Journal of Semiconductors
JOS iopscience.iop.org/jos
www.jos.ac.cn

A 28 nm 576K RRAM-based computing-in-memory macro featuring hybrid programming with area efficiency of 2.82 TOPS/mm^2

Siqi Liu, Songtao Wei, Peng Yao, Dong Wu, Lu Jie, Sining Pan, Jianshi Tang, Bin Gao, He Qian, and Huaqiang Wu

Citation: S Q Liu, S T Wei, P Yao, D Wu, L Jie, S N Pan, J S Tang, B Gao, H Qian, and H Q Wu, A 28 nm 576K RRAM–based computing–in–memory macro featuring hybrid programming with area efficiency of 2.82 TOPS/mm^2[J]. *J. Semicond.*, 2025, 46(6), 062304.

View online: https://doi.org/10.1088/1674–4926/24100017

ARTICLES

Journal of Semiconductors, 2025, 46, 062304
DOI: 10.1088/1674-4926/24100017
CSTR: 32376.14.1674-4926.24100017

A 28 nm 576K RRAM-based computing-in-memory macro featuring hybrid programming with area efficiency of 2.82 TOPS/mm^2

Siqi Liu[1], Songtao Wei[1], Peng Yao[1, †], Dong Wu[1], Lu Jie[1], Sining Pan[1], Jianshi Tang[1], Bin Gao[1, 2], He Qian[1], and Huaqiang Wu[1, 2]

[1]School of Integrated Circuits, Tsinghua University, Beijing 100083, China

[2]International Innovation Center of Tsinghua University, Shanghai 200062, China

Abstract: Computing-in-memory (CIM) has been a promising candidate for artificial-intelligent applications thanks to the absence of data transfer between computation and storage blocks. Resistive random access memory (RRAM) based CIM has the advantage of high computing density, non-volatility as well as high energy efficiency. However, previous CIM research has predominantly focused on realizing high energy efficiency and high area efficiency for inference, while little attention has been devoted to addressing the challenges of on-chip programming speed, power consumption, and accuracy. In this paper, a fabricated 28 nm 576K RRAM-based CIM macro featuring optimized on-chip programming schemes is proposed to address the issues mentioned above. Different strategies of mapping weights to RRAM arrays are compared, and a novel direct-current ADC design is designed for both programming and inference stages. Utilizing the optimized hybrid programming scheme, 4.67× programming speed, 0.15× power saving and 4.31× compact weight distribution are realized. Besides, this macro achieves a normalized area efficiency of 2.82 TOPS/mm^2 and a normalized energy efficiency of 35.6 TOPS/W.

Key words: computing-in-memory; on-chip programming scheme; hybrid programming; resistive random access memory; matrix-vector-multiplication acceleration

Citation: S Q Liu, S T Wei, P Yao, D Wu, L Jie, S N Pan, J S Tang, B Gao, H Qian, and H Q Wu, A 28 nm 576K RRAM-based computing-in-memory macro featuring hybrid programming with area efficiency of 2.82 TOPS/mm^2[J]. *J. Semicond.*, 2025, 46(6), 062304. https://doi.org/10.1088/1674-4926/24100017

1. Introduction

Computing-in-memory (CIM) has advantages in data-intensive tasks as it saves the latency and energy consumption of frequent data transfer between computation and storage units[1, 2]. Compared to digital counterparts, analog CIM is superior due to higher parallelism and energy efficiency[3, 4]. Among different memory types, RRAM-based analog CIM is promising due to its high storage density, non-volatility, low operating power, compatibility with CMOS process, and so on. Varied RRAM-based analog accelerators have been extensively surveyed[5–14] to unleash its potential in inference applications based on deep neural networks.

However, several challenges still exist in prior arts based on RRAM-CIM. Previous work mainly focuses on improving the energy efficiency (EF) and area efficiency (AF) in terms of inferance, without comprehensive research of macro-level on-chip programming. In traditional memory applications, RRAM writing only cares about the conductance of 1T1R. However, in CIM applications, RRAM writing needs to consider the state of 2T2R weights[15], leading to the desire for different on-chip programming methods. Presently, the actual RRAM-based CIM macros still suffer from low programming speed, high programming power consumption and limited programming accuracy. Besides, an energy-efficient analog-to-digital converter (ADC) featuring accurate current sampling and quantization has yet to be explored to improve the EF of the overall system.

In this paper, a 28 nm RRAM-based macro is fabricated to address the above issues. Different from prior works[15, 16], a novel weight-mapping flow based on ADC reusing scheme is adopted. Instead of verifying the conductance of RRAM with independent sense amplifier circuits, the ADC module in the calculation stage is also employed to sense RRAM states. This resuing scheme keeps the same resistive and capacitaive loads and activates the ADC modules for both read and calculation states, leading to accurate weight mapping and reliable matrix-vector-multiplication (MVM) operations. In addition, a dual-switch direct-current ADC (DSDC-ADC) for precise sampling and quantization for both the read and calculation stages is designed, which features an opamp-less structure to further improve the overall energy efficiency. Based on the fabricated macro, this paper explores different on-chip programming schemes for CIM applications, i.e., 1T1R, 2T2R, and hybrid mode, respectively. Meanwhile, the programming speed, accuracy and power consumption are investigated and compared in terms of different programming modes.

2. Macro architecture and circuitry implementation

This architecture of the proposed RRAM-based macro is presented in Fig. 1. The 2T2R array consists of a total of 576 ×

Correspondence to: P Yao, pyao@mail.tsinghua.edu.cn
Received 12 OCTOBER 2024; Revised 23 DECEMBER 2024.

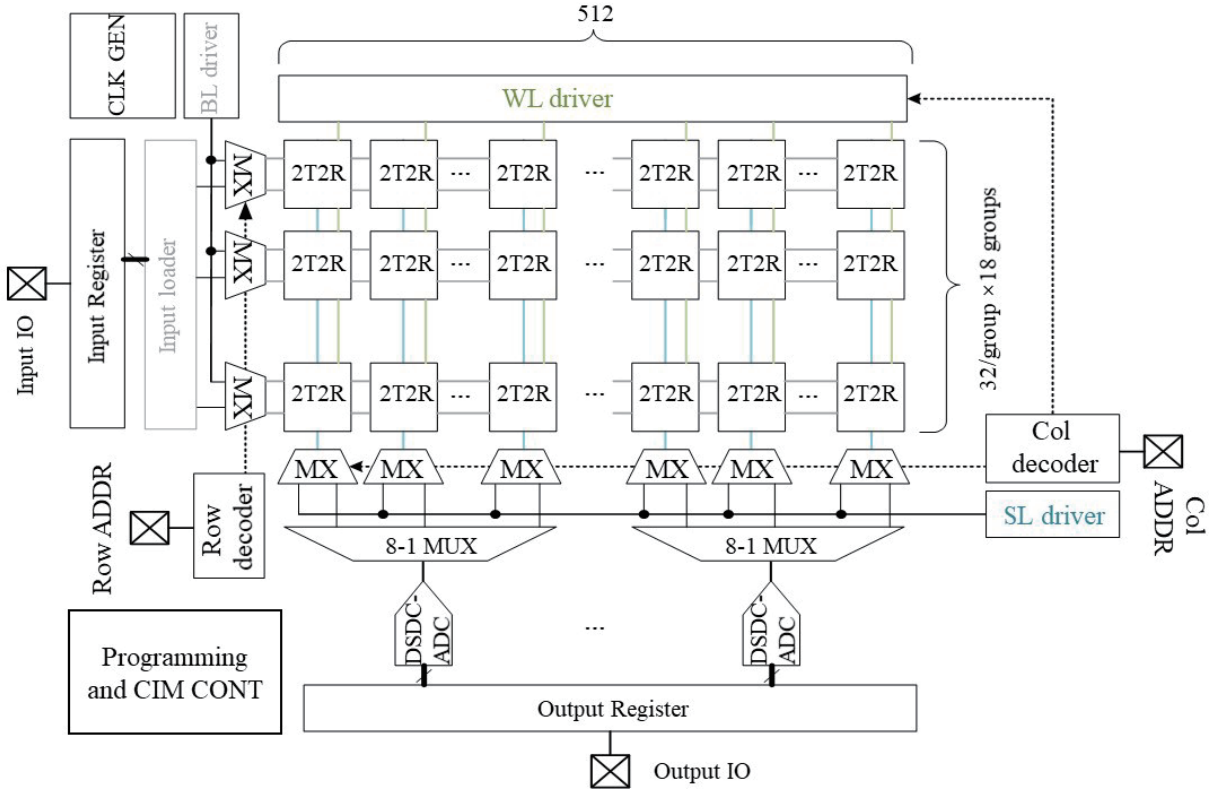

Fig. 1. (Color online) 576k macro architecture and configuration.

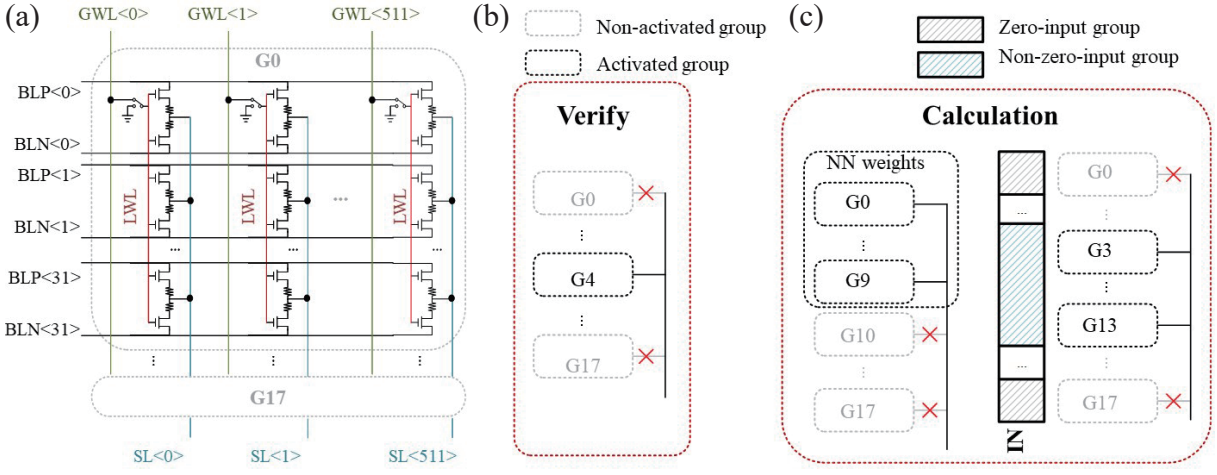

Fig. 2. (Color online) (a) Proposed 576k 2T2R array and WL segmentation diagram; finer control across column direction in (b) verify mode and (c) calculation mode.

512 2T2R cells, each of which represents a signed analog weight. The 2T2R array is divided into 18 groups along the column direction out of the consideration of flexible configuration of parallelism in the computation phase, which will be detailed in Section 2.1. The input loader applies up to 576 pairs of signed 1b-input concurrently to support fully parallel computing across input channels. 64 DSDC-ADCs are equipped to sample and quantize current from 64 columns from the 2T2R array. It takes 8 cycles to finish quantization across all 512 columns. The row decoder, BL driver, and MUXs ensure that under the guidance of signal Row ADDR, specific bit lines (BLs) in the array are selected in the programming mode to apply the programming voltage or read voltage. Similarly, in the column direction, the column decoder, driver, and MUXs guarantee that specific source lines (SLs)/word lines (WLs) in the array are selected under the guidance of signal Col ADDR in the programming mode to apply the programming voltage or read voltage. Besides, input and output registers are used to implement the interaction between the core circuits and I/O. Furthermore, the CLK generator, the programming and the CIM controller (CONT) are applied to guarantee proper working flow.

2.1. Array and segmented WL structure (SWS)

Fig. 2(a) depicts the overall structure of the proposed 576K 2T2R array. There is a total number of 576 pairs of BLs and 512 SLs, which are perpendicular to each other. One 2T2R cell is located at the intersection of one pair of BL and SL to represent a signed multibit weight and alleviate the IR drop effect resulting from considerable accumulated DC current on SL[15]. Global WLs (GWLs) are parallel with SLs and seg-

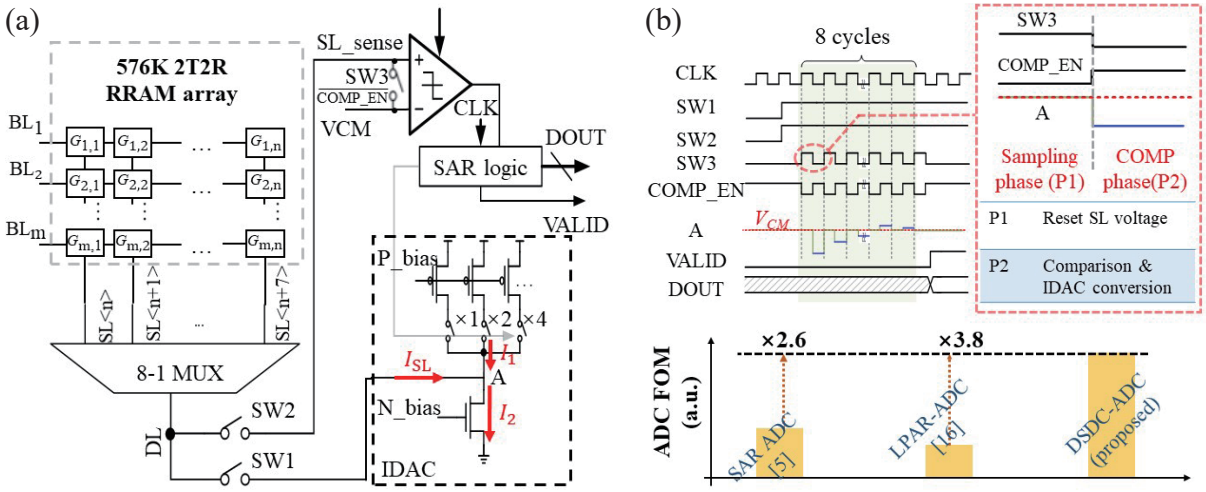

Fig. 3. (Color online) (a) Structure of DSDC-ADC. (b) Timing diagram of the proposed DSDC-ADC and comparison.

mented into 18 local WLs (LWLs), which enables finer-grained control over the cells of the selected column. In the Verify phase, one cell can be selected in a two-step manner. The segment containing the selected cell is first activated according to LWL<17:0>. Then one specific cell is selected according to the voltages of BLs in this group and generates a current indicating its conductance state on SL. As a result, the unselected cells in other segments contribute no additional current, which improves the reliability of single cell verification, as depicted in Fig. 2(b). In the Calculation phase, selected segment(s) can be activated to perform multiplication-and-accumulation (MAC). As a result, the unselected segments will not affect the current outputs on SLs. On the other hand, when the network scale is smaller than the array size or the input is sufficiently sparse, as shown in Fig. 2(c), where a certain number of all-zero input group exist, SWS helps to improve the computing accuracy by eliminating the current generated from inactivated segments. On the other hand, a trade-off can be achieved between computation accuracy and throughput under different computing parallelism configurations by SWS.

2.2. Implementation of DSDC-ADC

Fig. 3(a) presents the structure of the proposed DSDC-ADC, which mainly consists of a comparator, a current DAC (IDAC), SAR logic and several switches. This IDAC is bidirectional, in which I_2 remains at full-scale current (IFS), while the value of I_1 is controlled by the SAR logic. Besides, the bias of this IDAC can be tuned to allow the range of the proposed ADC adjustable.

The operation of the proposed ADC can be divided into 2 phases, which is shown in Fig. 3(b). During the sampling phase, SW1–SW3 are closed, resulting in the voltage of selected SL (DL, data line) being equal to the common mode voltage V_{CM}. As a consequence, the current on SL (I_{SL_P1}) represents the MAC results between inputs and weights in this column as depicted in Eq. (1). The size of SW1 is large out of the consideration of avoiding the IR drop on this switch, which alleviates the degradation of sampling accuracy. During the comparison phase, SW3 is open and the current at node A satisfies the relationship of Eq. (2) according to Kirchhoff's current law. If the SL current sampled (I_{SL_P1}) before is substituted into I_{SL_P2} in Eq. (2) and makes this relationship invalid, the voltage at point A will deviate from V_{CM}. The subsequent comparison result between SL_sense and V_{CM} will be obtained and indicate the conversion of IDAC. These 2 phases repeat 8 times and the final digital outputs are generated when VALID becomes high.

$$I_{SL\_P1} = \sum_{i=1}^{m} \mathrm{VBL}_i \times G_i, \tag{1}$$

$$I_{SL\_P2} + I_1 + I_2 = 0. \tag{2}$$

The comparison between the proposed DSDC-ADC and ADCs in the prior art of RRAM-based CIM works is also presented in Fig. 3(b). A figure-of-merit (FOM) is defined to take speed, area, power and resolution into consideration, as shown in Eq. (3). The proposed ADC in this macro has achieved a 3.8× improvement compared to Ref. [16], and a 2.6× improvement compared to Ref. [5], validating this ADC's feasibility in boosting the overall energy efficiency.

$$\mathrm{FOM} = \frac{\text{resolution of ADC}}{\text{latency} \times \text{power} \times \text{area}}. \tag{3}$$

3. CIM programming strategy-implemenatation and assessment

3.1. Configrantion and implemantation of different programming modes

Three programming modes are supported in this fabricated macro: 1T1R mode, 2T2R mode, and the proposed hybrid programming mode. In the 1T1R mode, the current is generated by activating a single 1T1R and monitoring its current. As depicted in Fig. 4(a), if the programming error tolerance is set to 0.5 × G_{tol} for each 1T1R units, the corresponding conductance of the 2T2R cell becomes:

$$G_{2T2R} = \left(G_{tar\_P} \pm 0.5 \times G_{tol}\right) - \left(G_{tar\_N} \pm 0.5 \times G_{tol}\right) = G_{tar\_P} - G_{tar\_N} \pm G_{tol}. \tag{4}$$

It demonstrates that the error tolerance for programming 1T1R units is doubled in the final 2T2R programming outcome, significantly limiting the achievable accuracy of the

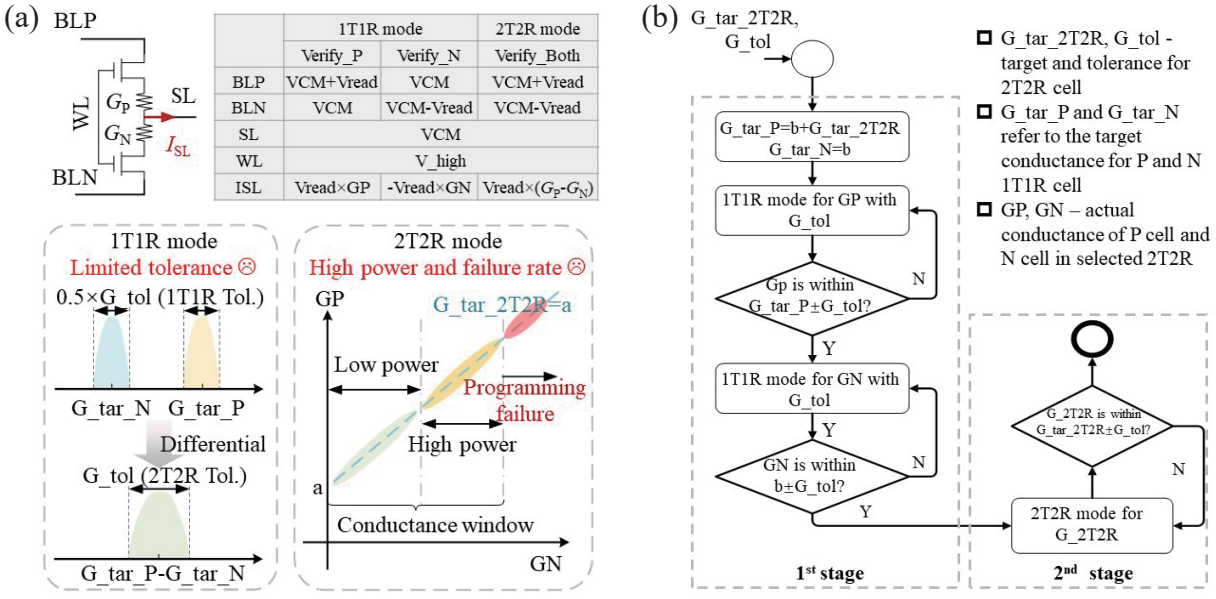

Fig. 4. (Color online) (a) Drawbacks of 1T1R and 2T2R programming modes. (b) Working flow of hybrid programming mode.

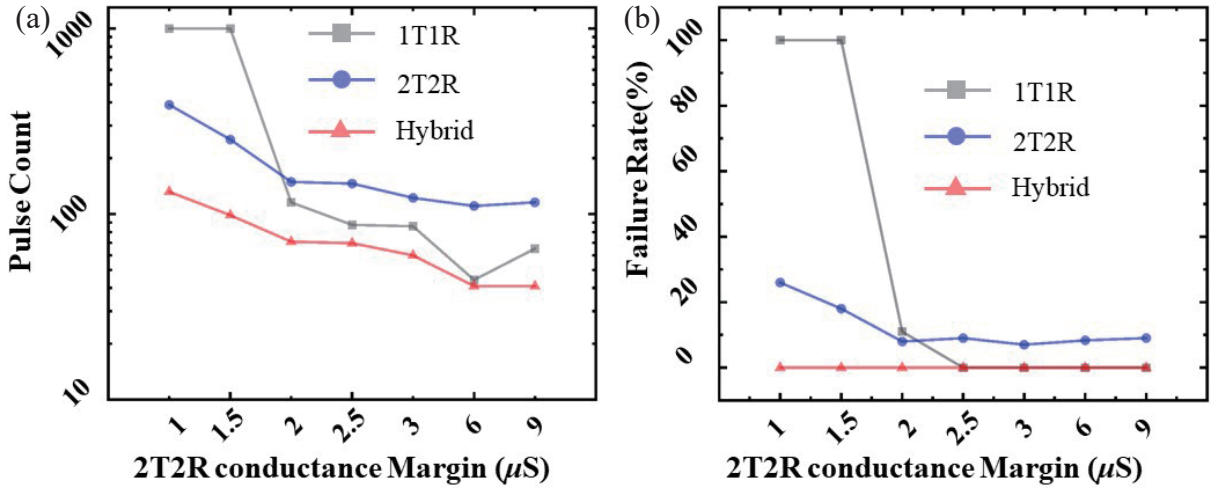

Fig. 5. (Color online) (a) Pulse number required for all tested devices under different modes. (b) Failure rate under different programming modes.

1T1R mode. Different from that in 1T1R mode, the differential current is obtained by activating both 1T1Rs in 2T2R mode, which can be regarded as a specific case of computation when input parallelism equals 1. Compared to the 1T1R mode, the 2T2R configuration offers double programming tolerance, resulting in faster programming speed. However, the conductance of each 1T1R cell is inherently uncertain and uncontrollable. Excessively high conductance values make it challenging to reset the 2T2R conductance back to the target range under limited pulse constraints, leading to increased programming failures and power consumption.

To address the constraints of the 2 programming modes mentioned above, a hybrid programming mode has been proposed, which can be described as a 2-step approach shown in Fig. 4(b). In the first step, 1T1R mode is employed for programming a single 1T1R cell with 2T2R programming tolerance. In the second step, 2T2R mode is used for programming differential 2T2R cells and ensuring the final 2T2R cell conductance converges within the target range. Consequently, the controllability of each 1T1R as well as large programming tolerance can be achieved simultaneously, enabling high programming speed, low programming failure rate and power consumption in the proposed hybrid programming mode.

3.2. Assessment of CIM programming modes

In order to compare the speed of different programming modes, the average programming pulse count (max. 1000 pulses) for all tested cells is used to assess the overall runtime under different programming modes. The comparison results in terms of programming speed are illustrated in Fig. 5(a). It is observed that when the 2T2R conductance tolerance is within the range of 1–9 μS, the hybrid programming mode requires 4.31 times fewer programming pulses compared to the 1T1R programming mode, and 2.50 times fewer programming pulses compared to the 2T2R programming mode, validating its advantage in programming speed. The measured programming failure rates under 3 programming modes are also depicted in Fig. 5(b), which is represented as the proportion of 2T2R cells whose conductance fall outside the predefined range (target ± tolerance) after programming. The programming failure rate of the hybrid mode remains the lowest (equal to zero) among the 3 programming modes when the conductance tolerance of 2T2R lies in 1–9 μS.

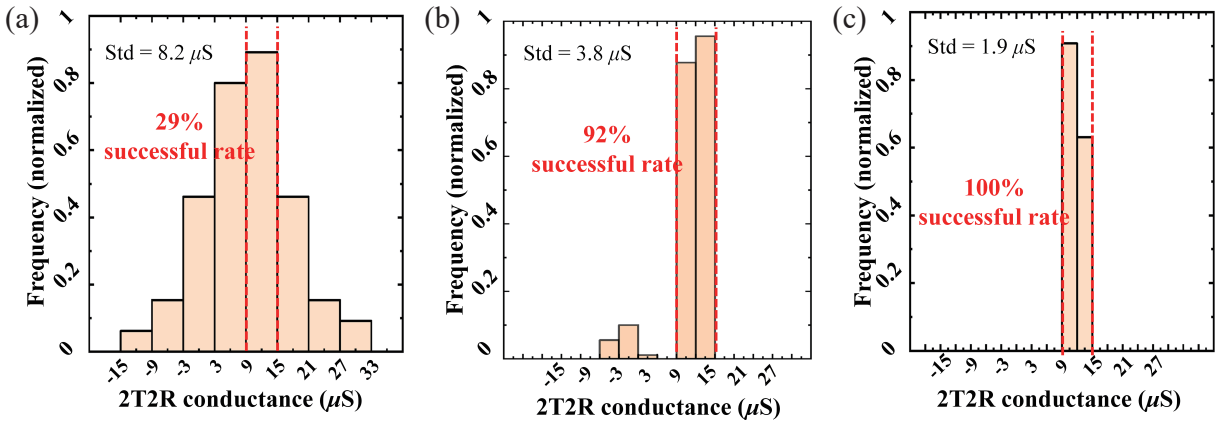

Fig. 6. (Color online) Distribution of 2T2R conductance with 6 μS tolerance setting utilizing (a) 1T1R, (b) 2T2R, and (c) hybrid programming mode.

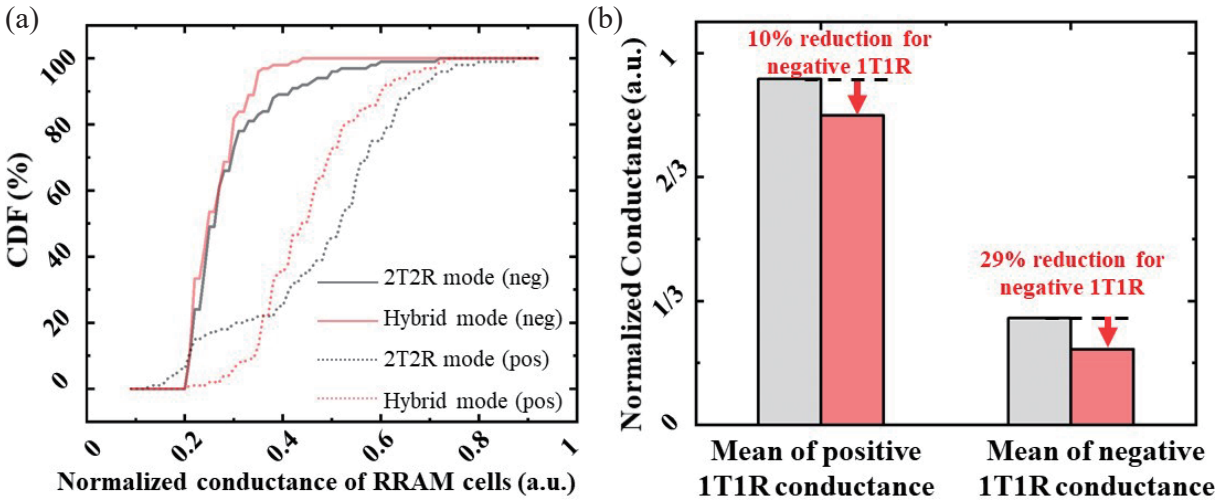

Fig. 7. (Color online) (a) Conductance values of positive 1T1R after programming. (b) Conduactance values of negative1T1R after programming.

Fig. 6 shows the measured results of the 3 programming modes in terms of programming accuracy with 12 μS 2T2R programming target and 6 μS 2T2R programming tolerance. The 2T2R unit conductance value realized by the 1T1R programming mode has a significant deviation from the theoretical value, which causes a large difference between the subsequent calculation results and the theoretical expectations. In contrast, the conductance distributions for the 2T2R and hybrid modes are more concentrated. The latter performs better in terms of programming consistency as all measured cells fall within the target conductance range, validating the superiority of the hybrid programming mode in enhancing programming accuracy.

The programming power consumption is primarily indicated by the conductance value of the 1T1R cell at the end of the programming. The larger conductance means a larger programming current and larger power consumption during programming. The target value of the 2T2R and the 2T2R tolerance are set the same for both 2T2R and hybrid programming modes. Fig. 7(a) presents the cumulative distribution of conductance of positive and negative 1T1R cells after hybrid and 2T2R programming modes. It is shown that the conductance of 1T1R cells after hybrid programming mode is consistently lower than those observed in the 2T2R mode, whether in terms of positive 1T1R cells or negative 1T1R cells. As shown in Fig. 7(b), the hybrid programming mode has 10% and 29% less mean value of 1T1R cells' conductance compared to the 2T2R mode in terms of positive and negative 1T1Rs, respectively. The result demonstrates the significant advantage of hybrid programming over the 2T2R mode in power consumption.

Fig. 8(a) depicts the relaxation of RRAM cells within 50 s under different programming modes. The relaxation of RRAM devices at time t is defined in Eq. (5) as relative deviation (RD), where G_{MAX} and G_{MIN} represent the bounds of the conductance window of 2T2R cells, $G_{i,0}$ and $G_{i,t}$ means the conductance value of the i_{th} 2T2R cell at the initial time and the time t.

$$RD_t = \frac{1}{G_{max} - G_{min}} \sqrt{\frac{\sum_{i=0}^{N} \left(G_{i,t} - G_{i,0}\right)^2}{N}} \times 100\%. \quad (5)$$

The measurement results show that when the 2T2R conductance target equals 12 μS and tolerance is set to 6 μS, the 1T1R programming mode exhibits the best performance in terms of relaxation characteristics, while the 2T2R mode shows the worst-case behavior. The hybrid programming mode falls in between these two modes. The reason for this result may be due to the programming conditions of the final step[17]. The higher the set voltage in the final programming step, the worse the relaxation of the device presents. Conversely, if the set voltage is lower, the relaxation perfor-

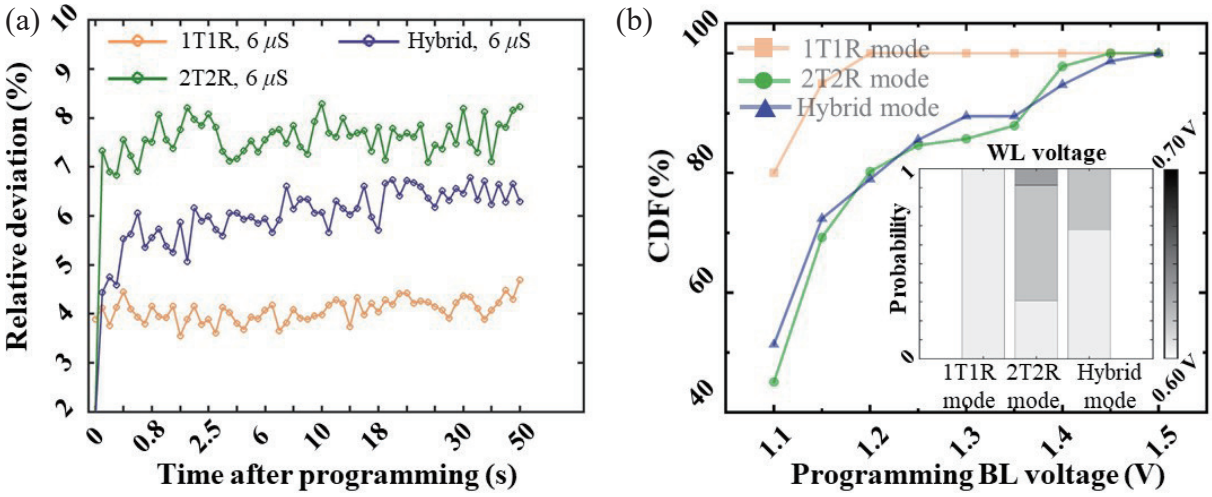

Fig. 8. (Color online) (a) RRAM relaxation under 3 programming modes. (b) Distribution of voltages across RRAM cells of the final step of SET programming.

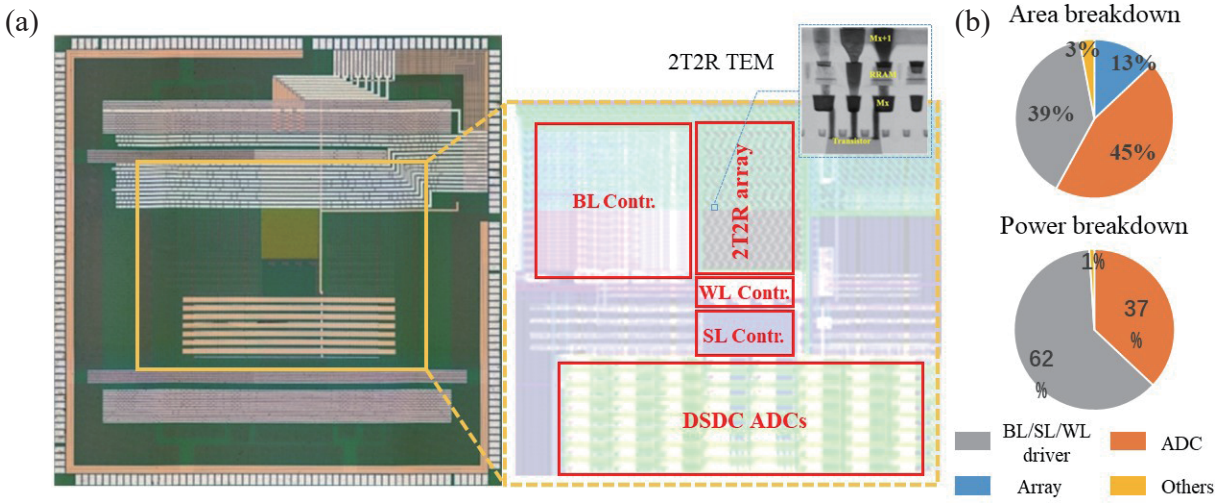

Fig. 9. (Color online) (a) Die photo and core layout. (b) Power and area breakdown.

mance of the device will be better. Fig. 8(b) shows that the cumulative distribution function (CDF) of SET voltages of the final programming step across RRAM cells, which indicates the operational voltages (BL voltages) across RRAM in 1T1R mode are lower compared to those in 2T2R and hybrid modes. Besides, the WL voltages applied in the last operation are also depicted, which indicates that the WL voltages of the final programming step hybrid mode are lower than those of the 2T2R mode. As a result, the relaxation characteristic of RRAM cells after the hybrid programming mode is superior to that after the 2T2R mode.

4. Macro overview and meansurement results

4.1. Macro overview and comparison with prior art

This 28 nm fabricated 576K macro occupies an area of 1.96 mm^2. The area and power breakdown are presented in Fig. 9. The ADC and BL driver are the two modules that take up the largest proportion in terms of both area (71%) and power consumption (99%) for accurate input voltage setup, reliable output current sampling and quantization. This macro achieves a high normalized AF of 2.82 TOPS/mm^2 and normalized EF of 35.6 TOPS/W under 1.5b-input and 1.5b-weight configuration, indicating the feasibility of this macro in data-intensive AI applications.

4.2. Multi-bit programming test and comparison of differernt programming modes

Fig. 10(a) presents the multi-bit programming result of the proposed 576K macro. A total of 5760 2T2R cells are evenly divided into 16 groups within ±48 μS and 6 μS 2T2R programming tolerance is set, aiming to test the qualification of 2T2R cells to represent 4-bit weights. It is obvious that the 2T2R cells in this macro realize the representation of 4-bit weights using hybrid programming mode, validating the reliability and efficacy of the hybrid programming mode in multi-bit array programming. Besides, a comparison of three programming modes is also summarized in Fig. 10(b). Compared to the other two modes, the hybrid programming mode demonstrates superiority in speed, power consumption, programming accuracy, and successful rate, making it the optimal solution among the three. This validates the advantages of the hybrid programming mode for large-scale programming at the macro level.

4.3. MVM testing under different parallelism

MVM measurement results are presented in Fig. 11. 256 random signed 1-bit inputs are generated under different par-

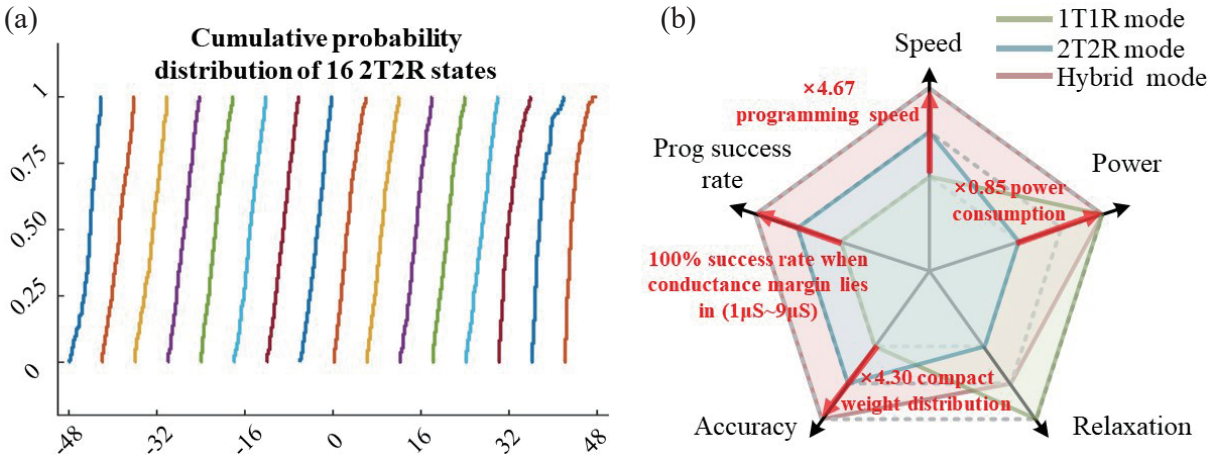

Fig. 10. (Color online) (a) Multi-bit programming results using hybrid programming mode; (b) comparison of three programming modes.

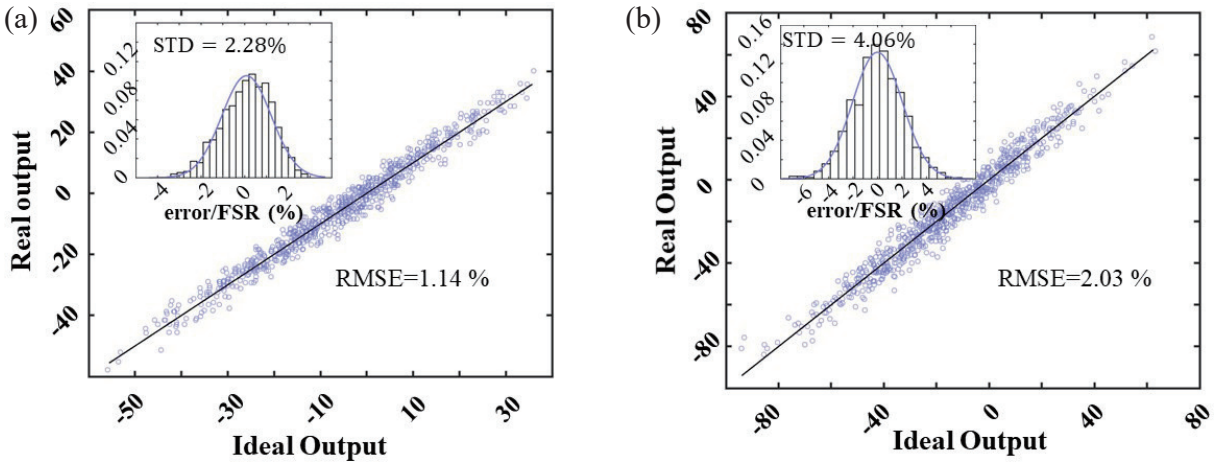

Fig. 11. (Color online) (a) MVM test results with 32 input parallelism. (b) MVM test results with 128 input parallelism.

allelism. Signed 1-bit weights are also randomly generated and mapped on the proposed 576K macro using the hybrid programming strategy. The MVM results between verified results and inputs are regarded as the ideal outputs, whereas the actual outputs refer to the tested outputs generated by applying inputs to BLs. It is shown that the root mean squared error (RMSE) between the actual output and the ideal output is 1.14% and 2.03% under 32 and 128 input parallelism configurations, respectively. This indicates the tradeoff between input parallelism and computing accuracy. Besides, the distributions of relative error compared to full-scale range (FSR) under 32 and 128 input parallelism are also presented.

5. Conclusion and future outlook

This work is the first macro-level exploration of various on-chip programming strategies in terms of RRAM-based CIM. Contributions at the circuitry level such as WL segmentation and DSDC ADC further boost the EF, AF and computing accuracy of the whole system. Different programming modes have been thoroughly investigated on the 28 nm fabricated 576K RRAM-based CIM macro, indicating the advantages of the hybrid programming mode in terms of programming speed, power, and accuracy. Besides, hybrid programming has better programming reliability due to finer operation in the last programming step compared to the 2T2R programming mode. Finally, high AF and EF have been measured, validating the qualification of the proposed macro in AI applications with reliable inference and programming. Future work will focus on realizing more sophisticated SoC systems based on the proposed macro, making RRAM-based CIM capable of executing more complex tasks. Besides, the power, area and speed of the interface circuits will be further optimized to improve the overall performance such as AF, EF and computing reliability.

Acknowledgments

This work was supported in part by the National Natural Science Foundation of China (62422405, 62025111, 62495100, 92464302), the STI 2030-Major Projects (2021ZD0201200), the Shanghai Municipal Science and Technology Major Project, and the Beijing Advanced Innovation Center for Integrated Circuits.

References

[1] Sebastian A, Le Gallo M, Khaddam-Aljameh R, et al. Memory devices and applications for in-memory computing. Nat Nanotechnol, 2020, 15(7), 529

[2] Jia H Y, Valavi H, Tang Y Q, et al. A programmable heterogeneous microprocessor based on bit-scalable in-memory computing. IEEE J Solid State Circuits, 2020, 55(9), 2609

[3] Lee K, Kim J, Park J. 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 J Solid State Circuits, 2024, 59(6), 1926

[4] Song J H, Tang X Y, Luo H Y, et al. A 4-bit calibration-free comput-

ing-In-memory macro with 3T1C current-programed dynamic-cascode multi-level-cell eDRAM. IEEE J Solid State Circuits, 2024, 59(3), 842

[5] Wan W, Kubendran R, Schaefer C, et al. A compute-in-memory chip based on resistive random-access memory. Nature, 2022, 608(7923), 504

[6] Yao P, Wu H Q, Gao B, et al. Fully hardware-implemented memristor convolutional neural network. Nature, 2020, 577, 641

[7] Chi P, Li S C, Xu C, et al. PRIME: A novel processing-in-memory architecture for neural network computation in ReRAM-based main memory. 2016 ACM/IEEE 43rd Annual International Symposium on Computer Architecture (ISCA), 2016, 27

[8] Shafiee A, Nag A, Muralimanohar N, et al. ISAAC: A convolutional neural network accelerator with in situ analog arithmetic in crossbars. 2016 ACM/IEEE 43rd Annual International Symposium on Computer Architecture (ISCA), 2016, 14

[9] Le Gallo M, Khaddam-Aljameh R, Stanisavljevic M, et al. A 64-core mixed-signal in-memory compute chip based on phase-change memory for deep neural network inference. Nat Electron, 2023, 6(9), 680

[10] Huang W H, Wen T H, Hung J M, et al. A nonvolatile AI-edge processor with 4MB SLC-MLC hybrid-mode ReRAM compute-in-memory macro and 51.4-251TOPS/W. 2023 IEEE International Solid-State Circuits Conference (ISSCC), 2023, 15

[11] Liu Y Y, Gao B, Tang J S, et al. Architecture-circuit-technology co-optimization for resistive random access memory-based computation-in-memory chips. Sci China Inf Sci, 2023, 66(10), 200408

[12] Zhou Y, Gao B, Zhang Q T, et al. Application of mathematical morphology operation with memristor-based computation-in-memory architecture for detecting manufacturing defects. Fundam Res, 2022, 2(1), 123

[13] Spetalnick S D, Chang M Y, Konno S, et al. A 2.38 MCells/mm^2 9.81−350 TOPS/W RRAM compute-in-memory macro in 40nm CMOS with hybrid offset/IOFF cancellation and ICELLRBLSL drop mitigation. 2023 IEEE Symposium on VLSI Technology and Circuits (VLSI Technology and Circuits), 2023, 1

[14] Correll J M, Jie L, Song S, et al. An 8-bit 20.7 TOPS/W multi-level cell ReRAM-based compute engine. 2022 IEEE Symposium on VLSI Technology and Circuits (VLSI Technology and Circuits), 2022, 264

[15] Liu Q, Gao B, Yao P, et al. 33.2 A fully integrated analog ReRAM based 78. 4TOPS/W Compute-In-Memory Chip with Fully Parallel MAC Computing2020 IEEE International Solid- State Circuits Conference-(ISSCC), 2020, 500

[16] Zhang W B, Yao P, Gao B, et al. Edge learning using a fully integrated neuro-inspired memristor chip. Science, 2023, 381(6663), 1205

[17] Jiang Z X, Xi Y, Tang J S, et al. COPS: An efficient and reliability-enhanced programming scheme for analog RRAM and on-chip implementation of denoising diffusion probabilistic model. 2023 International Electron Devices Meeting (IEDM), 2023, 1

Siqi Liu received her B.S. degree from Harbin Institute of Technology, Harbin, China, in 2021, and her M.S. degree from Tsinghua University, Beijing, China, in 2024. She is currently a Ph.D. student at the Institute of Neuroinformatics, University of Zurich and ETH Zurich, Zurich, Switzerland, with a primary research interest in NVM-based computing-in-memory circuits design.

Peng Yao received his B.S. degree in microelectronics from Xi'an Jiaotong University, Xi'an, China, in 2014, and his Ph.D. degree from Tsinghua University, Beijing, China, in 2020. His research interests include in-memory and neuromorphic computing, and he has authored or coauthored several papers in Nature, Science, Nature Communications, ISSCC, IEDM, and VLSI.