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

Rui Xiao, *Student Member, IEEE*, Kejie Huang, *Senior Member, IEEE*, Yewei Zhang, *Student Member, IEEE* and Haibin Shen

Abstract—A mass of data transfer between the processing and storage units has been the leading bottleneck in modern Von-Neuman computing systems, especially when used for Artificial Intelligence (AI) tasks. Computing-in-Memory (CIM) has shown great potential to reduce both 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, CIM schemes with digitized input data and weights have been proposed for high reliable computing. However, the properties of the digital memory and input data are not fully utilized. This paper presents a novel low power CIM scheme to further reduce the power consumption by using a Modified Radix-4 (M-RD4) 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 ratio of 1×1 by 78.5% on LeNet and 80.2% on AlexNet, and improve the computing efficiency by 41.6% in average. The computing-power rate at the fixed-point 8-bit is 60.68 TOPS/s/W.

Index Terms—Non-volatile Memory, In-memory Computing, Charge Redistribution Integrator, Radix-4 Booth Recoding, Canonical-Signed-Digit.

I. INTRODUCTION

ALONG with computer technology unceasing development, the Artificial Intelligence (AI) has been widely applied in various fields to perform specific tasks, such as transportation, education, healthcare, security, finance, etc [1]. With the support of massive data and high-performance hardware, the Deep Neural Network (DNN), a particular kind of machine learning, achieves excellent power and flexibility by learning to represent the world as a nested hierarchy of concepts [2]. However, due to the limited on-chip memory and memory bandwidth, a mass of intermediate data generated by DNN has to be transferred frequently between the separated computing units and storage units in conventional von-Neumann machines, resulting in a tremendous amount of power and propagation delay, which is treated as the "Von Neumann bottleneck" [3].

Inspired by the cranial nerve structure and information processing mechanism from the brain science research, the artificial intelligence and system are breaking through the

conventional computing architecture and promoting to the next generation of computing paradigm. The computing-in memory [4] scheme is formed by a large number of interconnected low-power computing units (neurons) and re-configurable storage units (synapses), which can perform the Multiplication-and-Accumulation (MAC) operations in the memory to significantly reduce the data movement. The emerging Resistive Random Access Memory (RRAM) is one of the best candidates in CIM design [5]–[7]. The resistance value of the memory can be used for weight storage and MAC operation. The memory cells are organized into crossbar arrays for high density storage, low power consumption, and fully parallel computing [4].

Analog computing with multi-level resistive memory is widely used to achieve a massive parallel low power computing [8]–[10]. However, data storage and transmission between computing cores require digital signals since analog signals are sensitive to noise. Most architectures require Digital-to-Analog Converters (DACs) and Analog-to-Digital Converters (ADCs) at the interface, which consume large area and high power consumption. Moreover, most of them have overlooked the defects of the resistive Non-volatile Memory (NVM), such as nonlinearity, stochasticity, asymmetry, etc [11]. To address the issues mentioned earlier, [12] proposes to use multiple binary RRAMs to emulate one synapse. Moreover, DACs are also moved to neurons to reduce the high driving power and the non-linearity caused by the analog input voltage. However, high-performance amplifiers are used to achieve high computing speed and 8-bit resolution, resulting high power consumption. The high power dissipation amplifiers are removed in [13] by regulating the voltage before the passive integral neurons. However, 2's complementary code is used in synapses, resulting in balanced '1's and '0's. Moreover, the uncertainty of the memory resistance in the MAC array with 2's complementary code may cause a big jump between the most negative and the most positive values. Differential weights with Modified Canonical Signed Digital (M-CSD) are proposed to leverage the unbalanced '1's and '0's in weights to address above issues. Modified radix-4 (M-RD4) booth algorithm is also used to further reduce the percentage of '1's in the computation. The simulation results show that the total power consumption is reduced by more than 41.55%. The performance-power ratio is 57.53 TOPS/s/W with 8b precision. The main contributions of this paper include:

1) The inputs are encoded with M-RD4 codes, the amount

The authors are with the College of Information Science & Electronic Engineering, Zhejiang University, 38 Zheda Road, Hangzhou, China, 310027, email: xiaor@zju.edu.cn; huangkejie@zju.edu.cn; yeweizhang@zju.edu.cn; shen_hb@zju.edu.cn.

K. Huang and H. Shen are also with Zhejiang Lab, Building 10, China Artificial Intelligence Town, 1818 Wenyi West Road, Hangzhou City, Zhejiang Province, China.

of '1's is halved since the encoding length of radix-4 booth codes is only half of binary encodes.

2) The weights are stored differentially with the M-CSD code to significantly reduce the number of '1's, which can complete the MAC operation with 41.55% less power computation.

3) Differential charge redistribution passive integrator and Successive Approximation Register (SAR) ADC are proposed to enable in-memory computing with M-RD4 and M-CSD algorithms.

The rest of the paper is organized as follows: Section II introduces the related works of the resistive non-volatile memory based in-memory computing circuits and architectures. Section III discusses the detailed design of the proposed CIM core, including M-RD4, M-CSD, the integration scheme to perform MAC operations, and the corresponding circuits. Section IV provides the circuit level and system level simulation results. Finally, the conclusion is drawn in Section V.

II. RELATED WORKS

Computing near memory and computing in memory are the two typical schemes to shorten the distance between the processing and storage units. Computing near memory such as IBM TrueNorth [14] and Intel Loihi [15] can only access the memory by one row each time, thus the processing speed is minimal. Besides, excessive charge and discharge of the bit lines will cost high power consumption. The CIM scheme could simultaneously access the whole array to perform the MAC operations, thus significantly reducing the latency and power for computing and memory access. Resistive NVMs such as memristor [16], Phase Change Memory (PCM) [5], [17], and RRAM [10] are the potential candidates to achieve the high-density CIM schemes. 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. The CIM schemes can be divided into two groups: CIM with analog memory and input signals, and CIM with digitized memory and input signals.

A. Analog Computing-in-Memory

In the analog CIM schemes, the multiplication is usually achieved by multiplying the conductance of the multi-level memory and input analog voltage based on Ohm's law [18], [19], which will output the current. The accumulation is usually done by converging output currents from different multiplications based on Kirchhoff's Current Law (KCL). The analog signals are difficult to be preserved and also sensitive to noises. Therefore, the converged current has to be converted to voltage signals for analog-to-digital conversion. The digital inputs are also converted to analog signals for analog computing. The DACs and ADCs will consume enormous power and area, significantly limiting the efficiency of the scheme.

A. Shafiee *et al.* [20] proposed the RRAM-based ISAAC scheme to perform 16-bit fixed-point MAC operation for CNN inference, where eight 4-level RRAM cells are used to store one 16-bit weight. As shown in Fig. 1, it takes 16 cycles to perform the 16-bit digital-to-analog conversion by 1-bit DACs

Fig. 1. The CIM core of the ISAAC architecture. The S/H voltage of each column is quantized by an individual 8-bit ADC at each cycle, and then the quantized results are shifted and added to achieve 16-bit output for MAC operations.

Fig. 2. The CIM core of the MBRAI architecture. The n-bit input data are sequentially computed in the integral multiplier and weighted at the output neurons

instead of 16-bit high-cost DACs in 1 cycle. In each cycle, the analog outputs are converted to digital signals by eight 8-bit ADCs, which are then shifted and added to generate 16-bit output. X. Qiao *et al.* [21] proposed AtomLayer to support 16-bit fixed-point CNN training and inference. The AtomLayer accesses the ability of training by processing one network layer each time. B esides, the data are redused to improve the efficiency. However, there are still some shortcomings in ISAAC and AtomLayer.

1) The S/H structure without an amplifier will seriously affect the analog computation accuracy due to the varying hold voltage.

2) The accuracy after ADC is far less than 8-bit due to the nonlinearity of multi-level RRAM cells and the loss of precision.

3) The shift-and-add operation will further reduce the accuracy because ADC's quantization error is magnified after the shift operation.

4) The eight ADCs and 16 cycles' conversion for 16-bit MAC operation leads to high power consumption.

B. Digitized RRAM based CIM Cores

Several single level RRAM based CIM cores have been proposed to avoid the nonlinearity issue of multi-level RRAM. M. Courbariaux *et al.* [22] and M. Rastefari *et al.* [23] used binary weight and 1-bit input for the recognition tasks on MNIST and CIFAR-10 datasets. However, 1-bit weight and 1-bit input will lose a lot of information when applied to large networks. C. Xue *et al.* [24] proposed a BL-IN-OUT (BLIOMC) scheme with Scrambled 2's Complement Weight Mapping (S2CWM), which exploits 4-bit inputs by 4-level read voltage and 4-bit weight represented by four single-level RRAM cells. The Dual-bit-Small-Offset Current-mode Sense Amplifier (DbSO-CSA) with two I_{REF} works as 2-bit ADC. It achieves an efficiency with 28.9 TOPS/s/W at 4-bit input and 4-bit weight. However, the structure of this design limits its application to some extent:

1) The 4-level read voltage $V_{RD}, 2/3V_{RD}, 1/3V_{RD}, 0$ at the input will vary memory resistance during the read operation, which will affect the accuracy of MAC operation.
2) The 2b sensing amplifier will greatly limit the total precision of the MAC output. Adding multiple outputs will average the quantization error and noise, but the increased precision is halved.
3) It needs multiple cycles to finish the Vector-Matrix operation, which will significantly reduce the computation speed.

To address the issues mentioned above and further improve the energy efficiency, S. Zhang *et al.* [12] proposed a Multiple Binary RRAM with Active Integrator (MBRAI) core architecture. As shown in Fig. 2, multiple binary-RRAM cells are used to represent an 8-bit weight instead of a multi-level RRAM cell. The core uses binary code at the input instead of a time signal 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 to achieve a wide dynamic range, which consume more than 95% power in the scheme. The computing efficiency of the CIM core is limited to 0.61 TOPS/s/W. To address this issue, Y Zhang *et al.* [13] proposed an 8-bit In Resistive Memory Computing Core with Regulated Passive Neuron and Bit Line Weight Mapping (RPN & BLM) scheme. RPN & BLM uses passive integral circuits without amplifiers to decrease power consumption. The regulators in the bit lines are used to improve the linearity of the integration process.

C. Differential Weight based CIM Cores

The uncertainty of the memory resistance in the MAC array with 2's complementary code may cause a big jump between the most negative value (i.e., 8'b10000000) and the most positive value (i.e., 8'b01111111). Differential weights [6], [25], [26] could be used to avoid this issue. Recently, P. Yao *et al.* [27] proposed a memristor-based hardware system with reliable multi-level conductance states for a five-layer mCNN for MNIST digit image recognition (MBHS-mCNN). As shown in Fig. 3, the neural processing unit consists of multiple memristor tiles and each tile contains four memristor cores.

Fig. 3. The architecture of the memristor-based neural processing unit and relevant circuit modules.

The MUX controller is used to select the positive and negative computing results. However, there are still some weaknesses in this scheme:

1) It requires 32 times of analog-to-digital conversions and Shift & add operations to finish one MAC operation, which consume about 92.14% energy in the system.
2) The quantization error is amplified by the shift and add operation, and thus it cannot achieve the desired precision.

III. PROPOSED IN-MEMORY COMPUTING CORE

In this paper, we propose a booth encoded differential core for low-power parallel MAC operations. M-RD4 and M-CSD algorithms are proposed at the input and weights respectively to reduce the power consumption of MAC operations. The overall structure of the proposed scheme is shown in Fig. 4, which consists of six components, including M-RD4 generator, differential RRAM array, regulator, integrator, controller, and differential ADC. To be simplified, only 8×8 crossbar memory cells are illustrated in Fig. 4. It can be extended to 8×N×N memory cells for the real application. Each memory cell is comprised of a 1R1T pair. The binary inputs are firstly converted to the stimulus of the CIM core by using an M-RD4 booth algorithm. The stimulus will turn on the transistor in 1R1T to generate the current to pass through RRAM cells and accumulated at the integrators to enable the massive parallel MAC computation. Regulator [13] is used before the integrator to minimize the voltage variation caused by the channel length modulation during the integration. Finally, the analog voltage at the neuron is converted to the digital signals using the charge

Fig. 4. The overall architecture of OUR proposed CIM core.

Binary Bits				Radix-4 Bit	M-RD4 Bit
t_{i+3}	t_{i+2}	t_{i+1}	t_i		z_j
0/1	0	0	0	0	0
0/1	0	0	1	1	1
0/1	0	1	0	1	1
0 / 1	0	1	1	2	2 / -2
0 / 1	1	0	0	-2	2 / -2
0/1	1	0	1	-1	-1
0/1	1	1	0	-1	-1
0/1	1	1	1	0	0

(a) Radix-4   (b) M-RD4

Fig. 5. The example of (a) radix-4 code, (b) M-RD4 code. The proposed M-RD4 code can effectively reduce the number of '1's in the input data to a minimum

redistribution differential SAR ADC. Only one 8-bit ADC is required by eight integrators for high density and low power. The details of each block will be introduced in the rest of this section.

A. Modified Radix-4 Booth Code

Unsigned fixed point data can be used as the CIM core input because there is no negative data after the ReLU activation function. The input data can be expressed as an n-bit unsigned fixed-pointed data X_k

$$X_k = 2^{n-1}x_{k,n-1} + ... + 2^i x_{k,i} + ... + 2^0 x_{k,0} \quad (1)$$

Radix-4 booth code [28] 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. The algorithm of recoding an n-bit binary number (X) to a radix-4 booth number (Z) is as follows. Firstly append a '0' to the right of the Least Significant Bit (LSB) of the X, and then extend the sign bit one position if necessary to ensure that n is even. After that, every three binary bits (with 1 bit overlap) are encoded as one radix-4 bit from the LSB to the Most Significant Bit (MSB).The eight cases of the radix-4 code are tabulated in Table I. By using the radix-4 algorithm, the length of the input code is halved (i.e. 01111111 is encoded to $200\bar{1}$). The number of '1's can also be reduced compared with binary codes, which means the power consumption can be reduced since more multiplications can be bypassed in the MAC calculations.

However, the radix-4 code sometimes leads to more '1's than that in binary codes. Fig.5(a) shows an example to encode a binary code '01010010' to the radix-4 code '$111\bar{2}$', where the number of '1's is increased in radix-4 code. To reduce the number of '1's in radix-4 code, we propose an M-RD4 code to get the least '1's at the input. The M-RD4 algorithm is illustrated in Algorithm 1 . The proposed M-RD4 algorithm

will observe one more bit at the left. If the sequence is '0100', it will be turned into '0011'. If the sequence is '1011', it will be turne into '1100'. After that, the right three bits will be encoded by using Eq (2).

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

where $i = 0, 2, 4, ..., j = \frac{i}{2}$, t_i is the i_{th} bit of T, T is defined in Algorithm 1. The cases are tabulated in Table I. The M-RD4 code can further reduce the number of '1's in input data. Fig. 5(b) is used as an example to illustrate our M-RD4 algorithm. The M-RD4 code of '01010010' is changed to '1102' instead of '$111\bar{2}$'.

Fig. 6 shows the M-RD4 booth recoding circuit implementation, which is composed of the MUX block, converter block and encoder block. The MUX block 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. In this way, the MUX block outputs the raw data for M-RD4 from the LSB to MSB. The converter block converts the raw data for encoding according to the M-RD4 algorithm. $a_{i+3}, a_{i+2}, a_{i+1}$ are the outputd of the MUX block, and a_i is generated by the converter. As shown in Fig.5(b), in the first clock, $a_i = 0$, and then a_i is determined by the output of the converter(t_{i+2}) in the last clock. Therefore, a_i is '0,0,0' in the next three clocks. According to Algorithm 1, we set

$$F = \overline{a_{i+3}}a_{i+2}\overline{a_{i+1}}a_i \quad (3)$$

$$G = a_{i+3}\overline{a_{i+2}}a_{i+1}a_i \quad (4)$$

Algorithm 1 M-RD4 Booth Code

Input: Binary n-bit $X = x_{n-1}x_{n-2}...x_i...x_0$.

Output: M-RD4 Booth Encoded m-bit data $Z = z_{m-1}z_{m-2}...z_j..z_0$, where $m = \lceil \frac{n}{2} \rceil$.

1: // Extend a '0' at the most left to ensure that n is even
2: // Append a '0' to the right of the Least Significant Bit (LSB)
3: **if** n is even **then**
4:     $T[n:0] \Leftarrow x_{n-1}x_{n-2}...x_i...x_0 0$
5: **else**
6:     $T[n+1:0] \Leftarrow 0x_{n-1}x_{n-2}...x_i...x_0 0$
7: **end if**
8: $i \Leftarrow 0, j \Leftarrow 0$
9: **while** $i \leq n - 2$ **do**
10:     // Observe one more bit per time and transfer the sequence if necessary.
11:     **if** $t_{i+3}t_{i+2}t_{i+1}t_0 == '0100'$ **then**
12:         $t_{i+3}t_{i+2}t_{i+1}t_0 \Leftarrow '0011'$
13:     **else if** $t_{i+3}t_{i+2}t_{i+1}t_0 == '1011'$ **then**
14:         $t_{i+3}t_{i+2}t_{i+1}t_0 \Leftarrow '1100'$
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

Fig. 6. The circuit implementation of the proposed M-RD4 booth recoding circuit. The MUX block selects the bits of the binary input, and the converter processes the input under the rules of the proposed M-RD4 algorithm. The encoder recodes the input to M-RD4 codes and outputs it into the neuron circuit.

then we can get the output of the converter block

$$t_{i+2} = G + \overline{F}a_{i+2} \tag{5}$$
$$t_{i+1} = F + \overline{G}a_{i+1} \tag{6}$$
$$t_i = F + \overline{G}a_i \tag{7}$$

The output of the converter is sent to the encoder block for recoding. The 3-bit binary codes are recoded to 1-bit M-RD4 code by combination circuit according to Table I. The encoder output log can be shown as

$$Z_2 = \overline{t_{i+2}}t_{i+1}t_i \tag{8}$$
$$Z_{-2} = t_{i+2}\overline{t_{i+1}}\overline{t_i} \tag{9}$$
$$Z_1 = \overline{t_{i+2}}(t_{i+1} \oplus t_i) \tag{10}$$
$$Z_{-1} = t_{i+2}(t_{i+1} \oplus t_i) \tag{11}$$

where Z_2, Z_{-2}, Z_1, and Z_{-1} represent four values of z_j (2, -2, 1, -1) in Table I. When z_j is encoded to zero, the multiplication result is always zero. Therefore, there are only four output terminals from the combination logic circuit, and only one of them will be activated at a time. If $z_j = 1$, then the voltage of Z_1 is high and the others are low, and the other cases can be speculated.

To make the M-RD4 code and its corresponding circuit clearer, we use the binary code '01010010' as an example. In the first clock, $S_A = 0, S_B = 1$, then $a_{i+3}a_{i+2}a_{i+1} = x_2x_1x_0(010)$, and $a_i = 0$. According to Eq (4), we can get $F = 1, G = 0$. The outputs $(t_{i+2}t_{i+1}t_i)$ of the converter are 011. The M-RD4 result is 2, thus $Z_2 = 1$, $Z_{-2} = 0$,

$Z_1 = 0$, and $Z_{-1} = 0$. In the second clock, $S_A = 0, S_B = 1$, then $a_{i+3}a_{i+2}a_{i+1}a_i = x_4x_3x_2Q(1000)$, where Q equals t_{i+2} at the last clock. $F = 0$, $G = 0$, then $t_{i+2}t_{i+1}t_i = 000$, therefore all of the outputs are 0 . In the third clock, $S_A = 1$, $S_B = 0$, then $a_{i+3}a_{i+2}a_{i+1}a_i = x_6x_5x_4Q(1010)$. $F = 0$, $G = 0$, then $t_{i+2}t_{i+1}t_i = 010$, therefore $Z_1 = 1$. In the fourth clock, $S_A = 1$, $S_B = 1$, then $a_{i+3} = gnd(0)$, and $a_{i+2}a_{i+1}a_i = x_7x_6Q(010)$. $F = 0$, $G = 0$, then $t_{i+2}t_{i+1}t_i = 010$. According to the third clock, $Z_1 = 1$. Therefore, the M-RD4 output is '1102'. The four output bits, which are either at VDD or ground, are directly used in the in-memory computing. The weights of 1, -1, 2, and -2 will be employed in the neuron circuit, which will be discussed in Section III. C.

B. Modified CSD Weights

2's complementary code representation is widely used in the arithmetic logic and operation. However, it may not be the best form to minimize the power consumption for the neural network computing. Fig. 7(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 inputs follow a half-normal distribution, because all negative values have been forced to be zero after the ReLU activation function. If the weights and inputs are qualified to 8-bit binary data, there are 40% - 50 % of '1's in the weights and about 20% - 30% of '1's in the inputs. If 2's complement is used, as shown in Fig. 7(b), the number of '1's and '0's will be balanced and the probability of 1×1 is about 10%, which is not optimized for low power computing. What's more, the 2's complementary may cause a big jump between the most negative value (10000000) and the most positive value (01111111) due to the uncertainty of the memory resistance. The leap will significantly influence the accuracy of in-memory computing.

Differential weights can be used to address the above mentioned issues, which can be represented as

$$w = w_p - w_n = 2^{n-1}(b_{n-1} - c_{n-1}) + ... + 2^0(b_0 - c_0) \tag{12}$$

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

Fig. 7. The simplified distribution curve of (a) weights, (b) weights in 2's complement, (c) differential weight.

Fig. 8. The data representation of our proposed differential weight system.

of a weight, respectively. For example, w_p = 8'b00000000 and w_n=8'b01110111 represent weight -119. As shown in Fig.7(c), the red line indicates a positive value, and the blue line indicates a negative value. The digits 1 and $\bar{1}$ 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 the power consumption by around 50%.

However, it doesn't fully utilize both parts of a differential weight. If we could represent W with fewer non-zero digits, we could reduce the in-memory computing power consumption. CSD representation [29] is widely used to reduce the non-zero digits by introducing a new digit $\bar{1}$ into the number to form a ternary number system. The pair b_i and c_i in Eq (12) can be used to represent the digit set {1,0,$\bar{1}$} for a CSD code. A simple approach to encode a binary code 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$\bar{1}$). The process may need to be repeated several times to make sure there is no string of '1's. CSD representation still suffers from some shortcomings:

1) In a CSD number, two consecutive non-zero 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$\bar{1}$) doesn't reduce the number of '1's.

An M-CSD representation is proposed to address the above issues. The strings '11' and '$\bar{1}\bar{1}$' are allowed in M-CSD. The main idea of M-CSD is shown in Algorithm 2. Strings containing three or more '1's will be replaced by 10...0$\bar{1}$ and three or more '$\bar{1}$'s will be replaced by $\bar{1}$0...01. If the MSB of the binary code is contained in a string, then the string will not be replaced with the M-CSD representation. In this way, the maximum value is extended to 219 (11011011). As shown in Fig. 8, 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. To comply with the CSD design rule, w_p and w_n in the above example will be changed to 8'b00001001 and 8'b10000000, respectively. Therefore, the number of '1's is significantly reduced.

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: $j \Leftarrow 0$
4: $k \Leftarrow 0$
5: //String containing MSB will not be replaced.
6: **while** i $>$0 & Flag == 0 **do**
7:    **if** w_i==0 **then**
8:       Flag $\Leftarrow$ 1
9:    **end if**
10:    $i \Leftarrow i - 1$
11: **end while**
12: // From LSB to w_i do the M-CSD.
13: **while** j $<$i **do**
14:    **if** $w_{j+4}...w_j$ == 11011 **then**
15:       $w_{j+2}w_{j+1}w_j \Leftarrow 10\bar{1}$
16:       $j \Leftarrow j + 2$
17:    **else if** $w_{j+4}...w_j$ == $\bar{1}\bar{1}0\bar{1}\bar{1}$ **then**
18:       $w_{j+2}w_{j+1}w_j \Leftarrow \bar{1}01$
19:       $j \Leftarrow j + 2$
20:    **else if** $w_{j+2}w_{j+1}w_j$ == 111 **then**
21:       $k \Leftarrow j + 2$
22:       **while** w_k == 1 **do**
23:          $k \Leftarrow k + 1$
24:       **end while**
25:       $w_kw_{k-1}...w_j \Leftarrow 10...\bar{1}$
26:       $j \Leftarrow k$
27:    **else if** $w_{j+2}w_{j+1}w_j$ == $\bar{1}\bar{1}\bar{1}$ **then**
28:       $k \Leftarrow j + 2$
29:       **while** w_k == $\bar{1}$ **do**
30:          $k \Leftarrow k + 1$
31:       **end while**
32:       $w_kw_{k-1}...w_j \Leftarrow \bar{1}0...1$
33:       $j \Leftarrow k$
34:    **else**
35:       $j \Leftarrow j + 1$
36:    **end if**
37: **end while**
38: **return** W

C. Neuron Circuit

The integral multiplier in the proposed CIM core is designed for massive parallel MAC operations and data transmission from digital to analog. [12] uses operational amplifiers to finish the integral operation. However, the static power consumption of the amplifier is not optimized for low power computing. Therefore, regulated passive neuron taken from [13] is adopted in our scheme to propose a differential passive integrator.

Fig. 9. The block diagram of the integration scheme in the integral multiplier. It implements digital input/weight and analog MAC operations, and completes the analog-to-digital conversion output by a SAR ADC.

As shown in Fig. 9, digital inputs and digital weights are differentially multiplied and accumulated at the neurons. The proposed integration scheme contains three phases: positive integration, negative integration, and charge redistribution. The integration phases are used to perform non-weighted MAC operations for inputs and weights. Therefore, each integrator has the same integral voltage for different input bits and weight bits. The charge redistribution phase is used to perform the weighting process for M-RD4 digits ($4^0, 4^1, ..., 4^{m-1}$ from LSB to MSB, where m is the length of M-RD4 code, and $m = \lceil \frac{n}{2} \rceil$).

The integral neuron is designed as a symmetrical structure complete the positive and negative MAC operations separately. The differential integrator is illustrated in Fig. 10(a). The M-RD4 inputs are sequentially sent to the word lines from LSB to MSB. The RRAM model used in the 1R1T cells is around 10 GΩ in High Resistance State (HRS) and 10 M Ω in Low Resistance State (LRS) [30], [31]. The 1R1T cells are used in pairs to store b_i and c_i mentioned in Eq (12). The positive circuit is used for MAC operations whose results are positive ($I_p \times W_p + I_n \times W_n$), while the negative circuit is used for MAC operations with negative results ($I_p \times W_n + I_n \times W_p$), 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. What's more, the positive and negative circuits are compensated to each other, effectively reducing the influence of parasitic parameters. Therefore, the 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_3, and S_4 control the integration phase and the charge redistribution phase. S_5 controls the sample phase and

the conversion phase of the ADC.

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 data ($IN_p = 1$, and$IN_n = -1$), 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 Cp_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 make sure the integrator are separated. S_3 and S_N are closed to clear the charge in negative capacitors. After that, S_1 is closed with the input $IN_p = -1$, and $IN_n = 1$. The phase complete 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 [13], the positive or negative integration voltage after one step of the charge redistribution phase is

$$V_S = V_S^- - k(2^{-1} \sum_{i=0}^{p-1} A_i G_{i,n-1} + 2^{-2} \sum_{i=0}^{p-1} A_i G_{i,n-2} + ...$$
$$+ 2^{-n+1} \sum_{i=0}^{p-1} A_i G_{i,0})$$

(13)

where V_S represents V_{Sp} or V_{Sn}, and V_S^- represents the initial integral voltage. $k = \frac{V_B T}{C_f}$, 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 time period for each integration. G_i is the conductance of each binary-RRAM cell, which is $1/R_H$ and $1/R_L$ when it is in the HRS and LRS, respectively.

(a)

(b)

(c)

Fig. 10. The proposed circuit schematics. (a) Integral multiplier with a symmetrical structure. The regulators are used to maintain the drain voltage of the 1R1T cells. All regulators have the same bias current. (b) The control logic of 1-bit input data. (c) The weighting process for the weight bit and the input data are completed simultaneously in the charge redistribution phase.

In the 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 $A_{i,m-1} = 4^1 A_{i,m-2} = ...4^{m-1} A_{i,0}$, which means each bit needs two steps of charge redistribution operation to achieve the weighting process for input data. As shown in Fig. 10(b), four integration phases (two positive and two negative) and two charge redistribution phases are needed to complete 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. 10(c), the first charge redistribution phase uses the sampling capacitor C_S to complete the weighting process for input data. Let $C_S = C_f$, the charge on the capacitors $C_{n-1}C_{n-2}...C_0$ and C_S is 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

$$V_{p,a} = \frac{1}{2}(V_{Sp,a} + V_p^-) \qquad (14)$$

where V_p^- represents the previous positive voltage in C_S, $V_{Sp,a}$ 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

$$V_p = \frac{1}{2}(V_{Sp,b} + V_{p,a}) = \frac{1}{2}V_{Sp,b} + \frac{1}{4}V_{Sp,a} + \frac{1}{4}V_p^- \qquad (15)$$

where $V_{Sp,b}$ is the positive integration voltage for layers with input '2' and '-2', V_p^- is the positive output voltage after the last input bit is computed. Eq (15) described the for loop process for each bit of the input data. Therefore the input data is weighted by $4^m - 1, 4^m - 2, ..., 4^0$ from LSB to MSB. Initially V_p is reset to Vdd. After m-bit input data are computed, it can be expressed as

$$V_p = 4^{-m}V_{dd} + 4^{-m}V_{Sp,0} + ... + 4^{-1}V_{Sp,m-1}, \qquad (16)$$

where $V_{Sp,i} = V_{Sp,a,i} + 2V_{Sp,b,i}$, the change of the V_p is

$$\Delta V_p = V_{dd} - V_p = 4^{-m}\sum_{i=0}^{m-1} 4^i \Delta V_{Sp,i} \qquad (17)$$

where $\Delta V_{Sp,i} = \Delta V_{Sp,a,i} + 2\Delta V_{Sp,b,i}$, $V_{Sp,i}$ 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

$$V_{out} = \Delta V_p - \Delta V_n = 4^{-m}\sum_{i=0}^{m-1} 4^i (\Delta V_{Sp,i} - \Delta V_{Sn,i}) \quad (18)$$

D. Mapping

There are several methods to implement the convolution layers and fully connected layers on cross-point arrays [32]–[34]. To estimate the network level energy efficiency of the proposed scheme, the mapping method in [12] is adopted. Both convolution kernel in convolution layers and weight matrix in fully connected layers are mapped into the cores. A convolution kernel whose size is $C_{in} * k * k * C_{out}$ is firstly transform it to a 2D matrix with size $(C_{in} * k * k) \times C_{out}$. The proposed scheme

has a cross-point array size of 256×512, and can implement a 256×256 matrix. Therefore, the number cores to implement the kernel is $\lceil \frac{C_{in}*k*k}{256} \rceil \times \lceil \frac{C_{out}}{256} \rceil$. The adders are integrated in the router unit to sum the results of different cores if the kernel size is larger than 256. For an $M * N$ fully connected layer, the weight matrix can be mapped into $\lceil \frac{M}{256} \rceil \times \lceil \frac{N}{256} \rceil$ cores, respectly.

IV. SIMULATION RESULTS

In this section, both circuit-level and network-level evaluation results are provided. The circuit-level simulation verifies the circuit's functionalities and shows the energy and accuracy benefits of the proposed core. The network-level evaluation presents the performance comparison with other related works. The circuit-level simulations are done in Cadence Analog Mixed Signal (AMS) with a 45nm generic Process Design Kit (PDK). The RRAM model proposed by [35] is adopted in the circuit simulations. The network-level simulations are done on the PyTorch platform.

A. Functionality

The transient simulation is performed to verify the correct function of the circuit. A random input 125 (binary representation: 8'b01111101, M-RD4 representation: 2, 0, -1, 1) is sent to the CIM core to complete the MAC operation with a random weight 123 (binary representation: 8'b01111011, differential representation: 8'b10000000-8'b00000101). Fig.11 (a) shows the whole MAC operations. The input bits is computed from LSB to MSB. From 0 ns to 130 ns, the circuit completes the MAC operation for the M-RD4 bit '1'. As shown in Fig. 11 (b), $V c_{p,7}$ is the integration voltage of the positive capacitor $C_{p,7}$, which is reset to 1 V when S_2 is closed. From 16 ns to 31 ns, S_P is closed and $V c_{p,7}$ is decreased to 745.4 mV linearly to complete the multiplication of 1×1. $V c_{n,0}$ is the integration voltage of the negative capacitor $C_{n,0}$, which is reset when S_3 is closed. The multiplication of 1×-1 is completed from 47 ns to 62 ns where $V c_{n,0}$ is decreased to 745.3 mV linearly. From 64 ns to 70 ns, S_4 is closed to complete the charge redistribution phase, and the output voltage V_{out} is 61.19 mV. From 66 ns to 124 ns, $V c_{p,7}$ and $V c_{n,0}$ are kept at 1 V since no data is input. After the second charge redistribution phase, the output voltage V_{out} is halved to 30.49 mV. The computing of the M-RD4 input '1' is completed. Using the difference as output can effectively reduce the impact of parasitic parameters on accuracy. After 8 cycles of integration and charge redistribution phase, the output voltage V_{out} is 59.73 mV. The digital result is 8'b00111011. The theoretical results are 59.89 mV and 8'b00111011, respectively. Therefore, the proposed scheme achieves its design requirement.

B. Robustness Analysis

Fig. 12 shows the relationship between analog output V_{out} ($\Delta V_p - \Delta V_n$) and (a) the digital input, (b) the number of input lines, and (c) the digital weight. The results show that the proposed scheme achieves high linearity and accuracy. Fig. 13 (a), (b), and (c) show the Differential Non-linearity (DNL) of

Fig. 11. The transient simulation results of (a) the computing progress for 1-bit M-RD4 input, and (b) the whole MAC operation for M-RD4 input '2,0,-1,1' and differential weight 8'b10000000-8'b00000101.

is reliable with different variations of the process, voltage, and temperature.

TABLE III
CIM CORE PERFORMANCE COMPARISON BETWEEN MBRAI, RPN&BLM AND THE PROPOSED

	MBRAI [12]	RPN&BLM [13]	Proposed	
Supply	1.1 V	1 V	M-RD4 Recoder	0.6 V
			Neuron Circuit	1 V
Computing speed	1.85 M/s	1.85 M/s	1.85 M/s	
SFDR	67.42 dB	59.13 dB	63.41 dB	
SNDR	45.48 dB	46.13 dB	46.48 dB	
ENOB	7.26 bit	7.37 bit	7.42 bit	

TABLE IV
ENERGY COST COMPARISON BETWEEN THE PROPOSED CORE AND OTHERS

		MBRAI [12]	MBHS-mCNN [27]	RPN&BLM [13]	Proposed
Technology		45 nm	65 nm	45 nm	45 nm
Supply		1.1 V	-	1 V	0.6/1 V
System Frequency		16.7 MHz	20 MHz	16.7 MHz	16.7 MHz
Core Size		256*256	128*256	256*256	256*512
Power	Amplifier	0.22 mW	-	-	-
	ADC	4.04 uW	25.47 uW	4.04 uW	3.99 uW
	Regulator	-	-	1.11 uW	0.55 uW
	Core	199.68 mW	7.44 mW	3.61 mW	2.00 mW

C. Performance

Table III shows the dynamic performance comparison between the MBRAI [12], RPN&BLM [13] and the proposed scheme. The M-RD4 recoder has a supply voltage of 0.6 V to further decreases the power consumption. The neuron circuit's supply voltage is 1 V to ensure the robustness of our proposed scheme. The computing speed, SFDR, SNDR, Effective Number of Bits (ENOB) of our proposed scheme are 1.85 M/s, 63.41 dB, 46.48 dB, and 7.42 bit, which are slightly better than the others. Table IV gives the energy cost comparison of MBRAI, MBHS-mCNN [27], RPN&BLM, and our proposed scheme. MBRAI consumes 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% energy in MBHS-mCNN, while the power consumption for 128×256 core is 7.44 mW. RPN&BLM uses regulators, with 1.11 uW 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 MBRAI, MBHS-mCNN, and RPN&BLM, 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 other CIM core schemes is shown in Table V. The simulation results show that our proposed design achieves energy efficiency as high as 60.68 TOPS/s/W in 8-bit input 8-bit weight pattern, 371.49 TOPS/s/W in 4-bit input 4-bit weight pattern, 941.55 TOPS/s/W in 3-bit input 2-bit weight pattern, 1418.44 TOPS/s/W in 2-bit input 2-bit weight pattern, and 1325.22 TOPS/s/W in 3-bit input 1-bit weight pattern.

the proposed scheme with different (a) input value, (b) input lines, and (c) weight value, respectively. Fig. 13 (d),(e), and (f) show the Integration Non-Linearity (INL), respectively. The simulated DNLs (INLs) in terms of the digital input, the digital weight, and the number of input lines are +0.464/-0.073 LSB (-0.047/-0.809 LSB), +0.055/-0.291 LSB (+1.772/-1.061 LSB), and +0.111/-0.445 LSB (0.205/-0.673 LSB).

TABLE II
PVT SIMULATION ON ENOB

Process	ff		ss		tt
Temprature (°C)	-40	80	-40	80	27
Voltage (V)	1				7.42
	0.9	7.41	7.28	7.29	7.21
	1.1	7.11	7.16	7.25	7.11

Different process, voltage and temperature are chosen to do the PVT simulation to verify the robustness of the circuit. ENOBs, as shown in Table II, are all greater than 7.1 bits in different PVT combinations. Therefore, the proposed scheme

Fig. 12. The output voltage V_{out} at various (a)input data and RRAM weights, (b) input lines and input data, (c) RRAM weights and input lines in the proposed scheme

Fig. 13. The simulated DNL in terms of (a)input value, (b)input lines, (c) weight value and INL in terms of (d)input value, (e)input lines, (f)weight value in the proposed scheme

Compared with the other schemes, our proposed scheme achieves much higher efficiency. In the 8-bit input 8-bit weight pattern, our proposed scheme achieves an efficiency which is 99.47 $\times$, 5.44 $\times$, and 1.80 $\times$ more efficient than MBRAI, MBHS-mCNN, and RPN&BLM schemes. Compared with other CIM schemes, our proposed CIM core achieves better energy efficiency.

D. Network-Level Estimation

To estimate the accuracy and energy estimate of our proposed scheme, the model of LeNet [40] on the dataset MNIST and the models of AlexNet [41], ResNet34 [42] and VGG16 [43] on ILSVRC2012 are evaluated with the mapping method mentioned in section III.D. The estimated accuracy is shown in Table VI. Our proposed scheme achieves an accuracy better than MBHS-mCNN in LeNet, and roughly equivalent to MBRAI and RPN&BLM in LeNet and AlexNet. The energy estimation between the proposed scheme and other RRAM based schemes is shown in Table VII. The model of LeNet on the dataset MNIST is used to test the performance of the schemes in small-scale networks. The models of AlexNet, ResNet34 and VGG16 on ILSVRC2012 are used to evaluate the performance in large-scale networks. Our proposed scheme reduces the ratio of 1×1 by 78.5% on LeNet, 80.2% on AlexNet, 70.4% on ResNet34 and 82.9% on VGG16. Therefore, the power consumption is greatly reduced. The inference energy per image is reduced by 98.9% compared with MBRAI, more than 81.5% compared with MBHS-mCNN, and more than 43.6% compared with

TABLE V
CORE-LEVEL COMPARISON BETWEEN THE PROPOSED CORE AND OTHERS

Structure	Technology	Crossbar-size	Weight/data bit	Throughput (GOPS)	Power (mW)	Efficiency (TOPS/s/W)
SINWP [36]	55 nm	256*512	fixed-3/fixed-1	-	-	53.17
			fixed-3/fixed-2	-	-	21.9
MBRAI [12]	45 nm	256*256	fixed-3/fixed-1	1524	19.6	77.76
			fixed-3/fixed-2	1040	26.8	38.8
			fixed-8/fixed-8	121.4	199.68	0.61
MBHS-mCNN [27]	65 nm	128*256	fixed-8/fixed-8	81.82	7.348	11.15
7nm SRAM Macro [37]	7 nm	4 K	fixed-4/fixed-4	186.2	1.06	175.5
RPN & BLM [13]	45 nm	256*256	fixed-2/fixed-2	1092.2	1.975	553.01
			fixed-4/fixed-4	546.1	2.66	205.30
			fixed-8/fixed-8	121.4	3.61	33.63
Synapses Integrated Analog Processor [38]	180 nm	2 M	analog	0.33	15.8	20.7
	40 nm	4 M	analog	0.66	9.9	66.5
Fully Integrated Analog Chip [39]	130 nm	4 K	fixed-1/tenary	-	-	78.4
Proposed	45 nm	256*512	fixed-3/fixed-1	1524	1.15	1325.22
			fixed-2/fixed-2	1092.2	0.77	1418.44
			fixed-3/fixed-2	1092.2	1.16	941.55
			fixed-4/fixed-4	546.1	1.47	371.49
			fixed-8/fixed-8	121.4	2.00	60.68

TABLE VI
ACCURACY ESTIMATE OF DIFFERENT RRAM-BASED SCHEMES

Network	Structure	Top-1 Error Rate
LeNet on MNIST	Software Based	0.90 %
	MBRAI	0.97 %
	MBHS-mCNN	2.44 %
	RPN&BLM	0.90 %
	Proposed	**0.91 %**
AlexNet on ILSVRC12	Software Based	42.70 %
	MBRAI	44.16 %
	RPN&BLM	43.60 %
	Proposed	**43.10 %**
ResNet34 on ILSVRC12	Software Based	26.70 %
	Proposed	**27.80 %**
VGG16 on ILSVRC12	Software Based	28.40 %
	Proposed	**29.30 %**

RPN&BLM on different nerworks. Therefore, the inference energy is significantly reduced in our proposed scheme by abandoning the amplifiers and adopting M-RD4 and M-CSD codes.

Fig. 14. The energy cost comparison between different code combination.

As shown in Fig. 14, the energy cost and the ratio of 1×1 of different codes are simulated to verify the superiority of out proposed M-RD4 and M-CSD. The core size is set to be 256*512, and the ratio of 1×1 is obtained on LeNet with the dataset MNIST. Take RPL&BLM as the standard, the ratio of 1×1 with binary input and binary weight is 14.7% and the

power consumption is 3.61 mW. The ratio of $1\times$ decreases to 13.3% by using radix-4 input. What's more, the ratio of 1×1 is further decreased to 11.2% by using our M-RD4 in the input, and the power consumption is decreased by 26.46%, respectively. Applying the M-RD4 input and CSD weight, the ratio of 1×1 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 1×1 to 2.2% and the power consumption to 2.00 mW. Therefore, for a 256*512 core, our proposed scheme saves 41.55% of power consumption compared with RPN& BLM.

V. CONCLUSION

In this paper, a low power in-memory multiplication and accumulation array with modified radix-4 input and canonical-signed-digit weights has been proposed. Modified radix-4 booth code is used to reduce the number of '1's in the input data, and differential memory pairs with modified canonical-signed-digit are used to reduce the '1's in weight. The proposed two coding schemes efficiently reduce the ratio of 1×1 by 85.0% on LeNet, 79.7% on AlexNet, 70.4% on ResNet34 and 82.9% on VGG16. The simulation results has shown that our proposed CIM core achieves 2.00 mW on power consumption with 256*512 in 8-bit input and 8-bit weight pattern. The computing-power rate at the fixed-point 8-bit is 60.68 TOPS/s/W, which is $99.47\times$, $5.44\times$, and $1.80\times$ than that of MBRAI, MBHS-mCNN and RPN&BLM schemes, respectively. The core is very robust with an ENOB of 7.42-bit whose SFDR and SNDR achieve 63.41 dB and 46.48 dB. The network-level estimation has shown that the proposed core achieves 0.91% top-1 error rate with 7.59E-3 uJ/img on LeNet, 43.60% top-1 error rate with 13.36 uJ/img on AlexNet, 27.80% top-1 error rate with 77.79 uJ/img on ResNet34, and 29.30% top-1 error rate with 297.88 uJ/img on VGG16, respectively. The core achieves very low inference energy cost and high accuracy, which are much better than other schemes. The linearity and PVT simulation has been done to verify the robustness of the circuit. The energy efficiency comparison has shown that the proposed scheme achieves much lower power consumption than others.

TABLE VII
ENERGY ESTIMATE OF DIFFERENT RRAM-BASED SCHEMES

Network	Number of Operations	Structure	Ratio of 1×1	System Frequency	Data Bit	Crossbar Size	Energy (uJ/img)	Saving %
LeNet on MNIST	0.42 M	MBRAI [12]	0.147	25 MHz	8	256*256	0.71	**98.9 %**
		MBHS-mCNN [27]		25 MHz	8	128*256	0.039	**81.6 %**
		RPN & BLM [13]		16.7 MHz	8	256*256	0.013	**44.6 %**
		Proposed	0.022	16.7 MHz	8	256*512	**7.19E-3**	-
AlexNet on ILSVRC2012	720 M	MBRAI [12]	0.143	25 MHz	8	256*256	1.23E+03	**98.9 %**
		MBHS-mCNN [27]		25 MHz	8	128*256	68.56	**81.5 %**
		RPN & BLM [13]		16.7 MHz	8	256*256	22.46	**43.6 %**
		Proposed	0.029	16.7 MHz	8	256*512	**12.66**	-
ResNet34 on ILSVRC2012	4 G	MBRAI [12]	0.125	25 MHz	8	256*256	6.92E+03	**98.9 %**
		MBHS-mCNN [27]		25 MHz	8	128*256	390.07	**81.1 %**
		RPN & BLM [13]		16.7 MHz	8	256*256	141.95	**48.1 %**
		Proposed	0.037	16.7 MHz	8	256*512	**73.73**	-
VGG16 on ILSVRC2012	16 G	MBRAI [12]	0.129	25 MHz	8	256*256	2.77E+04	**98.9 %**
		MBHS-mCNN [27]		25 MHz	8	128*256	1.56E+03	**81.9 %**
		RPN & BLM [13]		16.7 MHz	8	256*256	567.8	**50.3 %**
		Proposed	0.022	16.7 MHz	8	256*512	**282.35**	-

REFERENCES

[1] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, pp. 436–444, 2015.

[2] W. Liu, Z. Wang, X. Liu, N. Zeng, Y. Liu, and F. E. Alsaadi, "A survey of deep neural network architectures and their applications," *Neurocomputing*, vol. 234, pp. 11–26, 2017.

[3] G. W. Burr, R. M. Shelby, A. Sebastian, S. Kim, S. Kim, S. Sidler, K. Virwani, M. Ishii, P. Narayanan, A. Fumarola *et al.*, "Neuromorphic computing using non-volatile memory," *Advances in Physics: X*, vol. 2, no. 1, pp. 89–124, 2017.

[4] Q. Xia and J. J. Yang, "Memristive crossbar arrays for brain-inspired computing," *Nature materials*, vol. 18, no. 4, pp. 309–323, 2019.

[5] K. Huang, Y. Ha, R. Zhao, A. Kuma, and Y. Lian, "A low active leakage and high reliability phase change memory (pcm) based non-volatile fpga storage element," *Circuits and Systems I: Regular Papers, IEEE Transactions on*, vol. 61, no. 9, pp. 2605–2613, 2014.

[6] P. Chi, S. Li, C. Xu, T. Zhang, J. Zhao, Y. Liu, Y. Wang, and Y. Xie, "Prime: A novel processing-in-memory architecture for neural network computation in reram-based main memory," in *2016 ACM/IEEE 43rd Annual International Symposium on Computer Architecture (ISCA)*, 2016, pp. 27–39.

[7] Y. Pan, P. Ouyang, Y. Zhao, W. Kang, S. Yin, Y. Zhang, W. Zhao, and S. Wei, "A mlc stt-mram based computing in-memory architecture for binary neural network." in *2018 IEEE International Magnetics Conference (INTERMAG)*. IEEE, 2018, pp. 1–1.

[8] A. Irmanova and A. P. James, "Multi-level memristive memory with resistive networks," in *2017 IEEE Asia Pacific Conference on Postgraduate Research in Microelectronics and Electronics (PrimeAsia)*, 2017, pp. 69–72.

[9] M. Hu, C. E. Graves, C. Li, Y. Li, N. Ge, E. Montgomery, N. Davila, H. Jiang, R. S. Williams, J. J. Yang *et al.*, "Memristor-based analog computation and neural network classification with a dot product engine," *Advanced Materials*, vol. 30, no. 9, p. 1705914, 2018.

[10] E. Giacomin, T. Greenberg-Toledo, S. Kvatinsky, and P. Gaillardon, "A robust digital rram-based convolutional block for low-power image processing and learning applications," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 66, no. 2, pp. 643–654, 2019.

[11] A. Chen and M.-R. Lin, "Variability of resistive switching memories and its impact on crossbar array performance," in *2011 International Reliability Physics Symposium*. IEEE, 2011, pp. MY–7.

[12] S. Zhang, K. Huang, and H. Shen, "A robust 8-bit non-volatile computing-in-memory core for low-power parallel mac operations," *IEEE Transactions on Circuits and Systems I: Regular Papers*, 2020.

[13] Y. Zhang, K. Huang, R. Xiao, and H. Shen, "An 8-bit in resistive memory computing core withregulated passive neuron and bit line weight mapping," *arXiv preprint arXiv-2008.11669*, 2020.

[14] P. A. Merolla, J. V. Arthur, R. Alvarez-Icaza, A. S. Cassidy, J. Sawada, F. Akopyan, B. L. Jackson, N. Imam, C. Guo, Y. Nakamura *et al.*, "A million spiking-neuron integrated circuit with a scalable communication network and interface," *Science*, vol. 345, no. 6197, pp. 668–673, 2014.

[15] M. Davies, N. Srinivasa, T.-H. Lin, G. Chinya, Y. Cao, S. H. Choday, G. Dimou, P. Joshi, N. Imam, S. Jain *et al.*, "Loihi: A neuromorphic manycore processor with on-chip learning," *IEEE Micro*, vol. 38, no. 1, pp. 82–99, 2018.

[16] X. Zhang, A. Huang, Q. Hu, Z. Xiao, and P. K. Chu, "Neuromorphic computing with memristor crossbar," *physica status solidi (a)*, vol. 215, no. 13, p. 1700875, 2018.

[17] G. W. Burr, R. M. Shelby, S. Sidler, C. Di Nolfo, J. Jang, I. Boybat, R. S. Shenoy, P. Narayanan, K. Virwani, E. U. Giacometti *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 Transactions on Electron Devices*, vol. 62, no. 11, pp. 3498–3507, 2015.

[18] P. M. Sheridan, F. Cai, C. Du, W. Ma, Z. Zhang, and W. D. Lu, "Sparse coding with memristor networks," *Nature nanotechnology*, vol. 12, no. 8, p. 784, 2017.

[19] Y. Jiang, P. Huang, D. Zhu, Z. Zhou, R. Han, L. Liu, X. Liu, and J. Kang, "Design and hardware implementation of neuromorphic systems with rram synapses and threshold-controlled neurons for pattern recognition," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 65, no. 9, pp. 2726–2738, 2018.

[20] A. Shafiee, A. Nag, N. Muralimanohar, R. Balasubramanian, J. P. Strachan, M. Hu, R. S. Williams, and V. Srikumar, "Isaac: A convolutional neural network accelerator with in-situ analog arithmetic in crossbars," in *2016 ACM/IEEE 43rd Annual International Symposium on Computer Architecture (ISCA)*, 2016, pp. 14–26.

[21] X. Qiao, X. Cao, H. Yang, L. Song, and H. Li, "Atomlayer: A universal reram-based cnn accelerator with atomic layer computation," in *2018 55th ACM/ESDA/IEEE Design Automation Conference (DAC)*, 2018, pp. 1–6.

[22] M. Courbariaux, I. Hubara, D. Soudry, R. El-Yaniv, and Y. Bengio, "Binarized neural networks: Training deep neural networks with weights and activations constrained to+ 1 or-1," *arXiv preprint arXiv:1602.02830*, 2016.

[23] M. Rastegari, V. Ordonez, J. Redmon, and A. Farhadi, "Xnor-net: Imagenet classification using binary convolutional neural networks," in *European conference on computer vision*. Springer, 2016, pp. 525–542.

[24] C.-X. Xue, T.-Y. Huang, J.-S. Liu, T. Chang, H.-Y. Kao, J. Wang, T. Liu, S.-Y. Wei, S.-P. Huang, W.-C. Wei *et al.*, "15.4 a 22nm 2mb reram compute-in-memory macro with 121-28tops/w for multibit mac computing for tiny ai edge devices," *2020 IEEE International Solid-State Circuits Conference - (ISSCC)*, pp. 244–246, 2020.

[25] M. V. Nair, L. K. Muller, and G. Indiveri, "A differential memristive synapse circuit for on-line learning in neuromorphic computing systems," *Nano Futures*, vol. 1, no. 3, p. 035003, 2017.

[26] V. Joshi, M. Le Gallo, S. Haefeli, I. Boybat, S. R. Nandakumar, C. Piveteau, M. Dazzi, B. Rajendran, A. Sebastian, and E. Eleftheriou, "Accurate deep neural network inference using computational phase-change memory," *Nature Communications*, vol. 11, no. 1, p. 2473, May 2020.

[27] P. Yao, H. Wu, B. Gao, J. Tang, Q. Zhang, W. Zhang, J. J. Yang, and H. Qian, "Fully hardware-implemented memristor convolutional neural network," *Nature*, vol. 577, no. 7792, pp. 641–646, 2020.

[28] A. D. Booth, "A signed binary multiplication technique," *The Quarterly Journal of Mechanics and Applied Mathematics*, vol. 4, no. 2, pp. 236–240, 1951.

[29] A. Avizienis, "Signed-digit numbe representations for fast parallel arithmetic," *IRE Transactions on Electronic Computers*, vol. EC-10, no. 3, pp. 389–400, 1961.

[30] T. Ahmad, W. Devulder, K. Opsomer, M. Minjauw, U. Celano, T. Hantschel, W. Vandervorst, L. Goux, G. S. Kar, and C. Detavernier, "Influence of the chalcogen element on the filament stability in cuin(te,se,s)2/al2o3 filamentary switching devices," *ACS Applied Materials & Interfaces*, vol. 10, no. 17, pp. 14 835–14 842, 2018, pMID: 29652471. [Online]. Available: https://doi.org/10.1021/acsami.7b18228

[31] Y.-J. Huang and S.-C. Lee, "Graphene/h-bn heterostructures for vertical architecture of rram design," *Scientific Reports*, vol. 7, no. 1, p. 9679, Aug 2017. [Online]. Available: https://doi.org/10.1038/s41598-017-08939-2

[32] L. Gao, P. Chen, and S. Yu, "Demonstration of convolution kernel operation on resistive cross-point array," *IEEE Electron Device Letters*, vol. 37, no. 7, pp. 870–873, 2016.

[33] X. Wang, Q. Wang, F.-H. Meng, S. H. Lee, and W. D. Lu, "Deep neural network mapping and performance analysis on tiled rram architecture," in *2020 2nd IEEE International Conference on Artificial Intelligence Circuits and Systems (AICAS)*. IEEE, 2020, pp. 141–144.

[34] F. Cai, J. M. Correll, S. H. Lee, Y. Lim, V. Bothra, Z. Zhang, M. P. Flynn, and W. D. Lu, "A fully integrated reprogrammable memristor–cmos system for efficient multiply–accumulate operations," *Nature Electronics*, vol. 2, no. 7, pp. 290–299, 2019.

[35] Z. Jiang, Y. Wu, S. Yu, L. Yang, K. Song, Z. Karim, and H.-S. P. Wong, "A compact model for metal–oxide resistive random access memory with experiment verification," *IEEE Transactions on Electron Devices*, vol. 63, no. 5, pp. 1884–1892, 2016.

[36] C. Xue, W. Chen, J. Liu, J. Li, W. Lin, W. Lin, J. Wang, W. Wei, T. Chang, T. Chang *et al.*, "24.1 a 1mb multibit reram computing-in-memory macro with 14.6ns parallel mac computing time for cnn based ai edge processors," in *2019 IEEE International Solid- State Circuits Conference - (ISSCC)*, 2019, pp. 388–390.

[37] Q. Dong, M. E. Sinangil, B. Erbagci, D. Sun, W. Khwa, H. Liao, Y. Wang, and J. Chang, "15.3 a 351tops/w and 372.4gops compute-in-memory sram macro in 7nm finfet cmos for machine-learning applications," in *2020 IEEE International Solid- State Circuits Conference - (ISSCC)*, 2020, pp. 242–244.

[38] R. Mochida, K. Kouno, Y. Hayata, M. Nakayama, T. Ono, H. Suwa, R. Yasuhara, K. Katayama, T. Mikawa, and Y. Gohou, "A 4m synapses integrated analog reram based 66.5 tops/w neural-network processor with cell current controlled writing and flexible network architecture," in *2018 IEEE Symposium on VLSI Technology*, 2018, pp. 175–176.

[39] Q. Liu, B. Gao, P. Yao, D. Wu, J. Chen, Y. Pang, W. Zhang, Y. Liao, C. Xue, W. Chen, J. Tang, Y. Wang, M. Chang, H. Qian, and H. Wu, "33.2 a fully integrated analog reram based 78.4tops/w compute-in-memory chip with fully parallel mac computing," in *2020 IEEE International Solid- State Circuits Conference - (ISSCC)*, 2020, pp. 500–502.

[40] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[41] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," *Communications of the ACM*, vol. 60, no. 6, pp. 84–90, 2017.

[42] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[43] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 2014.

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), since 2016. Prior to joining ZJU, he has spent five years at the IC design industry, including Samsung and Xilinx, two years in the Data Storage Institute, Agency for Science Technology and Research (A*STAR), and another three years in the Singapore University of Technology and Design (SUTD), Singapore. He has authored or coauthored 40 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 low power circuits and systems design using emerging non-volatile memories, architecture and circuit optimization for reconfigurable computing systems and neuromorphic systems, machine learning, and deep learning chip design. He currently serves as the Associate Editor of the IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS-PART II: EXPRESS BRIEFS.

Yewei Zhang (Student Member, IEEE) recieved the bachelor's degree from College of Information Science & Electronic Engineering, Zhe Jiang University in 2018. He is currently studying for a master's degree at College of Information Science & Electronic Engineering, Zhe Jiang University. He is interested in in-memory computing and non-volatile memories.

Haibin Shen is currently a Professor with Zhejiang University, 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. His research interests include learning algorithm, processor architecture, and modeling. His research achievement has been used by many authority organizations. He has published more than 100 papers on academic journals, and he has been granted more than 30 patents of invention. He 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.

Rui Xiao (Student Member, IEEE) received the Bechalor degree from the College of Information Science Electronic Engineering, Zhejiang University in 2019. Currently she is pursuing the Ph.D degree in the School of Information Science and Electronic Engineering, Zhejiang University under the supervision of Prof. Huang. Her research interests include in-memory computing circuits and systems design using emerging resistive non-volatile memories, deep learning accelerators, and embedded system design.