Improving the Efficiency of In-Memory-Computing Macro with a Hybrid Analog-Digital Computing Mode for Lossless Neural Network Inference

Qilin Zheng[1], Ziru Li[1], Jonathan Ku[1], Yitu Wang[1], Brady Taylor[1],
Deliang Fan[2], Yiran Chen[1],
[1]Duke University, [2]Johns Hopkins University
qilin.zheng@duke.edu

ABSTRACT

Analog in-memory-computing (IMC) is an attractive technique with a higher energy efficiency to process machine learning workloads. However, the analog computing scheme suffers from large interface circuit overhead. In this work, we propose a macro with a hybrid analog-digital mode computation to reduce the precision requirement of the interface circuit. Considering the distribution of the multiplication and accumulation (MAC) value, we propose a nonlinear transfer function of the computing circuits by only accurately computing low MAC value in the analog domain with a digital mode to deal with the high MAC value with smaller possibility. Silicon measurement results show that the proposed macro could achieve 160 $GOPS/mm^2$ area efficiency and 25.5 TOPS/W for 8b/8b matrix computation. The architectural-level evaluation for real workloads shows that the proposed macro can achieve up to 2.92× higher energy efficiency than conventional analog IMC designs.

KEYWORDS

Processing-in-Memory, SRAM, Machine Learning Acceleration

1 INTRODUCTION

In-memory-computing (IMC) technology has attracted a lot of attention in recent years as a solution for low power neural network processing [1–9]. In particular, static random access memory (SRAM) based analog in-memory computing schemes are showing great potential to improve the energy efficiency by moving the key operation, multiplication and accumulation (MAC) into analog domain [10? –15]. Typically, the IMC macro supports vector-matrix multiplication (VMM) as the computation primitive. The elements of the matrix are stored in the memory array as logical values, and two or more transistors are added to the memory cell to convert the logical values to the analog currents in the SRAM array. Then, each element of the vector is converted into the voltage generated by the wordline driver and applied to the corresponding wordline of the array. The accumulation result can be represented as the accumulated current on the bitlines or other customized accumulation lines. The current is then converted to digital domain by an analog-to-digital converter (ADC) for the subsequent process. Since multiple wordlines are activated in parallel, the normalized throughput and energy efficiency can be extremely improved comparing with conventional digital MAC unit. However, the high efficiency of the analog IMC computing scheme sacrifices the computation accuracy, due to non-ideal effect of the analog computing units and interface ADCs. Consequently, existing analog IMC macros are usually used for highly customized neural networks [16, 17], such as binary, ternary neural networks.

To integrate analog IMC into general purpose NN inference accelerators, one key design metric is to support lossless MAC operations. To achieve this specific requirement, the analog accumulation process should be robust to PVT variations and enough margin should be provided for successive sensing, and an ADC with enough effective number of bits (ENOB) is required to accurately generate the accumulation results. Prior efforts to improve the robustness include investigating the usage of advanced analog computing schemes such as capacitive coupling [18]. However, to implement the capacitive coupling scheme, one or more capacitors are required to be added to each memory cell, and multiple transistors are required to implement the switches. For example, [18] uses 10T-1C SRAM cell to implement the in-memory charge sharing scheme. The computation accuracy is improved at the expense of memory density. In addition, the advanced analog computing scheme scheme only resolves the robustness issue, while the ENOB requirement of the ADC can not be eliminated. There will still be considerable energy cost of the interface circuits to convert the analog value into digital domain.

An alternative approach to solve the problem is to design a fully-digital IMC macro [3, 19, 20]. The basic digital IMC implementations is based on local AND computation and a in-memory adder trees. The local AND is implemented by a NOR gate with two reversed inputs to improve the area efficiency. A 4T-NOR gate is added to each memory cell, and the output of the NOR gate is connected to a local adder tree, which counts the bits "1" of the 4T-NOR gate and generate a bit-wise accumulation results. The results are sent to the same local PE for successive post processing, which is same as the analog IMC macro. Since all the computation is in the digital domain, the proposed IMC scheme is robust to PVT variations with arbitrary bitwidth. However, this specific scheme has large hardware overhead due to additional digital circuits, where the local adder tree consumes about 10× area compared to the memory cell, so the memory density will be extremely degraded. In summary, to achieve higher energy and area efficiency of analog IMC macros,

DAC '24, June 23–27, 2024, San Francisco, CA, USA

ACM ISBN 979-8-4007-0601-1/24/06.
https://doi.org/10.1145/3649329.3658472

we need to find a solution that can improve the sensing margin and reduce the ENOB requirements of the ADC at the same time.

In this paper, we propose an alternative solution to the lossless SRAM-based analog IMC macro, instead of focusing on pure circuit level innovation to build up robust computing schemes. The basic idea is to leverage the sparsity in typical NN workloads, where the computation results will mainly be small values. Simulation results show that, when 64 rows are activated in parallel, 90% accumulation results are within 8 instead of following a uniform distribution. Different from the conventional analog IMC design methodology where all accumulation results in the analog domain are treated equally, we propose an analog computing scheme based on a nonlinear transfer function which only covers an accurate computation for the low MAC value region. Significant energy efficiency improvement can be observed as the ENOB requirement for the interface ADC can be reduced. Our contributions can be summarized as follows:

- We propose a partial sum distribution aware computation scheme with a nonlinear transfer function to improve the sensing margin by 7× as well as a 3bit ENOB reduction for the interface ADC.
- We further propose a hybrid digital/analog computation scheme with an additional digital data path to provide lossless computation capability with only 13% area overhead.
- We verify our ideas on a fabricated silicon in 65nm technology. The result show that the energy efficiency can be improved by 2.92× higher than conventional current domain IMC macros.

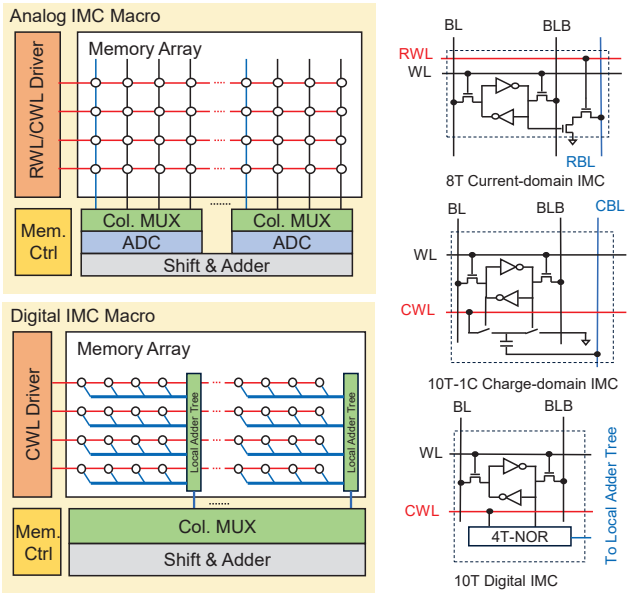

Figure 1: Illustration of analog IMC macro, digital IMC macro and related cell design.

2 BACKGROUND

2.1 Analog IMC

Analog IMC is characterized by the use of specific memory technology to perform MAC operations. It leverages the values stored in memory to directly modulate analog input signals into weighted analog output signals. A typical analog IMC macro contains a wordline driver, one or multiple memory arrays, readout circuitry (including column multiplexers (Col. MUXs) and ADC), and a shift adder for multi-bit accumulation. Depending on the computing scheme, the design methodology of the interface circuits and memory cells may vary slightly. Here, we review two representative approaches to implement analog domain computation.

2.1.1 Current Domain. One representative approach to implement the analog computation is based on 8T memory cell [10]. The logical value is stored in the conventional 6T cells and two additional transistor is used as a current source. We first present the basic computation scheme where the input and weights are both 1 bit. Before the computation is performed, each RBL is first pre-charged to the supply voltage V_{DD}. When the computation starts, the input vector is encoded as a binary value and a pulse will be applied on the read wordline (RWL) in the macro. When the logical value stored in the memory array is 1 and the RWL is driven to high voltage for a short period (T_0), 2 serial-connected NMOS will form a current source to discharge the RBL (assume the current is I_{DS}). The final voltage at RBL depends on the number of activated 2T current-sources which represents the bit-wise multiplication and accumulation results. The voltage at RBL will then be sampled and held on a column capacitor (C_{RBL}), and then converted by ADCs located in each column. The relationship between voltage at RBL and desired output (transfer function between analog output and input vector) can be written as follows:

$$V_{RBL} = V_{DD} - \int_0^{T_0} \frac{\sum W_i X_i * I_{DS} * dt}{C_{RBL}} \tag{1}$$

If V_{RBL} is large enough to keep I_{DS} as a constant, the equation can be reformulated as follows:

$$V_{RBL} = V_{DD} - \frac{\sum W_i X_i * I_{DS} * T_0}{C_{RBL}}. \tag{2}$$

Here W_i and X_i represent the logical values of weights and inputs. To deal with the multi-bit operands, the input vector and weight matrices are stored as a 2's complementary value. Consequently, we need to further perform a post accumulation to add the ADC output from different column. A shift-adder is used to add the results at different column for multi-bit accumulation purpose.

$$P_{sum} = \sum \sum Dout_{i,j} 2^{i+j} * (-1)^{i==7 | j==7} \tag{3}$$

The -1 term is used to implement negative weights or input.

Assume the analog computing results are represented as voltage with a full range of V_{Full}, the vector length to be accumulated in the analog domain is 2^N, and the bitwidth of the input vector is 1. In this case, the possible analog accumulation results would range from $0 - 2^N$, and the sensing margin will be $V_{Full}/2^N$. As shown in the transfer function, two terms will significantly generate variations to affect the sensing margin. The first term is the current generated by each cell (I_{DS}), which will be affected by threshold

voltage variations. The second term is the pulse period T_0, which will be affected by the pullup/pulldown drivability of the driver in the RWL. In a practical design example, V_{Full} is about 0.7V and N is 6. The sensing margin will be around 10mV and an ADC with 6-bit ENOB is required. To achieve this sensing margin, the 3σ variation of I_{DS} and T_0 should be less than 2%, However, simulation results show that the 3σ I_{DS} variation can reach 20% under 1.0V voltage supply at 65nm with W/L=240n/120n. Despite the current domain accumulation scheme is simple, it is generally challenging to implement lossless computation based on current domain accumulation.

2.1.2 Charge Domain. Another representative approach is based on capacitive coupling scheme [18]. The memory cell is shown in Figure 1. Before the computation is performed, each CBL is first pre-discharged to the ground. When the computation start, the input vector is encoded as a binary value and a DC voltage will be applied on the read wordline (CWL) in the macro. When the logical value stored in the memory array is 1 and the CWL is driven to high voltage, charges V_{CWL}/C_i will be accumulated on the CBL, while there will be zero charge if CWL is 0 or logical value stored in the memory cell is 0. The final voltage at CBL will depend on the total charge which represents the bit-wise multiplication and accumulation results. The successive conversion and post processing is similar as current domain. The relationship between voltage at CBL and desired output (transfer function analog output and input vector) can be written as follows:

$$V_{CBL} = \frac{\sum Q_i V_{CWLi} * C_i}{\sum C_i}. \quad (4)$$

Here, the major term that affects the computing accuracy is the capacitance of C_i, and V_{CWL} can be designed with an accurate voltage reference. The capacitance variance can be reduced to 1% with a 2fF customized metal-oxide-metal capacitor. The charge domain design provides some potential to implement lossless analog domain IMC. However, the area and energy efficiency is relatively lower than current domain as multiple capacitors are added to each CWL to ignore the parasitic capacitance, and the complexity of the cell structure is large due to the presence of switches.

2.2 Digital IMC

To overcome the ADC overhead and inaccurate computation of analog IMCs, researchers also presented multiple digital implementation of the IMC macro [3, 19]. In the literature, one representative approach of digital IMC implementations is based on local adder trees. The local AND operation is implemented by a NOR gate with two reversed inputs to improve the area efficiency. As shown in Figure 1, in the digital IMC, a 4T-NOR gate is added to each memory cell, and the output of the NOR gate is connected to a local adder tree. Each local adder tree counts the bits "1" of the 4T-NOR gate and generate a bit-wise accumulation results. The results are sent to the same local PE for successive post processing, which is same as the analog IMC macro. Since all the computation is in the digital domain, the proposed IMC scheme is robust to PVT variations with arbitrary bitwidth. However, this scheme has one main drawback. The local adder tree consumes about 10× area overhead compared to the memory cell, so the memory density will be reduced extremely.

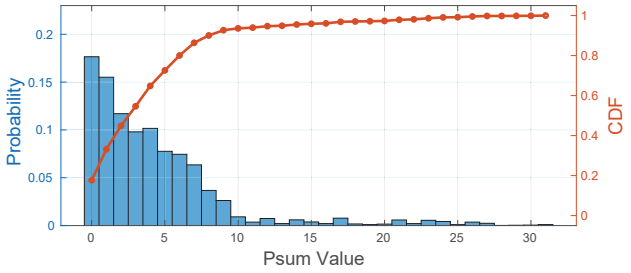

Figure 2: Psum value distribution of Resnet-20 on CIFAR-10 dataset.

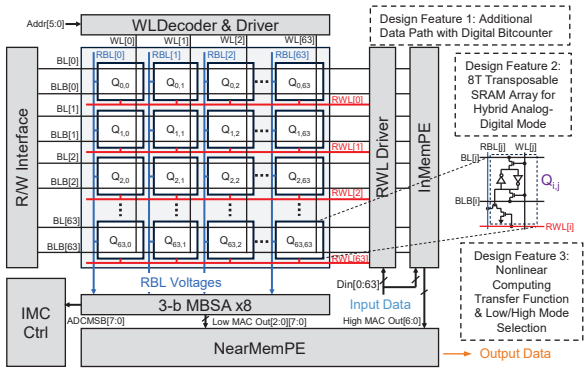

Figure 3: The overall architecture of our proposed macro.

3 DESIGN METHODOLOGY

3.1 Partial Sum Aware Accumulation

In a typical neural network model, the activations and weights are naturally sparse. Thus, the bit-wise accumulation results are usually small due to the sparsity. To understand the partial sum distribution, we choose ResNet-20 model on CIFAR-10 dataset, and calculate the partial sum value distribution under an assumption that 64 rows are activated in parallel. As shown in Figure 2, over 90% MAC results at each bitline are within 8 when 64 rows are activated in parallel. Based on this observation, we are motivated to design two different data paths for the low MAC value results (less than a specific threshold value) and high MAC value results (higher than a specific threshold value). Here, we choose the Low/High threshold as 8 to achieve a better trade-off between efficiency and implementation cost. For the low MAC value case, we can directly use the conventional analog IMC design method to implement the MAC operation in current domain. Since the low MAC value is less than 8, we only need to differentiate 8 possible states instead of 64 states in conventional analog IMC design. The sensing margin can be increased to 100mV when the full range is 0.7V, and the ENOB requirement of the ADC will be reduced to 3 bits. For the high MAC value case, we can design an additional data path based on digital IMC. Since only less than 10% MAC value is high, we can minimize the area overhead by sharing the digital path among a number of columns.

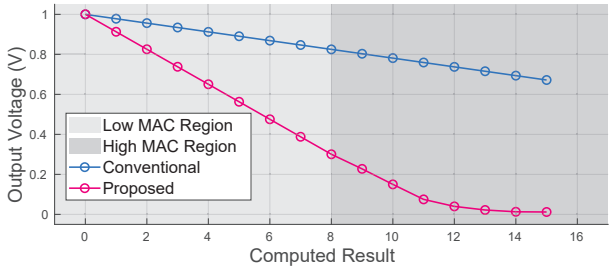

Figure 4: Simulated transfer function of proposed macro and conventional macro.

3.2 Macro Overview

Figure 3 shows the overall architecture of the proposed macro. Our macro is based on conventional analog current domain IMC design, with several moderate modifications. The proposed macro contains a wordline decoder and drivers, read/write interface (R/W Interface), several multi-bit sense amplifiers (MBSAs), one RWL driver and one or multiple memory array(s). Beyond the standard macro components, our macro has three key modifications to support the hybrid analog-digital computing mode. Firstly, we add an additional digital data path for high MAC value computing, motivated by the digital IMC computing scheme [21], to provide a simple and robust digital accumulation for high MAC value. The InMemPE is composed of several AND gates, a digital bitcounter, which is used to perform a lossless bit-wise MAC no matter what the MAC value is. One input for the InMemPE is the input vector and the other input is directly connected to BL/BLB. Secondly, the memory cell in the proposed design is implemented as a transposed 8T cell, where the RWL(RBL) is vertical to the standard WL(BL/BLB). This transposing design enables two separated data paths for both analog and digital computing. Thirdly, the IMC controller (IMC Ctrl) circuit is designed with additional functionalities to switch the analog-digital computing mode. The ADCMSB signal connected to the IMC Ctrl is used to indicate whether a specific MBSA detects a high MAC value. The NearMemPE is still a shift-adder to receive the low MAC results from the MBSAs, with an additional data path to receive the high MAC output from the InMemPE.

3.3 Nonlinear Transfer Function

Instead of forcing the desired output and analog voltages at RBL to be linear, we make the transfer function into two regions, as shown in the following equations:

$$V_{RBL} = \begin{cases} V_{DD} - \dfrac{\sum W_i X_i * I_{DS} * T_0}{C_{RBL}}, & \sum W_i X_i < N \\ 0, & \sum W_i X_i >= N \end{cases} \quad (5)$$

Here, N is a pre-defined threshold value to differentiate low and high MAC regions. For the low MAC region, we keep the original linear transfer function, where the desired computing results are linear to the voltage generated at RBL. For the high MAC region, we make the output voltage to a near zero value. Figure 4 shows the simulated transfer function of our proposed IMC macro and conventional IMC macro. Since we only need to convert the MAC results in the low MAC region accurately, the sensing margin can be improved by up to 7 × comparing with conventional design with linear transfer function.

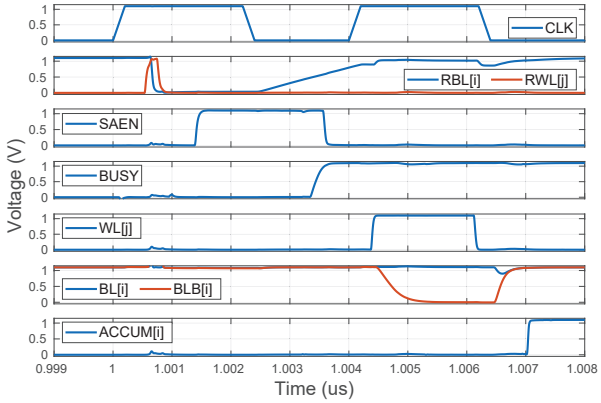

Figure 5: Simulated timing diagram of analog mode and digital mode computation.

It is worth noting that the nonlinear transfer function of the analog computation scheme is naturally supported. The only modification is to reduce the value of C_{RBL} or increase the value of I_{DS} and T_0. In this case, the I_{DS} can be designed as a constant only less than N current sources are activated, denote to the low MAC value region. Otherwise, the I_{DS} will naturally goes to near zero as the current source turns to the linear region.

3.4 MAC in Analog Domain

Before the computing starts, the RBL is pre-charged to V_{DD}. A input bit vector is sent to the RWL driver, and the trigger clock will generate a short pulse on each RWL if the corresponding input bit is 1. The current at each RBL will discharge the capacitor C_{RBL} to a certain voltage level, following the nonlinear transfer function we implemented. After the voltage at C_{RBL} is stable, a positive edge of the sense-enabling signal (SAEN) will trigger the interface circuits to sense the V_{RBL}. we use a MBSA with 7 different voltage reference to differentiate 8 different voltage level generated by the analog current summation. An additional comparator with another voltage reference adjusted to differentiate whether the output voltage crosses the ranges of low MAC value. If the output is within the low MAC region, the converted digital output will be directly sent to the NearMemPE to perform the bit-wise accumulation. Otherwise, the column index will be sent to the IMC Ctrl for successive MAC in digital domain.

The first cycle in Figure 5 shows a simulated timing diagram of proposed analog computing scheme when the computation results fall into a high MAC value region. After SAEN is triggered, the digital components (BUSY) will be activated due to a high MAC value is detected. In this case, the computation of the next cycle will goes to digital domain for high MAC value. Otherwise, the next cycle will be used for analog computation for the other columns or input bit vector, and there will be no throughput loss.

3.5 MAC in Digital Domain

When a high MAC value is detected, the IMC Ctrl circuit will store the column index of the high MAC value column and switch the

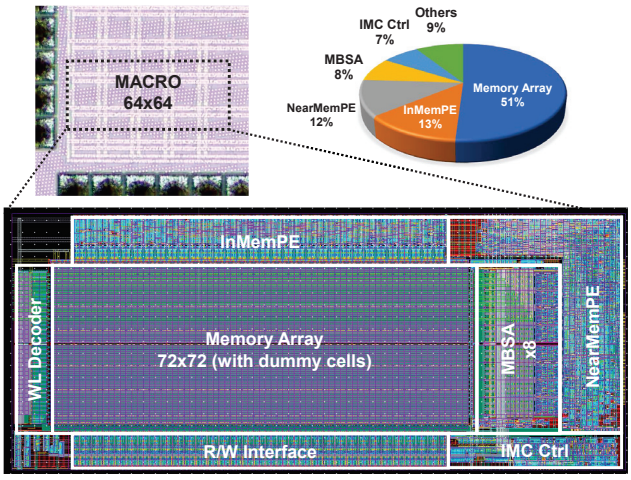

Figure 6: The die photograph, layout of our proposed macro, and the area breakdown results.

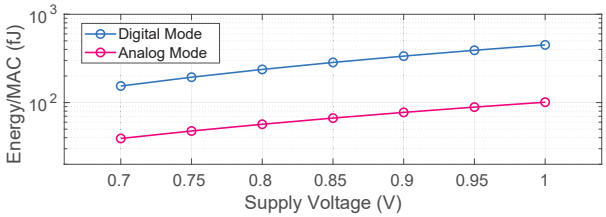

Figure 7: Measured energy consumption for digital mode and analog mode.

macro into the digital mode. The second cycle in Figure 5 illustrates the timing diagram of the digital computing scheme. A specific address will be generated in the IMC Ctrl to activate the wordline driver based on the column index detected to compute a high MAC value. Once the wordline is activated, the weight data stored in one column of the memory array will be read out through BL/BLB discharging process. The input vector is directly sent to the InMemPE to the weight data at BL/BLB will be sent to the InMemPE to perform a local digital computing with the input vector. At the same time, the NearMemPE is configured to receive the 7-bit output from the InMemPE. IMC Ctrl also generates the shift value for this accumulation depending on the column index.

4 EXPERIMENTS

4.1 Circuit-Level Measurement Results

We fabricated the SRAM IMC macro in GP 65nm technology node. The die photo and the detailed layout of the macro are shown in Figure 6. The proposed macro occupies 0.025 mm^2, with a 2.4 × 0.9um^2 8T SRAM bitcell. Most of the area is the memory array (51%), and the digital/analog interface only occupies 0.002 mm^2, and it takes about 8% of the whole macro, since we reduced the ENOB to only 3bit. Both of the InMemPE and the NearMemPE occupy 13% of the whole macro.

4.1.1 Energy Consumption. Figure 7 depicts the measured energy consumption for both digital mode and analog mode. When the

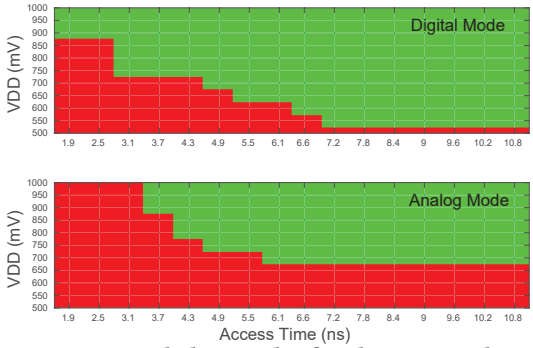

Figure 8: Measured Shmoo plot for the proposed macro for the digital mode and analog mode.

supply voltage is 1.0V, the digital mode requires about 450fJ to execute an 8-bit MAC, where it will be reduced to 154 fJ when the supply voltage is 0.7V. The effective energy efficiency is 2.22-6.89 TOPS/W for the digital mode computation. In contrast, the analog mode can provide significant energy reduction comparing with the digital mode. The energy consumption is 101 and 39.2 fJ when the supply voltage is 1.0 and 0.7 V, respectively. In this case, the effective energy efficiency in this mode is around 9.9-25.5 TOPS/W.

4.1.2 Speed. Figure 8 presents the measured shmoo plot for the proposed macro. The digital mode computation could achieve up to 2.3ns access time without any computation loss under 1.0V voltage supply. Even with the supply voltage at 0.6V, the access time could still achieve 4.9 ns to provide an accurate computing results. For the analog mode, the macro can reach 3.7ns under 1.0V supply voltage and 6ns under 0.7V supply voltage.

4.1.3 Comparative Results. Table 1 shows the comparative results with state-of-the-art IMC design. [10] is a conventional current domain design with a linear transfer function. Our design can achieve higher energy efficiency because our design does not require an ADC with large ENOB. The design in [10] shows higher area efficiency, which sacrifices the capability to provide a 8b lossless computation. Comparing with charge domain design in [11], our design also achieves higher energy efficiency and higher area efficiency due to the current-domain computation scheme.

We also compare our design with some designs implemented in advanced nodes, [22] and [3]. [22] is an extended version of [10] and their computing scheme is similar. Our design can achieve 2.65 × higher energy efficiency and 5.33× area area efficiency than this scheme. The current domain implementation has a low energy and area efficiency as a result of an interface ADC with a 6bit ENOB to support lossless computation. [3] is one representative work of digital IMC. All the analog IMC macro can achieve higher energy efficiency and memory density, as the local adder tree extremely increases the area overhead. In particular, our design could reach around 2× energy efficiency improvement over the fully digital design.

4.2 Architecture-Level Simulation Results

The computation flow and weight mapping scheme follows the principle shown in [1] and [2] for convolution neural networks and mobile convolution neural networks. We use an event-driven simulation framework in [23] to generate the area normalized throughput (denote to area efficiency) and energy efficiency for running 6

Table 1: Comparison with State-of-the-art IMC Design

Work	**This work**	JSSC19 [10]	JSSC21 [22]	ISSCC22 [3]	JSSC19 [11]
Technology Node	65	55	28	28	65
Speed (ns)	4	10.2	8.4	3.3	150
Area Efficiency ($GOPS/mm^2$)	160	572	120	200	126
8b Lossless	Yes	No	Yes	Yes	No
Energy Efficiency (TOPS/W)	25.5 (8b-8b)	18.37 (4b-5b)	16.63 (8b-8b)	27.4 (8b-8b)	28.1 (8b-1b)
Cell Structure	8T	T8T	6T+LCC	6T+LCC	10T-1C
Computing Scheme	hybrid current-digital	current	current	digital	charge

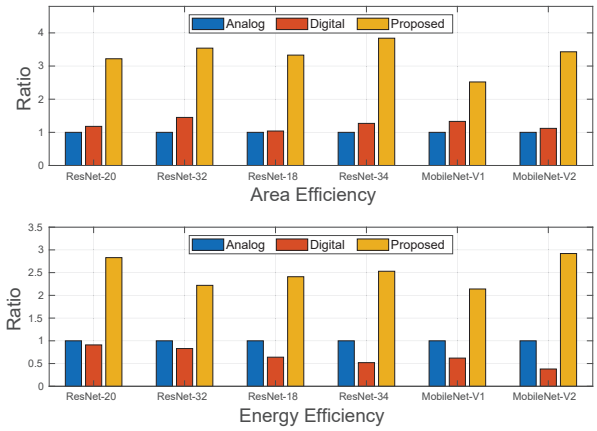

Figure 9: Architecture simulation results for area efficiency and energy efficiency on various machine learning model.

different models trained on CIFAR-10 and ImageNet datasets, based on the circuit-level evaluation results. We used a fully analog IMC macro in current domain in [22], and a fully digital IMC macro in [3] to build the same architecture, and scale our area efficiency and energy efficiency results into the same technology node.

The architecture-level simulation results are shown in Figure 9. The architecture based on our proposed IMC macro could achieve 2.52× to 3.84× improvement of the area efficiency comparing with the architecture based on a fully analog IMC macro. In terms of the energy efficiency, the architecture based on our proposed IMC macro could achieve 2.14× to 2.92× improvement comparing with the architecture based on a fully analog IMC macro.

5 CONCLUSION

In this paper, we implement lossless 8b MAC operation with a hybrid digital/analog computing scheme. We first propose a nonlinear transfer function of the analog accumulation to reduce the ENOB and improve the sensing margin of the conventional current domain IMC scheme for low MAC values. We further propose a in memory digital computing scheme to compensate the error generated by analog computing. Chip measurement results show that the proposed macro could achieve 160 $GOPS/mm^2$ area efficiency and 25.5 TOPS/W for 8b/8b matrix computation. The architectural-level evaluation for real workloads shows that the proposed macro can achieve up to 2.92× higher energy efficiency than conventional analog IMC designs.

ACKNOWLEDGMENTS

This work was supported by National Science Foundation under grant 2328805 and 2112562.

REFERENCES

[1] Qilin Zheng et al. Lattice: an adc/dac-less reram-based processing-in-memory architecture for accelerating deep convolution neural networks. In *DAC*, 2020.
[2] Qilin Zheng et al. Mobilattice: A depth-wise dcnn accelerator with hybrid digital/analog nonvolatile processing-in-memory block. In *ICCAD*, 2020.
[3] Bonan Yan et al. A 1.041-mb/mm 2 27.38-tops/w signed-int8 dynamic-logic-based adc-less sram compute-in-memory macro in 28nm with reconfigurable bitwise operation for ai and embedded applications. In *ISSCC*, 2022.
[4] Kodai Ueyoshi et al. Diana: A n end-to-end energy-efficient digital and analog hybrid neural network soc. In *ISSCC*, 2022.
[5] Pouya Houshmand et al. Diana: An end-to-end hybrid digital and analog neural network soc for the edge. *JSSC*, 2022.
[6] Ziru Li et al. Asters: adaptable threshold spike-timing neuromorphic design with twin-column reram synapses. In *DAC*, 2022.
[7] Qilin Zheng et al. Artificial neural network based on doped hfo 2 ferroelectric capacitors with multilevel characteristics. *EDL*, 2019.
[8] Zongwei Wang et al. Self-activation neural network based on self-selective memory device with rectified multilevel states. *TED*, 2020.
[9] Richard Linderman et al. Apparatus for performing matrix vector multiplication approximation using crossbar arrays of resistive memory devices, 2015. US Patent 9,152,827.
[10] Xin Si et al. A twin-8t sram computation-in-memory unit-macro for multibit cnn-based ai edge processors. *JSSC*, 2019.
[11] Avishek Biswas et al. Conv-sram: An energy-efficient sram with in-memory dot-product computation for low-power convolutional neural networks. *JSSC*, 2018.
[12] Yen-Cheng Chiu et al. A 4-kb 1-to-8-bit configurable 6t sram-based computation-in-memory unit-macro for cnn-based ai edge processors. *JSSC*, 2020.
[13] Xin Si et al. 15.5 a 28nm 64kb 6t sram computing-in-memory macro with 8b mac operation for ai edge chips. In *ISSCC*, 2020.
[14] Win-San Khwa et al. A 65nm 4kb algorithm-dependent computing-in-memory sram unit-macro with 2.3 ns and 55.8 tops/w fully parallel product-sum operation for binary dnn edge processors. In *ISSCC*, 2018.
[15] Edward Choi et al. A 133.6 tops/w compute-in-memory sram macro with fully parallel one-step multi-bit computation. In *CICC*, 2022.
[16] Xiaoxuan Yang et al. Improving the robustness and efficiency of pim-based architecture by sw/hw co-design. In *ASP-DAC*, 2023.
[17] Qilin Zheng et al. Enhance the robustness to time dependent variability of reram-based neuromorphic computing systems with regularization and 2r synapse. In *ISCAS*, 2019.
[18] Jinseok Lee et al. Fully row/column-parallel in-memory computing sram macro employing capacitor-based mixed-signal computation with 5-b inputs. In *VLSI*, 2021.
[19] Yu-Der Chih et al. 16.4 an 89tops/w and 16.3 tops/mm 2 all-digital sram-based full-precision compute-in memory macro in 22nm for machine-learning edge applications. In *ISSCC*, 2021.
[20] Chia-Fu Lee et al. A 12nm 121-tops/w 41.6-tops/mm2 all digital full precision sram-based compute-in-memory with configurable bit-width for ai edge applications. In *VLSI*, 2022.
[21] Qilin Zheng et al. Accelerating sparse attention with a reconfigurable non-volatile processing-in-memory architecture. In *DAC*, 2023.
[22] Xin Si et al. A local computing cell and 6t sram-based computing-in-memory macro with 8-b mac operation for edge ai chips. *JSSC*, 2021.
[23] Qilin Zheng et al. Pimulator-nn: An event-driven, cross-level simulation framework for processing-in-memory-based neural network accelerators. *TCAD*, 2022.