A Low-Power In-Memory Multiplication and Accumulation Array With Modified Radix-4 Input and Canonical Signed Digit Weights

Rui Xiao, Graduate Student Member, IEEE, Yawei Zhang, Student Member, IEEE, Bo Wang, Senior Member, IEEE, Yanfeng Xu, Jicong Fan, Haibin Shen, and Kejie Huang, Senior Member, IEEE

Abstract—Data transfer between the processing and storage units has become a significant bottleneck in modern von Neumann computing systems for artificial intelligence (AI) tasks. Conventional von Neumann computing (CIM) has emerged as a promising candidate for lowering latency and power consumption. However, the conventional analog CIM schemes are suffering from reliability issues, which may significantly degenerate the accuracy of the computation. Recently, digitized input data and weights have been utilized for high-reliable in-memory computing. However, the properties of the digital memory and input data are not fully utilized. This article presents a novel low-power CIM scheme to further reduce the power consumption by using a modified radix-4 booth algorithm at the input and a modified canonical signed digit (M-CSD) for the network weights. The simulation results show that M-RD4 and M-CSD reduce the number of nonzero activation bits by 24.2% and the number of nonzero weight bits by 36.0% in AlexNet, respectively. The power consumption can be reduced by 41.6% on average. The computing-power ratio at the fixed-point 8 bit is 60.7 tera operations per second per watt (TOPS/W), and the density is 0.177 TOPS/mm².

Index Terms—Canonical signed digit, charge redistribution integrator, in-memory computing, nonvolatile memory (NVM), radix-4 booth recoding.

I. INTRODUCTION

T
HE unceasing development of artificial intelligence (AI) and deep neural networks (DNNs) demands computing systems that can handle massive parameters and operations [1]. Conventional von Neumann architecture suffers from issues of separated processor and memory, limited on-chip memory, and limited memory bandwidth, resulting in significant energy consumption and latency due to data transmission and memory access [2]. The computing-in-memory (CIM) scheme, which can perform the multiplication-and-accumulation (MAC) operations within the memory to significantly reduce the data movement, has become a promising candidate to surpass the “von Neumann bottleneck” [3], [4]. As shown in Fig. 1, a conventional CIM scheme is formed by the off-chip memory, numerous interconnected low-power computing cores, and their buffers. The CIM cores, which are the crucial component of a CIM scheme, typically consist of input buffer, memory array, CIM neuron, and output buffer. The emerging resistive nonvolatile memory (NVM), such as spin-torque-transfer memory (STT-MRAM), phase change memory (PCM), and resistive random access memory (RRAM) [5], [6], [7], [8], are widely used to implement memory arrays. The NVM cells are organized into crossbar arrays, and the weight is represented by the cell conductance, and according to Kirchhoff’s law, the weighted sum of the MAC operation is represented by the current on each bitline (BL).

Multilevel cells (MLCs) and single-level cells (SLCs) are two types of NVM cells commonly used in CIM cores. MLCs are capable of storing multiple bits of data per cell, while SLCs can store only a single bit per cell. MLCs exhibit good performance in terms of high density and high parallelism [9], [10], [11], [12], [13]. However, they also suffer from issues related to slow speed, low reliability, and complex peripheral circuits [14]. In light of these limitations, SLCs are widely used in recent studies to achieve high speed and reliability.

Fig. 1. General CIM architecture; the CIM scheme comprises off-chip memory, interconnected computing cores, and their buffer. The computing core is organized with the memory crossbar array, CIM neurons, and buffers.
Some works adopt 1-bit weights to achieve high density [15], [16], [17], but the performance is significantly compromised in complex applications due to the limitation of precision. Therefore, using multiple SLCs to achieve multibit weight is widely used to achieve high precision [18], [19], [20], [21], [22], [23].

The 2’s complementary code is widely used in synapses with SLCs to represent multibit weights [18], [19], [20]. However, the uncertainty of the memory resistance may cause a big jump from the most negative to the most positive values. Differential weight can address this issue by using a pair of SLCs to represent 1-bit weight [21], [22], [23]. However, this approach leads to an unbalanced distribution of nonzero bits, resulting in significant integration differences among different BLs. Therefore, an integration circuit with a high-dynamic range is required, which results in higher power consumption and lower reliability. In this article, differential weights with modified canonical signed digital (M-CSD) code are proposed to leverage the unbalanced nonzero and zero bits in weights to address the above issues. The modified radix-4 (M-RD4) booth algorithm is also used to further reduce the percentage of nonzero bits in the computation. The simulation results show that the total power consumption is reduced by more than 41.55%. The performance-power ratio is 60.7 tera operations per second per watt (TOPS/W) with 8-b precision.

The rest of this article is organized as follows. Section II introduces the background and related works of the resistive NVM-based CIM architectures. Section III discusses the detailed design of our proposed CIM core, including M-RD4, M-CSD, the integration scheme to perform MAC operations, and the corresponding circuits. Section IV provides simulation results of our proposed core. Finally, the conclusion is drawn in Section V.

II. BACKGROUND AND RELATED WORKS

A. CIM Core Architectures

The majority of computations in neural networks, matrix MAC operations, can be efficiently implemented using the CIM crossbar. CIM schemes can simultaneously access the whole array to perform the MAC operations, thus significantly reducing the latency and power for computing and memory access. Hu et al. [24], Shafiee et al. [25], and Jiang et al. [26] utilize the conductance ($G_i$) of MLCs to achieve analog weights, and analog input voltage ($V_{in}$) is used to implement activations. Therefore, the output current ($I_{out}$) can be expressed as $I_{out} = \sum V_{in}G_i$, where multiplications are realized by Ohm’s law, and accumulations are realized by Kirchhoff’s current law. The output current will be converted to analog voltage amplifiers with the feedback resistor or sample and hold (S&H) circuits. The analog signals are difficult to be preserved and also sensitive to noises. Therefore, analog computing cores require digital-to-analog converters (DACs) and analog-to-digital converters (ADCs) to convert signals at the input and output interfaces, which will consume enormous power and area, significantly limiting the efficiency of the scheme. In light of these limitations, [25] proposed the ISAAC scheme to use 1-bit DACs with 16 cycles instead of 16-bit high-cost DACs in one cycle. Moreover, the 16-bit digital output is generated by eight 8-bit ADCs and the shift and add circuits. However, the output precision is far less than 8 bit due to the nonlinearity of multilevel RRAM cells and the variety of the S&H voltage. Moreover, the shift and add operation will further reduce the accuracy, because ADC’s quantization error is magnified after the shift operation.

Several SLC-based CIM cores have been proposed to avoid the nonlinearity issue of MLCs. Wan et al. [27] proposed a voltage-mode sensing scheme with 1-bit weights and 3-bit activations, and Li et al. [16] proposed an ADC-less CIM core with 1-bit weight and activations. These schemes achieve high density and energy efficiency, but the limited precision will result in significant information loss when applied to large-scale networks. To address the abovementioned issues and improve the energy efficiency, [18] proposed a multiple binary RRAM with active integrator (MBRAI) core scheme. As shown in Fig. 2, multiple single-level RRAM cells are used to represent an 8-bit weight instead of a multilevel RRAM cell. The core uses binary code at the input instead of a time or analog signal. The $n$-bit data are sequentially computed in the integral multiplier and weighted at the output neurons. However, the amplifiers in the neurons are power-hungry components, which consume more than 95% of power in the scheme. The computing efficiency of the CIM core is limited to 0.61 TOPS/W. Hung et al. [19] proposed a dc-current-free time-space (DCFTS)-based CIM core with a voltage-to-time converter, hidden-latency time-to-digital converter, and timing calibration table. This scheme achieves an energy efficiency of 21.6 TOPS/W with 8-bit precision. However, the reliability is highly dependent on the threshold voltage of the transistors. Moreover, it is also challenging to achieve the claimed precision through shift and add circuits. Zhang et al. [20] proposed a CIM core with a regulated passive neuron and bitline weight mapping (RPN-BWM) scheme, where the regulators in the bit lines are used to improve the linearity of the integration process, and the weight mapping scheme is used to address the resistance inconsistency.

The sign bit holds a crucial position in the 2’s complementary code. Due to the uncertainty of the memory resistance, there will be a big jump in the weight value if the sign bit is incorrect, especially for the most negative value (i.e., $8'\text{b}10000000$) and the most positive value...
(i.e., 8'b01111111). Therefore, some researchers use differential weights to represent positive and negative weights. Yao et al. [22] proposed a memristor-based hardware system with reliable multilevel conductance states for a five-layer memristor-based CNN for MNIST digit image recognition. It achieves energy efficiency with 11.15 TOPS/W at 8-bit activations and 8-bit weights. However, it requires 32 times analog-to-digital conversions to finish one MAC operation, which consumes about 92.14% energy in the system. Moreover, it still suffers from the drawbacks of MLCs and shift and add circuits. Liu et al. [23] proposed a signed-over, it still suffers from the drawbacks of MLCs and resolution-adjustable ADC. The peak energy efficiency is as high as 78.4 TOPS/W with 1-bit activations and ternary weights. Nevertheless, the resolution-adjustable ADC converts 1-bit per clock, leading to much more latency for high precision.

B. RRAM

RRAM is a two-terminal element with a metal oxide layer with variable conductance. The conductance responds to filament growth and destruction [2]. Since all resistive NVMs have high write power, the network weights are usually trained offline on the server and then sent to the CIM cores for inference. In our proposed scheme, single-level RRAM is used as the memory cell due to its high ratio of resistance in the high resistance state (HRS) and low resistance state (LRS). The RRAM model used in this scheme is realized by Verilog-A language, and the parameters of the RRAM model are based on the experimental results of Al2O3/Cu-based RRAM in [28]. The mathematical RRAM model is proposed by Jiang et al. [29] and can be expressed as follows:

$$\frac{dg}{dt} = -v_0 \times e^{-\frac{t}{\tau}} \times \sinh \left( \gamma \times \frac{a_0}{t_{ox}} \times \frac{qV}{kT} \right)$$

$$\gamma = \gamma_0 - \beta \times g^\alpha$$

$$I = I_0 \times e^{-\frac{t}{\tau}} \times \sinh \left( \frac{V}{V_0} \right)$$

where $g$, $v_0$, $E_A$, $a_0$, $t_{ox}$, $V$, and $\gamma$ are the parameters of the cell; $\gamma_0$, $\alpha$, and $\beta$ are fitting parameters; and $I_0$, $g_0$, and $V_0$ are determined by the experimental results. Fig. 3 shows the $I-V$ curve of the RRAM in [28] and the fitted parameters and $I-V$ curve of the fitted RRAM model. The reading voltage in our proposed scheme is 0.36 V, where the HRS/LRS ratio is around 9.8G/11M = 890. The high HRS/LRS makes the RRAM cells more reliable to represent the weights.

III. PROPOSED IN-MEMORY COMPUTING CORE

In this article, we propose a low-power MAC core with M-RD4 activations and M-CSD weights. M-RD4 is proposed to reduce the nonzero bits of activations. M-CSD is used to balance the distribution of nonzero bits and reduce the nonzero bits of weights. The overall structure of our proposed scheme is shown in Fig. 4, which consists of six components, including an M-RD4 generator, differential RRAM array, regulator, integrator, controller, and differential ADC. To be simplified, only $8 \times P \times 1$ crossbar memory cells are illustrated in

$$X_k = 2^{-n}x_{k,n-1} + \cdots + 2^{i}x_{k,i} + \cdots + 2^{0}x_{k,0}.$$
Algorithm 1 M-RD4 Booth Code

Input: Binary n-bit $X = x_{n-1}x_{n-2}...x_1x_0$.
Output: M-RD4 m-bit $Z = z_{m-1}z_{m-2}...z_1z_0$, $m = \lceil \frac{n}{2} \rceil$

1: // Append a '0' to the right of the LSB
2: // Extend a sign-bit at the left of the MSB if necessary
3: if $n$ is even then
4: $T[0:0] = x_{n-1}x_{n-2}...x_1x_0$ 0
5: else
6: $T[1:0] = x_{n-1}x_{n-2}...x_1x_0$ 0
7: end if
8: $i \leftarrow 0$, $j \leftarrow 0$
9: while $i \leq n - 2$ do
10: // Observe 4-bits with 2-bit overlap and transfer the sequence if necessary.
11: if $t_{i+3}t_{i+2}t_{i+1}t_i = '0100'$ then
12: $t_{i+3}t_{i+2}t_{i+1}t_i \leftarrow '0111'$
13: else if $t_{i+3}t_{i+2}t_{i+1}t_i = '1011'$ then
14: $t_{i+3}t_{i+2}t_{i+1}t_i \leftarrow '1100'$
15: end if
16: // Get the M-RD4 code $Z$ from LSB to MSB.
17: $z_j \leftarrow -2t_{i+2} + t_{i+1} + t_i$
18: $i \leftarrow i + 2$
19: $j \leftarrow j + 1$
20: end while
21: return $Z$

Radix-4 booth code is a modified booth code used for high-speed and low-power computing, widely used to design the multipliers to halve the number of partial products. However, the radix-4 code sometimes leads to more nonzero bits than that in binary codes (i.e., 01010010 is encoded to 1112). We propose the M-RD4 code based on radix-4 code to further reduce the number of nonzero bits in activations. The M-RD4 algorithm is illustrated in Algorithm 1. First, append a ‘0’ to the right of the least significant bit (LSB) of the X and then extend the sign bit to one position if necessary to ensure that $n$ is even. Afterward, every four binary bits (with 2 bits overlapped) are observed to do the data convention. If the sequence is “0100,” it will be turned into “0011.” Else if the sequence is “1011,” it will be turned into “1100.” After that, the right three bits will be encoded by using

$$z_j = -2t_{i+2} + t_{i+1} + t_i$$ (5)

where $i = 0, 2, 4, ..., j = \lceil i/2 \rceil$, $t_i$ is the $i$th bit of $T$, and $T$ is the temporary string defined in Algorithm 1. The activations represented by M-RD4 code can be expressed as follows:

$$X_k = 4^{m-1}x_{k,m-1} + \cdots + 4^jx_{k,j} + \cdots + 4^0x_{k,0}$$ (6)

where $m = \lceil (n/2) \rceil$ and $x_{k,i} = 0$ or ±1 or ±2. The data range of 4-bit M-RD4 code is $-170$ to $170$ ($2222$ to $2222$). Therefore, our proposed M-RD4 can represent both positive and negative values. The M-RD4 code can further reduce the number of nonzero bits in activations. For example, the M-RD4 code of “01010010” is encoded to “1102” in the M-RD4 code instead of “1112” in the radix-4 code.

The circuit implementation of the M-RD4 code is shown in Fig. 5. It comprises an in-MUX block, converter block, encoder block, and out-MUX block. The in-MUX consists of three 4-to-1 multiplexers and one quaternary counter. The counter generates the control signals ($S_A$ and $S_B$) to select the output of each multiplexer. The MUX block outputs the raw data ($a_{i+1}, a_{i+2},$ and $a_{i+3}$) for M-RD4 from the LSB to most significant bit (MSB). In the first clock, $a_i = 0$, and then, $a_i$ is determined by the output of the converter ($t_{i+2}$) in the previous clock. The converter block converts the raw data and sends the outputs ($t_{i+2}, t_{i+1}$, and $t_i$) to the encoder block. The encoder block generates and outputs the M-RD4 code. The value of the output [$z_j$ in (5)] is represented by $Z_2$, $Z_2$, $Z_1$, and $Z_1$. If $z_j = 1$, then the voltage of $Z_1$ is high, and the others are low, and the other cases can be speculated. When $z_j$ is encoded to zero, the multiplication result is always zero, and all four output voltages are low. Therefore, one of them will be activated at a time. Finally, the out-MUX selects the encoded code with two signals ($S_{Z1}$ and $S_{Z0}$) and sends the outputs into the neuron circuit. The waveform of our proposed M-RD4 recoding circuit will be implemented in Section IV-A.

B. Modified CSD Weights

The 2’s complementary code representation is widely used in arithmetic logic and operation. However, it may not be the best form to minimize the power consumption for neural network computing. Fig. 6(a) shows the simplified distribution curve of the weights in a neural network. In unpruned DNN networks, the weight values often follow a normal distribution. Similarly, the activations follow a half-normal distribution, because all negative values have been forced to be zero after the ReLU activation function. If the weights and activations are qualified to 8-bit binary data using the 2’s complementary code, as shown in Fig. 6(b), the number of nonzero and zero bits will be balanced, which is not optimized for low-power computing. In contrast, differential weights can efficiently reduce the number of nonzero bits in weights. As shown in Fig. 6(c), the red line indicates positive values, and the blue line indicates negative values. Therefore, the weight can be represented as follows:

$$w = w_p - w_n = 2^{n-1}(b_{n-1} - c_{n-1}) + \cdots + 2^0(b_0 - c_0)$$ (7)

where $w_p$ and $w_n$ are the unsigned number representation, and $b_i$ and $c_i$ are the bits in the positive part and negative part of the weight.
Fig. 6. Simplified distribution curve of (a) weights, (b) weights in 2’s complement, and (c) differential weight.

Fig. 7. Number of consecutive ones in M-CSD weights. For the weights between \([-219, 219]\), at most two consecutive ones are allowed; for weights greater than 219 or smaller than \(-219\), more consecutive ones are allowed.

part of weight, respectively. For example, \(w_p = 8'b00000000\) and \(w_n = 8'b01110111\) represent weight \(-119\). As shown in Fig. 6(c), digits 1 and 10 are placed in the positive and negative parts of the weight, respectively. In this way, the majority of bits in the weights are 0, which could bypass the in-memory computing to save power consumption.

However, it does not fully utilize both parts of a differential weight. In addition, the unbalanced distribution of nonzero bits results in significant integration differences among different BLs. Consequently, a high-dynamic range integration circuit is required, leading to increased power consumption and decreased reliability. CSD representation is widely used to reduce the nonzero digits by introducing a new digit \(\overline{1}\) into the number to form a ternary number system. The pair \(b_i\) and \(c_i\) in (7) can be used to represent the digit set \{1,0,\overline{1}\} for a CSD code. A simple approach to encode a binary coding to a CSD code is to search the binary code from LSB to MSB, find a string of “1’s” followed by “0” (i.e., 0111), and replace them with the CSD representation (1000 \(\overline{1}\)). The process may need to be repeated several times to ensure there is no string of “1’s.” CSD representation still suffers from some shortcomings.

1) In a CSD number, two consecutive nonzero bits are not allowed. Thus, the maximum value of 8-bit CSD is limited to 170 (10101010). For those 8-bit binary numbers greater than 170, an extra bit is needed to represent them in CSD representation.

2) For string “011,” CSD representation (10\(\overline{1}\)) does not reduce the number of nonzero bits. Therefore, converting “011” to “10\(\overline{1}\)” is useless.

An M-CSD representation is proposed to compact the above limitations. The strings “11” and “\(1\overline{1}\)” are allowed in M-CSD. In this way, the maximum value is extended to 219 (11011011). As shown in Fig. 7, to achieve the same range as the binary code, more consecutive “1’s” will be allowed if the weight is greater than 219 or smaller than \(-219\). In this way, the M-CSD code perfectly fits the differential weight scheme. The main idea of M-CSD is shown in Algorithm 2. Strings containing three or more “1’s” will be replaced by 10, \(\overline{1}\), 0, and three or more “\(1\overline{1}\)’s” will be replaced by 01. Furthermore, a string containing the MSB will not be replaced. Therefore, the M-CSD representation of the above example \((-119)\) will be \(1001001\), and the number of nonzero bits is significantly reduced.

In our proposed scheme, the M-CSD weights are stored in the 1R1T pairs. As shown in Fig. 4, the black 1R1T cells represent the positive part \((b_i)\), and the blue cells represent the negative part \((c_i)\). An \(n\)-bit weight is represented by \(n\) pairs of 1R1T cells, and each pair represents 1 bit of the M-CSD code. The positive cell will be set to LRS, while the bit value is 1. Otherwise, the negative cell will be set into LRS, while the value is \(\overline{1}\). To comply with the M-CSD design rule, the weight 119 in the above example will be represented by \(w_p = 8'b00001001\) and \(w_n = 8'b1000000\), respectively. The weighting process for different weight bits is required to implement multibit weights by single-level 1R1T cells, and charge redistribution operation is adopted in our proposed scheme, which will be discussed in Section III-C.

Algorithm 2 Modified CSD Representation

**Input:** \(n\)-bit differential \(W_i = w_{n-1}w_{n-2}...w_0\).

**Output:** \(n\)-bit modified CSD \(W'_i = w_{n-1}w_{n-2}...w_0\).

1. \(Flag \leftarrow 0\)  //Mark the string containing the MSB.
2. \(i \leftarrow n - 1\)
3. //String containing MSB will not be replaced.
4. while \(i > 0\) & \(Flag \equiv 0\) do
5. if \(w_i = 0\) then
6. \(Flag \leftarrow 1\)
7. end if
8. \(i \leftarrow i - 1\)
9. end while
10. // From LSB to \(w_i\) do the M-CSD.
11. \(j \leftarrow 0, k \leftarrow 0\)
12. while \(j < i\) do
13. if \(w_{j+1}...w_j = 11011\) then
14. \(w_{j+1}...w_j \leftarrow 1\overline{01}\)
15. \(j \leftarrow j + 2\)
16. else if \(w_{j+1}...w_j = 1\overline{01}0\overline{1}\) then
17. \(w_{j+1}...w_j \leftarrow 1\overline{0}1\overline{0}\)
18. \(j \leftarrow j + 2\)
19. else if \(w_{j+1}...w_j = 111\) then
20. \(k \leftarrow j + 2\)
21. while \(w_k = 1\) do
22. \(k \leftarrow k + 1\)
23. end while
24. \(w_i = 10...1\)
25. \(j \leftarrow k\)
26. else if \(w_{j+1}...w_j = 1\overline{0}\overline{1}\) then
27. \(k \leftarrow j + 2\)
28. while \(w_k = \overline{1}\) do
29. \(k \leftarrow k + 1\)
30. end while
31. \(w_i = \overline{0}10...1\)
32. \(j \leftarrow k\)
33. else
34. \(j \leftarrow j + 1\)
35. end if
36. end while
37. return \(W'_i\)
the weighting process for M-RD4 digits (4 MAC operations for activations and weights. Therefore, each redistribution. The integration phases perform nonweighted
the neurons. Our proposed integration scheme contains three
analog-to-digital conversion is completed by a differential SAR ADC.
Fig. 8. Block diagram of the integration scheme. Digital activations/
weights are multiplied and accumulated in the integral multiplier, and the
analog-to-digital conversion is completed by a differential SAR ADC.

C. Neuron Circuit
The integral multiplier in our proposed CIM core is
designed for massive parallel MAC operations and data trans-
mission from digital to analog. As shown in Fig. 8, activations
and weights are differentially multiplied and accumulated at
the neurons. Our proposed integration scheme contains three
phases: positive integration, negative integration, and charge
redistribution. The integration phases perform nonweighted
MAC operations for activations and weights. Therefore, each
integrator has the same integral voltage for different activation
bits and weight bits. The charge redistribution phase performs
the weighting process for M-RD4 digits \((4^4, 4^3, \ldots, 4^m-1)\)
from LSB to MSB, where \(m\) is the length of M-RD4 code, and \(m=[(n/2)]\).

The integral neuron is designed as a symmetrical structure to
complete the positive and negative MAC operations separately.
The differential integrator is illustrated in Fig. 9(a). The positive
and negative circuits are compensated to each other, effectively
reducing the influence of parasitic parameters. The M-RD4 inputs are sequentially sent to the word lines from
LSB to MSB. The 1R1T cells are used in pairs to store
\(b_i\) and \(c_i\) mentioned in (7). The positive circuit is used for
MAC operations whose results are positive \((I_p \times W_p\)
and \(I_n \times W_n)\), while the negative circuit is used for MAC
operations with negative results \((I_p \times W_n\) and \(I_n \times W_p)\),
where \(I_p, I_n, W_p,\) and \(W_n\) are the positive input, negative input,
positive weight, and negative weight, respectively. In this way,
the number of the discharge path is reduced. Therefore, our
proposed integrator can achieve higher accuracy with lower
power. \(S_1\) controls the data input, \(S_p\) controls the positive
integral operation, and \(S_n\) controls the negative operation.
\(S_2, S_1\), and \(S_0\) control the integration phase and the
charge redistribution phase. \(S_5\) controls the sample phase and the
conversion phase of the ADC.

Fig. 9(b) shows the brief flowchart of different phases.
During the positive integration phase, \(S_4\) is open to separate
each integrator. After that, \(S_2\) and \(S_p\) are closed to clear the
charge in positive integral capacitors. Then, \(S_1\) is closed to
input the M-RD4 bits \((I_p = 1, \text{ and } I_n = -1)\), and \(S_2\) is open
to complete the 1-bit MAC of \((1 \times 1) + (-1 \times -1)\). After
the positive integration phase, \(S_p\) is open to keep the charge in
\(C_{p,i}\), and \(S_1\) is open to ensure no power is consumed by
the 1R1T cells. During the negative integration phase, \(S_4\) is
still open to ensure the integrator is separated. \(S_1\) and \(S_n\) are
closed to clear to charge in negative capacitors. After that,
\(S_1\) is closed with the input \(I_p = -1\) and \(I_n = 1\). The phase
completes the 1-bit MAC of \((-1 \times 1) + (1 \times -1)\). After two
integration phases, \(S_4\) and \(S_5\) are closed to complete the charge
redistribution phase, where the equivalent analog voltage \((V_p
for positive and \(V_n\) for negative) is generated. According to
the derivation process of [20], the positive or negative integration
voltage after one step of the charge redistribution phase is
\[
V_S = V_S^p - k \left( 2^{-1} \sum_{i=0}^{p-1} A_i G_{i,n-1} + 2^{-2} \sum_{i=0}^{p-1} A_i G_{i,n-2} \right)
+ \cdots + 2^{-n+1} \sum_{i=0}^{p-1} A_i G_{i,0} \right) \tag{8}
\]

where \(V_S\) represents \(V_{Sp}\) or \(V_{Sn}\), and \(V_S^p\)
represents the initial integral voltage. \(k = V_B T/C_f, p\) is the number
of the input layers, \(A_i\) is 1-bit M-RD4 input of the \(i\)th input line, and
\(T\) is the fixed period time for each integration. \(G_i\) is the
conductance of each binary-RRAM cell, which is \(1/R_H\) and
\(1/R_L\) in the HRS and LRS, respectively.

In our proposed scheme, the input pulse has only two
possible values, which can effectively reduce the 1R1T cells’
reading variation. Therefore, 1-bit M-RD4 data with different
values are computed sequentially. The bits in M-RD4 have the relationship \(A_{i,m-1} = 4^m A_{i,m-2} = \cdots = 4^{m-n} A_{i,0}\), which means
each bit needs two steps of charge redistribution operation
to achieve the weighting process for input data. As shown in
Fig. 9(b), four integration phases (two positive and two
dependent) and two charge redistribution phases are required
to perform the computing and weighting process for 1-bit
M-RD4 data. The first two integration phases mentioned above
compute the layers whose input is “1” or “-1.” As shown in
Fig. 9(c), the first charge redistribution phase uses the
sampling capacitor \(C_S\) to perform the weighting process for
input data. Let \(C_S = C_f/;\) the charge on the capacitors
\(C_{n-1} C_{n-2}, \ldots, C_0\) and \(C_S\) are equally divided after the
charge redistribution operation. Taking the positive integrator as an
example, the voltage \(V_{p,a}\) of \(C_S\) can be expressed as follows:
\[
V_{p,a} = \frac{1}{2} (V_{Sp,a} + V_{p}^-) \tag{9}
\]
where \(V_{p}^-\) represents the previous positive voltage in \(C_S\) and
\(V_{Sp,a}\) represents the positive integration voltage for layers
with input “1” and “-1.” In the second two integration phases, the
layers with input “2” or “-2” are input and computed. The
positive voltage of \(C_S\) after the second charge redistribution
phase is
\[
V_p = \frac{1}{2} (V_{Sp,b} + V_{p,a}) = \frac{1}{2} V_{Sp,b} + \frac{1}{4} V_{Sp,a} + \frac{1}{4} V_{p}^- \tag{10}
\]
where \(V_{Sp,b}\) is the positive integration voltage for layers
with input “2” and “-2” and \(V_{p}^-\) is the positive output voltage
after the last input bit is computed. Equation (10) described
the for-loop process for each bit of the input data. Therefore,
the input data are weighted by \(4^m - 1, 4^m - 2, \ldots, 4^0\) from
bits and weight bits are completed. The \( V_{\text{exponential linear unit (ELU), MaxOut, and more.}} \) handle various activation functions, such as Leaky ReLU, lookup tables. Therefore, our proposed scheme can effectively functions can be efficiently approximated and computed using approximations, such as lookup tables, can be employed to 8-bit binary code by the differential SAR ADC. Efficient evaluations are done on the PyTorch platform.

In this way, the weighting processes for different activation bits and weight bits are completed. The \( V_{\text{out}} \) is then converted to 8-bit binary code by the differential SAR ADC. Efficient approximations, such as lookup tables, can be employed to express any activation function with high accuracy and precision. Furthermore, due to the ability of both 8-bit digital output and our proposed M-RD4 code to represent positive and negative values, they can effectively represent the inputs and outputs of various activation functions. These activation functions can be efficiently approximated and computed using lookup tables. Therefore, our proposed scheme can effectively handle various activation functions, such as Leaky ReLU, exponential linear unit (ELU), MaxOut, and more.

IV. Simulation Results

In this section, we first verify the functionality of our proposed CIM core. Then, we evaluate the performance of our proposed core, including energy cost evaluation, area estimation, dynamic performance, and comparisons on both the core level and network level. Afterward, we give an analysis of our proposed M-RD4 code and M-CSD code. Finally, we analyze the functionality of the circuit. The circuit-level simulations are done in cadence analog mixed signal (AMS) with a 45-nm generic process design kit (PDK). The network-level evaluations are done on the PyTorch platform.

LSB to MSB. Initially, \( V_p \) is reset to \( V_{dd} \), and after \( m \)-bit input data are computed, it can be expressed as follows:

\[
V_p = 4^{-m} V_{dd} + 4^{-m} V_{Sp,i} + \ldots + 4^{-1} V_{Sp,m-1}
\]  

where \( V_{Sp,i} = V_{Sp,a,i} + 2V_{Sp,b,i} \), and the change of the \( V_p \) is

\[
\Delta V_p = V_{dd} - V_p = 4^{-m} \sum_{i=0}^{m-1} 4^i \Delta V_{Sp,i}
\]

where \( \Delta V_{Sp,i} = \Delta V_{Sp,a,i} + 2\Delta V_{Sp,b,i} \) is the \( i \)-th positive integration voltage, and \( \Delta V_{Sp,i} \) is the change of \( V_{Sp} \) in the \( i \)-th integration. Therefore, the output voltage is

\[
V_{out} = \Delta V_p - \Delta V_n = 4^{-m} \sum_{i=0}^{m-1} 4^i (\Delta V_{Sp,i} - \Delta V_{Sn,i})
\]

In this way, the weighting processes for different activation functions and outputs of various activation functions. These activation functions can be efficiently approximated and computed using lookup tables. Therefore, our proposed scheme can effectively handle various activation functions, such as Leaky ReLU, exponential linear unit (ELU), MaxOut, and more.

A. Functionality Verification

The transient simulation is conducted to validate the functionality of the circuit. Fig. 10 shows the waveform of recoding the binary input to our proposed M-RD4 code. The encoding process for four inputs (10101010, 01111111, 01111101, and 00000000) is presented to demonstrate the functionality of the encoder. In the first clock, \( S_A = 0, S_B = 0, a_i + 3a_i + 2a_i + a_i = x_2x_1x_0, \) and \( a_i = 0 \). Then, the outputs \( (t_i + 2t_i + t_i) \) of the converter are generated and sent into the encoder to generate the M-RD4 output. In the second clock, \( S_A = 0, S_B = 1, a_i + 3a_i + 2a_i + a_i = x_4x_3x_2Q, \) where \( Q \) equals \( t_i + 2t_i + t_i \) at the last clock. In the third clock, \( S_A = 1, S_B = 0, a_i + 3a_i + 2a_i + a_i = x_6x_5x_4Q. \) In the fourth clock, \( S_A = 1, S_B = 1, a_i + 3a_i + 2a_i + a_i = x_7x_6Q. \) The four output bits, either at \( V_{dd} \) or ground, are sent into the RRAM array from LSB to MSB and directly used for in-memory computing.

Fig. 11 illustrates the transient waveform of 1-bit MAC operations. First, the input “1” is sent into the positive memories, and “−1” is sent into the negative memories to generate the BL currents. Subsequently, \( S_A \) is closed to reset the positive integral capacitors, and \( S_p \) is closed to complete the multiplication of “1 × 1” and “−1 × −1.” The difference of
the integration voltage of the positive and negative is increased to 280.2 mV linearly. Following that, input “1” is directed to negative memories, and “−1” is directed to positive memories. $S_3$ and $S_4$ are then closed to complete the multiplication of “1 × −1” and “−1 × 1.” After then, $S_4$ is closed to complete the charge redistribution phase, and the output voltage $V_{out}$ is 140.3 mV. Subsequently, the computations of “2 × 2,” “−2 × −2,” “2 × −2,” and “−2 × 2” are sequentially performed. Finally, $S_4$ is closed again to conduct the second charge redistribution, thereby completing the weighting process of different input values. Using the difference as output can effectively mitigate the impact of parasitic parameters on accuracy. Fig. 12 shows the complete MAC operations of four input and weight combinations: “max” denotes input = 2222, weight = 11111111, “min” denotes input = 2222, weight = 11111111, “R1” denotes random input = 2011, random weight = 10000101, “R2” denotes random input = 1021, and random weight = 01100101.

Finally, $S_4$ is closed again to conduct the second charge redistribution, thereby completing the weighting process of different input values. Using the difference as output can effectively mitigate the impact of parasitic parameters on accuracy. Fig. 12 shows the complete MAC operations of four input and weight combinations: “max” denotes input = 2222, weight = 11111111, “min” denotes input = 2222, weight = 11111111, “R1” denotes random input = 2011, random weight = 10000101, “R2” denotes random input = 1021, and random weight = 01100101. After four cycles of 1-bit MAC operations shown in Fig. 11, the results of MAC are represented by the analog voltage $V_{out}$, which will be subsequently converted to digital signals by the ADC. Fig. 13 presents the output voltage with different input and weight combinations. The $V_{out}$ closely aligns with the baseline and exhibits remarkable linearity, providing compelling evidence of the circuit’s functional correctness.

B. Performance

Table I gives the energy cost comparison of [18], [20], [22], and our proposed scheme. Zhang et al. [18] consume 0.22 mW on amplifiers for stable read voltage, which means that amplifiers consume more than 90% power, resulting in total power consumption is 199.68 mW. The ADCs consume more than 85% of energy in [22], while the power consumption for 128 × 256 core is 7.44 mW. Zhang et al. [20] use regulators with 1.11-µW power consumption to keep the read voltage stable, and the total power consumption is 3.61 mW. In contrast, the power consumption of our proposed core is only 2.00 mW. Compared with [18], [22], and [20], the power consumption of our proposed scheme is reduced by 98.9%, 73.1%, and 44.6%, respectively.

The core-level comparison between our proposed scheme and the conventional CIM core schemes is shown in Table II. Our proposed core has a capacity of 64 kb and a data precision of 8 bit. The energy efficiency is as high as 60.7 TOPS/W in 8-bit precision, which is much greater than most conventional designs. Yoon et al. [12] achieves 118.4-TOPS/W peak energy efficiency with ternary weights. However, the reliability is constrained by the shift and add circuits. Moreover, it is challenging to achieve high-precision applications. Table III shows the area configuration of our proposed core. To estimate the area of our proposed scheme, the metal–oxide–metal (MOM) capacitors proposed in [31] are used to implement the integrator and ADC, which can achieve a capacitance density of 2.75 fF/µm². The area of each 1R1T cell is based on [23], which is 0.2025 µm². Our proposed core achieves a density of 0.177 TOPS/mm² with an area of 0.686 mm².

The scalability of our proposed core can be analyzed through several key aspects, including array size, precision, and process. In our proposed scheme, the array size is set at 256 × 256 × 16, with each pair of points storing an individual weight. Increasing the size of the array results in larger...
TABLE III

| Encoder     | Area comp (μm²) | Total Area (μm²) |
|-------------|-----------------|-----------------|
| 1R1T Array  | 0.2025          | 0.2123          |
| Integrator  | 95.38           | 0.3907          |
| Regulator   | 0.1395          | 0.0007          |
| ADC         | 52.02           | 0.0824          |
| Sum         | -               | 0.6861          |

BL currents, which necessitates the use of larger capacitors in the integrator or shorter integration times. However, due to the need to maintain a high signal-to-noise ratio (SNR), further reduction of capacitor size to accommodate small BL currents becomes difficult. Therefore, it is essential to select a predetermined array size in order to achieve the desired SNR with the minimum possible capacitor size. Our proposed scheme uses 8-bit inputs and 8-bit weights. Expanding the precision of the weight leads to an exponential increase in the capacitor area with a power of two. The use of shift and add circuitry introduces additional bit widths that are not reflective of actual data, with the lower bits mostly comprising noise. The precision of the activations can be adjusted by changing the number of stimuli applied to each layer. Nonetheless, increasing the stimuli count results in a long operation time and necessitates an ADC redesign. Therefore, it requires a trade-off among energy efficiency, density, and precision. Furthermore, our proposed scheme can be utilized in various processes, such as RRAM, SRAM, and ferro-electric field effect transistor (FeFET), among others.

To estimate the accuracy of our proposed core, the quantization error and the impact of resistance distribution of transistors are taken into consideration. The activations and weights are first quantized from 32-bit floating codes to 8-bit fixed point and then encoded into our proposed M-RD4 code and M-CSD code, respectively. The quantization noise can be expressed as $Q_E = (\text{LSB}^2 / 12)$. To integrate the RRAM with the network, we established the nonideal model of the adapted 1R1T cells. The thermal noise of the cells can be expressed as a normal distribution. Fig. 14 gives the normalized distribution of the RRAM array. The reading method is proposed by Zhang et al. [20], where 1400 runs of Monte Carlo simulations are used to evaluate the compact of the transistors, and the thermal noises of the RRAM cells are represented by normalized Gaussian distribution with $\mu = 0$ and $\sigma = 0.2$. The quantization noise and thermal noise are injected into the weights during the forward pass with the method proposed by Rekhi et al. [32]. During the forward pass, the weights of every layer except the last layer are injected with the noises. Finally, the models of LeNet [33], AlexNet [34], and VGG-16 [35] with new weights are simulated on the PyTorch platform to evaluate the accuracy of the network.

Table IV shows the accuracy estimation of different RRAM-based schemes. On the MNIST dataset, our proposed core achieves an accuracy of 98.89% with LeNet. On the CIFAR-10 dataset, the accuracy is 91.85% on VGG-16. On the imagenet large scale visual recognition challenge (ILSVRC) 12 dataset, the accuracy is 56.90% on AlexNet. Our proposed scheme achieves an accuracy better than [16] on MNIST and CIFAR-10 and roughly equivalent to [18] and [20] on MNIST and ILSVRC2012. Compared with the software-based results, our proposed scheme has a slight accuracy reduction.

C. M-RD4 and M-CSD Analysis

According to the characteristics of RRAM, the array consumes energy only when both activations and weights are nonzero. Therefore, our proposed M-RD4 and M-CSD codes are dedicated to reducing the number of nonzero bits, thereby reducing power consumption. Fig. 15 shows the distribution of nonzero bits of the activations and weights in AlexNet. Both our proposed M-RD4 code and M-CSD code can significantly reduce the nonzero bits in activations and weights.
Fig. 16. Histogram of nonzero bits of (a) binary weights with 2’s complementary code, (b) differential weights, and (c) proposed M-CSD codes. The high ratio of nonzero bits in binary weights results in high-power consumption. The large distribution range of differential weights also implies the requirement of a high-dynamic integration circuit. Our proposed M-CSD weight achieves low-power consumption and high reliability by ensuring a small distribution range while maintaining a ratio of nonzero bits.

TABLE IV
ACCURACY ESTIMATION OF DIFFERENT RRAM-BASED SCHEMES

| Dataset | Scheme | TCAD-F B-20 [18] | TSVLSI22 [20] |
|---------|--------|-----------------|--------------|
| MNIST   | Accuracy | 99.10% | 99.10% |
| CIFAR10 | 93.66% | 93.55% | 93.55% |
| LSRC12  | 93.55% | 93.55% | 93.55% |
| Proposed | 99.25% | 99.25% |

Using LeNet for MNIST, VGG-16, and AlexNet for LSRC12.

TABLE V
POWER COMPARISON BETWEEN DIFFERENT CODE COMBINATIONS

| Code Combination | Binary | RD4 | M-RD4 | M-RD4+CSD | M-RD4+M-CSD |
|------------------|--------|-----|------|----------|-------------|
| Ratio of 1*1 (%) | 14.7%  | 13.3% | 11.2% | 3.9%     | 2.2%        |
| Power (mW)       | 3.61   | 2.80 | 2.66 | 2.21     | 2.00        |
| Saving (%)       | -22.46 | 26.19 | 38.69 | 41.55    |

* ‘1*1’ means both the activation and weight are non-zero.

Compared with binary codes, the length of activation is reduced by half. Using radix-4 code, the total number of nonzero bits of activations can be reduced by 22.0%, while our proposed M-RD4 code can further reduce it to 24.2%. Compared with the 2’s complementary weights, our proposed M-CSD code can reduce the number of nonzero bits by 36.0% in AlexNet. Therefore, power consumption can be significantly reduced. Table V gives the power consumption and the ratio of nonzero activations \( \times \) nonzero weights (ratio of 1 \( \times \) 1) of different codes. Take [20] as the standard, whose ratio of 1 \( \times \) 1 with binary input and binary weight is 14.7%, and the power consumption is 3.61 mW. The ratio of 1 \( \times \) 1 decreases to 13.3% by using radix-4 input. Furthermore, the ratio of 1 \( \times \) 1 is further decreased to 11.2% by using our M-RD4 in the input, and the power consumption is decreased by 26.19%. Applying the M-RD4 input and CSD weight, the ratio of 1 \( \times \) 1 decreases to 3.9%, and the power consumption decreases to 2.21 mW. Our proposed scheme with M-RD4 input and M-CSD weight further decreases the ratio of 1 \( \times \) 1 to 2.2% and the power consumption to 2.00 mW. Therefore, for a 256 \( \times \) 256 \( \times \) 16 core, our proposed scheme saves 41.55% of power consumption compared with [20], which uses 2’s complement code.

When implementing a DNN (i.e., AlexNet) with our proposed CIM core, each row corresponds to an input activation, and each eight columns correspond to the synaptic weights of the neuron. For a convolution layer, each neuron has \( N \times k \times k \) synapse and \( M \) neurons, where \( N \) is the input feature map, \( k \) is the kernel size, and \( M \) is the output feature map. Therefore, the weights are partitioned into multiple 256 \( \times \) 256 blocks and employed in our proposed CIM core. As the number of nonzero bits is directly proportional to the power consumption, the distribution of nonzero bits in each 1 \( \times \) 256 column from MSB to LSB is counted and plotted as histograms. As shown in Fig. 16, for the binary weights with 2’s complementary code, the ratio of nonzero bits is almost higher than 35.9% and has a mean greater than 46.0%. As a result, the power consumption using binary code would be substantial. Differential weights can reduce the ratio of nonzero bits. However, the distribution range of weights is significantly widened (4.7%–61.7%), resulting in a demand for integration circuits with a large dynamic range. In contrast, our proposed M-CSD weight has a low ratio (<42.3%) of nonzero bits as well as a small distribution of weights (11.0%–56.0%). Therefore, our proposed M-CSD weight achieves low-power consumption and high reliability.

D. Robustness

Differential nonlinearity (DNL) and integration nonlinearity (INL) are important parameters to evaluate the linearity of the output voltage. The linearity of different input values is illustrated in Fig. 17(a). The DNL of different input values ranges from \(-0.073\) to \(+0.464\) LSB, and the INL has a range of \(-0.809\) to \(-0.047\) LSB. Since the ideal range of DNL is \(-1\) to 1 LSB, our proposed scheme shows excellent linearity. The linearity of different input lines and weight values is shown in Fig. 17(b) and (c), respectively. The DNL and INL are also in the ideal range.

Table VI shows the dynamic performance comparison among [18], [20], and our proposed scheme. The M-RD4 encoder has a supply voltage of 0.6 V to further decrease...
than the others. Different processes, voltages, and temperatures (PVT) are chosen to do the simulation to verify the robustness of the circuit. ENOBs, as shown in Table VII, are all greater than 7.1 bits in different PVT combinations. Therefore, our proposed scheme is reliable with different variations of the process, voltage, and temperature.

The 1400 runs of Monte Carlo simulation are used to evaluate the robustness of our proposed scheme. Fig. 18 shows the histogram of integration error at different input/weight combinations. According to the statistical results, our proposed scheme shows great robustness in process variations and local mismatch.

V. CONCLUSION AND DISCUSSION

In this article, we have proposed the M-RD4 and canonical-signed-digit codes to reduce the ratio of nonzero bits in both activations and weights. The multibit weights are represented by multiple single-level 1R1T cells, which efficiently reduce the impact of the nonlinearities of the RRAM cells. Furthermore, using SLCs makes our proposed scheme friendly to various processes, such as RRAM, SRAM, FeFET, and so on. According to the simulation results, the ratio of nonzero bits of activations and weights is significantly reduced. Therefore, our proposed scheme has shown high energy efficiency. Moreover, our proposed M-CSD encoding also balances the distribution of nonzero bits in the weights to improve reliability. Our proposed scheme has shown strong robustness with various processes and local mismatch. The core is very robust with an ENOB of 7.42 bit, whose SFDR and SNDR achieve 63.41 and 46.48 dB. In this core, a large number of capacitors in the integrators consume a large area; therefore, further research is required to reduce the area while maintaining high accuracy. Our proposed scheme has adopted the minimum capacitors for integrators and ADC while maintaining the SNDR, which can support larger arrays without causing integration overflow.

REFERENCES

[1] Y. Long, T. Na, and S. Mukhopadhyay, “ReRAM-based processing-in-memory architecture for recurrent neural network acceleration,” IEEE Trans. Very Large Scale Integr. (VLSI) Syst., vol. 26, no. 12, pp. 2781–2794, Dec. 2018.

[2] W. Chen, Z. Qi, Z. Akhtar, and K. Siddique, “Resistive-RAM-based in-memory computing for neural network: A review,” Electronics, vol. 11, no. 22, p. 3667, Nov. 2022.

[3] Q. Xia and J. J. Yang, “Memristive crossbar arrays for brain-inspired computing,” Nature Mater., vol. 18, no. 4, pp. 309–323, Apr. 2019.

[4] S. Kundu, P. B. Ganganaiik, J. Louis, H. Chalamalasetty, and B. P. Rao, “Memristors enabled computing correlation parameter in-memory system: A potential alternative to von Neumann architecture,” IEEE Trans. Very Large Scale Integr. (VLSI) Syst., vol. 30, no. 6, pp. 755–768, Jun. 2022.

[5] X. Zhang, A. Huang, Q. Hu, Z. Xiao, and P. K. Chu, “Neuromorphic computing with memristor crossbar,” Phys. Status Solidi (A), vol. 215, no. 13, Jul. 2018, Art. no. 1700875.

[6] K. Huang, Y. Ha, R. Zhao, A. Kumar, and Y. Lian, “A low active leakage and high reliability phase change memory (PCM) based non-volatile FPGA storage element,” IEEE Trans. Circuits Syst. I, Reg. Papers, vol. 61, no. 9, pp. 2605–2613, Sep. 2014.

[7] G. W. Burr et al., “Experimental demonstration and tolerancing of a large-scale neural network (165 000 synapses) using phase-change memory as the synaptic weight element,” IEEE Trans. Electron Devices, vol. 62, no. 11, pp. 3498–3507, Nov. 2015.

[8] S. Jain, A. Ranjan, K. Roy, and A. Raghunathan, “Computing in memory with spin-transfer torque magnetic RAM,” IEEE Trans. Very Large Scale Integr. (VLSI) Syst., vol. 26, no. 3, pp. 470–483, Mar. 2018.
[9] M. Hu et al., “Memristor-based analog computation and neural network classification with a dot product engine,” *Adv. Mater.*, vol. 30, no. 9, Mar. 2018, Art. no. 1705914.

[10] E. Giacomin, T. Greenberg-Toledo, S. Kvatinsky, and P.-E. Gaillardon, “A robust digital RRAM-based convolutional block for low-power image processing and learning applications,” *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 66, no. 2, pp. 643–654, Feb. 2019.

[11] R. Xiao, W. Jiang, and P. Y. Chee, “An energy efficient time-multiplexing computing-in-memory architecture for edge intelligence,” *IEEE J. Explor. Solid-State Comput. Devices Circuits*, vol. 8, no. 2, pp. 111–118, Dec. 2022.

[12] J.-H. Yoon, M. Chang, W.-S. Khwa, Y.-D. Chih, M.-F. Chang, and A. Raychowdhury, “A 40-nm 118.44-TOPS/W voltage-sensing compute-in-memory RRAM macro with write veriﬁcation and multi-bit encoding,” *IEEE J. Solid-State Circuits*, vol. 57, no. 3, pp. 845–857, May 2022.

[13] M. Bettayeb, F. Zayer, H. Abunahla, G. Gianini, and B. Mohammad, “An efﬁcient in-memory computing architecture for image enhancement in AI applications,” *IEEE Access*, vol. 10, pp. 48229–48241, 2022.

[14] M. Liu and K. Chakrabarty, “Online detection in ReRAM-based computing systems for inﬁering,” *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 30, no. 4, pp. 392–405, Apr. 2022.

[15] Y. Zhao et al., “A 0.02% accuracy loss voltage-mode parallel sensing compute-in-memory RRAM macro with write veriﬁcation and multi-bit encoding,” *IEEE J. Solid-State Circuits*, vol. 57, no. 3, pp. 845–857, May 2022.

[16] M. E. Sinangil et al., “A 7-nm compute-in-memory SRAM macro with write veriﬁcation and multiplexing computing-in-memory architecture for edge intelligence,” *Proc. IEEE Symp. VLSI Technol.*, Jun. 2022.

[17] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner, “Gradient-based learning applied to document recognition,” *Proc. IEEE*, vol. 86, no. 11, pp. 2278–2324, Nov. 1998.

[18] A. Krizhevsky, I. Sutskever, and G. E. Hinton, “ImageNet classiﬁcation with deep convolutional neural networks,” *Commun. ACM*, vol. 60, no. 6, pp. 84–90, May 2017.

[19] K. Simonyan and A. Zisserman, “Very deep convolutional networks for large-scale image recognition,” *2014, arXiv:1409.1556*.
Yanfeng Xu received the bachelor’s degree in applied electronic technology from Jilin University, Changchun, China, in 2001, and the master’s degree in software engineering from Dongnan University, Nanjing, China, in 2011.

His research interests include architecture and circuit design for reconfigurable computing systems, field-programmable gate array (FPGA) architecture design, and embedded memory design.

Jicong Fan received the master’s degree in microelectronics and solid-state electronics from Anhui University, Hefei, China, in 2013.

His research interests include architecture and circuit optimization for reconfigurable computing systems, field-programmable gate array (FPGA) architecture design and evaluation, and floating-point unit design.

Haibin Shen is currently a Professor with Zhejiang University, Hangzhou, China, a member of the second level of 151 talents project of Zhejiang Province, and a member of the Key Team of Zhejiang Science and Technology Innovation. He has published more than 100 papers on academic journals and has been granted more than 30 patents of invention. His research interests include learning algorithm, processor architecture, and modeling. His research achievement has been used by many authority organizations.

Dr. Shen was a recipient of the First Prize of Electronic Information Science and Technology Award from the Chinese Institute of Electronics and has won a second prize at the provincial level.

Kejie Huang (Senior Member, IEEE) received the Ph.D. degree from the Department of Electrical Engineering, National University of Singapore (NUS), Singapore, in 2014.

He has been a Principal Investigator with the College of Information Science Electronic Engineering, Zhejiang University (ZJU), Hangzhou, China, since 2016. Prior to joining ZJU, he has spent five years at the IC design industry, including Samsung, Hangzhou, and Xilinx, Singapore, two years in the Data Storage Institute, Agency for Science, Technology and Research, Singapore, and another three years in the Singapore University of Technology and Design (SUTD), Singapore. He has authored or coauthored more than 70 scientific articles in international peer-reviewed journals and conference proceedings. He holds four granted international patents, and another eight pending ones. His research interests include in-memory computing circuits and systems design, software/hardware co-optimization for deep learning accelerators and cryptoprocessors.

Dr. Huang currently serves as an Associate Editor for the IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS—PART II: EXPRESS BRIEFS.