversion curve falls below half of VDD. The corresponding QB node voltage value is then used as the SNM evaluation indicator for the SRAM design. The simulation results for SNM are shown in Fig. 16, and the values for each PVT corner are shown in Fig. 17. The worst SNM value of 174.1 mV occurs when the PVT corner is FS, 720 mV at 80 °C.

The dynamic noise margin (DNM) of the SRAM cell is shown in Fig. 18. With a pulse width of 60 ps and a pulse voltage of 0.3 V, the DNM shows that it can withstand any noise as long as the VDD is 0.8 V at an operating clock frequency of 1 GHz to evaluate its performance under extreme conditions. The purpose of these simulations is to confirm the stability and dependability of the SRAM circuit in a range of conditions and operating parameters.

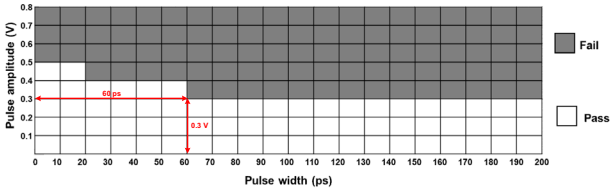


Figure 18 DNM simulations for the 6T single-ended SRAM cell.

3.2 Functional Simulations of the Proposed CIM

Multiplication and addition cannot be performed simultaneously by the CIM that is being given. Figures 19 and 20 display the post-layout simulation results of the entire CIM circuit, carried out under simulation conditions of a system voltage VDD = 0.8 V, the corner of the TT process, and a temperature of 25 °C. The addition and multiplication results are displayed in Figs. 19 and 20, respectively, for user-defined inputs applied serially on the input pin AData_in, first operand X=[11101010], sign of the first operand sign X=[01010101], and second operand Y=[10111100] and sign of the second operand sign Y=[01010101] at a clock frequency of 500 MHz.

Figure 21 shows the random data test results under the conditions of VDD = 0.8 V, FF process corner, and a temperature of 0 °C. Figure 22 shows the random data test results under the conditions of VDD = 0.8 V, SS process corner, and a temperature of 80 °C. The simulation results in Figs. 21 and 22 demonstrate that the CIM system can maintain normal operation and achieve correct computational functionality under varying process corners, temperature environments, and random data input conditions.

Several assessment measures that capture energy, area, and efficiency are necessary for a thorough comparison of

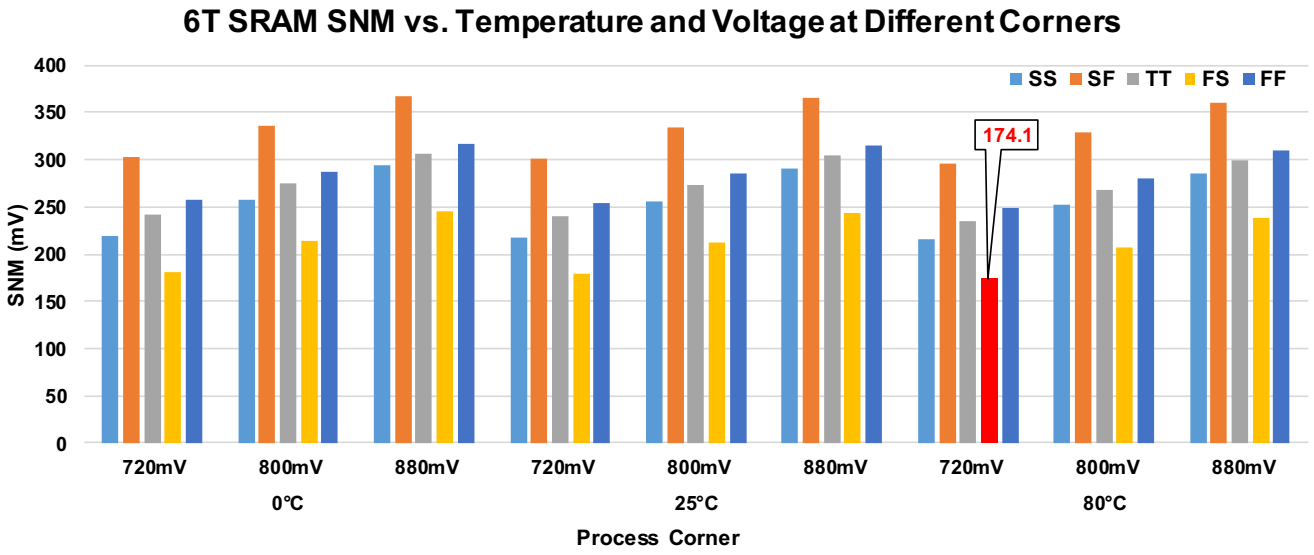


Figure 17 All-PVT-corner simulations of SRAM cell.

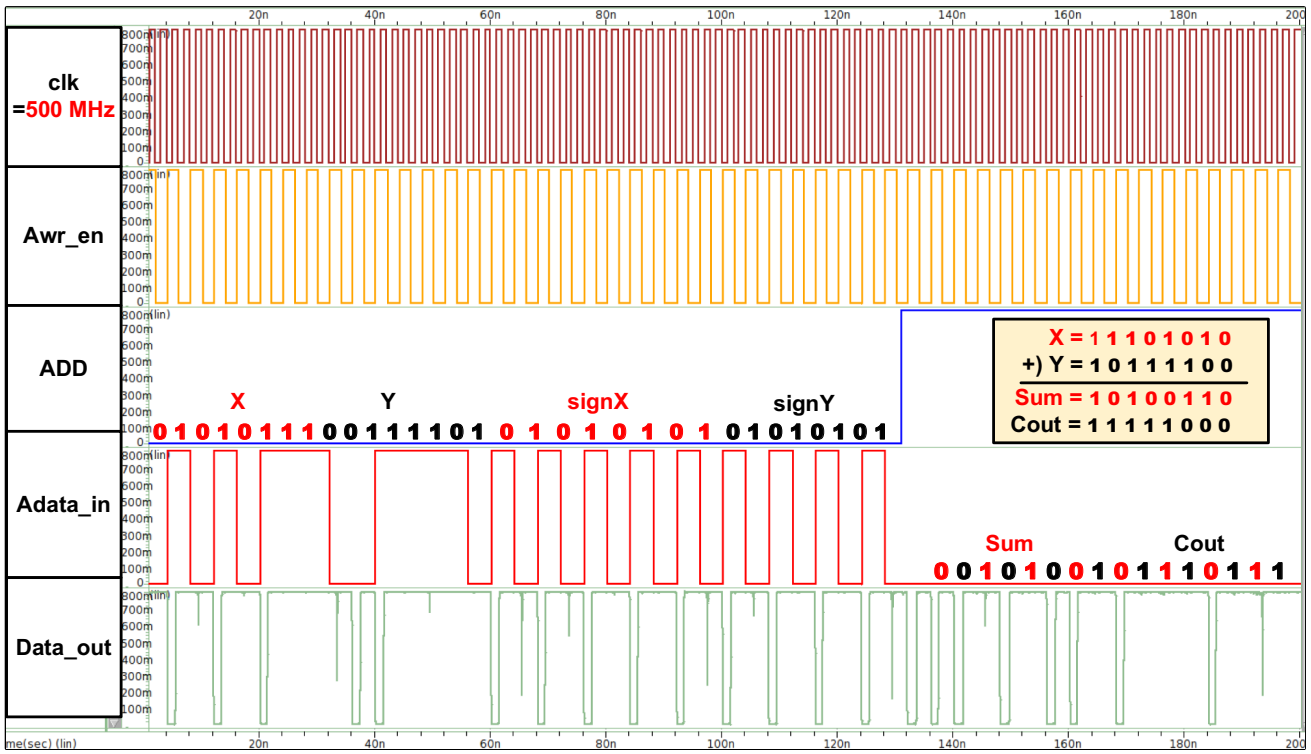


Figure 19 Post-layout simulation results of the addition operation for user defined inputs.

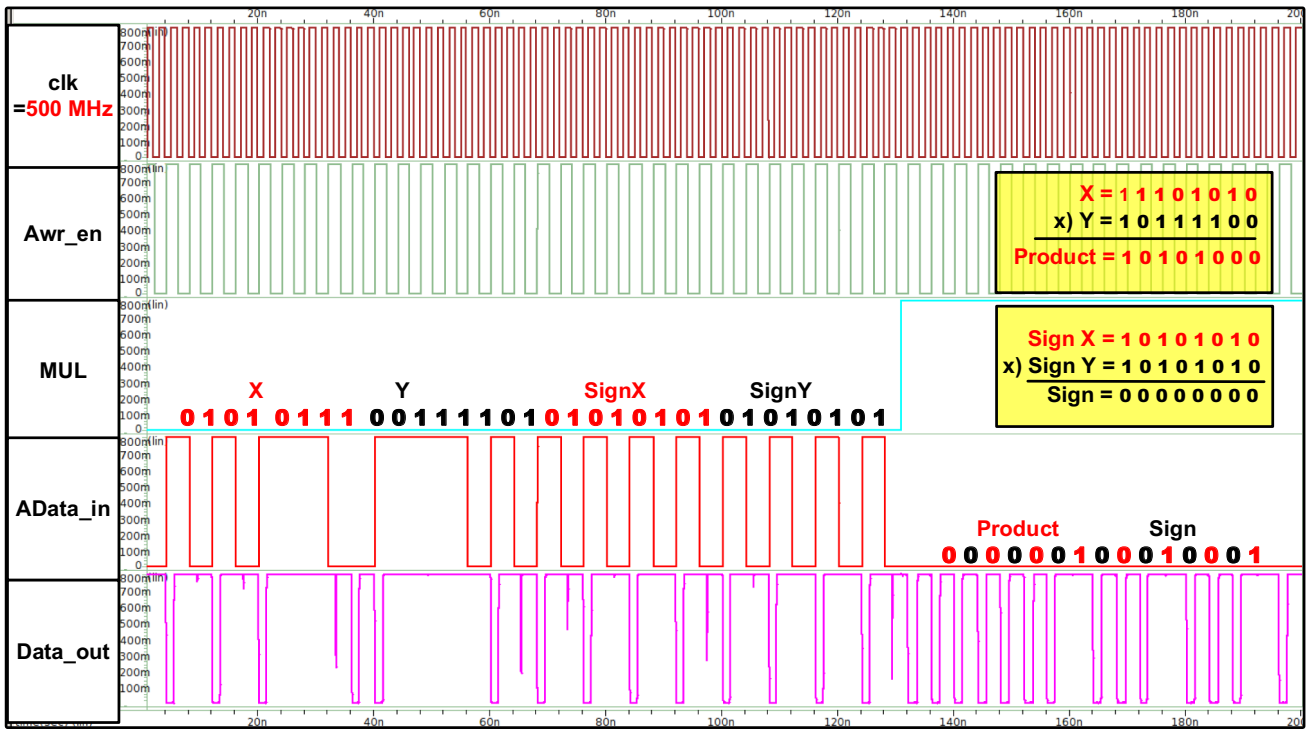


Figure 20 Post-layout simulation results of the multiplication operation for user defined inputs.

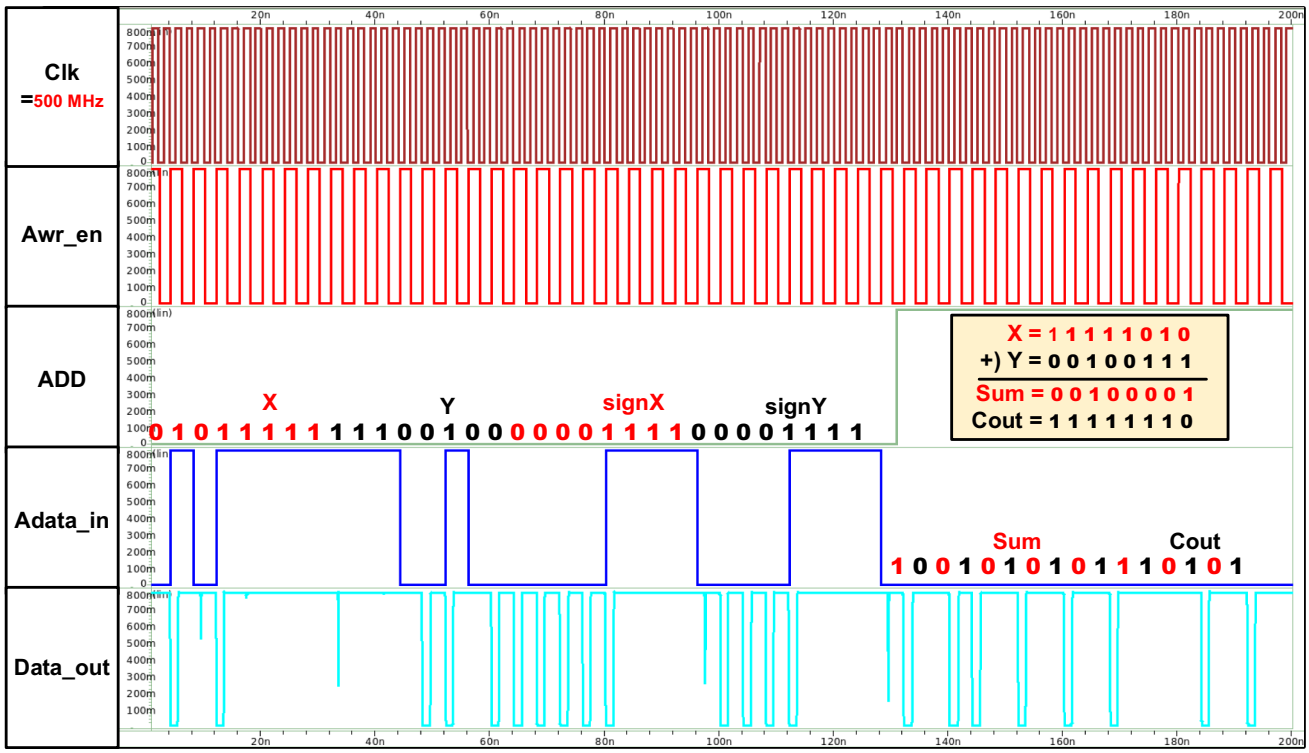


Figure 21 Post-layout simulation results of the CIM for random data inputs-first set.

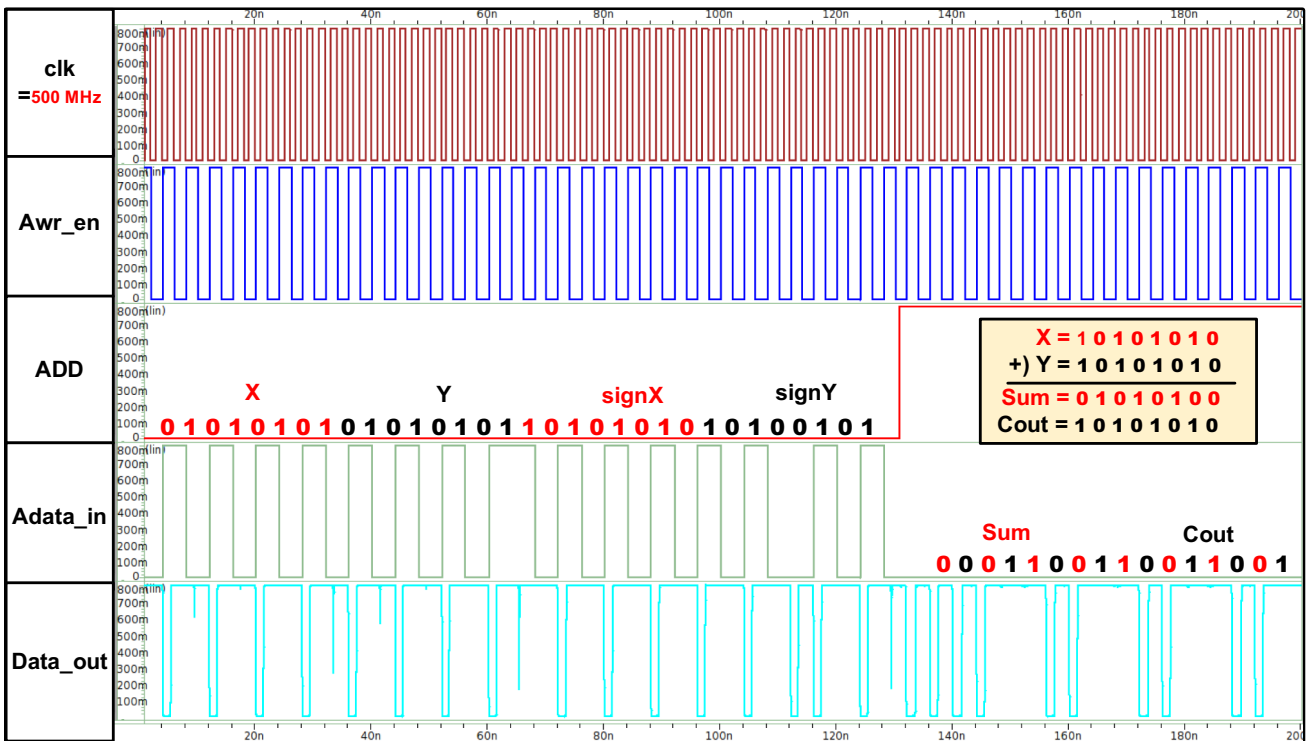


Figure 22 Post-layout simulation results of the CIM for random data inputs-second set.

SRAM-based CIM designs. The following definitions apply to the important performance measurements utilized in this study:

1. Energy Efficiency: Considers the impact of power consumption on computing performance.
2. Bitwise Energy Efficiency: Evaluates the relationship between the input and weight effective bits and power consumption. The calculation formula us given by Eq. 5.

$$\text{Bitwise Energy Efficiency} = \frac{\text{TOPS}}{\text{W}} \times \text{Input Bits} \times \text{Weight Bits} \tag{5}$$

3. Area efficiency: Evaluates the computational power relative to the chip area.
4. Bitwise Area Efficiency: Provides a comprehensive analysis based on the input effective bits and chip area. The calculation formula is given by Eq. 6.

$$\text{Bitwise Area Efficiency} = \frac{\text{TOPS}}{\text{mm}^2} \times \text{Input Bits} \times \text{Weight Bits} \tag{6}$$

5. The Energy FOM: By normalizing bitwise energy efficiency with the process node and supply voltage, as specified in Eq. 7, this metric quantifies energy efficiency and allows fair comparisons across technologies.

$$\text{FOM}_{\text{Energy}} = \frac{\text{Bitwise Energy Efficiency}}{\text{Process Node (nm)} \times \text{Supply Voltage (V)}} \tag{7}$$

Table 2 compares our suggested design with earlier SRAM-based CIM architectures using these metrics. Our design achieves an energy efficiency of 75.47 TOPS/W, which is much better than previous generations and surpasses earlier works [21–23]. It is also competitive with the current design (71.82 TOPS/W) [28].

Our design outperforms all previous designs, including [21] and [23], in terms of bitwise energy efficiency, reaching 4830.08 TOPS×bits^2/W, demonstrating greater energy efficiency at high precision levels. The bitwise area efficiency (8.98 TOPS×bits^2/mm^2) is the greatest of all studied works, showing a favorable precision-to-area ratio, while the area efficiency (140.25 GOPS/mm^2) demonstrates a slight trade-off in layout density.

Finally, our design achieves the Energy Figure of Merit ($\text{FOM}_{\text{Energy}}$) of 377.35 TOPS×bits^2/(W V nm), significantly higher than all prior SRAM-based CIM efforts, demonstrating superior normalized energy efficiency across process nodes and supply voltages. The evolution of energy efficiency trends across different designs is illustrated in Fig. 23, highlighting the advancements in SRAM-based CIM architectures over time.

Table 2 Comparison of prior SRAM-based CIM designs.

Parameter	[21]	[22]	[23]	[28]	Our work
Year	2023	2024	2025	2025	2026
Process Node (nm)	40	28	40	28	16
Supply Voltage (V)	0.7–1.1	0.7–0.9	1	0.9	0.8
Verification	Simu	Simu	Simu	Simu	Simu
SRAM Cell	6T	6T + ISC	8T + LCC	7T	6T(SE)
Array Size (kB)	12	32	16	16	4
Input Bits (bits)	8	4	8	4	8
Weight Bits (bits)	8	4	8	4	8
Energy Efficiency (TOPS/W)	1.97	62.90	33.10	71.82	**75.47**
Bitwise Energy Efficiency (TOPS×bits^2/W)	126.08	1006.40	2118.40	1149.12	**4830.08**
Area Efficiency (GOPS/mm^2)	117.00	NA	116.36	NA	**140.25**
Bitwise Area Efficiency (TOPS×bits^2/mm^2)	7.49	NA	7.45	NA	**8.98**
$\text{FOM}_{\text{Energy}}$ (TOPS×bits^2/ (W V nm))	2.87	44.93	52.96	45.60	**377.35**

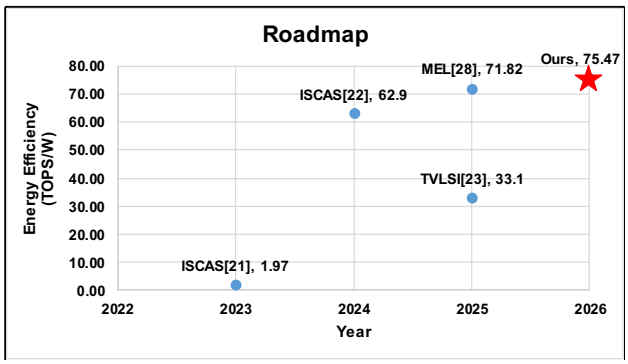


Figure 23 Energy Efficiency roadmap of SRAM-based CIMs.

4 Conclusion

This work focused on design of a high-efficiency computing-in-memory (CIM) based on an innovative single-ended 4-kb SRAM array has been presented and realized using a 16-nm 1P9M FinFET process.The integration of ultra-low threshold voltage (ULVT) transistors and the proposed 6T SRAM cell minimize circuit latency and energy consumption, allowing for effective support of convolutional neural network (CNN) functions such as addition, signed multiplication, and logical operations.

The design offers a competitive energy efficiency of 75.47 TOPS/W and a superior bitwise energy efficiency with moderate area efficiency of 140.25 GOPS/mm^2 at a maximum

operating frequency of 500 MHz. Additionally, the proposed design achieves a remarkable $\mathrm{FOM}_{\mathrm{Energy}}$ of 377.35 TOPS bits^2/(W V nm), demonstrating its exceptional normalized energy efficiency over supply voltage scaling and process nodes.Overall, the design achieves foremost-edge efficiency within a dense and advanced framework.

Acknowledgements This work has been partially funded by the National Science and Technology Council, Taiwan, subject to grants NSTC 112-2923-E-006-003-MY3, NSTC 115-2811-E-110-001, NSTC 112-2221-E-110-063-MY3 and NSTC 114-2923-E-110-001-. Taiwan Semiconductor Research Institute, National Institutes of Applied Research is acknowledged for the assistance of EDA tool support.

Author Contributions Author 1, 2, 5 wrote the main manuscript text. Author 3 and 4 provided the design specs. Author 6 takes charge of the project.

Data Availability No datasets were generated or analysed during the current study.

Declarations

Competing interests The authors declare no competing interests.

References

1. Patterson, D. A., & Hennessy, J. L. (2017). *Computer Architecture: A Quantitative Approach* (6th ed.). San Francisco: Morgan Kaufmann.
2. Santoro, G., Turvani, G., & Graziano, M. (2019). New logic-in-memory paradigms: An architectural and technological perspective. *Micromachines., 10*(6), 368. https://doi.org/10.3390/mi10060368
3. Gao, D., Reis, D., Hu, X.S., & Zhuo, C. (2020). Eva-cim: A system-level performance and energy evaluation framework for computing-in-memory architectures. *IEEE Trans. on Computer-Aided Design of Integrated Circuits and Syst.,39*(12), 5011–5024. https://doi.org/10.1109/TCAD.2020.2966484
4. Mutlu, O., Ghose, S., Gómez-Luna, J., & Ausavarungnirun, R. (2019). Invited: Enabling practical processing in and near memory for data-intensive computing. In *Proceedings of the 56th ACM/IEEE Design Automation Conference (DAC),* pp. 1–4.
5. Asifuzzaman, K., Miniskar, N.R., Young, A.R., Liu, F., & Vetter, J.S. (2023). A survey on processing-in-memory techniques: Advances and challenges. *Memories - Materials, Devices, Circuits and Syst.,4*, 100022. https://doi.org/10.1016/j.memori.2022.100022
6. Yu, S., & Chen, P.-Y. (2016). Emerging memory technologies: Recent trends and prospects. *IEEE Solid-State Circuits Magazine, 8*(2), 43–56. https://doi.org/10.1109/MSSC.2016.2546199
7. Rathi, N., Kumar, A., Gupta, N., & Singh, S.K. (2023). A review of low-power static random access memory (sram) designs. In *Proceedings of the IEEE Devices for Integrated Circuit (DevIC),* pp. 455–459. https://doi.org/10.1109/DevIC57758.2023.10134887
8. Dong, Q., Jeloka, S., Saligane, M., Kim, Y., Kawaminami, M., Harada, A., Miyoshi, S., Yasuda, M., Blaauw, D., & Sylvester, D. (2018). A 4 + 2t sram for searching and in-memory computing with 0.3-V V_{DDmin}. *IEEE Journal of Solid-State Circuits,53*(4), 1006–1015. https://doi.org/10.1109/JSSC.2017.2776309
9. Agrawal, A., Jaiswal, A., Lee, C., & Roy, K.: X-sram: Enabling in-memory boolean computations in cmos static random access memories. *IEEE Transactions on Circuits and System I: Regular Papers,65*(12), 4219–4232. https://doi.org/10.1109/TCSI.2018.2848999
10. Dhakad, N.S., Chittora, E., Raut, G., Sharma, V., & Vishvakarma, S.K. (2024). In-memory computing with 6t sram for multi-operator logic design. *Circuits, Systems, and Signal Processing,43*, 646–660. https://doi.org/10.1007/s00034-023-02481-5
11. Wang, C.-C., Tolentino, L.K.S., Huang, C.-Y., & Yeh, C.-H. (2021). A 40-nm cmos multifunctional computing-in-memory (cim) using single-ended disturb-free 7t 1-kb sram. *IEEE Transactions on Very Large Scale Integration (VLSI) Systems,29*(12), 2172–2185. https://doi.org/10.1109/TVLSI.2021.3115970
12. Wang, C.-C., Sulistiyanto, N., Tsai, T.-Y., & Chen, Y.-H. (2020). Multifunctional in-memory computation architecture using single-ended disturb-free 6t sram. In *Advances in Electronics Engineering,* pp. 49–57. Springer, Singapore. https://doi.org/10.1007/978-981-15-1289-6_5
13. Khan, S., & Hamdioui, S. (2010). Trends and challenges of sram reliability in the nano-scale era. In *Proceedings of the 5th International Conference on Design & Technology of. Integrated Systems in Nanoscale Era, Hammamet, Tunisia,* pp. 1–6. https://doi.org/10.1109/DTIS.2010.5487565
14. Gul, W., Shams, M., & Al-Khalili, D. (2022). Sram cell design challenges in modern deep sub-micron technologies: An overview. *Micromachines., 13*(8), 1332. https://doi.org/10.3390/mi13081332
15. Choi, S., Cho, S., & Nam, B.-G. (2021). An area-efficient memory-based multiplier powering eight parallel multiplications for convolutional neural network processors. *Electronics Letters, 57*(15), 573–575. https://doi.org/10.1049/ell2.12206
16. Karimi, K., Fardoost, A., & Javanmard, M. (2024). Comprehensive review of finfet technology: History, structure, challenges, innovations, and emerging sensing applications. *Micromachines., 15*(10), 1187. https://doi.org/10.3390/mi15101187
17. Yu, B., Chang, L., Ahmed, S., Wang, H., Bell, S., Yang, C.-Y., Tabery, C., Ho, C., Xiang, Q., King, T.-J., Bokor, J., Hu, C., Lin, M.-R., & Kyser, D. (2002). Finfet scaling to 10 nm gate length. In *Dig.-Int. Electron Devices Meeting (IEDM),* pp. 251–254. https://doi.org/10.1109/IEDM.2002.1175825
18. Wang, C.-C., Chodisetti, L.S.S.P.K., Ke, J.-Y., Lo, C.-Y., Lee, T.-J., & Tolentino, L.K.S. (2024). A 6-gbps 16-nm finfet cmos i/o buffer with variation insensitivity ensured by genetic algorithm. *IEEE Transactions on Circuits and Systems,* I: Regular Papers. *71*(11), 4961–4972. https://doi.org/10.1109/TCSI.2024.3419020
19. Mukhopadhyay, S., Mahmoodi, H., & Roy, K. (2005). Modeling of failure probability and statistical design of sram array for yield enhancement in nanoscaled cmos. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems,24*(12), 1859–1880. https://doi.org/10.1109/TCAD.2005.852295
20. Ensan, S.S., Moaiyeri, M.H., & Hessabi, S. (2018). A robust and low-power near-threshold sram in 10-nm finfet technology. *Analog Integrated Circuits and Signal Processing,94*, 497–506. https://doi.org/10.1007/s10470-018-1107-7
21. Mai, Y., Wang, M., Zhang, C., Zhong, B., & Yu, Z. (2023). A 1.97 tflops/w configurable sram-based floating-point computation-in-memory macro for energy-efficient ai chips. In *Proceedings of the 2023 IEEE International Symposium on Circuits and Systems (ISCAS),* pp. 1–5. https://doi.org/10.1109/ISCAS46773.2023.10182197
22. Wang, S., Ma, Y., & Kang, Y. (2024). Aft-cim: An energy efficient adc-free transpose computing-in-memory macro for mac operations. In *Proceedings of the 2024 IEEE International Symposium on Circuits and Systems (ISCAS),* pp. 1–5. https://doi.org/10.1109/ISCAS58744.2024.10558546

23. Dou, S., Gu, Z., You, H., Zhan, Y., Qiao, S., & Zhou, Y. (2025) A charge domain sram computing-in-memory macro with quantized interval-optimized adc and input bit-level sparsity-optimized p2o-dac for 8-b mac operation. *IEEE Transactions on Very Large Scale Integrated (VLSI) Systems,33*(5), 1467–1471. https://doi.org/10.1109/TVLSI.2024.3509432
24. Lin, Z., Long, M., Yang, Y., Wang, X., Li, H., Zhang, W., Chen, L., Liu, Y., Li, X., & Wu, X. (2025). A 28 nm 1.3 tflops/mm^2 floating-point sram-based cim macro with asynchronous normalization and parallel sorting alignment for ai-edge chip. *IEEE Transactions on Very Large Scale Integrated (VLSI) Systems,01*(01), 1–10. https://doi.org/10.1109/TVLSI.2025.3630648
25. Wang, H., Liu, R., Dorrance, R., Dasalukunte, D., Lake, D., & Carlton, B. (2023). A charge domain sram compute-in-memory macro with c-2c ladder-based 8-bit mac unit in 22-nm finfet process for edge inference. *IEEE Journal of Solid-State Circuits, 58*(4), 1037–1050. https://doi.org/10.1109/JSSC.2022.3232601
26. Su, J.-W., Chou, Y.-C., Liu, R., Liu, T.-W., Lu, P.-J., Wu, P.-C., Chung, Y.-L., Hong, L.-Y., Ren, J.-S., Pan, T., Jhang, C.-J., Huang, W.-H., Chien, C.-H., Mei, P.-I., Li, S.-H., Sheu, S.-S., Chang, S.-C., Lo, W.-C., Wu, C.-I., ... Chang, M.-F. (2023). An 8-b-precision 6t sram computing-in-memory macro using segmented-bitline charge-sharing scheme for ai edge chips. *IEEE Journal of Solid-State Circuits,58*(3), 877–892. https://doi.org/10.1109/JSSC.2022.3199077
27. Lee, K., Kim, J., & Park, J. (2024). A 28-nm 50.1-tops/w p-8t sram compute-in-memory macro design with bl charge-sharing-based in-sram dac/adc operations. *IEEE Journal of Solid-State Circuits,59*(6), 1926–1937. https://doi.org/10.1109/JSSC.2023.3334566
28. Dai, C., Zhang, J., Wang, R., Chen, J., Hu, W., Hao, L., Lu, W., Lin, Z., Peng, C., & Wu, X. (2025). A pvt-insensitive 7t sram cim macro for multibit multiplication with dynamic matching quantization circuits. *Microelectronics Journal,161*, 106703. https://doi.org/10.1016/j.mejo.2025.106703
29. Hsieh, S.-E., Wei, C.-H., Xue, C.-X., Lin, H.-W., Tu, W.-H., Chang, E.-J., Yang, K.-T., Chen, P.-H., Liao, W.-N., Low, L.L., Lee, C.-D., Lu, A.-C., Liang, J., Cheng, C.-C., & Kang, T.-H. (2023). 7.6 a 70.85-86.27 tops/w pvt-insensitive 8b word-wise acim with post-processing relaxation. In *Proceedings of the 2023 IEEE International Solid-State Circuits Conference (ISSCC),* pp. 136–138. https://doi.org/10.1109/ISSCC42615.2023.10067335
30. Guo, A., Si, X., Chen, X., Dong, F., Pu, X., Li, D., Zhou, Y., Ren, L., Xue, Y., Dong, X., Gao, H., Zhang, Y., Zhang, J., Kong, Y., Xiong, T., Wang, B., Cai, H., Shan, W., & Yang, J. (2023). A 28nm 64-kb 31.6-tflops/w digital-domain floating-point-computing-unit and double-bit 6t-sram computing-in-memory macro for floating-point cnns. In *Proceedings of the 2023 IEEE International Solid-State Circuits Conference (ISSCC),* pp. 128–130. https://doi.org/10.1109/ISSCC42615.2023.10067260
31. Wang, C.-C., Chodisetti, L.S.S.P.K., Liao, B.-H., Vellanki, P., & Lee, T.-J. (2024). A 1–6.5 gbps dual-loop cdr design with coarse-fine tuning vco and modified dqfd. *Microelectronics Journal,151*, 106355. https://doi.org/10.1016/j.mejo.2024.106355
32. Wang, C.-C., Chodisetti, L.S.S.P.K., Kamarajugadda, D.S., Alvarez, O.L.J.J., & Vellanki, P. (2024). A 15.13 mw 3.2 ghz 8-bit carry look-ahead adder using single-phase all-n-transistor logic. *Integration,98*, 102234. https://doi.org/10.1016/j.vlsi.2024.102234

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Springer Nature or its licensor (e.g. a society or other partner) holds exclusive rights to this article under a publishing agreement with the author(s) or other rightsholder(s); author self-archiving of the accepted manuscript version of this article is solely governed by the terms of such publishing agreement and applicable law.