A 4-bit Calibration-Free Computing-In-Memory Macro With 3T1C Current-Programed Dynamic-Cascode Multi-Level-Cell eDRAM

Jiahao Song, *Member, IEEE*, Xiyuan Tang, *Member, IEEE*, Haoyang Luo, Haoyi Zhang, *Graduate Student Member, IEEE*, Xin Qiao, *Graduate Student Member, IEEE*, Zixuan Sun, *Graduate Student Member, IEEE*, Xiangxing Yang, *Member, IEEE*, Zihan Wu, *Graduate Student Member, IEEE*, Yuan Wang, *Member, IEEE*, Runsheng Wang, *Member, IEEE*, and Ru Huang, *Fellow, IEEE*

***Abstract*— Analog computing-in-memory (CIM) has been widely explored for computing neural networks (NNs) efficiently. However, most analog CIM implementations trade compute accuracy for energy efficiency. The low accuracy restricts the practical application of analog CIM. In this article, a current-programming CIM that unifies the weight programming and computing in the current domain is proposed to address this dilemma. The enabled technique is a novel 3-transistor 1-capacitor (3T1C) embedded dynamic random access memory (eDRAM) cell. The current-programming mechanism and the dynamic-cascode read structure of the 3T1C cell make it immune to transistor-level non-idealities, including nonlinear I–V, threshold voltage variations, and short-channel effect. Therefore, the cell enables multi-level-cell (MLC) operations without any calibration, supporting eight current-weight levels (0–700 nA). In addition, a voltage–current two-step programming scheme is proposed to boost the sub-microamphere current-weight writing speed. To support signed 4-b weights, a pseudo-differential CIM cell composed of two 3T1C MLCs is developed. Fabricated in a 65-nm CMOS, the prototype demonstrates 2.2× reduction in macro-level variation through current programming. Benefiting from sub-microamphere compute currents, the prototype achieves the 4-b energy efficiencies of 233–304 TOPS/W. With a refresh interval of 0.4 ms, the macro achieves >90% inference accuracy on CIFAR10.**

***Index Terms*— Analog, computing-in-memory (CIM), current programming, dynamic cascode, embedded dynamic random access memory (eDRAM), multi-level-cell (MLC), neural network (NN), variation.**

Manuscript received 8 July 2023; revised 19 October 2023; accepted 26 November 2023. Date of publication 15 December 2023; date of current version 27 February 2024. This article was approved by Associate Editor Yan Lu. This work was supported in part by the Joint Funds of National Natural Science Foundation of China under Grant U20A20204 and Grant 62304009 and in part by the 111 Project under Grant B18001. *(Corresponding authors: Xiyuan Tang; Yuan Wang.)*

Jiahao Song, Haoyi Zhang, Xin Qiao, Zixuan Sun, and Zihan Wu, are with the Key Laboratory of Microelectronic Devices and Circuits (MOE), School of Integrated Circuits, Peking University, Beijing 100871, China.

Xiyuan Tang and Haoyang Luo are with the Institute for Artificial Intelligence and the School of Integrated Circuits, Peking University, Beijing 100871, China (e-mail: xitang@pku.edu.cn).

Xiangxing Yang is with pSemi Corporation, Austin, TX 78704 USA.

Yuan Wang, Runsheng Wang, and Ru Huang are with the Key Laboratory of Microelectronic Devices and Circuits (MOE), School of Integrated Circuits, Peking University, Beijing 100871, China, and also with the Beijing Advanced Innovation Center for Integrated Circuits, Beijing 100871, China (e-mail: wangyuan@pku.edu.cn).

Color versions of one or more figures in this article are available at https://doi.org/10.1109/JSSC.2023.3339887.

Digital Object Identifier 10.1109/JSSC.2023.3339887

I. Introduction

DEEP neural network (DNN) has achieved unprecedented success in various tasks, such as vision and speech processing [1], [2]. The deployment of the DNN model to edge brings local intelligence, addressing the increasing concerns of communication bandwidth, latency, and privacy. However, the performance of DNN relies on its growing computational complexity, which is dominated by matrix-vector multiplications (MVMs) and corresponding data movement/accessing [3]. The intensive MVMs and data movement of the recent DNN model have pushed the conventional hardware accelerator to their energy-efficiency limits. With limited computing resources and energy budget, the smart edge devices desire a new hardware solution.

The analog computing-in-memory (CIM) architecture holds great promise as an energy-efficient solution for local inference [4]. With the 2-D weight matrix stored in 2-D bit-cell array and the input vectors given via word lines, analog CIM macro performs the MVM inside the memory array, and the MVM results are developed directly on bitlines. In contrast to conventional memory that only allow one-row access, the CIM architecture enables multi-row access, effectively amortizing the bitline switching energy [5]. In addition, by implementing the multiply-and-accumulate (MAC) function using compact bit cells, the computing density is significantly enhanced. Early CIM exploits the standard 6T/8T static random access memory (SRAM) bit cells to charge/discharge bitline for analog computing [6], [7], [8], [9]; thus, a high storage density is achieved. These CIM designs, which utilize the transistor currents in the bit cells to perform computing, are classified as current-based CIM. However, the compute signal-to-noise ratio (SNR) in current-based CIM is adversely affected by large variations of the minimum-size transistors used in the bit cells and the read-disturb issue. Specifically, as more rows are simultaneously activated for computing, the SNR in current-based CIM experiences degradation. In other words, there exists a fundamental energy efficiency versus compute accuracy trade-off in the analog CIM [10], [11], [12], as illustrated in Fig. 1. Consequently, achieving both high energy efficiency and high compute accuracy simultaneously becomes challenging for the analog CIM macro.

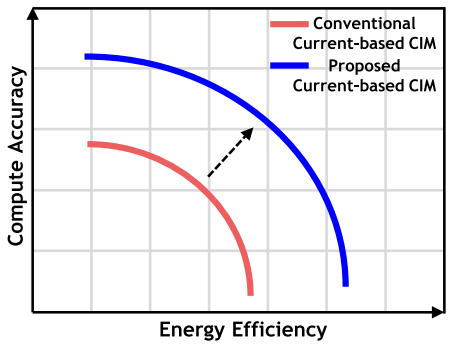


Fig. 1. Fundamental energy efficiency versus compute accuracy trade-off in analog CIM and the design target of the proposed design.

Efforts have been made to improve the energy efficiency, SNR, and density of analog CIM. The capacitor-based bit cells are proposed for high-SNR analog computing [13], [14], [15], [16]. Due to the decoupled write–read ports and robust capacitor-based computing in these bit cells, the read-disturb issue is addressed, and more rows can be activated simultaneously, bringing higher energy efficiency and process–voltage–temperature (PVT) robustness. However, the custom-designed 6T+ bit cell in capacitor-based CIM introduces a significant area overhead, trading density for energy efficiency, and SNR. The embedded dynamic random access memory (eDRAM), especially with multi-level cell (MLC), provides a high-density solution with only 1–4 transistors per cell [17], [18], [19], [20], [21], [22]. This makes eDRAM an attractive option for analog CIM designs. However, similar to SRAM-based analog CIM, the challenge of simultaneously improving energy efficiency and accuracy persists in eDRAM-based analog CIM.

In summary, SRAM/eDRAM current-based designs, which utilize the transistor currents within the compact cells for computing, provide area- and energy-efficient solutions for analog CIM. However, improving the computing accuracy of current-based CIM faces challenges due to threshold voltage (V_t) variations of the transistors in cells. This compute accuracy limitation in current-based CIM is rooted in the inconsistency of weight representations during programming and computing: weights are programed and stored as fixed voltages, while transistor currents are used for computation. To fundamentally surmount this dilemma faced by current-based CIM, we propose a current-programming eDRAM CIM that unifies the weight programming and computing in the current domain. The enabling technique is a novel 3-transistor 1-capacitor (3T1C) current-programed dynamic-cascode MLC eDRAM design. It confers several key merits.

1) The eDRAM cell is programed by the weight current directly with the self-calibrated voltage generated on the storage capacitor; it essentially stores the weight current instead of a fixed voltage, thus mitigating V_t variation and nonlinear transistor I–V impacts.
2) A dynamic-cascode read structure is proposed to significantly reduce the V_{RBL} sensitivity while not requiring any bias voltage.
3) Thanks to the accurately programed cell, it supports eight current levels ranging from 0 to 700 nA in a single cell without any calibration, largely boosting computation density.
4) A voltage–current two-step programming scheme significantly boosts the sub-microamphere current-weight writing speed.

Combining these merits, the proposed MLC eDRAM CIM is naturally immune to transistor-level non-idealities, thus allowing a small LSB weight current of only 100 nA. To support 4-b signed weight, a 4-b CIM cell composed of two MLCs is developed, containing 15 current levels (from −700 to 700 nA).

This article is an extension of [23] and is organized as follows. Section II reviews the recent CIM designs and analyzes the conventional voltage-programed cell and the proposed current-programed cell. Section III presents the overall design of current-programming CIM. Section IV shows the measured results. Finally, Section V concludes this article.

II. CIM Review and Design Analysis

A. CIM Macro Review

To compute the DNN more efficiently and accurately, various CIM macros have been proposed in recent years. These works can be categorized into two groups based on whether the macro performs MAC operations using digital or analog circuit, as illustrated in Fig. 2.

In digital CIM macro, the logic gates are placed near the memory cells to perform bitwise multiplications [24], [25], [26], [27], [28]. The accumulations are performed using an adder tree. Furthermore, multi-bit MACs can be extended through the utilization of near-memory shift-and-add circuits. Digital CIM offers excellent robustness due to its fully digital operations. However, its density is constrained by the large area occupied by the adder tree. In addition, the digital CIM only exhibits energy efficiency benefits at advanced technology nodes [27], [28], as it relies on full-swing digital logic for computations.

On the other hand, analog CIM demonstrates higher energy efficiency due to the utilization of high-parallelism analog MACs. However, the analog MACs are susceptible to transistor-level non-idealities, which limit the computing accuracy. In analog CIM, accessing more rows simultaneously enhances energy efficiency but reduces the voltage swing of each MAC output, resulting in a lower compute SNR. This is the fundamental energy efficiency versus compute accuracy trade-off in analog CIM; the knob is the number of simultaneous accessed rows. For current-based CIM, the maximum achievable SNR is limited by V_t variations of the minimum-size transistors within the bit cell. Similarly, in capacitor-based CIM, the maximum achievable SNR is limited by capacitor mismatches. Compared with the current mismatch of small access transistors in the bit cell, capacitors exhibit better matching, resulting in a higher SNR. However, the utilization of custom-designed 6T+ bit-cells in capacitor-based CIM leads to a significant area overhead, which limits its density. In contrast, current-based CIM can be implemented using compact 6/8T SRAM [6], [7], [8], [9], [29], [30] or 2/3T gain-cell eDRAM [19], [21]. With the goal of

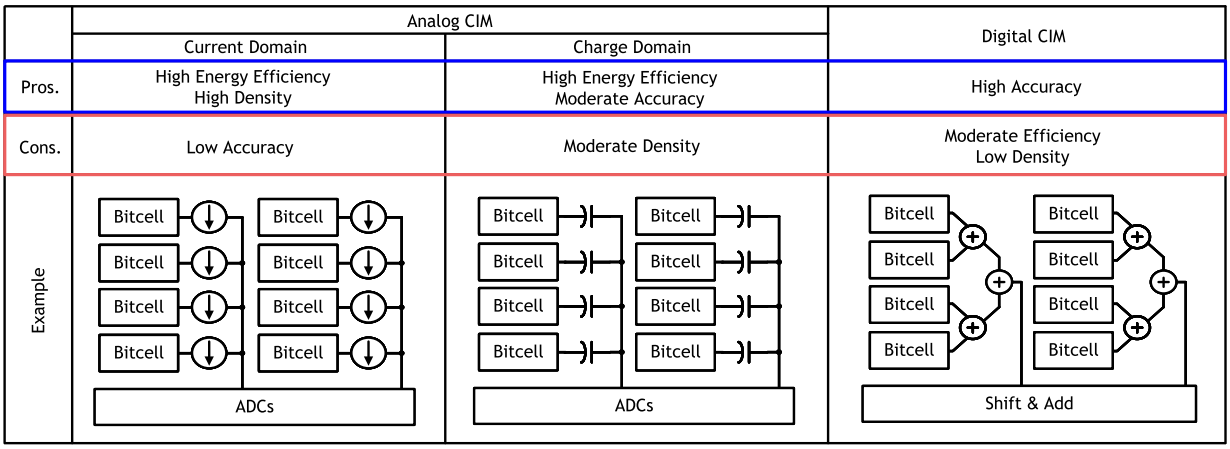


Fig. 2. Comparison of recent analog and digital CIMs.

achieving high-density analog CIM, we focus on current-based CIM and address its SNR limitations in this article.

B. SNR Limit of Voltage-Programed Cell

As mentioned before, in analog CIM, the trade-off between energy efficiency and compute accuracy is determined by the number of simultaneously accessed rows. Specifically, for current-based CIM, this trade-off is also controlled by the compute currents. In current-based CIM, all compute currents discharge the bitline capacitance to perform accumulation. With the same bitline capacitance, smaller compute currents allow more cells to be activated without clipping errors, thus brings higher energy efficiency. However, when operated in small currents, the compute SNR is severely limited due to transistor V_t variations. The cause is rooted in the inconsistency of weight representations during programming and computing: weights are programed as fixed voltages V_{write}, while transistor currents $I_{\text{comp\_}v}$ are used for computation, as shown in Fig. 3(a). In the conventional memory, SRAM and eDRAM cells follow a voltage-programed and current-read style. For SRAM, the bit cells are biased to bi-stable state during programming, and the currents of access transistors are used for reading. For eDRAM, two-level voltages are written to the storage node, and its corresponding current is used for reading. Due to the high ON/OFF ratio of transistors and only one row is accessed, the cell value can be easily readout by the sense amplifier. However, in the presence of spatial V_t variations, the voltage-programed cells are no longer suitable for the analog CIM architecture, which is operated in a multi-row accessing mode. When the compute transistor operates in the near-threshold region to enable parallel access of more rows, the voltage-programming and current-computing process can be modeled as follows:

$$I_{\text{comp\_}v} = I_0 \cdot \exp\left(\frac{V_{\text{write}} - V_t}{nkT/q}\right) \tag{1}$$

where I_0 is proportional to the W/L of compute transistor, n is the slope factor, k is Boltzmann's constant, T is the absolute temperature, and q is the electron charge. In the presence of threshold voltage variations, (1) transforms into

$$I_{\text{comp\_}v} + i_{\text{comp\_}v} = I_0 \cdot \exp\left(\frac{V_{\text{write}} - (V_t + v_t)}{nkT/q}\right) \tag{2}$$

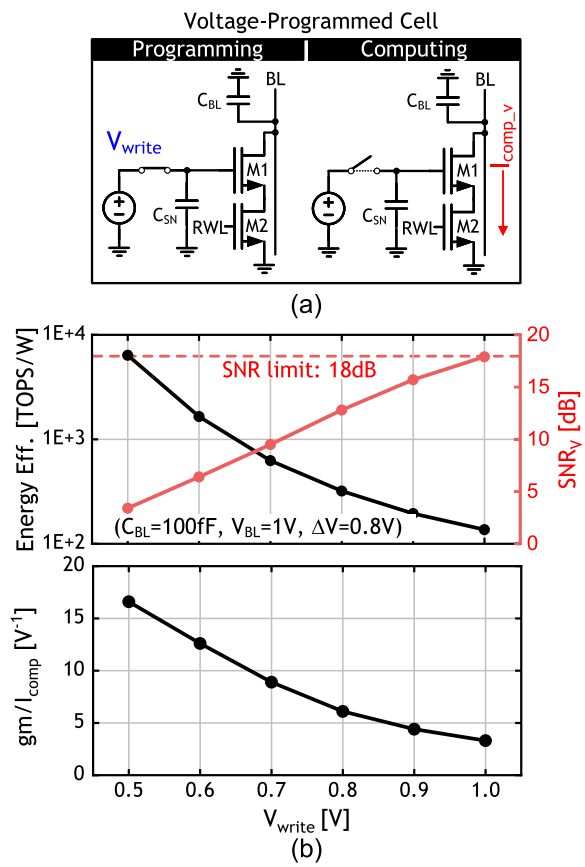


Fig. 3. (a) Concept of voltage programming. (b) Simulated energy efficiency, compute SNR, and g_m/I_{comp} versus V_{write} of current-based analog CIM.

where v_t is the threshold voltage variation, and $i_{\text{comp\_}v}$ is the corresponding current variation. Therefore, the compute SNR is degraded even if V_{write} can be programed precisely. We define $\text{SNR}_V = I^2_{\text{comp\_}v}/\sigma^2_{i_{\text{comp\_}v}}$ as the SNR of voltage-programming CIM. Using a 1st-order Taylor series expansion for (2), we get

$$i_{\text{comp\_}v} \approx -v_t \cdot \frac{\partial I_{\text{comp\_}v}}{\partial V_t} = -v_t \cdot \frac{I_{\text{comp\_}v}}{nkT/q} = -v_t \cdot g_m. \tag{3}$$

Then, the SNR_V can be calculated as follows:

$$\mathrm{SNR}_{V,\mathrm{dB}} = 10\lg\frac{I_{\mathrm{comp\_}v}^2}{\sigma_{i_{\mathrm{comp\_}v}}^2} = 10\lg\frac{1}{(g_m/I_{\mathrm{comp\_}v})^2}\cdot\frac{1}{\sigma_{v_t}^2} \quad (4)$$

where $g_m = (I_{\mathrm{comp\_}v}/(nkT/q))$ is the trans-conductance of compute transistor $M1$. As the transistor operates in the subthreshold region, $g_m/I_{\mathrm{comp\_}v}$ gradually increases, resulting in improved energy efficiency for computing. However, this brings a decreased SNR. Assuming ($C_{\mathrm{BL}} = 100$ fF, the read-word-line (RWL) pulsewidth = 200 ps, the bitline precharge voltage = 1.0 V, and the bitline dynamic range = 0.8 V), we conducted a simulation using a 65-nm CMOS process. Fig. 3(b) shows the simulated SNR_V and energy efficiency as a function of V_{write}. The results clearly demonstrate that V_{write} (or the corresponding compute current) plays a crucial role in controlling the trade-off between energy efficiency and SNR_V. When V_{write} lower than 0.6 V, a >1000 TOPS/W energy efficiency can be achieved, but SNR_V is lower than 10 dB. Moreover, even if V_{write} is set to 1.0 V, SNR_V is only 18 dB, which clearly demonstrates the restricted precision of the voltage-programming scheme.

C. Proposed Current-Programed Cell

To fundamentally address the V_t-variation-induced SNR degradation in current-based CIM, we proposed a current-programed eDRAM cell, which unifies the weight programming and computing in the current domain. This innovative approach significantly enhances the maximum achievable SNR of current-based CIM. The simplified model of the proposed current-programed cell is shown in Fig. 4(a). The operation of the current-programed eDRAM cell is as follows. In the programming phase, the transistor $M1$ is diode connected, and the programming current I_{write} flows through $M1$ with the corresponding V_{GS} developed. Different from a fixed $V_{\mathrm{GS}}/V_{\mathrm{write}}$ in voltage programming, this V_{GS} can be viewed as a self-calibrated value that tackles the V_t variation. Thus, the self-calibrated V_{GS}, essentially the voltage representation of the corresponding I_{write}, can be stored. Ideally, I_{comp} of $M1$ is consistent with the programed value I_{write}, regardless of the V_t variations. In practice, there are still noise sources that must be addressed to achieve a high SNR. The current-programming and current-computing process can be modeled as follows:

$$\begin{aligned} I_{\mathrm{comp\_}c} &= I_{\mathrm{write}} + i_{\mathrm{write}} + i_e \\ &= I_0\cdot\exp\left(\frac{(V_{\mathrm{cal}} + v_j + v_n + v_s) - (V_t + v_t)}{nkT/q}\right) \end{aligned} \quad (5)$$

where $I_{\mathrm{comp\_}c}$ is the compute current, I_{write} is the programming current, i_{write} is the variation of programming current, V_{cal} is the self-calibrated storage-node voltage, which depends on the $I_{\mathrm{write}} + i_{\mathrm{write}}$ and $V_t + v_t$, and i_e is the corresponding current-domain noise induced by the charge injection v_j, thermal noise v_n, and settling error v_s. The charge injection

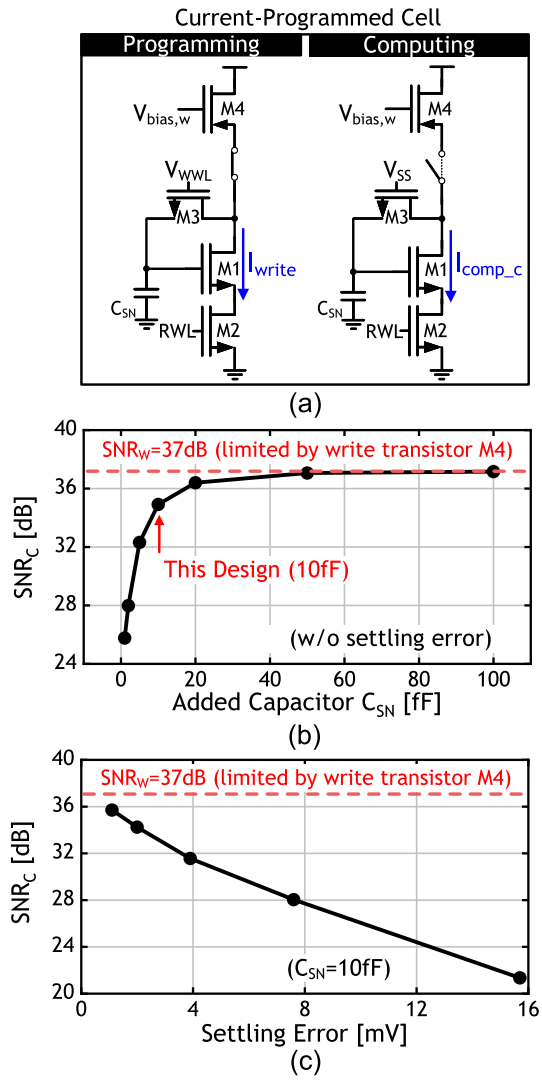


Fig. 4. (a) Concept of current programming. (b) Simulated SNR versus added capacitor at storage node. (c) Simulated SNR versus settling error.

and thermal noise can be derived as follows [31]:

$$v_j = p\frac{WLC_{\mathrm{ox}}(V_{WWL} - V_{\mathrm{cal}} - V_t)}{C_{\mathrm{SN}}}, \quad \sigma_{v_n} = \sqrt{\frac{kT}{C_{\mathrm{SN}}}} \quad (6)$$

where C_{SN} is the capacitance of storage node, $0 \le p \le 1$ is constant that depends on the transition speed and terminal impedances of the switch transistor $M3$, C_{ox} is the gate oxide capacitance per unit area, W and L are the width and length of $M3$, and V_{WWL} is the control-signal voltage of $M3$. We define SNR_C as the SNR of the current-programming CIM, which can be approximated by

$$\mathrm{SNR}_c \approx \frac{I_{\mathrm{write}}^2}{\sigma_{i_{\mathrm{write}}}^2 + \sigma_{i_e}^2} = \left[\frac{1}{\mathrm{SNR}_W} + \frac{1}{\mathrm{SNR}_E}\right]^{-1} \quad (7)$$

where $\mathrm{SNR}_W = I_{\mathrm{write}}^2/\sigma_{i_{\mathrm{write}}}^2$ is limited by write transistor $M4$, $\mathrm{SNR}_E = I_{\mathrm{write}}^2/\sigma_{i_e}^2$ indicates the impact due to sampling and settling errors. Due to the write transistor only occupying a small part of the macro area, up-sizing this transistor can suppress its current variation at a small cost. To tackle with charge injection and KT/C noise, the capacity of C_{SN} need

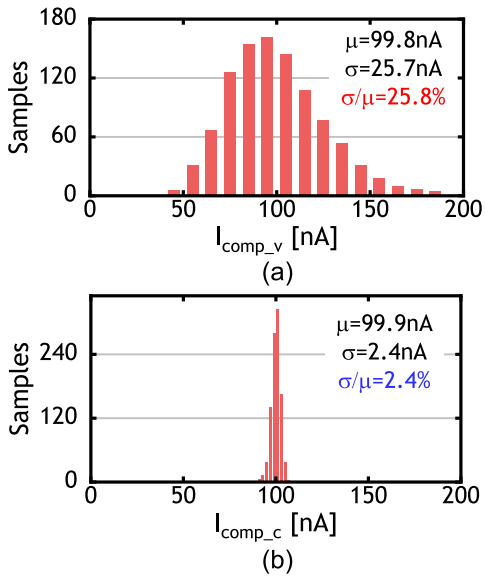


Fig. 5. Simulated 100-nA computing-current variation of (a) voltage-programed cell and (b) current-programed cell.

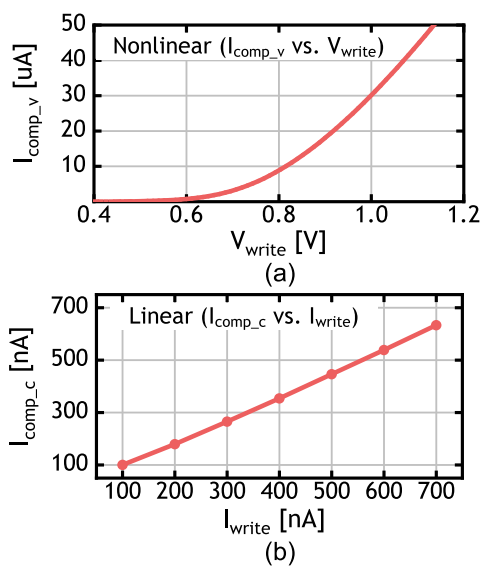


Fig. 6. Simulated programming linearity of (a) voltage-programed cell and (b) current-programed cell.

to be increased, as shown in Fig. 4(b). To achieve a high SNR_C without much area overhead, C_{SN} is set to 10 fF, which is implemented by the metal-oxide-metal (MOM) capacitor. In addition, the added C_{SN} can enhance the cell retention. SNR_C versus settling error is shown in Fig. 4(c). While a <4-mV settling error is guaranteed, SNR_C is higher 30 dB. To speed up the sub-microamphere-current programming and keep a small settling error, a voltage–current two-step write driver is proposed, the detailed operations of which are described in Section III-C. As shown in Fig. 5, when operated at a compute current of 100 nA, the current-programed cell presents a ~10× variation reduction (20-dB SNR improvement) compared with the conventional cell design.

In addition to improve the SNR of current computing, the current programming can also help improving the programming linearity of MLC, as shown in Fig. 6. For conventional

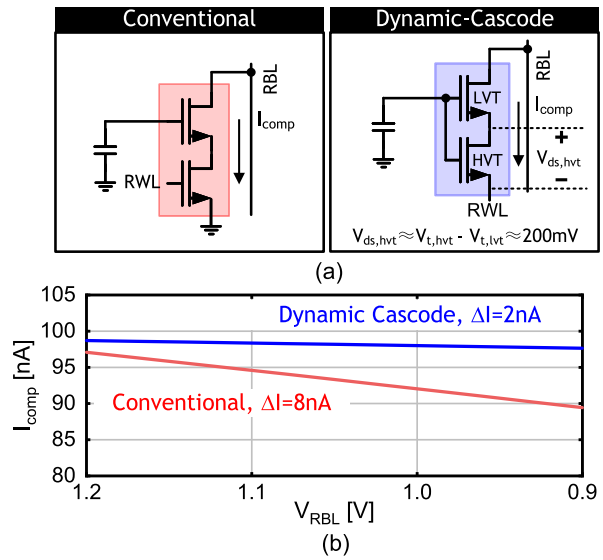


Fig. 7. (a) Schematic and (b) simulated I_{comp} versus V_{RBL} of the conventional and proposed dynamic-cascode read structure.

voltage-programming, the nonlinear transistor I–V poses challenges for the design of an MLC gain cell: calibration is required, bringing extra area and power overhead. In addition, the simple calibration that is only for transistor I–V nonlinearity is not sufficient to write the MLC precisely in the presence of V_t variation. With the proposed current programming technique, an eight-level (0–700 nA) MLC eDRAM cell is designed without calibration.

D. Dynamic-Cascode Read Structure

Another issue in current-based CIM is the I_{comp} sensitivity to V_{RBL} contributed by the short-channel effect (i.e., the limited output impedance), resulting in a nonlinear activation function. A common circuit technique to tackle this issue is the cascode stage. However, the conventional cascode stage requires dedicated biasing, which is impractical for the memory array. This issue is addressed by the proposed dynamic-cascode read structure, which consists of one low threshold voltage (LVT) cascode transistor and one high threshold voltage (HVT) main transistor with gates connected together [32], [33], as shown in Fig. 7. In the dynamic-cascode read structure, the compute currents flow through the HVT transistor and LVT transistor are equal, which can be formulated as follows:

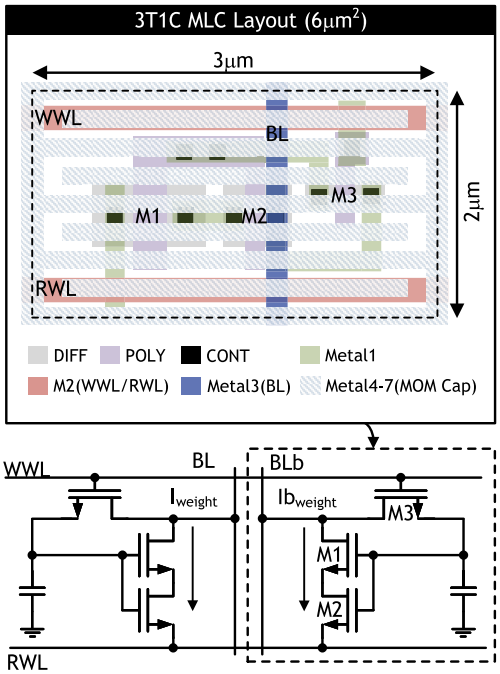
$$I_{\mathrm{comp}} = I_0 \cdot \exp\left(\frac{(V_{\mathrm{cal}} - V_{\mathrm{ds,hvt}}) - V_{t,\mathrm{lvt}}}{nkT/q}\right) = I_0 \cdot \exp\left(\frac{(V_{\mathrm{cal}} - V_{t,\mathrm{hvt}}}{nkT/q}\right) \tag{8}$$

where $V_{\mathrm{ds,hvt}}$ is the drain–source voltage of HVT transistor, $V_{t,\mathrm{lvt}}$ is the threshold voltage of LVT transistor, and $V_{t,\mathrm{hvt}}$ is the threshold voltage of HVT transistor. Then, we can get

$$V_{\mathrm{cal}} - V_{\mathrm{ds,hvt}} - V_{t,\mathrm{lvt}} = V_{\mathrm{cal}} - V_{t,\mathrm{hvt}} \tag{9}$$

$$V_{\mathrm{ds,hvt}} = V_{t,\mathrm{hvt}} - V_{t,\mathrm{lvt}}. \tag{10}$$

It can be seen that $V_{\mathrm{ds,hvt}}$ equal to the threshold difference between HVT and LVT transistors (~200 mV in



Activation (A)	RWL Width
0	0
1	T_0
…	…
15	$15T_0$

Weight (W)	I_{weight}	Ib_{weight}
+7	$7I_0$	0
…		
+1	I_0	0
0	0	0
-1	0	I_0
…		
-7	0	$7I_0$

Fig. 8. Schematic and operation table of 4-b CIM cell, which composed of two 3T1C MLCs; layout of 3T1C MLC.

65-nm CMOS), ensuring the HVT device's operation region. With the proposed dynamic-cascode stage, I_{comp} sensitivity to V_{RBL} is reduced by 4× compared with the conventional read structure.

III. Proposed Current-Programming CIM Macro

A. 3T1C MLC and 4-b CIM Cell

A 3T1C calibration-free eight-level MLC is developed by combining the current-programming and dynamic-cascode techniques, as shown in Fig. 8. Without additional bias voltages, the proposed MLC addresses key issues faced by conventional current-based CIM designs, including the nonlinear I–V, V_t variations, and short-channel effect. In addition, with the sub-microamphere compute currents, the energy efficiency of analog MAC is significantly boosted. The layout of the 3T1C cell occupies an area of 6 μm^2, which is dominated by the added 10-fF MOM capacitor from metal-4 to metal-7. Fig. 9 shows the simulated compute currents drifted from 0 to 0.4 ms at FF corner and 80 °C when 100-/400-/700-nA programming currents are applied. The added capacitor effectively brings more accurate compute currents and smaller drifts (from 17.1-57.5% to 1.5-20.4%) during 0.4-ms retention time. To support 4-b signed weight and 4-b unsigned input, a 4-b CIM cell composed of two MLCs is developed, containing 15 current levels (from −700 to 700 nA) and supporting 16 RWL pulsewidth levels. These two MLCs

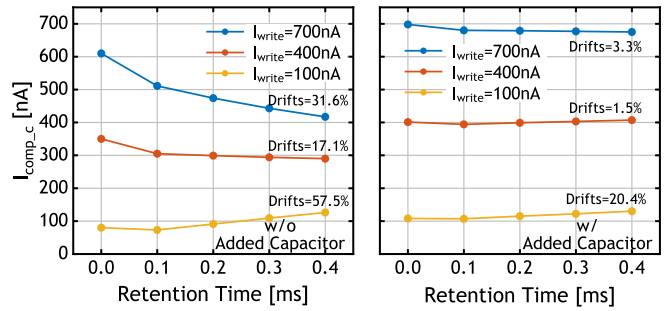


Fig. 9. Simulated computing currents drifted from 0 to 0.4 ms at FF corner and 80 °C when 100-/400-/700-nA programming currents are applied.

are pseudo-differentially combined, providing noise immunity against unwanted charge injection/coupling.

B. Overall Architecture, DTC, and ADC Design

Fig. 10 presents the overall architecture of the proposed eDRAM CIM macro, which comprises a 64 × 64.4-b cell array, the voltage–current two-step write drivers, the 5-b successive-approximation-register (SAR) analog-to-digital converters (ADCs), and 4-b digital-to-time converters (DTCs) for CIM operations, and control blocks for writing and computation (WCTRL, CCTRL). During one computing cycle (180 ns), the macro operations are divided into three phases. In the first phase, the BL capacitance and the capacitive digital-to-analog converter (CDAC) in SAR ADC are connected and precharged to V_{PCH}. Then, all DTCs are enabled and generate the corresponding RWL pulse width. These RWL pulses are multiplied by the stored current in the CIM cells. In each CIM column, 64 CIM cells discharges the BL/BLb, performing the current-based MACs. In the third phase, the sampled analog-MAC value is quantized by the 5-b SAR ADC.

In the CIM mode, the RWL pulsewidth is modulated based on the digital input code (A[3:0]) by a DTC. To reduce the RWL pulsewidth mismatch between 64 input channels, all DTCs generate the RWL pulses from the shared pulsewidth modulated (PWM) signal. The pulse generation and selection methodology are adapted from [34]. Five global PWM signals (TD_0–TD_{15}) are taped out from a tunable delay line, with its delay controlled by off-chip bias for testing purposes. The pulsewidth of each signal increments by 5 × t_0 and varies within the range of 0–15 × t_0. Here, t_0 represents the minimum possible pulsewidth, which is the delay of the tunable delay unit. Each DTC selects and generates the corresponding RWL pulse from (TD_0 to TD_{15}) using a 4:1 MUX and two 2:1 MUXs in two phases: the LSB phase and the MSB phase. In the LSB phase, the pulsewidth is incremented by 1 × t_0, while in the MSB phase, it gets incremented by 4 × t_0. A control signal (TD_{12}) switches between the two phases. The generated pulsewidth of the first phase is determined by two LSB bits of A, and the second phase is determined by two MSB bits. Compared with the one-phase architecture that selects 16 timing signals with a 16:1 MUX, the two-phase design reduces the number of global timing signals to route and the power consumption.

During computation, the common-mode voltage of analog-MAC varies as the CIM cells discharge the BL capacitance.

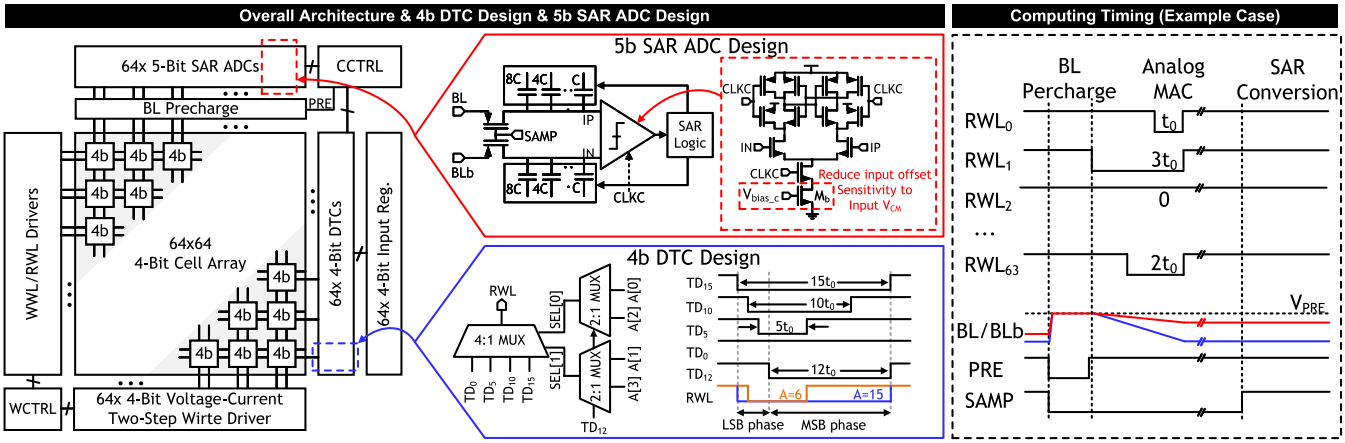


Fig. 10. Overall architecture and timing of the proposed current-programming CIM, and the schematic of ADC and DTC.

This variation can introduce offset changes in the comparator of the ADC and result in quantization errors. To reduce this effect, a current–source-assisted dynamic comparator is adopted to reduce offset sensitivity to input common-mode voltage [35]. This dynamic comparator incorporates a cascode-biased MOS transistor M_b at the bottom of the switch MOS. By operating M_b in the saturation region, the change of the drain–source voltage has only a slight influence on the drain current. As a result, it maintains the effective voltage of the input pair near a constant value even when the common-mode voltage changes, thereby minimizing the impact on offset.

C. Voltage–Current Two-Step Write Driver

Fig. 11 describes the schematic and operation of the voltage–current two-step write driver. The purpose of the driver is to reduce the latency caused by the small-current programming. Although the small operating current brings higher energy efficiency, driving the BL parasitics (50 fF) with a small current (100 nA) results in a long settling time (∼400 ns). To speed it up, the driver uses a two-step process: voltage-mode coarse writing followed by current-mode fine writing. In the first step, the voltage-write block drives the BL/BLb to V_W or V_{SS} (depending on the weight value) within 5 ns. V_W (0.52 V) is V_{GS} of computing transistors, when 100 nA I_{write} is applied. Following this, current-mode fine writing is used to accurately program the cell current to the target value. In addition, in order to avoid a slow startup of the current source and keep the current-write transistor in the saturated region, a replica cell is used to provide an operation point during write mode. As shown in Fig. 12, the simulation reveals that all current levels can be settled within 60 ns, indicating a >6× speed up of the current programming.

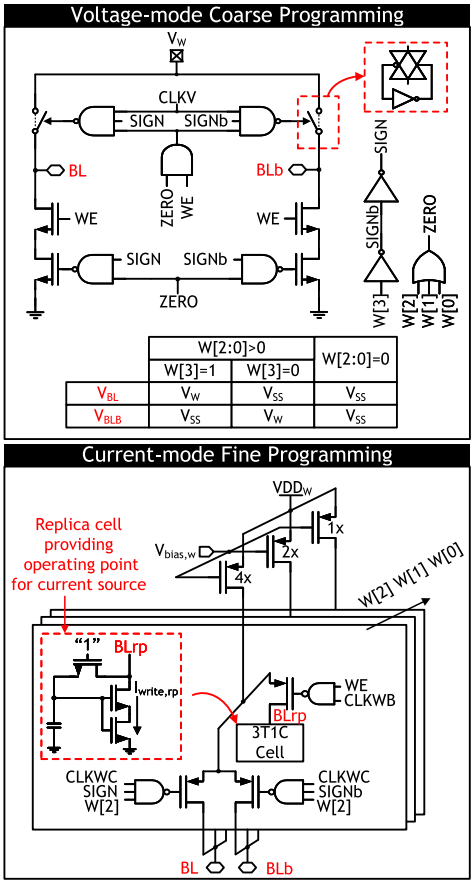


Fig. 11. Schematic of the proposed voltage–current two-step write driver.

IV. Measurement Results

The prototype is implemented in a 65-nm CMOS. Fig. 13 shows the micrograph of the test chip and the test environment. The CIM macro has a capacity of 16 kb at the footprint of 0.1536 mm^2. The operating voltages of macro are 1.0-V digital block (CCTRL and DTC), 0.9-V cell array (BL precharge), and 0.7-V ADC. In the CIM mode, each computing cycle lasts for 180 ns, during which the macro computes 64 independent 64-input MACs. Consequently, the throughput is calculated as $2 \times 64 \times 64/180$ ns = 45.5 GOPS (1 MAC = 2 operations). The computing density is 296.3 GOPS/mm^2. To measure the computing energy, all weights are written to "+7", and all inputs are configured to "+15"," which allows maximum discharge of the bitline. With the number of activated inputs are swept from 25% to 75%, the measured 4-b-MAC energy efficiencies are 233–304 TOPS/W, as shown in Fig. 14(a).

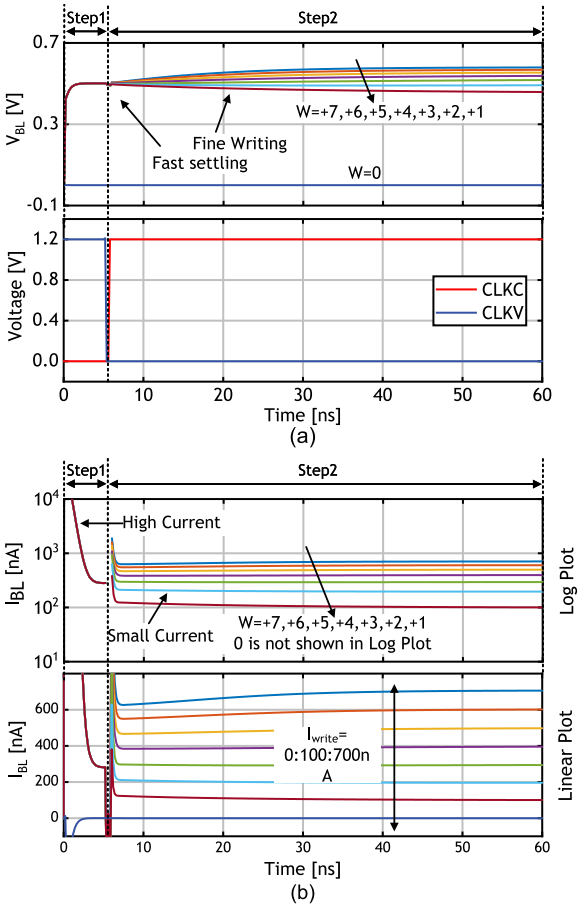


Fig. 12. Simulated waveforms of the voltage–current two-step write driver. (a) Simulated BL voltage and control signals. (b) Simulated BL current.

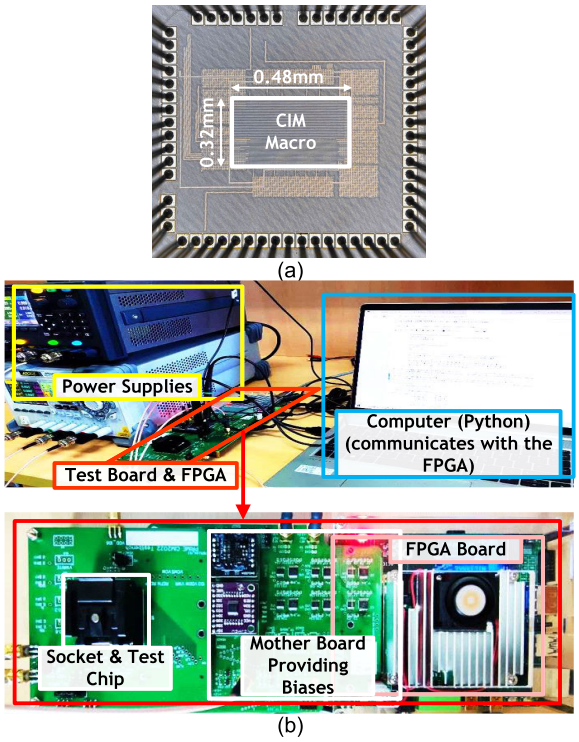


Fig. 13. (a) Die micrograph. (b) Test environment.

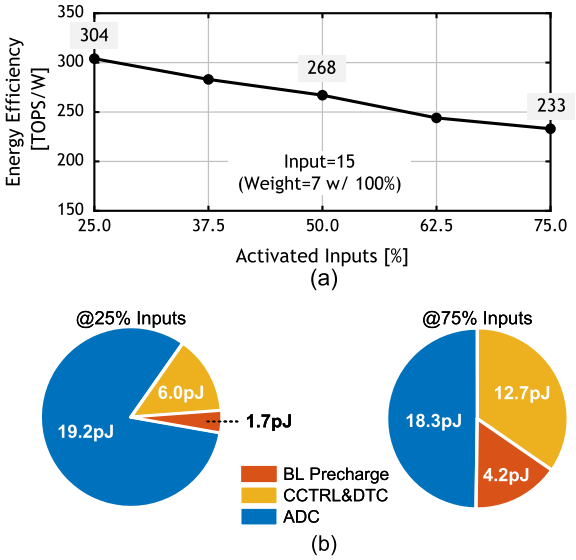


Fig. 14. Measured (a) energy efficiency as a function of activated inputs and (b) energy breakdown at 25% and 75% activated inputs.

Fig. 14(b) shows the energy breakdown of 25% and 75% activated inputs. The energy consumed by the ADC is ∼19 pJ in both 25% and 75% activated inputs. As the activated input varies from 25% to 75%, the energy consumed on the bitline increases from 1.7 to 4.2 pJ due to larger bitline swing. Similarly, the energy consumed by CCTRL and DTC are increased from 6 to 12.7 pJ for driving more RWL. Table I shows the comparison of recent eDRAM-CIM works and SRAM-CIM (current based) works for neural network (NN) acceleration. The proposed current-programming CIM achieves high energy efficiency.

A. Characteristics of Current Programming

To verify the effectiveness of the proposed programming scheme, the CIM transfer functions with voltage and current programming are measured and compared. The transfer functions are measured according to the following steps. First, the weight data in all CIM cells are written to "+1/−1" using the voltage–current two-step write drivers, and an activation pattern is applied to the input registers. Once the data are prepared, the CIM macro performs computation, generating the corresponding analog-MAC outputs. These outputs are then observed and quantized by the ADCs. This process is repeated for each activation pattern (from all "0" to all "15") to construct the CIM transfer function. During testing, the DTC output pulsewidth is adjusted by an off-chip bias to ensure that the full-scale range of the MAC output matches the dynamic range of the ADC. For current programming, the voltage–current two-step write driver program the CIM macro by two steps: a fixed voltage (0.52 V) is written to the eDRAM cell in the first step, followed by fine-tuning the storage-node voltage using a fixed current (100 nA) in the second step. For voltage programming, the current-domain fine-tuning step is omitted, and a fixed voltage (0.52 V) is stored in the storage node of the eDRAM. The transfer functions of different CIM columns with voltage and current programming performed on the same macro are shown in Fig. 15. As can be seen,

TABLE I
PERFORMANCE COMPARISON WITH STATE-OF-THE-ART CIM MACROS

	eDRAM CIM						SRAM CIM (Current-based)	
	This Work	ISSCC'23 S. Kim	VLSI'22 S. Xie	ISSCC'21 S. Xie	ISSCC'21 Z. Chen	TCAS I '21 C. Yu	ISSCC'20 Q. Dong	JSSC'22 J. Yue
Technology	**65 nm**	28nm	65 nm	65 nm	65 nm	65 nm	7nm	65nm
Programming	**Current**	Voltage					Voltage	
Computing	**Current**	Charge			Current		Current	
Cell Type	**3T1C eDRAM**	3T2C eDRAM	2T1C eDRAM	1T1C eDRAM	3T1C eDRAM	2T1C eDRAM	8T SRAM	8T SRAM
Macro Size	**16kb**	92kb	32kb	16kb	8kb	16kb	4kb	4kb
Fully-Parallel MAC	✓	✓	✕	✕	✓	✓	✓	✓
Multi-level Cell (MLC)	✓	✕	✕	✕	✓	✕	✕	✕
Calibration Free	✓	✓	✓	✓	✕	✕	✕	✕
OTA Free	✓	✓	✕	✕	✓	✓	✓	✓
Parallelism	**64**	288	27	1	64	64	64	8/16
Levels/CIM Cell	**15**	2	2	2	16	2	2	2
CIM Cell Area	**2×6μm²**	1.02μm²	N/A	22.08μm²	N/A	1.08μm²	0.053μm²	N/A
[1]CIM Column Input Precision	**4b**	1b	2b	8b	4b	1b	4b	2b
[1]CIM Column Weight Precision	**4b**	1b	1b	8b	4b	1.5b	4b	4b
Dataset	**CIFAR10**	CIFAR10	CIFAR10	CIFAR10	CIFAR10	CIFAR10	MNIST	CIFAR10
Classification Accuracy	**[2]90.78-90.96%**	89.5%	92.02%	80.1%	90.6%	82.8%	98.5%	86.62%
Computing Density (TOPS/mm²)	**[3]0.296 (I:4b W:4b)**	2.03 (I:4b W:5b)	0.4113 (I:2b W:1b)	0.00826 (I:8b W:8b)	N/A	309 (simulated) (I:1b W:1.5b)	116.4 (I:4b W:4b)	N/A
[4]Normalized Computing Density (TOPS/mm²)	**4.74**	40.6	0.8226	0.52864	N/A	463.5 (simulated)	1862.4	N/A
Energy Efficiency (TOPS/W)	**[5]233-304 (I:4b W:4b)**	115.8 (I:4b W:5b)	236 (I:2b W:1b)	4.76 (I:8b W:8b)	102.2 (I:4b W:4b)	552.5 (simulated) (I:1b W:1.5b)	262.3-610.5 (I:4b W:4b)	64.85 (I:2b W:4b)
[6]Normalized Energy Efficiency (TOPS/W)	**3728-4864**	2316	472	304.64	1635.2	828.75 (simulated)	4196.8-9768	518.8

[1] An CIM column includes the circuitry and computations that precede the input to a single ADC.
[2] Within 0.4ms retention time.
[3] Assuming 1OP=1 addition or 1 multiplication.
[4] Normalized computing density= input precision × weight precision × computing density.
[5] Measured with 25%-to-75% input ratio and assuming 1OP=1 addition or 1 multiplication.
[6] Normalized energy efficiency= input precision × weight precision × energy efficiency.

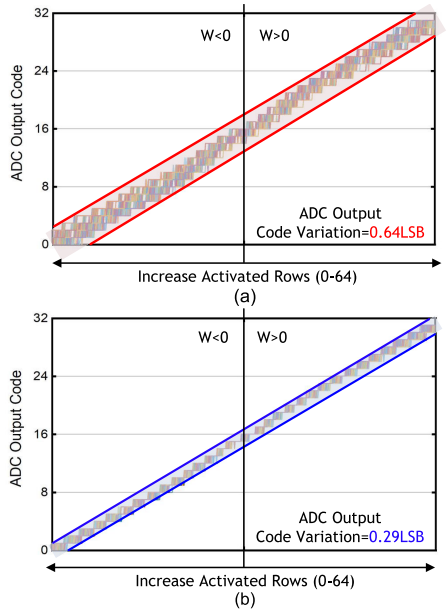


Fig. 15. Measured transfer functions of CIM columns with different programming methods performed on the same macro. (a) Voltage programming. (b) Current programming.

a 2.2× macro-level variation reduction is achieved with the proposed current-programming technique. To measure the programming speed of the proposed voltage–current two-step write driver, we program the 15-level weights to the CIM macro. The programming process involved a fixed voltage-domain coarse writing time of 5 ns, followed by different current-domain fine-tuning times ranging from 10 to 70 ns in increments of 10 ns. By sweeping the activations and collecting the average outputs of different columns, the corresponding transfer functions are measured, as shown in Fig. 16. For the large weights, the storage node of eDRAM cannot be charged and settled to target values due to insufficient current-domain fine-tuning time. Therefore, these transfer functions cannot achieve the target dynamic ranges. However, as the fine-tuning time increases, the transfer functions of different weights become more separated and gradually reach the target dynamic ranges. It can be seen that the transfer functions of all weights are settled at 65 ns, which match well with the simulation.

B. Retention Time and NN Characteristics

The analog weights in CIM cells suffer leakage, so we characterize the retention time of the CIM macro. The retention time is measured according to the following steps. First, the data in all CIM cells are written to "+7/−7". Subsequently, an immediate readout of the programed values is performed.

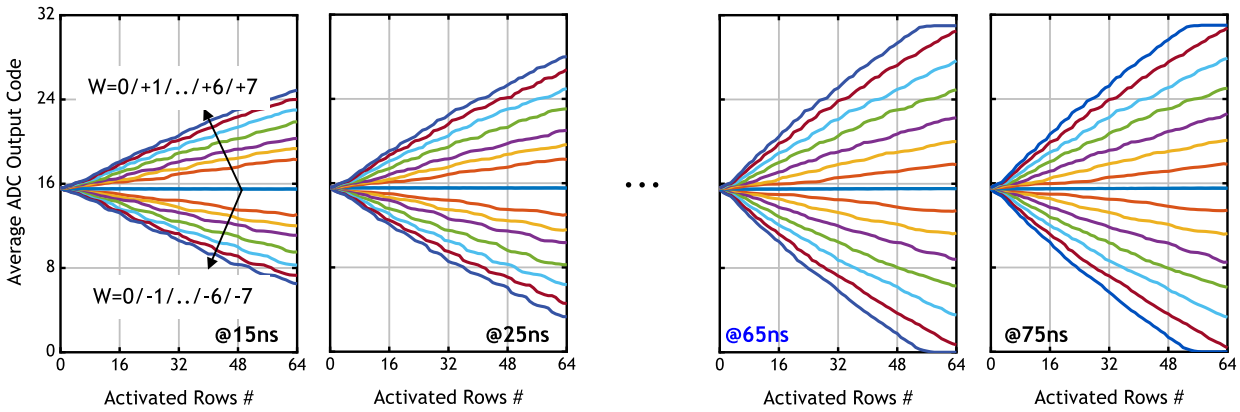


Fig. 16. Measured transfer functions of 15 weight levels with 15-/25-/65-/75-ns programming time.

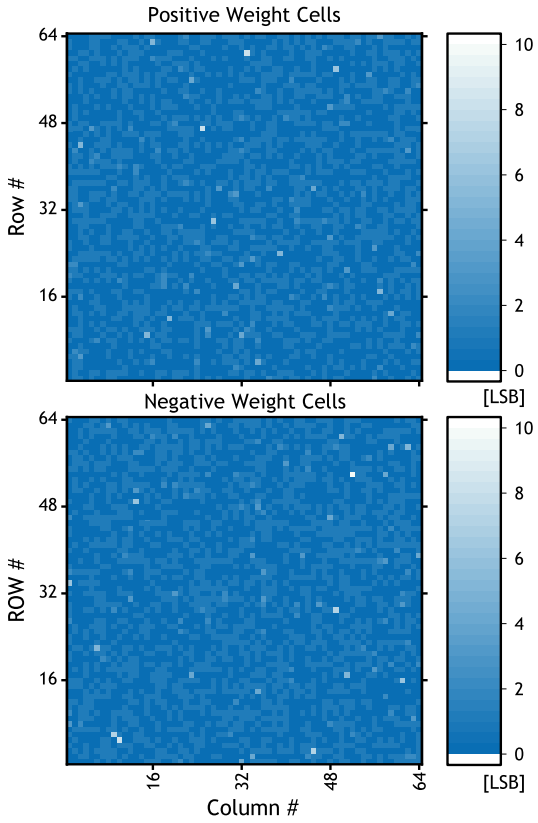


Fig. 17. Measured retention time bit map when positive and negative weights are programed.

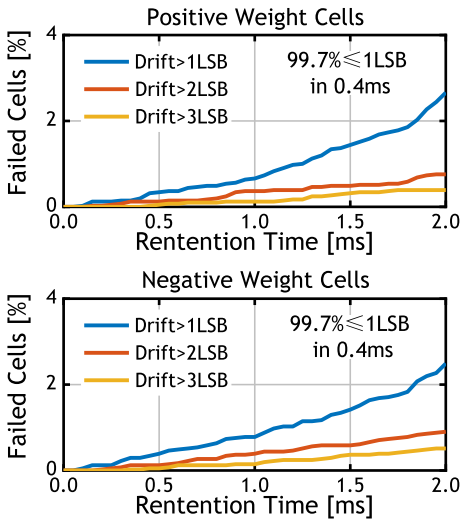
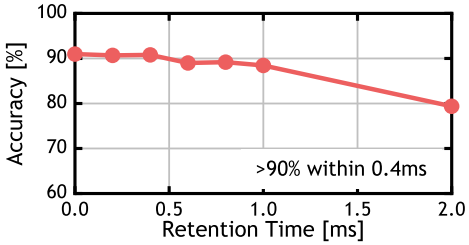


Fig. 18. Measured retention time distribution when positive and negative weights are programed.

Different from the multi-row accessing in CIM mode, the data in macro are readout row-by-row via the ADCs. In addition, the RWL pulse width is tuned to leverage the dynamic range of ADCs. Then, the data in the macro are readout again after a known retention time. By subtracting the two readout values, the corresponding drift value of the eDRAM cell during the known retention time is obtained. With the drift value calculated in this way, the error caused by ADC offset is canceled. The measured retention bit map and retention time distribution are shown in Figs. 17 and 18, respectively. Within 0.4 ms, 99.7% of cells realize less than 1-LSB drift.

To validate the inference capability of the analog CIM macro, a 4-b-quantized ResNet CNN [1] is trained to perform the CIFAR10 image classification. The reference software accuracy is 91. 67% on the CIFAR10 dataset, and our hardware achieves 90.78% under fresh state and >90% with 0.4 ms retention time, as shown in Fig. 19. The refresh overheads are also measured, as shown in Fig. 20. During one refresh cycle, the measured energy consumption and delay are 1204 pJ and 64×65 ns $= 4.16$ μs, respectively. To ensure 90% classification accuracy, a refresh interval of 0.4 ms is required, which allows the macro to perform 2.2k computing cycles (One computing cycle = 180 ns). Thus, the refresh overhead of throughput is only 4.16 μs/(400–4.16 μs) = 1.1%. In addition, within a 180-ns computing cycle, the macro simultaneously executes 64×64 MACs. Thus, the total number of operations performed by the macro during a single refresh interval amounts to $2.2k \times 2 \times 64 \times 64$ (1 MAC = 2 operations). Consequently, the refresh energy is amortized to 1204 pJ/($2.2k \times 2 \times 64 \times 64$) = 0.07 fJ/operation. Without the refresh energy, the measured 4-b-MAC energy efficiency is 233–304 TOPS/W, which is equivalent to 3.29–4.29 fJ/operation. When considering the refresh energy, the 4-b-MAC energy increased to 3.36–4.36 fJ/operation (i.e., 229–298 TOPS/W). In current-programming CIM, the storage-node capacitance controls the



Neural Network Paramaters	
Precision	IN:4b W:4b OUT:5b
Dataset	CIFAR-10
DNN Model	ResNet-20
Software Inferemce Accuracy (INT4)	91.67%
Measured Inference Accuracy (5K Image)	90.96% (0.0ms) 90.78% (0.4ms)

Fig. 19. DNN performance of prototype chip.

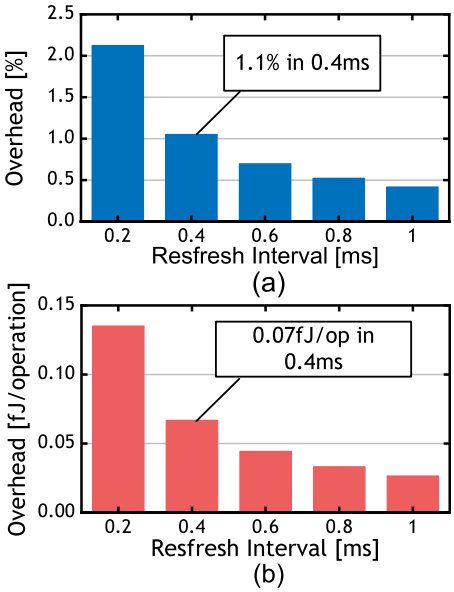


Fig. 20. Refresh overhead of (a) throughput and (b) energy consumption.

trade-off between compute accuracy/refresh overhead and density. In this prototype chip, the addition of a 10-fF MOM capacitor on the eDRAM storage node ensures an almost negligible refresh cost. However, the added MOM capacitor dominates the cell layout, resulting in a 2-μm^2/b normalized area. This cell area can be further optimized based on the required cell SNR and retention time.

V. Conclusion

In summary, this work presents and demonstrates a current-programming CIM macro with 3T1C MLC eDRAM cells. The current-programming technique enables 3T1C cell to operate at sub-micromaphere currents with reduced variations, which significantly improves the compute SNR and energy efficiency. Furthermore, this technique also allows for MLC programming without the need for calibration. The dynamic-cascode read structure in 3T1C cell reduces the computing-current sensitivity to bitline voltage. In addition, a voltage–current two-step write driver is proposed to speed up the sub-micromaphere-current programming. A 65-nm prototype demonstrates a 2.2× reduction in macro-level variation through current programming. With a refresh interval of 0.4 ms, the macro achieves >90% inference accuracy on CIFAR10. With input sparsity ranging from 25% to 75%, the macro achieves energy efficiencies of 233–304 TOPS/W for 4-b-MAC operations.

References

[1] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 770–778.

[2] D. Amodei et al., "Deep speech 2: End-to-end speech recognition in english and Mandarin," in *Proc. Mach. Learn. Res.*, 2016, pp. 173–182.

[3] M. Horowitz, "Computing's energy problem (and what we can do about it)," in *IEEE Int. Solid-State Circuits Conf. (ISSCC) Dig. Tech. Papers*, Feb. 2014, pp. 10–14.

[4] M. Kang, S. K. Gonugondla, and N. R. Shanbhag, "Deep in-memory architectures in SRAM: An analog approach to approximate computing," *Proc. IEEE*, vol. 108, no. 12, pp. 2251–2275, Dec. 2020.

[5] N. Verma et al., "In-memory computing: Advances and prospects," *IEEE Solid State Circuits Mag.*, vol. 11, no. 3, pp. 43–55, Summer 2019.

[6] 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.

[7] 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.

[8] W.-S. Khwa et al., "A 65nm 4Kb algorithm-dependent computing-in-memory SRAM unit-macro with 2.3ns and 55.8TOPS/W fully parallel product-sum operation for binary DNN edge processors," in *IEEE Int. Solid-State Circuits Conf. (ISSCC) Dig. Tech. Papers*, Feb. 2018, pp. 496–498.

[9] X. Si et al., "A twin-8T SRAM computation-in-memory macro for multiple-bit CNN-based machine learning," in *IEEE Int. Solid-State Circuits Conf. (ISSCC) Dig. Tech. Papers*, Feb. 2019, pp. 396–398.

[10] N. R. Shanbhag and S. K. Roy, "Comprehending in-memory computing trends via proper benchmarking," in *Proc. IEEE Custom Integr. Circuits Conf. (CICC)*, Apr. 2022, pp. 01–07.

[11] S. K. Gonugondla, C. Sakr, H. Dbouk, and N. R. Shanbhag, "Fundamental limits on energy-delay-accuracy of in-memory architectures in inference applications," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 41, no. 10, pp. 3188–3201, Oct. 2022.

[12] M. Kang, Y. Kim, A. D. Patil, and N. R. Shanbhag, "Deep in-memory architectures for machine learning–accuracy versus efficiency trade-offs," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 67, no. 5, pp. 1627–1639, May 2020.

[13] H. Valavi, P. J. Ramadge, E. Nestler, and N. Verma, "A 64-tile 2.4-mb in-memory-computing CNN accelerator employing charge-domain compute," *IEEE J. Solid-State Circuits*, vol. 54, no. 6, pp. 1789–1799, Jun. 2019.

[14] 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.

[15] H. Jia et al., "Scalable and programmable neural network inference accelerator based on in-memory computing," *IEEE J. Solid-State Circuits*, vol. 57, no. 1, pp. 198–211, Jan. 2022.

[16] H. Wang, R. Liu, R. Dorrance, D. Dasalukunte, D. Lake, and B. Carlton, "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 J. Solid-State Circuits*, vol. 58, no. 4, pp. 1037–1050, Apr. 2023.

[17] S. Xie, C. Ni, A. Sayal, P. Jain, F. Hamzaoglu, and J. P. Kulkarni, "eDRAM-CIM: Compute-in-memory design with reconfigurable embedded-dynamic-memory array realizing adaptive data converters and charge-domain computing," in *IEEE Int. Solid-State Circuits Conf. (ISSCC) Dig. Tech. Papers*, Feb. 2021, pp. 248–250.

[18] S. Xie, C. Ni, P. Jain, F. Hamzaoglu, and J. P. Kulkarni, "Gain-cell CIM: Leakage and bitline swing aware 2T1C gain-cell eDRAM compute in memory design with bitline precharge DACs and compact Schmitt trigger ADCs," in *Proc. IEEE Symp. VLSI Technol. Circuits (VLSI Technol. Circuits)*, Jun. 2022, pp. 112–113.

[19] C. Yu, T. Yoo, H. Kim, T. T. Kim, K. C. T. Chuan, and B. Kim, "A logic-compatible eDRAM compute-in-memory with embedded ADCs for processing neural networks," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 68, no. 2, pp. 667–679, Feb. 2021.

[20] S. Ha, S. Kim, D. Han, S. Um, and H.-J. Yoo, "A 36.2 dB high SNR and PVT/leakage-robust eDRAM computing-in-memory macro with segmented BL and reference cell array," *IEEE Trans. Circuits Syst. II, Exp. Briefs*, vol. 69, no. 5, pp. 2433–2437, May 2022.

[21] Z. Chen, X. Chen, and J. Gu, "A 65nm 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. 240–242.

[22] S. Kim et al., "DynaPlasia: An eDRAM in-memory-computing-based reconfigurable spatial accelerator with triple-mode cell for dynamic resource switching," in *IEEE Int. Solid-State Circuits Conf. (ISSCC) Dig. Tech. Papers*, Feb. 2023, pp. 256–258.

[23] J. Song et al., "A calibration-free 15-level/cell eDRAM computing-in-memory macro with 3T1C current-programmed dynamic-cascoded MLC achieving 233-to-304-TOPS/W 4b MAC," in *Proc. IEEE Custom Integr. Circuits Conf. (CICC)*, Apr. 2023, pp. 1–2.

[24] H. Kim, T. Yoo, T. T. Kim, and B. Kim, "Colonnade: A reconfigurable SRAM-based digital bit-serial Compute-In-Memory macro for processing neural networks," *IEEE J. Solid-State Circuits*, vol. 56, no. 7, pp. 2221–2233, Jul. 2021.

[25] D. Wang, C.-T. Lin, G. K. Chen, P. Knag, R. K. Krishnamurthy, and M. Seok, "DIMC: 2219TOPS/W 2569F2/b digital in-memory computing macro in 28nm based on approximate arithmetic hardware," in *IEEE Int. Solid-State Circuits Conf. (ISSCC) Dig. Tech. Papers*, Feb. 2022, pp. 266–268.

[26] Y.-D. Chih et al., "An 89TOPS/W and 16.3TOPS/mm^2 all-digital SRAM-based full-precision compute-in memory macro in 22nm for machine-learning edge applications," in *IEEE Int. Solid-State Circuits Conf. (ISSCC) Dig. Tech. Papers*, Feb. 2021, pp. 252–254.

[27] H. Fujiwara et al., "A 5-nm 254-TOPS/W 221-TOPS/mm^2 fully-digital computing-in-memory macro supporting wide-range dynamic-voltage-frequency scaling and simultaneous MAC and write operations," in *IEEE Int. Solid-State Circuits Conf. (ISSCC) Dig. Tech. Papers*, vol. 65, Feb. 2022, pp. 1–3.

[28] H. Mori et al., "A 4nm 6163-TOPS/W/b 4790 $-TOPS/mm^2/b$ SRAM based digital-computing-in-memory macro supporting bit-width flexibility and simultaneous MAC and weight update," in *IEEE Int. Solid-State Circuits Conf. (ISSCC) Dig. Tech. Papers*, Feb. 2023, pp. 132–134.

[29] J. Yue et al., "STICKER-IM: A 65 nm computing-in-memory NN processor using block-wise sparsity optimization and inter/intra-macro data reuse," *IEEE J. Solid-State Circuits*, vol. 57, no. 8, pp. 2560–2573, Aug. 2022.

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

[31] G. Wegmann, E. A. Vittoz, and F. Rahali, "Charge injection in analog MOS switches," *IEEE J. Solid-State Circuits*, vol. 22, no. 6, pp. 1091–1097, Dec. 1987.

[32] X. Tang, X. Yang, J. Liu, W. Shi, D. Z. Pan, and N. Sun, "A 0.4-to-40MS/s 75.7dB-SNDR fully dynamic event-driven pipelined ADC with 3-stage cascoded floating inverter amplifier," in *IEEE Int. Solid-State Circuits Conf. (ISSCC) Dig. Tech. Papers*, Feb. 2021, pp. 376–378.

[33] X. Tang et al., "A bandwidth-adaptive pipelined SAR ADC with three-stage cascoded floating inverter amplifier," *IEEE J. Solid-State Circuits*, vol. 58, no. 9, pp. 2564–2574, May 2023.

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

[35] C.-C. Liu, S.-J. Chang, G.-Y. Huang, and Y.-Z. Lin, "A 10-bit 50-MS/s SAR ADC with a monotonic capacitor switching procedure," *IEEE J. Solid-State Circuits*, vol. 45, no. 4, pp. 731–740, Apr. 2010.

Jiahao Song (Member, IEEE) received the B.Sc. degree (Hons.) from the University of Electronic Science and Technology of China (UESTC), Chengdu, China, in 2017, and the Ph.D. degree from Peking University, Beijing, China, in 2022.

He is currently a Post-Doctoral Researcher with Peking University. His interests include computing-in-memory (CIM), hardware security, and low-power sensor node.

Xiyuan Tang (Member, IEEE) received the B.Sc. degree (Hons.) from the School of Microelectronics, Shanghai Jiao Tong University, Shanghai, China, in 2012, and the M.S. and Ph.D. degrees in electrical engineering from The University of Texas at Austin, Austin, TX, USA, in 2014 and 2019, respectively.

He was a Design Engineer with Silicon Laboratories, Austin, from 2014 to 2017, where he was involved in the RF receiver design. From 2019 to 2021, he was a Post-Doctoral Researcher with The University of Texas at Austin. He is currently an Assistant Professor with Peking University, Beijing, China. His research interests include digitally assisted data converters, low-power mixed-signal circuits, and analog data processing.

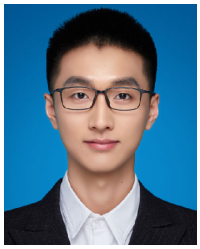
Dr. Tang was a recipient of the IEEE Solid-State Circuits Society Rising Stars in 2020, the Best Paper Award at Silicon Labs Tech Symposium in 2016, the National Scholarship in 2011, and the Shanghai Scholarship in 2010.

Haoyang Luo received the B.Sc. degree from the Beijing Institute of Technology, Beijing, China, in 2021. He is currently pursuing the Ph.D. degree with Peking University, Beijing.

His current research interests include hardware security, in-memory computing, and neural interfacing circuits.

Haoyi Zhang (Graduate Student Member, IEEE) received the B.S. degree in microelectronics from Beihang University, Beijing, China, in 2022. He is currently pursuing the Ph.D. degree in microelectronics with Peking University, Beijing.

His research interests include analog design automation and mixed-signal circuit design.



Xin Qiao (Graduate Student Member, IEEE) received the B.S. degree from the Beijing University of Posts and Telecommunications (BUPT), Beijing, China, in 2019. He is currently pursuing the Ph.D. degree with Peking University, Beijing.

His current research interests include computing-in-memory (CIM) and spiking neural network processor.

Zixuan Sun (Graduate Student Member, IEEE) is currently pursuing the Ph.D. degree with the School of Integrated Circuits, Peking University, Beijing, China.

His current research interests include reliability and modeling of nanoscale CMOS devices.

Xiangxing Yang (Member, IEEE) received the B.S. degree in electronics engineering from the University of Electronic Science and Technology of China, Chengdu, China, in 2016, and the M.S. and Ph.D. degrees from the Department of Electrical and Computer Engineering, The University of Texas at Austin, Austin, TX, USA, in 2021 and 2022, respectively.

He is currently with pSemi Corporation, Austin. His research interests include analog and mixed-signal circuit design for edge computing.

Zihan Wu (Graduate Student Member, IEEE) received the B.S. degree from the Huazhong University of Science and Technology, Wuhan, China, in 2019, and the master's degree from Fudan University, Shanghai, China, in 2021. He is currently pursuing the Ph.D. degree with Peking University, Beijing, China.

His current research interests include computing-in-memory (CIM), low-power circuit design, and novel computing paradigms.

Yuan Wang (Member, IEEE) received the Ph.D. degree in microelectronics from Peking University, Beijing, China, in 2006.

He is currently a Professor with the School of Integrated Circuits, Peking University. He has authored or coauthored more than 100 IEEE technical articles and conference contributions. His current research interests include CMOS memory circuits, neuromorphic computing, in-memory computing, and deep neural networks (DNNs).

Runsheng Wang (Member, IEEE) received the B.S. and Ph.D. degrees (Hons.) from Peking University, Beijing, China, in 2005 and 2010, respectively.

From November 2008 to August 2009, he was a Visiting Scholar with Purdue University, West Lafayette, IN, USA. He joined Peking University, in 2010, where he is currently a Full Professor with the School of Integrated Circuits and also he serves as the Associate Dean of the School of Electronics Engineering and Computer Science. He has authored or coauthored one book, four book chapters, and over 190 scientific articles, including more than 40 articles published in the International Electron Devices Meeting (IEDM) and Symposium on VLSI Technology (VLSI-T). He has been granted over 20 U.S. patents and over 30 Chinese patents. His current research interests include nanoscale CMOS devices, characterization and reliability, design-technology co-optimization (DTCO) and EDA, and emerging technologies and circuits for new-paradigm computing.

Dr. Wang was awarded the IEEE EDS Early Career Award by the IEEE Electron Device Society (EDS), the National Distinguished Young Scholars by the National Natural Science Foundation of China (NSFC), the Natural Science Award (First Prize) by the Ministry of Education (MOE) of China, and many other awards. He serves on the Editorial Board of IEEE TRANSACTIONS ON ELECTRON DEVICES and Science China Information Sciences. He has also served on the Technical Program Committee for many IEEE conferences, including IEDM, IRPS, EDTM, and IPFA.

Ru Huang (Fellow, IEEE) received the B.S. (Hons.) and M.S. degrees in electronic engineering from Southeast University, Nanjing, China, in 1991 and 1994, respectively, and the Ph.D. degree in microelectronics from Peking University, Beijing, China, in 1997. Since 1997, she has been a Faculty Member of Peking University, where she is currently a Professor. She is also serving as the President of Southeast University, since 2022. She has authored or coauthored five books, five book chapters, and more than 300 papers, including more than 100 articles in IEDM (46 IEDM papers from 2007 to 2021), VLSI Technology Symposium, IEEE EDL, and IEEE T-ED. She has delivered over 50 keynote/invited talks at conferences and seminars. She has been granted over 300 patents including 49 U.S. patents. Her research interests include nano-scaled CMOS devices, ultralow-power new devices, new devices for neuromorphic computing, emerging memory technology, and device variability/reliability.

Dr. Huang is an Elected Academician of the Chinese Academy of Science and an Elected Member of TWAS Fellow.