2-Bit-per-Cell RRAM based In-Memory Computing for Area-/Energy-Efficient Deep Learning

Wangxin He, *Student Member, IEEE,* Shihui Yin, *Student Member, IEEE,* Yulhwa Kim, *Student Member, IEEE,* Xiaoyu Sun, *Student Member, IEEE,* Jae-Joon Kim, *Member, IEEE,* Shimeng Yu, *Senior Member, IEEE,* Jae-sun Seo, *Senior Member, IEEE*

***Abstract*—In-memory computing (IMC) has emerged as a promising technique for enhancing energy-efficiency of deep neural networks (DNN). While embedded non-volatile memory such as resistive RAM (RRAM) is a good alternative to SRAM/ DRAM for IMC owing to high density, low leakage, and non-destructive read, most prior works have not demonstrated using multi-level RRAM devices for array-level IMC operations. In this work, we present an IMC prototype with 2-bit-per-cell RRAM devices for area-/energy-efficient DNN inference. Optimizations on four-level conductance distribution and peripheral circuits with input-splitting scheme have been performed, enabling high DNN accuracy and low area/energy consumption. The prototype chip that monolithically integrated 90nm CMOS and 2-bit-per-cell RRAM array achieves 87% (83%) CIFAR-10 accuracy and 25 (51) TOPS/W energy-efficiency at 1.2 V (0.9 V) supply. At 1.2V, a stable accuracy of ~87% is maintained throughout 108 hours.**

***Index Terms*—Deep neural networks, in-memory computing, multi-level cell, RRAM**

I. Introduction

With exponential growth in the sizes of deep/convolutional neural networks (DNN/CNN), demands for highly dense and energy-efficient memory devices have skyrocketed [1]. Compared to CMOS memory technologies such as SRAM/DRAM, embedded non-volatile memory such as RRAM has shown advantages in high density, low leakage power, non-volatility, and multi-level programming.

For DNN hardware accelerators, conventionally volatile and non-volatile memories were accessed in a row-by-row manner and data was communicated to/from separate multiply-and-accumulate (MAC) or computation engines. To resolve such data access/communication bottleneck, in-memory computing (IMC) has emerged as a promising technique [2]. By asserting multiple or all rows simultaneously, analog computations of MAC operations are performed inside the memory (e.g. along the bitline), substantially reducing the memory access energy and latency of row-by-row operations.

Several RRAM based in-memory computing prototypes have been presented, but most of them only employed single-level cell designs [3]-[5]. The device-level programming of 2-bit/3-bit per RRAM cell has been reported but was limited to row-by-row read-out [6][7] or simulation of multi-row read-out based on cell-by-cell measurement [5]. Recently, [8] reported IMC with four-level RRAM programming, but only demonstrated a simple two-layer multi-layer perceptron for a low 94.4% accuracy for MNIST dataset.

This paper demonstrates in-memory computing using 2-bit-per-cell RRAM array, towards dense and energy-efficient inference of large DNNs. We assert all rows of the 128×64 RRAM array, but use input-splitting scheme to simplify the area-/power-hungry analog-to-digital converters (ADCs) at the column periphery into single sense amplifiers (SAs). The prototype chip has been implemented in 90nm CMOS with monolithic integration of RRAM. We benchmarked three different CNNs for CIFAR-10 dataset, achieving up to 87% (83%) accuracy, and 25 (51) TOPS/W energy-efficiency at 1.2 V (0.9 V) supply. Compared to a 1-bit-per-cell RRAM design, we achieve 2.8-5.3% CNN accuracy improvement for the same area. We also evaluated the RRAM conductance distribution over 108 hours, and demonstrated robust CNN accuracy of ~87%.

Manuscript received May 15, 2020. We thank Winbond Electronics for RRAM chip fabrication support. This work is partially supported by JUMP CBRIC, JUMP ASCENT, NSF/SRC E2CDA program, SRC AIHW program, NSF grants 1652866/1715443/1740225, and Nano-Material Technology Development program by National Research Foundation of Korea.

Wangxin He, Shihui Yin, and Jae-sun Seo are with Arizona State University, AZ 85287 USA (e-mail: jaesun.seo@asu.edu).

Yulhwa Kim and Jae-Joon Kim are with Pohang University of Science and Technology, Pohang, South Korea.

Xiaoyu Sun and Shimeng Yu are with Georgia Institute of Technology, GA 30332 USA.

II. In-Memory Computing RRAM Macro Design

A. In-Memory Computing Design with Four-Level RRAM Devices

Our proposed RRAM macro design supports the multiplication of 2-bit weights (e.g. -3, -1, +1, +3) and 1-bit activation (e.g. -1, +1) in a single cycle. As shown in Fig. 1(a), we use two vertically-adjacent cells and differential wordlines (WLs) to represent one 2-bit weight. The activation of +1 makes top (bottom) WL to be 1 (0) and activation of -1 makes bottom (top) WL to be 1 (0). We set the four conductance levels as G_{LOW} (highest resistance state), $G_{HIGH}\times1/3$, $G_{HIGH}\times2/3$, and G_{HIGH} (lowest resistance state), and we program the two 1T1R cells differentially as [G_{LOW} and G_{HIGH}] or [$G_{HIGH}\times1/3$ and $G_{HIGH}\times2/3$], as shown in Fig. 1(a). This way, element-wise multiplication results of -3, -1, +1, and +3 will be mapped to RBL voltage (V_{RBL}) being pulled down with G_{LOW}, $G_{HIGH}\times1/3$, $G_{HIGH}\times2/3$, and G_{HIGH} conductance, respectively (Fig. 1(b)).

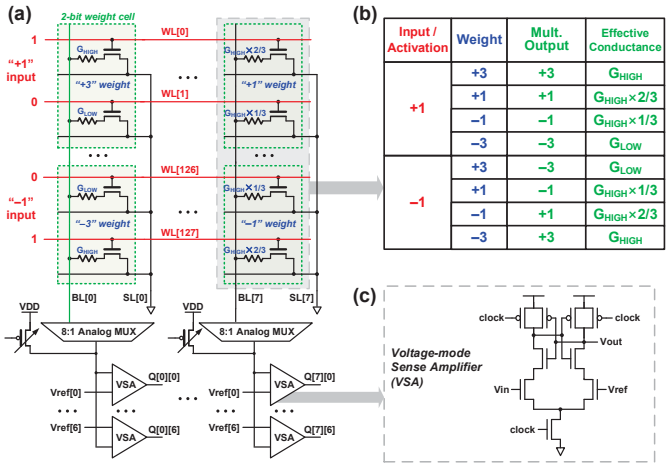

Fig. 1. In-memory computing operation with four-level RRAM devices.

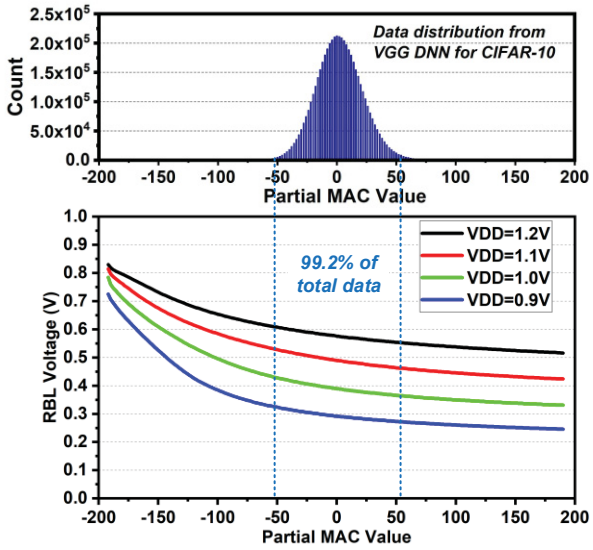

Fig. 2. (Top) Partial MAC data distribution. (Bottom) Simulated transfer curve of the RBL voltage at different supply voltages.

By simultaneously asserting all differential WLs of the RRAM array, all cells in the same column are computed in parallel. The RRAM cells that pull down RBL and the configurable PMOS header that pulls up RBL form a resistive divider, resulting in V_{RBL} that represents the 64-input partial sum between -192 and +192 (Fig. 2). The PMOS header is digitally configurable in 16 different strength values. Through 8-to-1 column multiplexers, RBL is connected to a group of SAs, which can be served collectively as a flash ADC or as individual SAs (Fig. 1(c)). One group of SAs is shared for every 8 columns. The area of the 1T1R bitcell that we used is ~0.5µm×0.5µm (~31 F^2), and thus one 2-bit RRAM cell occupies ~62 F^2 area, which is much smaller than two SRAM cells with 300-400 F^2.

B. Four-Level RRAM Programming

To achieve 2-bit RRAM, two intermediate conductance levels are inserted between the minimum and maximum conductance levels, where the conductance interval is kept identical between adjacent states. A write-verify programming scheme is iterated until <5% of 4kb (128×64) RRAM cells are outside the target conductance range for each of the four levels. First, we set the initial gate voltage (V_G) and apply a 100 ns SET pulse with 2.1V amplitude. If the resistance after SET is lower than the lower bound, a 200 ns RESET pulse with 3.8 V amplitude and V_G of 4.0 V is applied, followed with a SET pulse with

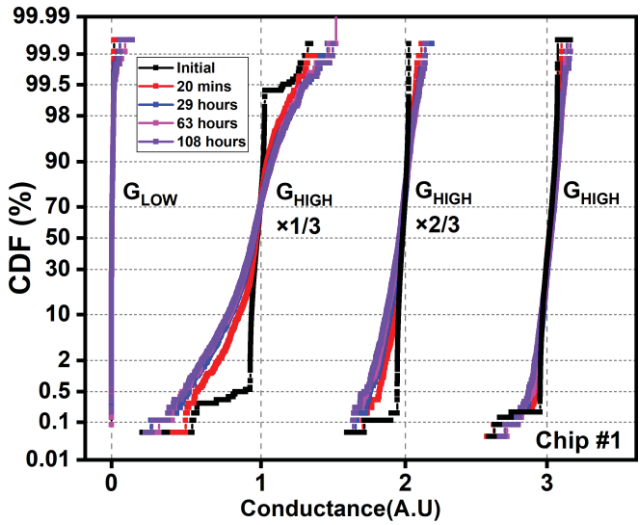

Fig. 3. Measured four-level conductance distribution over 108 hours.

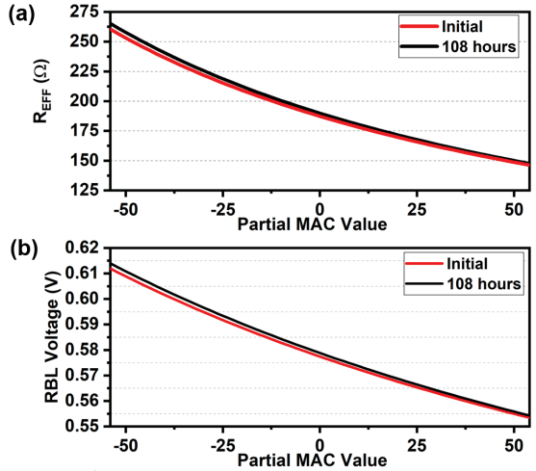

Fig. 4. (a) Effective resistance (R_{EFF}) and (b) V_{RBL} change between initial programming and 108 hours.

a new V_G lower by a 'lower ΔV'. If the resistance after SET is higher than the upper bound, a RESET pulse is applied, followed with a SET pulse with a new V_G higher by an 'upper ΔV'. After 15 write-verify iterations, if the resistance is still outside of the lower/upper bounds, we further reduce the lower/upper ΔV for finer adjustment.

Fig. 3 shows the four-level programming results of HfO_2-RRAM devices [10] and the distributions over time. While the minimum and maximum conductance levels maintain tight distributions over 108 hours, two intermediate conductance levels show moderate relaxation over time. In particular, G_{HIGH}×1/3 encounters more relaxation, due to relatively higher resistance value and stability from a weak filament in RRAM. This symptom needs to be evaluated for reliable IMC design.

To understand the effect of wider conductance distribution for IMC, we have calculated the effective resistance (R_{EFF}) of 64 parallel pull-down paths in one column, by randomly choosing each resistance value from the CDF data in Fig. 3. We also performed transistor-level simulation of eight columns with randomly selected resistances from Fig. 3 data and observed V_{RBL}. Fig. 4 shows the simulation results using conductance distributions after initial programming and after 108 hours. Since large relaxation only occurs to a small percentage of RRAM cells and the positive/negative relaxation cancels out, R_{EFF} and V_{RBL} only changes by up to 1.85% and 0.32%, respectively, across different MAC values over 108 hours. Therefore, we surmise that the effect of RRAM relaxation on IMC results will be insignificant. Further chip measurement results will be presented in Section III.

C. Column Sensing Optimization with Input-Splitting

In previous in-RRAM computing works [3], it has been shown that ADCs pose critical challenges for area and energy. Input-splitting was proposed to reduce the ADC overhead in IMC by splitting a large layer

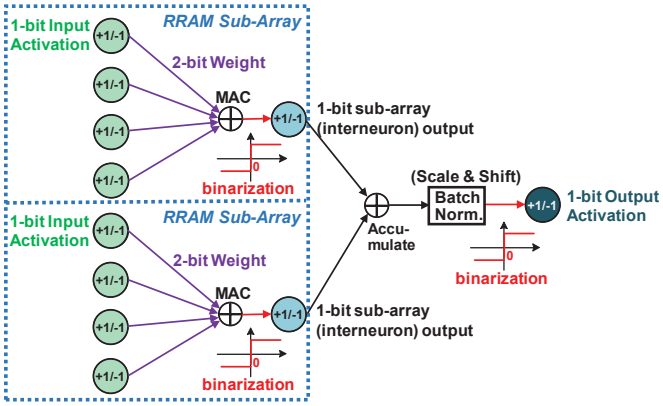

Fig. 5. Input-splitting scheme with 2-bit weights. DNNs are trained so that RRAM array outputs are binarized.

into small groups with binarized outputs [9], and has been applied to binary RRAM arrays in [5].

In this work, we re-designed the input-splitting algorithm to support 2-bit weights and 1-bit activations, as illustrated in Fig. 5. Input-splitting for binary DNNs [5] could employ a fixed scaling factor for partial sums since the distributions do not deviate considerably during training. However, in DNNs with 2-bit weights, the distributions of partial sums change dynamically, necessitating a trainable scaling factor. To that end, we trained the scaling factor for partial sums as a parameter using zero-mean batch normalization (BN), which completely removes biasing operations in BN assuming that the mean value of MAC results is 0. The scaling operation with the zero-mean BN stabilizes the gradients during the training stage. Note that the zero-mean BN can be removed in the inference stage as it does not affect the binarization of the MAC result, and this importantly allows using identical reference voltages (Vrefs) for all SAs for 64 columns.

III. Measurement Results and DNN Evaluation

The prototype chip (Fig. 6(a)) was fabricated in an industrial 90nm CMOS technology that monolithically integrates HfO_2-RRAM between M1 and M2 [10]. 128×64 RRAM array is integrated with the peripheral circuits including row decoder/drivers, eight groups of SAs (one group of SAs shared among eight columns), eight 8-to-1 column multiplexers, level-shifters and two 64-to-1 column decoders for RRAM cell-level high-voltage programming. The row decoder has two modes of operation: (1) it asserts all differential WL signals simultaneously for analog MAC operations, or (2) generates one-hot WL signals for cell-level programming. We performed measurement of two chips at room temperature. Fig. 6(b) shows the power breakdown of chip #1 at 1.2V supply. The power of decoder/driver modules and RRAM/SA modules were measured directly from the chip with separate power supplies. With the resistive divider formed by the PMOS header and the RRAM pull-down paths dissipating crowbar current, RRAM array dominates the power consumption.

A. IMC Measurement Results from RRAM Array

We first programmed the RRAM array with the values of a 64×64 weight submatrix from the trained DNN with 2-bit weights using the write-verify scheme described in Section II.B. 2,000 64-bit binary test vectors were then presented to the RRAM array, to perform MAC computations and obtain the 2,000×64 outputs. In total, 128,000 pairs of measured sum of seven SAs' outputs and target MAC values are used to estimate the joint distribution of these two, and the resultant 2-D histogram is shown in Fig. 7(a). The sum of seven SAs' output needs to be binarized to either +1 or -1 as the interneuron output. From the results of Fig. 7(a), we obtain the conditional probability for each MAC value being binarized to +1, as shown in Fig. 7(b). Probability of 1/0 in Fig. 7(b) corresponds to interneuron value of +1/-1.

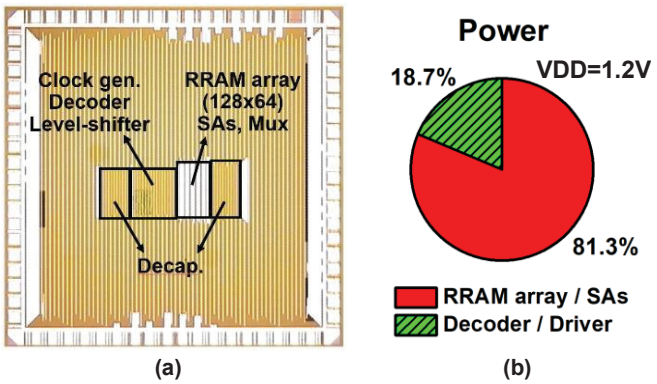

Fig. 6. (a) Die photo of prototype chip. (b) Power breakdown of chip #1.

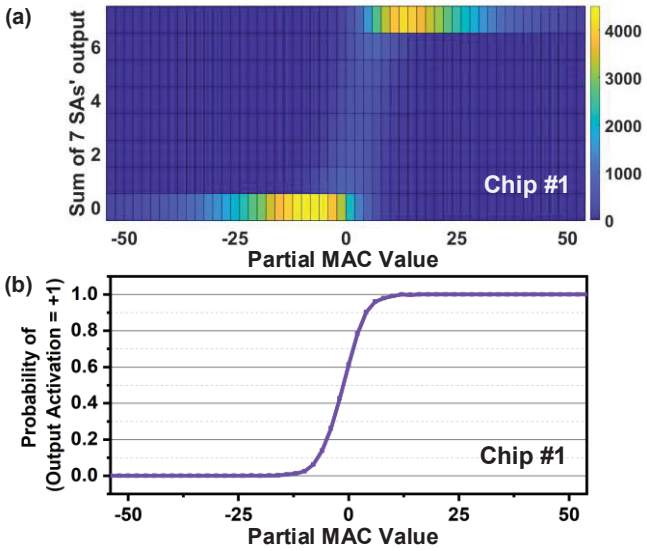

Fig. 7. (a) 2-D histogram of the partial sum and the measured SA output at time=108 hours. (b) Probability of interneuron output for partial MAC values.

B. DNN Evaluation

When we map DNNs onto RRAM arrays, the IMC computations of 64 inputs and 64×64 weights are first stochastically quantized to 1-bit (+1 or -1) according to the conditional probability distribution in Fig. 7(b). Subsequently, the accumulation of partial sums and non-MAC operations such as max-pooling are performed in digital simulation with high fixed-point precision, to obtain the DNN accuracy results.

We benchmarked the inference accuracy of the proposed 2-bit RRAM array for three DNNs (heavy-VGG, light-VGG, AlexNet-like CNN) for CIFAR-10 dataset (Table I). All convolution and fully-connected layers of DNNs are mapped onto multiple 2-bit RRAM instances, where weights for different input (output) channels are stored on different rows (columns), and weights within each convolution kernel (e.g. 9=3×3) are stored in different RRAM arrays.

TABLE I: DNN models used for evaluation for CIFAR-10 dataset.

DNN Model	DNN Layer Structure
Heavy-VGG	126C3-B-126C3-B-252C3-B-252C3-B-511C3-B-512C3-B-FC1024-B-FC1024-B-FC10-B
Light-VGG	126C3-B-126C3-B-189C3-B-189C3-B-252C3-B-256C3-B-FC512-B-FC512-B-FC10-B
AlexNet-like	91C3-B-M-252C3-B-M-378C3-B-378C3-B-256C3-B-M-FC1024-B-FC1024-B-FC10-B

* *n*C*m*: convolutional layer with *n* channels and *m*×*m* kernel, B: batch normalization layer, M: max-pooling (2×2), FC: fully-connected layer

C. Energy, Performance, and Accuracy Characterization

As shown in Fig. 8, the input-splitting algorithm and corresponding measurements incur minimal accuracy degradation for the three DNNs, and the two chips that we measured exhibit similar CIFAR-10 accuracy. Compared to binary RRAMs, in-memory computing with 2-bit-per-cell RRAMs achieves 2.8-5.3% DNN accuracy improvement for the same area, or 2X area reduction for the same accuracy. If we compare the accuracy between input-splitting algorithm and hardware measurement, this work shows considerably less accuracy degradation (-0.76% in average) than that of binary DNNs [5] (-2.61% in average), demonstrating that DNNs with 2-bit weights exhibit more robustness against hardware noise/variability of in-memory computing.

Our implementation of the input splitting algorithm allows using only one SA for RBL sensing. Since the RRAM array has seven SAs for every eight columns, we experimented using the seven independent

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/LSSC.2020.3010795, IEEE Solid-State Circuits Letters

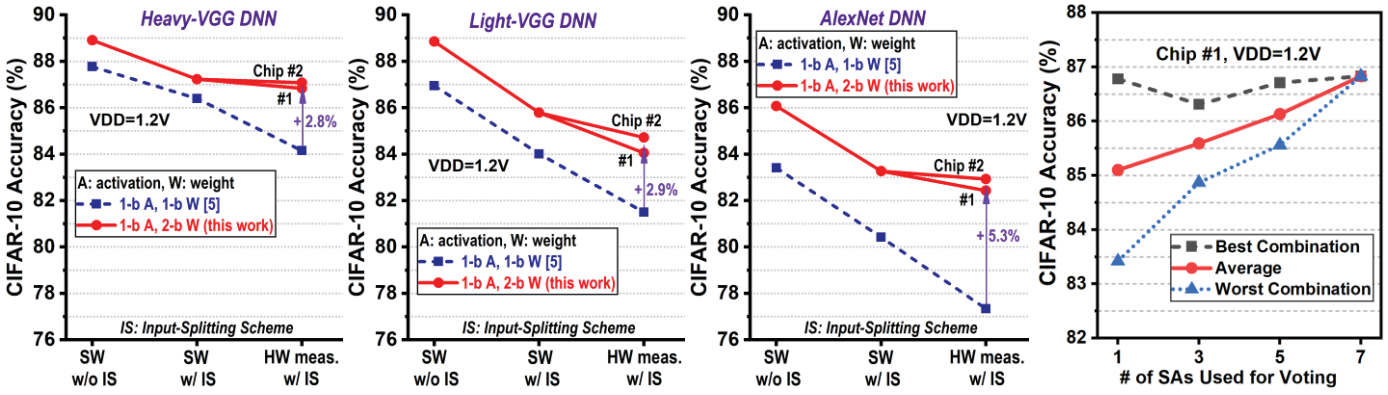

Fig. 8. Accuracy (software vs. measurements) of three DNNs for 1-bit/2-bit weights and without/with input-splitting.

Fig. 9. Number of SAs vs. accuracy.

SAs with identical Vref to vote majority and obtain the binary output for the interneuron. While the results in Fig. 8 used all seven SAs in the prototype chip, we also experimented using a small number of SAs. Fig. 9 shows that the best SA combination outputs show similar CIFAR-10 accuracy compared to the voting results of seven SAs. On average, using more SAs for voting results in improved CIFAR-10 accuracy, due to the averaging effect of hardware variability.

With dynamic voltage scaling (Fig. 10(a)), the power of both analog and digital modules are largely reduced, improving energy-efficiency from 25 TOPS/W at 1.2V to 51 TOPS/W at 0.9V. This is achieved by trading off the voltage margin of SAs, leading to small (1.0% for Light-VGG) or moderate (5.5% for Heavy-VGG) DNN accuracy loss, as shown in Fig. 10(b). At 1.2V/0.9V, the leakage power accounts for 2.1%/0.9% of the total power consumption.

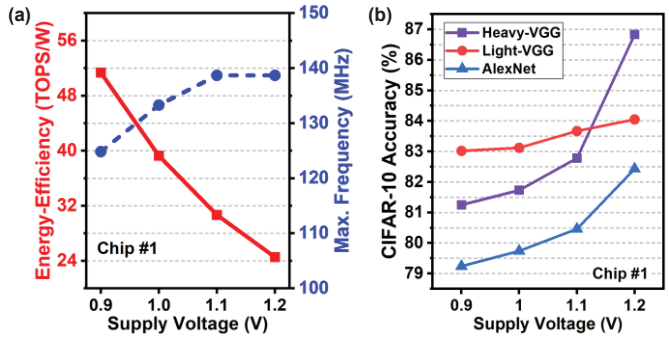

Fig. 10. (a) Measured energy/frequency results with voltage scaling. (b) Accuracy of three DNNs with voltage scaling.

TABLE II: Heavy-VGG CNN accuracy over time.

Time (hours)	0	15	29	43	63	87	108
Accuracy (%)	87.1	87.2	86.8	86.9	87.3	87.2	87.0

TABLE III: Comparison with prior works on RRAM-based in-memory computing demonstrated on CNNs for CIFAR-10.

	[3]	[4]	[7]	**This work**
CMOS Technology	55nm	150nm	130nm	**90nm**
Array Size	256×512	256×256	1Mb	**128×64**
# of bits per RRAM (B)	1	1	2-3	**2**
# of rows turned on (R)	9	2-16	1	**64**
Column sensing	4b ADC	Spike counting	N/A	**1b SA**
Energy-efficiency (TOPS/W)	53.2–21.9	16.9	N/A	**51.4–24.5**
FoM[1] (TOPS/W×B×R)	478.8	270.4	N/A	**6,579 (14X↑)**
CIFAR-10 Accuracy	81.8–88.5%	~80%	83.0%	**83.0–87.1%**

[1] FoM represents 1/(energy×delay×area).

To assess the robustness of IMC over time amidst RRAM relaxation (Fig. 3), we characterized the Heavy-VGG CNN accuracy over 108 hours, as shown in Table II. Similar relaxation in conductance has been reported in prior works [11]. Still, we observed that the effective resistance and RBL voltage remains relatively constant, and with Vref calibration for SAs, the CNN accuracy for CIFAR-10 is maintained stably around 87% over 108 hours (Table II). Table III shows the comparison with prior in-RRAM computing works. Our work is the first to demonstrate 2-bit-per-cell in-RRAM computing with assertion of a high number of rows (64) for large CNNs for CIFAR-10 dataset. Using the figure-of-merit (FoM) that represents the inverse of energy-delay-area product, our design achieves 14X higher FoM than that of [3].

IV. Conclusion

In this work, we present a 2-bit-per-cell RRAM based in-memory computing prototype in 90nm CMOS. Input splitting scheme replaced power-hungry ADCs with simple SAs. Three different DNNs were benchmarked, achieving CIFAR-10 accuracy of 87% (83%) and 24.5 (51.4) TOPS/W energy-efficiency at 1.2V (0.9V) supply. At 1.2V, a stable accuracy of ~87% is maintained throughout 108 hours.

References

[1] X. Xu *et al*., "Scaling for edge inference of deep neural networks," *Nature Electronics*, vol. 1, pp. 216-222, 2018.
[2] N. Verma *et al*., "In-memory computing: advances and prospects," *IEEE Solid-State Circuits Magazine*, vol. 11, no. 3, pp. 43-55, Summer 2019.
[3] C. Xue *et al*., "A 1Mb Multibit ReRAM Computing-In-Memory Macro with 14.6ns Parallel MAC Computing Time for CNN Based AI Edge Processors," *IEEE ISSCC*, 2019.
[4] B. Yan *et al*., "RRAM-based spiking nonvolatile computing-in-memory processing engine with precision-configurable in situ nonlinear activation," *IEEE Symp. on VLSI Technology*, 2019.
[5] S. Yin *et al*., "Monolithically integrated RRAM and CMOS based in-memory computing optimizations for efficient deep learning," *IEEE Micro*, vol. 39, no. 6, Nov./Dec. 2019.
[6] B. Q. Le *et al*., "Resistive RAM with multiple bits per cell: Array-level demonstration of 3 bits per cell, " *IEEE Trans. on Elec. Devices*, Jan. 2019.
[7] E. Hsieh *et al*., "High-density multiple bits-per-cell 1T4R RRAM array with gradual set/reset and its effectiveness for deep learning," *IEEE IEDM*, 2019.
[8] Q. Liu *et al*., "A fully-integrated analog ReRAM based 78.4 TOPS/W computing-in-memory chip with fully-parallel MAC computing," *IEEE ISSCC*, 2020.
[9] Y. Kim *et al*., "Input-splitting of large neural networks for power-efficient accelerator with resistive crossbar memory array," *IEEE ISLPED*, 2018.
[10] C. Ho *et al*., "Integrated HfO2-RRAM to achieve highly reliable, greener, faster, cost-effective, and scaled devices," *IEEE IEDM*, 2017.
[11] C. Wang *et al*., "Relaxation effect in RRAM arrays: demonstration and characteristics," *IEEE Electron Device Letters*, vol. 37, no. 2, Feb. 2016.