# Design and Implementation of a Hybrid, ADC/DAC-Free, Input-Sparsity-Aware, Precision Reconfigurable RRAM Processing-in-Memory Chip

Junjie Wang, Teng Zhang, Shuang Liu, Yihe Liu, Yuancong Wu, Shaogang Hu, *Member, IEEE*, Tupei Chen, Yang Liu, Yuchao Yang, *Member, IEEE*, and Ru Huang, *Fellow, IEEE*

***Abstract*— In this work, we design and implement a 1-Mb resistive random access memory (RRAM) processing-in-memory (PIM) chip based on a 180-nm CMOS technology. In this design, a time-division multiplexing (TDM) circuit along with sparsity-aware sense amplifier (SA) and asynchronous counter module (ACM) are proposed to free the chip from digital-to-analog converter (DAC) and analog-to-digital converter (ADC). A sparsity-aware input module (SAIM) is designed to improve computational efficiency for bit-level input sparsity detection. A technique based on quantization-aware training (QAT), dynamically reconfigurable shifters (RecSTRs), and tree adders (TAs) is used to achieve system reconfigurability for 1–8-bit input, 1–8-bit weight, and 6–22-bit output. With this technique, optimized quantization to 4-bit weight 4-bit activation (W4A4) can reduce the number of network parameters to 1/8 of that required for the 32-bit floating-point (FP32) version. The number of calculate cycles can also be reduced to 1/4 of that of the FP32 version. This design has achieved a weight density of 13.32 Mb/mm$^2$ normalized to the 22-nm node and an energy efficiency of 17.36 TOPS/W for 4-bit integer (INT4) activation and weight.**

***Index Terms*— Processing-in-memory (PIM), quantization-aware training (QAT), resistive random access memory (RRAM), sparsity-aware, time-division multiplexing (TDM).**

## I. Background

MANY types of memories, such as static random access memory (SRAM) [1], [2], resistive random access memory (RRAM) [3], [4], dynamic random access memory (DRAM) [5], [6], magnetic random access memory (MRAM) [7], [8], and ferro-electronic random access memory (FeRAM) [9], [10], have been used in processing-in-memory (PIM) configurations. PIM systems have been applied in many cognitive tasks, such as image recognition [11], speech recognition [11], [12], and text recognition [13] with prospective high efficiency and small chip area. Compared with the PIM architecture implemented by conventional CMOS technology, the architecture based on non-volatile memory (NVM) devices shows greater potential in artificial neural network computations due to high density and low power consumption. The MRAM-based PIM architecture not only enables basic logic operations but also enables the implementation of a full adder [7] and an approximate adder for energy-constrained applications [8], resulting in significant savings in terms of area and power consumption. Similarly, the FeFET-based PIM macro exhibits significant power savings compared with the SRAM-based macro, whether it is used for computing-in-memory or logic-in-memory operations [9], [10]. Among various NVM devices, the RRAM has the advantage of using materials that are common in semiconductor manufacturing [14]. In particular, RRAM-based PIM configurations show promising potential in artificial neural network computations due to high density and low power consumption [3]. A conventional RRAM PIM configuration is shown in Fig. 1. For a typical application, the median level of multiply–accumulate (MAC) computations in the CIFAR-10 dataset is greater than ~$10^4$, as shown in Fig. 2(a), which is largely beyond the precision of low-power analog-to-digital converter (ADC). In an analog RRAM PIM system, both the input and output of the memory array work in the analog domain. Digital-to-analog converter (DAC) [15] or spikes' encoder is required for the input, and ADC [2], [11], [16], [17], [18] or time-domain converter (TDC) [3], [19] is required

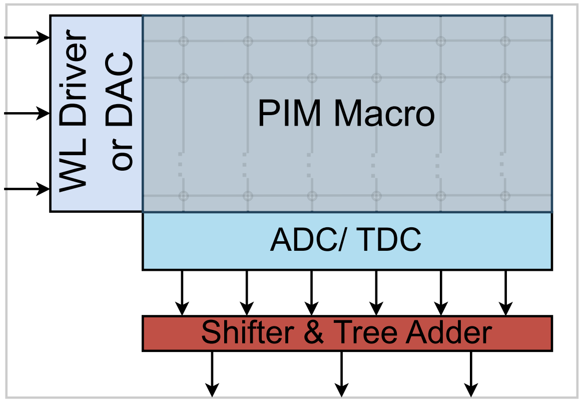


Fig. 1. Schematic illustration of a conventional PIM configuration.

Manuscript received 6 February 2023; revised 14 May 2023 and 18 July 2023; accepted 30 July 2023. Date of publication 22 August 2023; date of current version 30 January 2024. This article was approved by Associate Editor Meng-Fan Chang. This work was supported in part by the NSFC under Project 61925401, Project 92064004, Project 61927901, and Project 92164302; and in part by the 111 Project under Grant B18001. *(Junjie Wang and Teng Zhang contributed equally to this work.) (Corresponding author: Yang Liu.)*

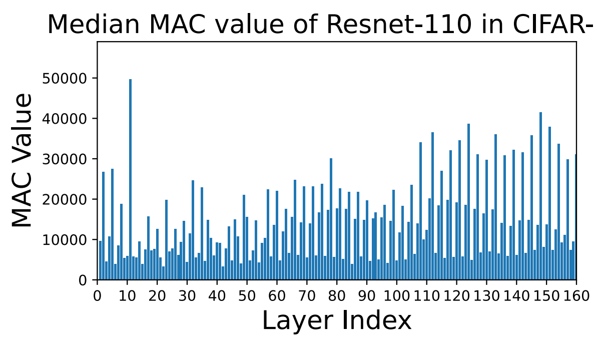
Junjie Wang, Shuang Liu, Yihe Liu, Yuancong Wu, Shaogang Hu, and Yang Liu are with the State Key Laboratory of Electronic Thin Films and Integrated Devices, University of Electronic Science and Technology of China, Chengdu 611731, China (e-mail: yliu1975@uestc.edu.cn).

Teng Zhang, Yuchao Yang, and Ru Huang are with the School of Integrated Circuits, Peking University, Beijing 100871, China (e-mail: yuchaoyang@pku.edu.cn).

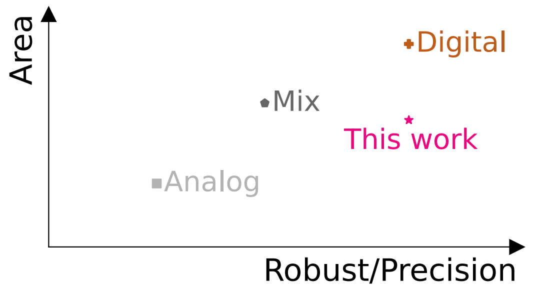
Tupei Chen is with the School of Electrical and Electronic Engineering, Nanyang Technological University, Singapore 639798.

Color versions of one or more figures in this article are available at https://doi.org/10.1109/JSSC.2023.3304174.

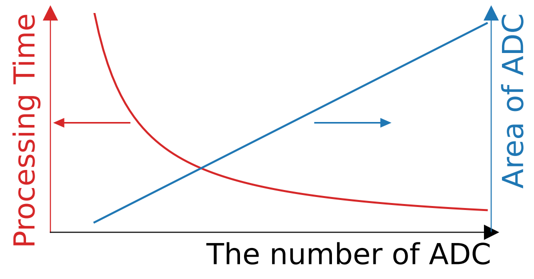
Digital Object Identifier 10.1109/JSSC.2023.3304174



(a)



(b)



(c)

Fig. 2. (a) MAC value needed for each layer in the Resnet-110 algorithm for dataset CIFAR-10, (b) chip area needed as a function of input data precision, and (c) processing time and chip area for ADC as functions of the number of ADCs. (a) Challenge 1: High MAC level. (b) Challenge 2: Area overhead of computational circuit. (c) Challenge 3: Low parallelism due to MUX.

for the output. Multi-level RRAMs are usually used in analog PIM designs so that the number of RRAM devices is less than that of the digital type of PIM. However, the precision is limited by RRAM state variation and the area/energy overhead of ADC/DAC, as shown in Fig. 2(b). Another type of PIM is a mix of digital and analog, as indicated in Fig. 2(b). In a mix PIM system, low-power ADC is connected to the output of the memory array, and thus, it is more robust and energy-efficient than an analog PIM system. However, the resolution of low-power ADC may be insufficient to deal with vector–matrix multiplication (VMM) directly. The mix PIM has to divide MAC into several parts to meet ADC requirement. Such a design decreases the degree of parallelism (DOP) and increases the power consumption of ADC and chip area, as shown in Fig. 2(c). Fully digital RRAM PIM configuration has high robustness as shown in Fig. 2(b). In addition, ADC/DAC-free configuration has the advantage of saving a large amount of area and power. Some works

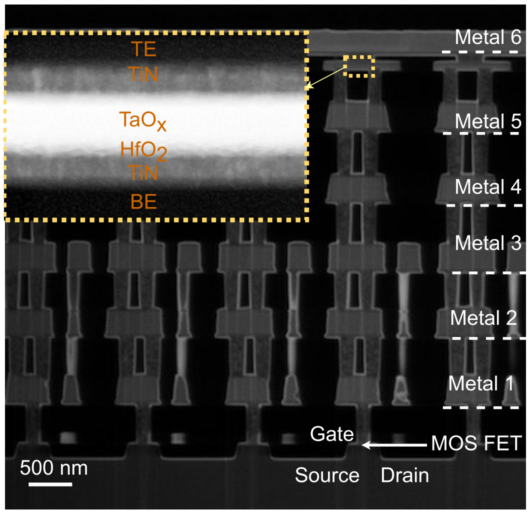


Fig. 3. Cross-sectional view of the proposed hybrid integration of RRAM and COMS circuits by scanning electron microscopy (SEM). Inset part shows cross section view of the RRAM device structure by transmissional electron microscopy (TEM).

on RRAM PIM chips were reported recently [16], [17], [18], [20], [21], [22], [23]. However, RRAM PIM design still faces several challenges: 1) most RRAM-based PIMs still use ADC and DAC, resulting in large area and power consumption overhead; 2) reconfigurability for input, weight, and output precisions is needed to deal with different types of artificial neural network algorithms efficiently; and 3) sparsity-aware input module (SAIM) is required for improving computational efficiency.

This work presents an ADC/DAC-free fully digital PIM system based on a 1-Mb RRAM array. The following four techniques are used in this work: 1) RRAM hybrid integration process for the integration of RRAM array with CMOS circuits; 2) software–hardware co-optimization which includes the quantization-aware training (QAT) and dynamic reconfigurable shifter and tree adder (TA) for realizing the reconfigurability of input, weight and output precisions; 3) input logic to detect the bit-level sparsity of input for speeding up the computation; and 4) asynchronous ripple counter (CNT) to save energy. This work has a weight density of 13.32 $\text{Mb/mm}^2$ normalized to the 22-nm node and an energy efficiency of 17.36 TOPS/W for 4-bit integer (INT4) activation and weight.

The remaining part of this article is organized as follows. Section II describes the hybrid integration of RRAM and CMOS circuits. Section III provides the reading and programming of the RRAM device. In Section IV, the design of PIM configuration and functional modules is presented. Section V gives the measurement results of RRAM array. Section VI discusses the algorithm mapping and evaluation of the PIM system. A conclusion is given in Section VII.

## II. Hybrid Integration of RRAM and CMOS Devices

Fig. 3 shows the hybrid one transistor one RRAM (1T1R) integrations. One 1T1R cell consists of a transistor whose drain is connected to an RRAM cell to serve as the selector. For the chip used in this work, the cell-select transistor and peripheral circuitry were fabricated in a commercial foundry using a

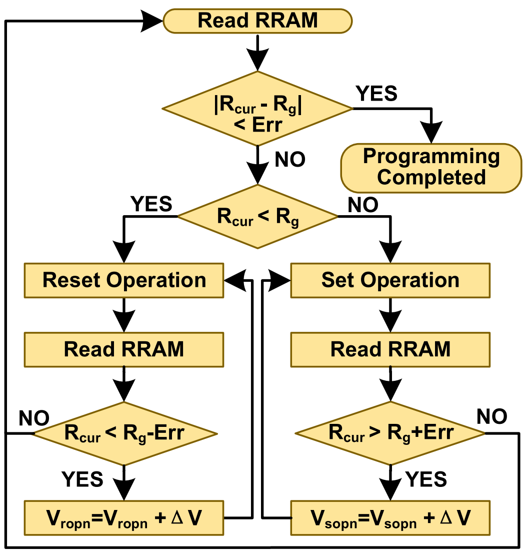


Fig. 4. Illustration of the procedure for programming RRAM devices.

TABLE I

VOLTAGES APPLIED TO THE RRAM DEVICE

| | READ | RESET | SET | FORMING |
|---|---|---|---|---|
| BL | 0.2 V | 0 V | 1 V ∼ 2 V | 3.5V ∼ 4.5 V |
| SL | 0 V | 1 V ∼ 2 V | 0 V | 0 V |
| WL | 1.8 V | 5 V | 5 V | 5 V |

mixed signal 180-nm CMOS process. The RRAM cell with a material stack of TiN/$HfO_2$/$TaO_x$/TiN was then deposited after Metal 5 in back end of line (BEOL). Via and the last metal layer were manufactured to finalize the whole 1-Mb PIM chip.

## III. Reading and Programming of the RRAM Device

In our design, the RRAM array can be programmed, erased, and read both externally and internally. The reading operation is conducted as follows: a certain small voltage is applied to an RRAM device and the corresponding current of the device is measured. Then the control unit converts the current value into the resistance value. To mitigate the cycle-to-cycle (C2C) and device-to-device (D2D) variations, we use a write-and-verify protocol for programming the RRAM array. The voltage values for RRAM read, write, and forming operations are presented in Table I. Fig. 4 illustrates the programming procedure of RRAM device. As there are 1 million RRAM devices in the array, programming of RRAM weight must be executed automatically. To set/reset the RRAM to the target resistance, stair-like spikes are used, with each stair step having a width of 50 $\mu$s. The selection MOSFETs are used to achieve current compliance, which helps prevent RRAM devices' breakdown during FORMING or SETTING. In the process of programming of RRAM weight, the RRAM resistance value $R_{\text{cur}}$ was first read out using the reading operation mentioned above. Then the resistance value is compared with the target value $R_g$ required for weight loading. If the deviation between $R_{\text{cur}}$ and $R_g$ is smaller than a preset value of error Err, the programming procedure is terminated. However, if the deviation between $R_{\text{cur}}$ and $R_g$ is larger than Err, reset or set operation is carried out. if $R_{\text{cur}}$ is smaller/greater than $R_g$, a reset/set voltage pulse

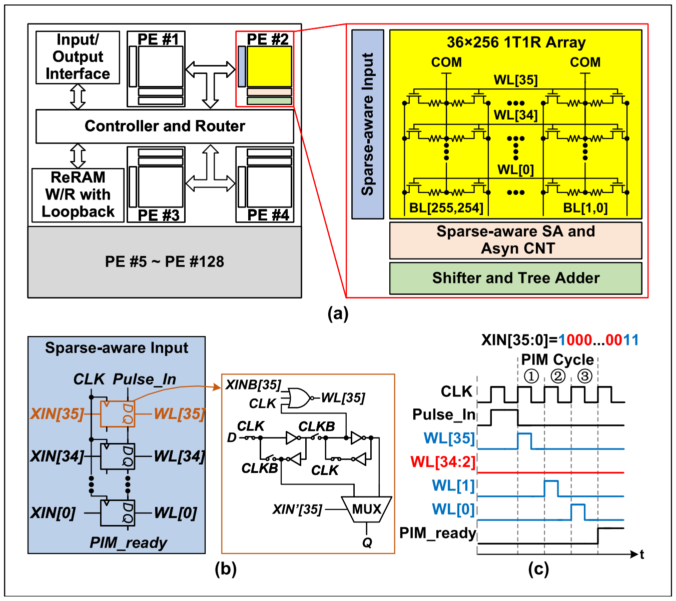


Fig. 5. (a) Schematic illustration of the dynamic-precision RRAM-based PIM structure and circuit schematic of PE, (b) circuit schematic of the SAIM, and (c) waveforms of PIM cycles.

series $V_{\text{ropn}}/V_{\text{sopn}}$ is applied to the RRAM device until the deviation is smaller than Err. Using the above procedure, the weights of the neural network can be updated automatically.

## IV. Design of PIM Configuration and Modules

Fig. 5(a) shows the schematic illustration of the proposed dynamic-precision RRAM-based PIM structure including 128 processing engines (PEs), an input–output interface, a system controller with routing capability, and an internal RRAM write/read (W/R) module with feedback control to modify RRAM resistance for weight updating. As shown in Fig. 5(a), each PE comprises a 36 × 256 1T1R array, an SAIM for the bit-level input sparsity detection [see Fig. 5(b)], a sparsity-aware sense amplifier (SA), an asynchronous counter module (ACM) for decreasing power consumption, and a shifter and TA block (STAB) for achieving reconfigurability and supporting up to 1–8-bit input, 1–8-bit weight, and 6–22-bit output. There are a total of 128 PIM macros in our design, resulting in a total of 1 179 648 bits (36 × 256 × 128). There are no explicit connections among the macros. Instead, the controller is responsible for distributing and receiving activations to and from the PIM macros. The selection of 36 rows is designed to better map the 3 × 3 convolution kernel. If the amount of channels in a certain activation exceeds 36 (for 1 × 1 2-D-convolution) or 4 (for 3 × 3 2-D-convolution), the convolution operation will be distributed into multiple PIM macros, and the partial sums (PSUMs) are accumulated in the controller, as shown in Fig. 6.

As shown in Fig. 5(c), during PIM computing, SAIM ("Pulse_in") inputs a pulse first. Then the word line (WL) of the 1T1R array works row by row to realize the MAC of 36 1-bit inputs based on 36 × 256 1-bit weights. In SAIM, 36 D-type flip flops (DFFs) and 36 multiplexers (MUXs) form a reconfigurable multi-bit shift register. Each DFF and MUX are connected to a three-input NOR gate. When one of the input IN[i] is "0," the output WL[i] of the NOR gate becomes "0," and the associated DFF is bypassed through the MUX.

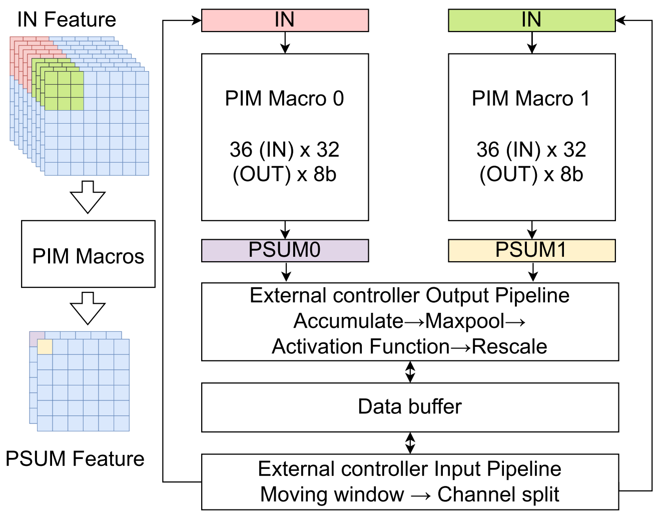
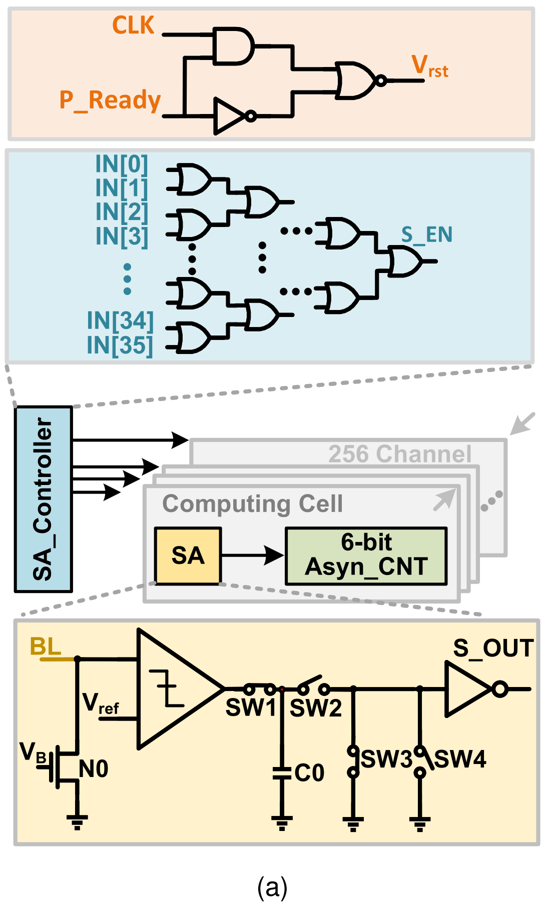


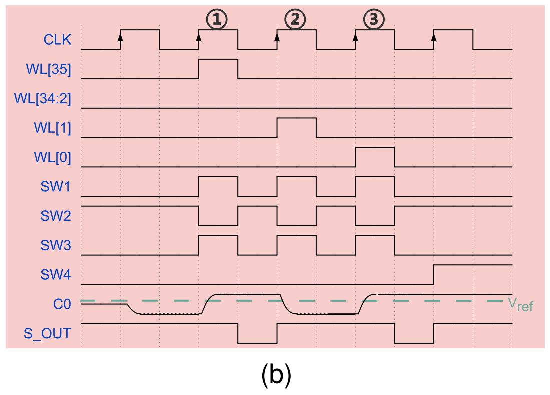
Fig. 6. Hardware mapping for 3 × 3 convolution kennel. PIM macros complete VMM, and external controller conducts other operations.

Therefore, the PIM cycles are positively correlated with the number of **1** in IN[35:0], and thus, detection of bit-level sparsity is realized. When PIM computation is completed, SAIM sends out a signal of "PIM_ready" to STAB for obtaining the MAC value as shown in Fig. 5(c).

Fig. 7(a) illustrates the design of the SA which comprises a comparator, a sampling capacitor (C0), an inverter, four switches (SW1–4), and a tail current source (N0). One of the challenges of using RRAM is the insufficient drive capacity of the SA because of its relatively high resistance. To address this, our design incorporates a preamplifier stage prior to sensing, which enhances the drive capability during the sense phase. As a result, the proposed SA, operating at 100 MHz, exhibits a faster readout speed than an SA without a preamp. The SA is designed specifically for a single-ended 1T1R circuit. Compared with the differential SA presented in [23], our SA saves approximately 30% of the area, thereby contributing to a more area-efficient design. The preamplifier and sample-and-hold circuits in our design realized as dynamic circuits. Therefore, akin to other latch-type mode SAs, our proposed SA exhibits low static and dynamic power consumption. In comparison to an ADC-based readout circuit, the SA is designed only for distinguishing "1" and "0," providing a larger margin than SAR-ADCs, and making it easier to achieve precise results. The asynchronous ripple CNT in our design is essentially a chain of DFFs. As such, the clock frequency halves with each stage of the DFF, meaning that only the first one or two DFFs need to operate at high frequency. This can be achieved using current mode logic (CML). In terms of area, a 10-bit ripple CNT occupies 70 $\times$ 14 $\mu$m (given that the area of a DFF is 7 $\times$ 14 $\mu$m) in the 180-nm technology, which is significantly smaller than 265 $\times$ 195 $\mu$m of an SAR-ADC fabricated using the same technology, operating at 100 MHz with 10-bit precision [24]. As technology nodes shrink, the area occupied by CNTs also reduces. However, the DAC submodule of the SAR-ADC does not benefit from this technology change. Furthermore, the speed of the comparator in the SAR-ADC is affected by its area, which means it cannot be made too small in high-speed designs. Therefore, the use of CNTs in our design does not negate the area savings achieved by eliminating ADC/DACs. Considering the C2C and D2D variation in the RRAM, the proposed scheme, which combines the SA with an adder tree, can achieve higher precision and exhibit better performance under PVT variations. SW1 and SW3 are controlled by "S_EN," and SW2 is controlled by the inversion of "S_EN" for sampling and holding, respectively. SW4 is controlled by the signal "PIM_Ready" for reset. There are 256 channel SAs in macro working in parallel, with each connected to a 6-bit asynchronous CNT ("ASYN_CNT"). As shown in Fig. 7(a), "SA_Controller" is an OR gate with 36 inputs which generates the control signal ("S_EN") of SA from WL[35:0]. The enable signal of the sparse sensing SA is different from the global clock, which only works when it detects the input-valid signal, realizing asynchronous operation and power reduction. The sampling and holding circuit is designed to eliminate overshoot on BL when WL is turned on. Fig. 7(b) shows the waveforms of SA operation cycles when WL[35], WL[1], and WL[0] are "1," while the remaining WLs are "0"; meanwhile, the RRAM in WL[1] is in the low-resistance state (LRS), while WL[35] and WL[0] are in the high-resistance state (HRS). As can be observed, in Cycle① and Cycle③, the LRS of RRAM is detected, while the HRS of RRAM is detected in Cycle②. When all WLs are at low, "S_EN" and "PIM_Ready" are low; SW1 and SW3 are turned off; SW2 and SW4 are turned on; and the voltage on C0 is reset. When any of WL [35:0] is high, as shown in Fig. 7(b), "S_EN" is pulled high in the first half cycle; and the comparator will charge or discharge C0 as shown in Fig. 7(a). In the second half cycle, SW2 is turned on; and the voltage of S1 is amplified by the inverter to send it to the final output "S_OUT." The layout of the PE comprising bit cell array, SA, ASYN_CNT, and programming circuit is shown in Fig. 7(c). Fig. 8 shows the process of VMM. Each bit of the input vector is calculated with time-division multiplexing (TDM). The least significant bit (LSB) of all the vector elements is fed into the RRAM array, followed by the second bit and so forth to the most significant bit (MSB). During computation, AND function operates between Input[j] and Weight[i], as shown in Fig. 8. The outcomes of AND operations between the selected row of weights and one input bit are then distributed across all the BLs. The weight precision determines the number of BLs used. For example, 4-bit weight needs four BLs, while 8-bit precision should use eight BLs. Eight adjacent RRAMs can form one weight in the 8-bit mode, while these RRAMs can also form two weights in the 4-bit mode. The CNT array connected to BLs receives the ripple signals and obtains the PSUM. The PSUMs belonging to one set of OUT data are shifted and fed into the TA. Since the number of bits in one weight is reconfigurable, RS and TA are required.

In the 8-bit mode, the weighted input activation is calculated as the sum of eight shifted PSUMs. This is achieved by configuring the eight reconfigurable shifters (RecSTRs) to perform bit shifts of 0, 1, 2, …, up to 7 bits. The MUX then reads the sixth input and outputs the corresponding weighted activation. In contrast, the 4-bit mode computes the weighted input activation as the sum of four shifted PSUMs. The array of RecSTRs is divided into two sections, each calculating one of the two weighted input activations. Each section of the RecSTR array is set to perform bit shifts of 0, 1, 2, and 3 bits. Subsequently, the MUX reads the fourth and fifth inputs in sequence. The bit sum (BSUM) for different input bits is shifted and accumulated. The accumulators obtain output





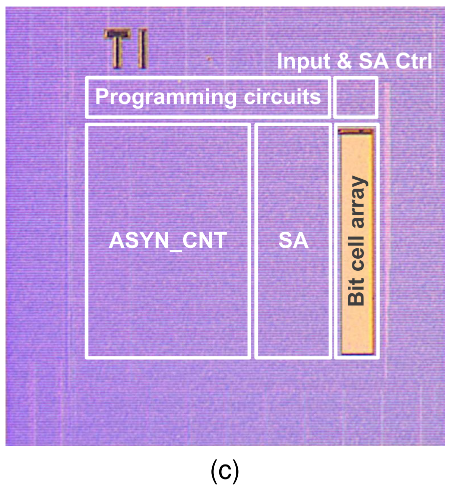


Fig. 7. (a) Circuit schematics of the SA controller and SA, (b) waveforms of the SA working flow, and (c) PE layout comprising bit cell array, SA, ASYN_CNT, and programming circuit.

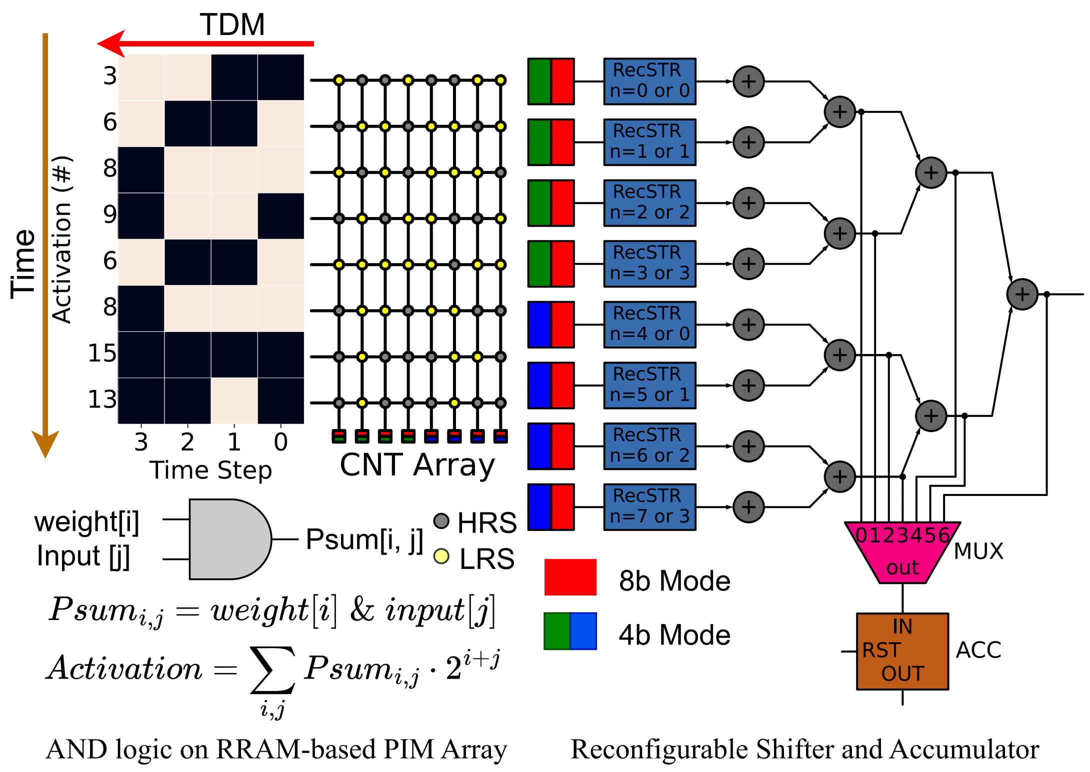


Fig. 8. Illustration of the process of VMM in the PIM chip.

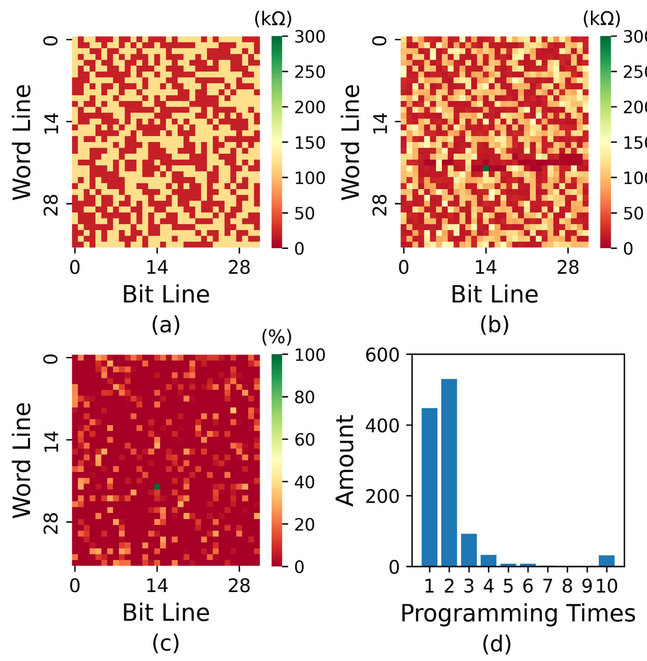


Fig. 9. Measured resistance distribution of the RRAM device in a single RRAM array. (a) Target resistance, (b) resistance after programming, (c) relative error between the target and actual resistance, and (d) number of programming attempts.

activation when all the bits of inputs are computed, as shown in Fig. 8.

## V. Measurement Results

The dimensions of a single RRAM device are 1.25 × 1.25 $\mu$m, and one PIM macro contains 36 × 256 RRAM devices. The size of the PIM macro, including SAs and RecSTRs, is 723 × 811 $\mu$m. The HRS and LRS are set to 30 and 300 kΩ, respectively, allowing the SA to easily determine the RRAM state. To verify the RRAM array resistance distribution, we programmed the RRAM array to a random resistance state and read out the resistance subsequently. The writing process is deemed complete when the relative error between the current resistance $R_{\text{cur}}$ and the goal resistance $R_g$ falls below 10%. To avoid device failure and excessive testing time, we limit the programming of each RRAM device to a maximum of ten attempts, regardless of whether the target is achieved. Fig. 9 illustrates the results from a 36 × 32 array within one PIM macros. Fig. 9(a) and (b) depicts the resistance distribution of the target resistance and the actual resistance post-writing, respectively. Fig. 9(c) represents the discrepancy between the target and actual values, and Fig. 9(d) indicates the number of programming attempts before the resistance difference falls below Err in a single 1T1R array comprising 1152 RRAM devices. Most RRAM devices attain the target value within three attempts, and the RRAM variation is maintained below 20%. The chip manufactured at the CMOS 180-nm node can operate at 1.8 V with the 100-MHz clock. As shown in Fig. 10, the Shmoo plot demonstrates that RRAM can be accessed in 3.19 ns at 1.4-V supply. The RRAMs are read at 0.2 V, and the energy consumption of the PIM macro is obtained by measuring the current flow through the PIM macro and multiplying the read voltage.

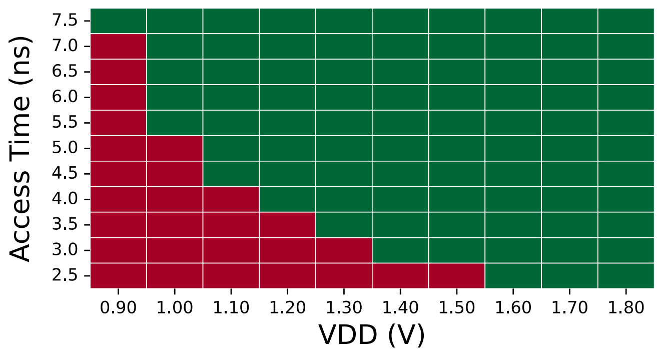


Fig. 10. Access time as a function of VDD voltage for the PIM chip.

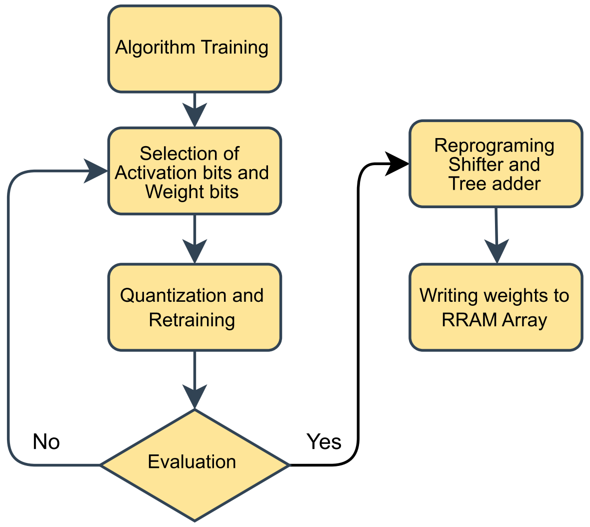

Algorithm Training
Selection of Activation bits and Weight bits
Quantization and Retraining
Evaluation
No
Yes
Reprograming Shifter and Tree adder
Writing weights to RRAM Array

Fig. 11. Flowchart of mapping an algorithm to the RRAM PIM chip.

Fig. 11 illustrates the flowchart of mapping an algorithm to the RRAM PIM chip. Software–hardware co-optimization is used to improve the inference accuracy. An algorithm is first trained in the software frame of PYTORCH.[1] Then activation bits and weight bits are selected for quantization and retraining. The performance of the algorithm is evaluated. If the accuracy of the result is poor, new activation bits and weight bits are used for quantization and retraining again until the accuracy meets the requirements.

## VI. Algorithm Mapping and Evaluation of the System

Fig. 12 shows the block diagram of the verification system for the RRAM PIM chip. An external module is used to read and program RRAM arrays (note that the internal P/E module can also be used). The read/program process is described in Session III. A personal computer (PC) is used to conduct training and control the verification system. An FPGA platform is used as an input and output interface between the PC and the PIM chip. The SRAM module in FPGA is used as the input data/output buffer. An Arduino micro-controller unit (MCU) is used to write the weight matrix to the RRAM array externally. As there are 1 million RRAM devices in the array, the system should be able to write weight to any selected RRAM device automatically, and the process can be controlled by either the MCU or the control unit inside the chip.

[1]Trademarked.

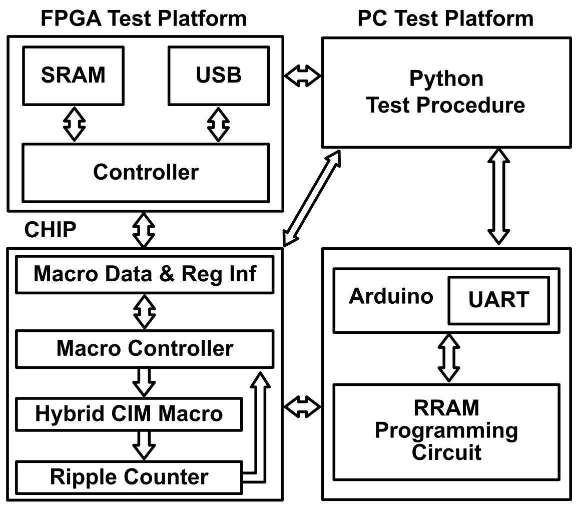


Fig. 12. Illustration of the verification system for the PIM chip.

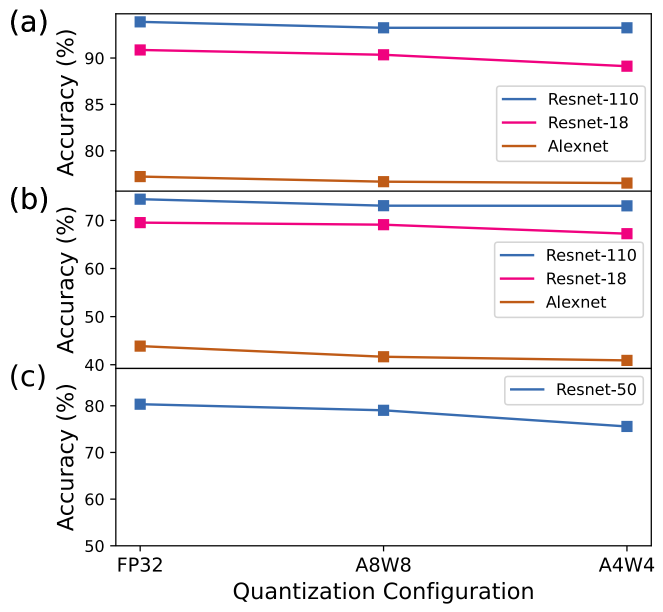


Fig. 13. Accuracy under different quantization configurations for various algorithms for datasets. (a) CIFAR-10. (b) CIFAR-100. (c) ImageNet.

Fig. 13 shows the accuracy under different quantization configurations for various algorithms and the datasets including CIFAR-10, CIFAR-100, and ImageNet. We tested the CIFAR-10 dataset using algorithms such as Resnet-18, which can be fully mapped onto our hardware, as well as Resnet-110 and AlexNet. The accuracies are presented in Fig. 13(a). The **FP32** result represents the accuracy obtained using 32-bit floating-point number through software. Fig. 13(a) shows that in the quantization configurations of 8-bit activation and 8-bit weight (A8W8) and 4-bit activation and 4-bit weights (A4W4), the accuracy can reach a comparable level to that of FP32. Fig. 13(b) shows quantization results for algorithms of Resnet-18, Resnet-110, and AlexNet with dataset CIFAR-100. Neglectable difference in accuracy between quantization configurations of A8W8 and A4W4 for algorithms of Resnet-110 is observed. However, for AlexNet, the accuracy decreases from 44% for A8W8 to 41% for A4W4. Fig. 13(c) shows the accuracy of Resnet-50 for the datasets of ImageNet with various quantization configurations. As can be observed, the quantization of A8W8 leads to an ignorable decrease in accuracy from 80% for FP32 to 79%, while the quantization of A4W4 shows an accuracy drop of 4%. The result shows that the quantization configurations can make the algorithms hardware-friendly while ensuring the accuracy.

The input sparse ratio is defined as $(n_0/(n_0+n_1))$, where $n_0$ and $n_1$ are the total sum of the number of "0" and "1" in each element of the input vector, respectively. For example, consider an input vector of (13, 24, 0, 15). The total number of "1"s in this vector is $3 + 2 + 0 + 4 = 11$. With a total of four elements, each consisting of 8 bits, there are $4 \times 8 = 32$ bits in the input vector. Therefore, the sparse ratio is calculated as 11/32, which equals 32.375%. Sparse ratio $\simeq$0.5 because bit "1" and bit "0" appear equally in the activation, and this hypothesis is verified in Resnet-18 on the CIFAR10 dataset. Fig. 14(a) shows the clock cycles needed for different algorithms with sparse input. As can be observed, the calculate cycles are substantially reduced with sparse input. In addition, the calculation can be further reduced by quantization. For example, for the algorithm of VGG19, the sparse input (8 bits) and sparse input (4 bits) reduce the calculate cycles from normal input (8 bits) $4.8 \times 10^6$ to $2.4 \times 10^6$ and $1.2 \times 10^6$ cycles, respectively. That

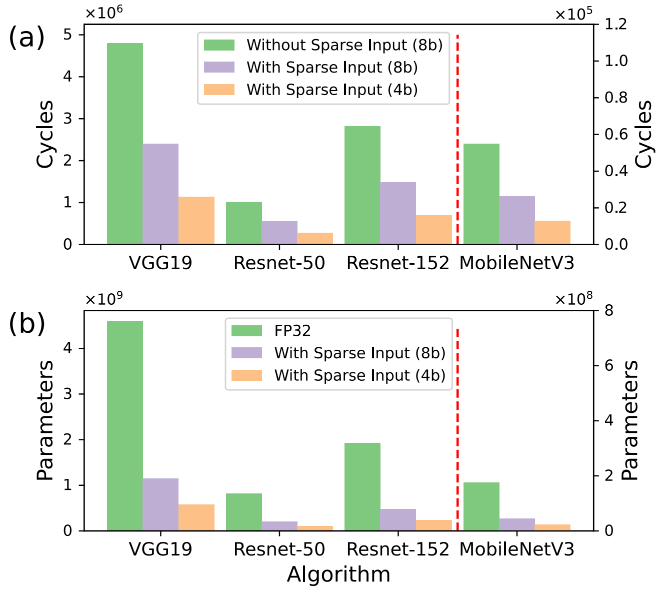


Fig. 14. (a) Number of operations needed for different large-scale algorithms under different input precisions. (b) Number of parameters needed for different large-scale algorithms under different precisions.

means the bit-level sparsity can reduce the clock cycles by half compared with that of the original 8-bit version. When the input activations are compressed to 4 bits, the clock cycles are further reduced to a quarter of the original 8-bit version. Fig. 14(b) shows the number of parameters required for various algorithms with different quantization precisions. As can be observed in Fig. 14(b), weight quantization can also reduce the network parameters required for the algorithms. For example, for VGG19, $4.8 \times 10^9$ parameters are needed in the FP32 mode; in contrast, only $1.1 \times 10^9$ and $0.6 \times 10^9$ parameters are required for 8-bit integer (INT8) and INT4 modes, respectively. A hybrid parameter mapping method is used for the models whose parameters' amount is beyond the capacity of our design. For smaller models such as Resnet-18, we quantized all the convolution layers and mapped a portion of the weight matrix channels to the RRAM-based PIM to

TABLE II
BENCHMARK BETWEEN THIS WORK AND PRIOR ARTS

| | **This work** | [21] | [22] | [23] | [16] | [17] | [18] |
|---|---|---|---|---|---|---|---|
| Technology node | **180 nm** | 55 nm | 22 nm | 22 nm | 40 nm | 40 nm | 40 nm |
| Capacity | **1.12 Mb** | 1 Mb | 2 Mb | 4 Mb | 2 Mb | 64 Kb | 16 Kb |
| 1T1R Storage density (normalized to 22 nm)* | **13.32 $\mathbf{Mb/mm^2}$** | 30.86 $\mathrm{Mb/mm^2}$ | N/A | N/A | N/A | N/A | N/A |
| Sparsity-aware capability | **Yes** | No | No | No | No | No | Yes |
| Computing method | **ADC/DAC free** | **ADC/DAC free** | **ADC/DAC free** | **ADC/DAC free** | ADC | ADC | ADC |
| Input bit | **1-8 bit** | 2 bit | 4 bit | 8 bit | 8 bit | 1-8 bit | 1 bit |
| Weight bit | **1-8 bit** | 3 bit | 4 bit | 8 bit | 8 bit | 1-8 bit | ternary weight |
| Output bit | **6-22 bit** | 3 bit | 11 bit | 26 bit | 19 bit | 20 bit | 3 bit |
| Throughput (GOPS) | **410** (4-bit input & 4-bit weight) | N/A | N/A | 260 (8-bit input & 8-bit weight) | 475 (8-bit input & 8-bit weight) | N/A | 24.45 (1-b input and ternary weight) |
| Energy efficiency (TOPS/W @ 4 bit) | **17.36** (14-bit output) | 21.9 (3-bit output) | 28.93 (11-bit output) | 25.1 (26-bit output) | 20.5 (19-bit output) | 14.1 (4-bit input & 4-bit weight) | 44.5 (1-b input and ternary weight) |
| Chip area** ($\mathrm{mm^2}$ normalized to 22 nm) | **4.31** | 1.2 | 6.0 | 18.0 | 5.44 | 0.47 | 0.076 |
| Area efficiency ($\mathrm{TOPS/mm^2}$ normalized to 22 nm) | **0.057** (14-bit output) | N/A | N/A | 0.029 (26-bit output) | 0.054 (19-bit output) | 0.976 (4-bit input & 4-bit weight) | 0.2 (1-b input and ternary weight) |
| FoM (IN-W-OUT)*** ($\times 10^6$) | **2.98 (4-4-14)** | N/A | N/A | 6.13 (8-4-22) | 6.70 (4-4-11) | N/A | 0.010 (1-2-3) |

* The Normalized Storage density is defined as the density/(Technology node/22 nm)$^2$
** The Normalized Chip area is defined as the Chip area/(Technology node/22 nm)$^2$
*** FoM = In precision $\times$ W precision $\times$ Output ratio $\times$ Throughput $\times$ Capacity/Normalized chip area

ensure that the weights could be stored in our 1-Mb RRAM array. For larger networks, such as Resnet-50, the amount of the weight parameters is much larger than the capacity of our RRAM array. Therefore, we mapped the first part of the CNN to the PIM chip, while the remaining part of the CNN was computed on a software platform, where the weights and activations were quantized using the same method as in the hardware implementation. In our approach, a moving window on the feature map provides the input for the PIM macro. For a 3 $\times$ 3 convolutional kernel, four channels are transferred to the PIM macro, which then calculates VMM for a 36 $\times$ 32 weight matrix and 36 inputs (3 $\times$ 3 $\times$ 4). For a 1 $\times$ 1 convolutional kernel, 32 channels are transferred to the PIM macro (with four rows of the PIM being unused), and the macro calculates the VMM for a 32 $\times$ 32 weight matrix and 32 inputs. The PSUM of four channels (for a 3 $\times$ 3 convolutional kernel) or 32 channels (for a 1 $\times$ 1 convolutional kernel) is accumulated in the external controller. The activation function, max pooling, shortcut connection, and quantization adjustment are also performed in the external controller. Thanks to the high fidelity of the PIM readout circuit and the digital accumulator, the calculation accuracy of the hardware closely matches the accuracy of the software. Fig. 15(a) shows the layout of the PIM chip where the 1-Mb 1T1R RRAM array was fabricated on top of the as-fabricated CMOS chip as discussed in previous session. Fig. 15(a) also shows the packaged chip mounted on the test PCB board. A 256-pin quad flat packaging (QFP) was used to pack the chip. The power breakdown and area breakdown are shown in Fig. 15(b) and (c), respectively.

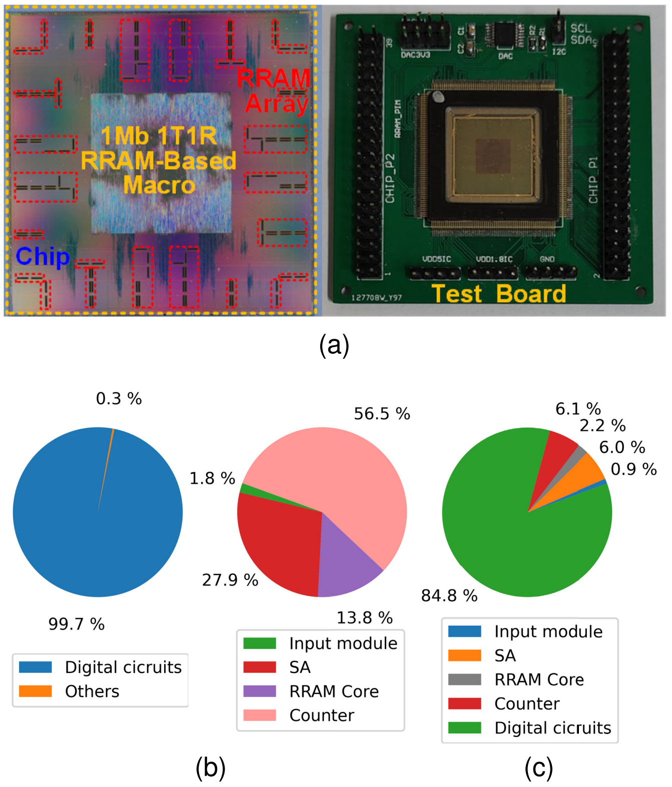


Fig. 15. (a) Layout of the PIM chip (left) and photograph of the PIM chip mounted on the test board (right). (b) Power breakdown, the right pie is the power breakdown of others in the left pie. (c) Area breakdown.

Table II shows the figure-of-merit (FoM) characteristic versus the technology node. The FoM is defined as follows, where "In precision" is the number of bits required to represent the input activation; "W precision" is the number of bits to required represent the weight; and "output ratio" can be presented by the sum of input precision and weight precision over output precision. In our work, output ratio =14/(4 + 4) for 4-bit inputs and 4-bit weights, because the accumulation is operated in the digital domain. Note that our design uses SA and ripple CNT implemented in a 180-nm process. Due to the limitation of the 180-nm process, the MOSFET selector must be placed in both n-well and p-well, resulting in an increased area for the selector transistor and storage density 13.32 Mb/mm$^2$. To make a fair comparison, we normalized the storage density to the 22-nm node. We developed a circuit using the 22-nm technology and determined the density of the RRAM array and the energy efficiency of the SA and ripple CNTs. The energy consumption of the 22-nm technology is 149.2 times lower than that of the 180-nm technology, which is a reduction beyond the square law of the decrease in technology node. The normalized area of 1T1R array is 69.57 $\times$ 6.38 $\mu$m under the 22-nm technology. Therefore, we believe that the scaling in the FoM comparison in our work is fair. We compare our work with the all-parallel RRAM PIM [25]. In their work, the read latency for a 3-bit output is reported as 214.17 ns, and the energy efficiency is measured as 30.26 TOPS/W. In contrast, our work achieves a read latency of 1280 ns ($10 \times 256 \times 0.5$) and an energy efficiency of 17.36 TOPS/W (using 4-bit weights) when assuming a sparsity of 0.5, which aligns with the actual activation of the neural network. It is important to note that considering the higher precision output of 14 bits, the read latency of the reference work would be 999.46 ns ($214.17/3 \times 14$), resulting in decreased energy efficiency. Furthermore, we observe that our work demonstrates superior latency compared with [25] for input sparsity values below 0.39. This highlights the benefits of our proposed scheme in scenarios with lower input sparsity. Table II presents the comparison of parameters between this work and previous works reported by other researchers [16], [17], [18], [21], [22], [23]. As can be observed in Table II, the FoM of our work is inferior to [16] and [23]. However, the throughput can improve as the technology node shrinks, leading to the FoM improvement. The measurement results of our design show an average energy efficiency including RRAM macro, SA, and ripple CNT, of 17.36 TOPS/W with 4-bit input and 4-bit weights, and the area efficiency achieves 0.057 TOPS/mm$^2$. The throughput and area efficiency in the 8-bit weight mode are half of those in the 4-bit weight mode, while the energy efficiency is a quarter of that in the 4-bit weight mode.

## VII. Conclusion

In conclusion, a hybrid 1-Mb RRAM PIM chip has been implemented based on a 180-nm CMOS technology. The ADC/DAC free design is realized by obtaining matrix multiplication results through counting the PSUM of all the rows using a ripple CNT. An SAIM is designed for the bit-level input sparsity to improve computation efficiency. Software–hardware co-optimization technique is used to conduct QAT. A reconfigurable shifter and TA are proposed for reconfigurability and quantization of the PIM system. Quantization with the proposed INT4 4-bit weight-4-bit activation (W4A4) scheme can reduce the number of network parameters to 1/8 of that required for the conventional FP32 scheme. In addition, with the INT4 W4A4 scheme, calculate cycles can also be reduced to 1/4 of that of the FP32 scheme. The PIM design based on the INT4 W4A4 scheme can achieve a weight density of 13.32 Mb/mm$^2$ normalized to the 22-nm node and an energy efficiency of 17.36 TOPS/W and area efficiency of 0.057 TOPS/mm$^2$.

## References

[1] X. Si et al., "A local computing cell and 6T SRAM-based computing-in-memory macro with 8-b MAC operation for edge AI chips," *IEEE J. Solid-State Circuits*, vol. 56, no. 9, pp. 2817–2831, Sep. 2021.

[2] C. Yu, T. Yoo, K. T. C. Chai, T. T. Kim, and B. Kim, "A 65-nm 8T SRAM compute-in-memory macro with column ADCs for processing neural networks," *IEEE J. Solid-State Circuits*, vol. 57, no. 11, pp. 3466–3476, Nov. 2022.

[3] J.-M. Hung et al., "8-b precision 8-Mb ReRAM compute-in-memory macro using direct-current-free time-domain readout scheme for AI edge devices," *IEEE J. Solid-State Circuits*, vol. 58, no. 1, pp. 303–315, Jan. 2023.

[4] Y. Chen, L. Lu, B. Kim, and T. T. Kim, "Reconfigurable 2T2R ReRAM architecture for versatile data storage and computing in-memory," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 28, no. 12, pp. 2636–2649, Dec. 2020.

[5] M. F. Ali, A. Jaiswal, and K. Roy, "In-memory low-cost bit-serial addition using commodity dram technology," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 67, no. 1, pp. 155–165, Jan. 2020.

[6] S. Roy, M. Ali, and A. Raghunathan, "PIM-DRAM: Accelerating machine learning workloads using processing in commodity DRAM," *IEEE J. Emerg. Sel. Topics Circuits Syst.*, vol. 11, no. 4, pp. 701–710, Dec. 2021.

[7] S. Prajapati, V. Nehra, and B. K. Kaushik, "High-performance computing-in-memory architecture based on single-level and multilevel cell differential spin Hall MRAM," *IEEE Trans. Magn.*, vol. 57, no. 9, pp. 1–15, Sep. 2021.

[8] K. Monga, N. Chaturvedi, and S. Gurunarayanan, "A dual-mode in-memory computing unit using spin Hall-assisted MRAM for data-intensive applications," *IEEE Trans. Magn.*, vol. 57, no. 4, pp. 1–10, Apr. 2021.

[9] G. Yin et al., "Enabling lower-power charge-domain nonvolatile in-memory computing with ferroelectric FETs," *IEEE Trans. Circuits Syst. II, Exp. Briefs*, vol. 68, no. 7, pp. 2262–2266, Jul. 2021.

[10] J. Hwang, S. Lim, G. Kim, S.-O. Jung, and S. Jeon, "Non-volatile majority function logic using ferroelectric memory for logic in memory technology," *IEEE Electron Device Lett.*, vol. 43, no. 7, pp. 1049–1052, Jul. 2022.

[11] W. Wan et al., "A compute-in-memory chip based on resistive random-access memory," *Nature*, vol. 608, no. 7923, pp. 504–512, Aug. 2022.

[12] S. Liu et al., "Efficient and reconfigurable reservoir computing to realize alphabet pronunciation recognition based on processing-in-memory," *Appl. Phys. Lett.*, vol. 119, no. 10, Sep. 2021.

[13] W.-H. Chen et al., "A 65 nm 1 Mb nonvolatile computing-in-memory ReRAM macro with sub-16 ns multiply-and-accumulate for binary DNN AI edge processors," in *IEEE Int. Solid-State Circuits Conf. (ISSCC) Dig. Tech. Papers*, Feb. 2018, pp. 494–496.

[14] A. Sebastian, M. Le Gallo, R. Khaddam-Aljameh, and E. Eleftheriou, "Memory devices and applications for in-memory computing," *Nature Nanotechnol.*, vol. 15, no. 7, pp. 529–544, Jul. 2020.

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

[16] W.-S. Khwa et al., "A 40-nm, 2 M-cell, 8 b-precision, hybrid SLC-MLC PCM computing-in-memory macro with 20.5–65.0 TOPS/W for tiny-Al edge devices," in *IEEE Int. Solid-State Circuits Conf. (ISSCC) Dig. Tech. Papers*, vol. 65, Feb. 2022, pp. 1–3.

[17] J.-H. Yoon, M. Chang, W.-S. Khwa, Y.-D. Chih, M.-F. Chang, and A. Raychowdhury, "A 40-nm, 64-kb, 56.67 TOPS/W voltage-sensing computing-in-memory/digital RRAM macro supporting iterative write with verification and online read-disturb detection," *IEEE J. Solid-State Circuits*, vol. 57, no. 1, pp. 68–79, Jan. 2022.

[18] W. Li, X. Sun, S. Huang, H. Jiang, and S. Yu, "A 40-nm MLC-RRAM compute-in-memory macro with sparsity control, on-chip write-verify, and temperature-independent ADC references," *IEEE J. Solid-State Circuits*, vol. 57, no. 9, pp. 2868–2877, Sep. 2022.

[19] P.-C. Wu et al., "A 28 nm 1 Mb time-domain computing-in-memory 6T-SRAM macro with a 6.6 ns latency, 1241 GOPS and 37.01 TOPS/W for 8b-MAC operations for edge-AI devices," in *IEEE Int. Solid-State Circuits Conf. (ISSCC) Dig. Tech. Papers*, vol. 65, Feb. 2022, pp. 1–3.

[20] K. Zhang et al., "Progress in rectifying-based RRAM passive crossbar array," *Sci. China Technological Sci.*, vol. 54, no. 4, pp. 811–818, Apr. 2011.

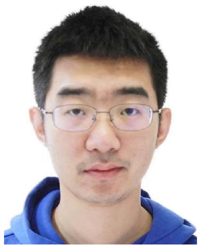
[21] C.-X. Xue et al., "A 1 Mb multibit ReRAM computing-in-memory macro with 14.6 ns parallel MAC computing time for CNN based AI edge processors," in *IEEE Int. Solid-State Circuits Conf. (ISSCC) Dig. Tech. Papers*, Feb. 2019, pp. 388–390.

[22] C.-X. Xue et al., "A 22 nm 2 Mb ReRAM compute-in-memory macro with 121–28 TOPS/W for multibit MAC computing for tiny AI edge devices," in *IEEE Int. Solid-State Circuits Conf. (ISSCC) Dig. Tech. Papers*, Feb. 2020, pp. 244–246.

[23] Y.-C. Chiu et al., "A 22 nm 4 Mb STT-MRAM data-encrypted near-memory computation macro with a 192 GB/s read-and-decryption bandwidth and 25.1–55.1 TOPS/W 8b MAC for AI operations," in *IEEE Int. Solid-State Circuits Conf. (ISSCC) Dig. Tech. Papers*, vol. 65, Feb. 2022, pp. 178–180.

[24] 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.

[25] L. Wang et al., “Efficient and robust nonvolatile computing-in-memory based on voltage division in 2T2R RRAM with input-dependent sensing control,” *IEEE Trans. Circuits Syst. II, Exp. Briefs*, vol. 68, no. 5, pp. 1640–1644, May 2021.



**Junjie Wang** received the Ph.D. degree in microelectronics from the University of Electronic Science and Technology of China, Chengdu, China.

He is currently a Post-Doctoral Researcher with the University of Electronic Science and Technology of China. His current research interests include digital circuit design, nonvolatile memory devices, and their applications in artificial intelligence.

**Teng Zhang** received the Ph.D. degree from the Institute of Microelectronics, School of Electronics Engineering and Computer Science, Peking University, Beijing, China.

He is currently a Post-Doctoral Researcher with the School of Integrated Circuits, Peking University. His research interests include emerging memory technologies and their applications in sensing, logic, and neuromorphic computing.

**Shuang Liu** received the B.S. degree in microelectronics from the University of Electronic Science and Technology of China, Chengdu, China, where he is currently pursuing the Ph.D. degree.

His current research interests include processing-in-memory circuits and neuromorphic systems.

**Yihe Liu** received the B.S. degree in microelectronics from the University of Electronic Science and Technology of China, Chengdu, China, where he is currently pursuing the Ph.D. degree.

His current research interests include processing-in-memory circuits and neuromorphic systems.

**Yuancong Wu** received the Ph.D. degree in microelectronics from the University of Electronic Science and Technology of China, Chengdu, China.

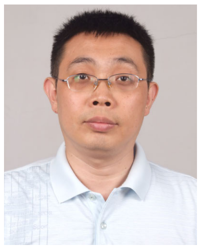
His current research interests include artificial intelligence chips and systems.

**Shaogang Hu** (Member, IEEE) received the Ph.D. degree in microelectronics from the University of Electronic Science and Technology of China, Chengdu, China.

He has been an Associate Professor with the University of Electronic Science and Technology of China since 2016. His current research interests include thin-film transistor, nonvolatile memory devices, and their applications in artificial intelligence.

**Tupei Chen** received the Ph.D. degree from The University of Hong Kong, Hong Kong, in 1994.

He is currently an Associate Professor with the School of Electrical and Electronic Engineering, Nanyang Technological University, Singapore.



**Yang Liu** received the B.Sc. degree in microelectronics from Jilin University, Changchun, China, in 1998, and the Ph.D. degree from Nanyang Technological University, Singapore, in 2005.

From May 2005 to July 2006, he was a Research Fellow with Nanyang Technological University. In 2008, he joined the School of Microelectronics, University of Electronic Science and Technology, Chengdu, China, as a Full Professor. He has authored or coauthored more than 130 peer-reviewed journal articles and more than 100 conference papers. He holds one U.S. patent and more than 30 Chinese patents. His current research includes memristor neural network system, neuromorphic computing ICs, and AI-RFICs.

Dr. Liu received the prestigious Singapore Millennium Foundation Fellowship in 2006.

**Yuchao Yang** (Member, IEEE) is currently a Boya Distinguished Professor with the School of Integrated Circuits, Peking University, Beijing, China. He also serves as the Deputy Dean of the School of Electronic and Computer Engineering and the Director of the Center for Brain Inspired Chips. He has authored more than 140 papers in high-profile journals and conferences, such as *Nature Electronics*, *Nature Reviews Materials*, *Nature Communications*, *Nature Nanotechnology*, *Science Advances*, *Advanced Materials*, *Nano Letters*, and International Electron Devices Meeting (IEDM). His research interests include memristors, neuromorphic computing, and in-memory computing.

Dr. Yang was invited to give more than 40 keynote/invited talks on international conferences and serves as the TPC chair or a member for nine international conferences.

**Ru Huang** (Fellow, IEEE) received the Ph.D. degree from Peking University, Beijing, China, in 1997.

Since 1997, she has been a Faculty Member with Peking University, where she is currently a Boya Chair Professor. She currently serves as the President at Southeast University, Nanjing. Her research interests include nanoscale CMOS technology, ultralow-power devices, emerging memory, neuromorphic computing, and reliability.

Dr. Huang is an elected Academician of the Chinese Academy of Sciences and a TWAS Fellow.