RRAM Based In-Memory Computing for Area-/Energy-Efficient Deep Learning

by

Wangxin He

A Dissertation Presented in Partial Fulfillment
of the Requirements for the Degree
Doctor of Philosophy

Approved June 2023 by the
Graduate Supervisory Committee:

Jae-sun Seo, Chair
Yu Cao
Deliang Fan
Matthew Marinella

ARIZONA STATE UNIVERSITY

August 2023

ABSTRACT

Deep neural networks (DNNs) have been successfully developed in many applications including computer vision, speech recognition, and others. As the complexity of DNN tasks increases, the number of weights or parameters in DNNs surges as well, leading to consistent demands for denser memories than SRAMs. Conventional DNN accelerator systems have used DRAM to store a large number of DNN weights, but DRAM requires cumbersome refresh operations and off-chip memory access consumes very high energy consumption. Instead of using off-chip memory, several recent accelerators employed embedded non-volatile memory (NVM) such as resistive RAM (RRAM) to store a large amount of weight fully on-chip and reduce the energy consumption for overall memory access.

Non-volatile resistive devices such as RRAM can naturally support in-memory computing (IMC) operations with multiple rows turned on, where the weighted sum current between the wordline voltage (representing DNN activations) and RRAM conductance (representing DNN weights) represents the dot-product result.

This dissertation first presents a circuit-/device-level optimization to improve the energy and density of RRAM-based in-memory computing architectures. experimental results are reported based on prototype chip design of 128×64 RRAM arrays and CMOS peripheral circuits, where RRAM devices are monolithically integrated in a commercial 90nm CMOS technology.

Next, this dissertation presents 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 an input-splitting scheme have been performed, enabling high DNN accuracy and low area/energy consumption.

Furthermore, this dissertation presents an investigation on the relaxation effects

on multi-level resistive random access memory-based in-memory computing for deep neural network inference.

Plus, this dissertation works on the Progressive-wRite In-memory program-VErify (PRIVE) scheme, which this thesis verify with an RRAM testchip for IMC-based hardware acceleration for DNNs. This dissertation optimizes the progressive write operations on different bit positions of RRAM weights to enable error compensation and reduce programming latency/energy while achieving high DNN accuracy.

For the ongoing project, this dissertation includes the progress of the RRAM-based hybrid in-memory computing process and the progress on the ferroelectric capacitive devices for next-generation AI hardware.

ACKNOWLEDGMENTS

I owe many thanks to my parents - Erjin He and Zeying Wang for their unwavering support and love throughout my Ph.D. education and my life. There is a long way to the doctoral degree and I cannot hold along without the support from my family.

Here I want to express my sincere gratitude to my committee chair Dr. Jae-sun Seo, for all the guidance throughout my Ph.D. career, and for all the knowledge, and assistance, in my research work, professional career, and daily life. My professor taught me with his patience and assistance not only in research study and skills, but also in personality, and attitude towards challenges and difficulties in both academics and more complicated environments in social life. When I decided to start my Ph.D. career, I wasn't expecting such a fantastic professor as my mentor toward my research goals. I have the great honor to become his student in Prof. Jae-sun Seo's Lab for my Ph.D. career.

I want to extend my gratitude to Dr. Yu Cao, Dr. Deliang Fan, and Dr. Matthew Marinella for their guidance in my research projects, comments and suggestions on my dissertation, and for taking the time to serve as committee members. I want to thank Dr. Shimeng Yu, Dr. Bipin Rajendran, Dr. Naresh Shanbhag and Dr. Ivan Sanchez Esqueda for the co-operation and help during the related research works.

I am indebted to my colleagues, Shihui Yin, Injune Yeo, Jian Meng, Fan Zhang, Xu Han, Sahra Afshari, Sai Kiran Cherupally, Shreyas Kolala Venkataramanaiah, Jyotishman Saikia, Han-Sok Suh, Yuan Liao, Ahmed Hassan, and Amitesh Sridharan for their invaluable contribution, discussions, and support throughout my study. I must also thank my graduate advisor James VanderPloeg and Toni Mengert for her active help with all the administrative procedures. Thanks to my friends, Yinan Wang, Yuantao Chen, Yiheng Chen, Qingzu He, Yilin He, Wenzhong Hu, Yuhong Lu,

Ting-an Yan, Ziyi Wang, Zhenyu Wang, Shuo Xie and many others, who were also crucial in the successful realization of this dissertation.

TABLE OF CONTENTS

CHAPTER Page

APPENDIX Page

LIST OF TABLES

LIST OF FIGURES

Figure Page

Figure Page

Chapter 1

INTRODUCTION

1.1 DNN Acceleration with Non-volatile Memory

Deep neural networks (DNNs) have been successfully developed in many applications including computer vision, speech recognition, and others. As the complexity of DNN tasks increases, the number of weights or parameters in DNNs surges as well, leading to consistent demands for denser memories than SRAMs. Conventional DNN accelerator systems have used DRAM to store a large number of DNN weights, but DRAM requires cumbersome refresh operations and off-chip memory access consumes very high energy consumption Sze *et al.* (2017). Instead of using off-chip memory, several recent accelerators employed embedded non-volatile memory (NVM) such as resistive RAM (RRAM) Li *et al.* (2021); Giordano *et al.* (2021) and magnetic RAM (MRAM) Rossi *et al.* (2021), to store a large amount of weights fully on-chip and reduce the energy consumption for overall memory access. Among the various choice of the NVMs, this dissertation focuses on the RRAM application for the DNN hardware accelerations.

Resistive random-access memory (RRAM) is a specific type of non-volatile memory that relies on the resistive switching behavior of certain materials. RRAM stores data by altering the resistance state of a memory cell, typically by applying electrical pulses. This technology offers several advantages, including high storage density, low power consumption, fast read/write speeds, and compatibility with existing complementary metal-oxide-semiconductor (CMOS) processes. RRAM has the potential to

revolutionize memory storage by providing a scalable and energy-efficient alternative to traditional memory technologiesYang *et al.* (2013). The integration of RRAM with DNNs and IMC can further enhance system performance and energy efficiency by leveraging the non-volatility and fast access times of RRAM.

One step forward in the on-chip memory, in-memory computing (IMC) is an approach that aims to improve computational efficiency by performing data processing and computation tasks directly within the memory subsystem. Traditional computing systems often suffer from the "von Neumann bottleneck" Arikpo *et al.* (2007), where data transfer between memory and processing units becomes a limiting factor. IMC seeks to alleviate this bottleneck by minimizing data movement, which reduces latency and enhances system performance. By storing and processing data in the same physical location, IMC can achieve significant speed-ups for various workloads. NVM technology aligns well with the IMC concepts, with benefits such as high data density, low power consumption, and fast access times, making it an attractive choice for various computing systems.

Despite its potential benefits, RRAM faces several challenges when applied to DNN-related applications. These challenges include:

- Device Variability: RRAM devices exhibit inherent variability due to variations in manufacturing processes and material properties. This variability can lead to inconsistent device performance, affecting the reliability and accuracy of DNN computations. Addressing device variability is crucial for achieving reliable and predictable performance in RRAM-based DNN systems.
- Endurance and Lifetime: RRAM devices have limited endurance, meaning that they can only endure a finite number of write and erase cycles before experiencing degradation. In DNN applications, which involve frequent read

and write operations, endurance can become a critical concern. Increasing the endurance and lifetime of RRAM devices is necessary to ensure the longevity and reliability of DNN systems.

- Energy Consumption: While RRAM is known for its low power consumption during read operations, energy efficiency during write and erase operations remains a challenge. The high energy required for switching resistance states in RRAM can limit its suitability for energy-constrained DNN applications. Reducing the energy consumption of RRAM write and erase operations is essential for enabling efficient and sustainable DNN systems.
- Scalability: RRAM faces scalability challenges in terms of achieving high-density memory arrays. As the size of memory arrays increases, issues such as sneak paths, cross-talk, and resistance drift become more prominent, leading to reduced device performance and reliability. Overcoming scalability limitations is crucial for realizing the full potential of RRAM in large-scale DNN systems.

This dissertation explores the higher-density design of the RRAM IMC array architecture while maintaining high device variability, endurance, energy consumption, and data density.

The key contributions of this thesis to DNN acceleration with non-volatile memory are

- We present circuit-/device-level optimizations to improve the energy and density of RRAM-based in-memory computing architectures.
- We present the experimental results based on prototype chip design of 128×64 RRAM arrays and CMOS peripheral circuits, where RRAM devices are monolithically integrated into the commercial 90nm CMOS technology.

- The optimization methods using the input-splitting scheme report high energy-efficiency and inference accuracy under different DNN models.
- Further, we present an IMC prototype with 2-bit-per-cell RRAM devices for area-/energy-efficient DNN inference.
- Four-level conductance distribution and peripheral circuits with input-splitting scheme have been performed, enabling high DNN accuracy and low area/energy consumption.
- We analyze the power performance and the inference accuracy for different supply voltages for the 2-bit-per-cell RRAM scheme.
- We notice the relaxation effect on the RRAM devices, which may potentially harm the correctness for the IMC outputs over time.
- Two mitigation schemes are proposed to recover the degraded accuracy for the RRAM relaxation: 1) at the circuit level, the reference voltage for RRAM IMC could be calibrated after 80 hours when the relaxation is saturated. 2) At the algorithm level, the weights are trained with lower percentages to be quantized to the two intermediate states. With both schemes applied, the accuracy could be recovered to 87.32% for long-term stability.
- We notice the bottlenecks on RRAM in the IMC training process for the DNN, where the programming inevitably induces high write latency and energy consumption.
- we present the Progressive-wRite In-memory program-VErify (PRIVE) scheme, which we verify with an RRAM testchip for IMC-based hardware acceleration for DNNs. For 5-bit precision DNNs, PRIVE reduces the RRAM programming energy by 1.82×, while maintaining high accuracy of 91.91% (VGG-7) and 71.47% (ResNet-18) on CIFAR-10 and CIFAR-100 datasets.

- Hybrid in-memory computing (HIMC) idea is discussed on RRAM.
- The progress of ferroelectric capacitive devices for in-memory computing is proposed for the next generation AI hardware. A prototype chip is tape-out for peripheral circuit verification, and the characterization of the ferroelectric devices is included in this thesis.

1.2 Thesis Organization

The outline of this thesis is as follows:

- **Chapter 2** presents a monolithically integrated CMOS design under 90nm technology with RRAM arrays. The high-level design logic and circuit-level optimization methods under different DNN models. The results are comprehensively discussed and compared with prior works.
- **Chapter 3** presents an an IMC prototype with 2-bit-per-cell RRAM devices for area-/energy-efficient DNN inference. On top of the previous chapter, this chapter focuses more on multi-level RRAM programming with a similar XNOR-RRAM architecture. The algorithm, optimization topology, and performance analysis are included in this chapter for this work.
- **Chapter 4** presents the characterization and mitigation for the relaxation effect happening on the multi-level RRAM devices.
- **Chapter 5** presents a new algorithm called PRIVE for energy-efficient RRAM programming.
- **Chapter 6** presents the progress on hybrid in-memory computing with non-volatile memory. This chapter includes the high-level idea for hybrid in-memory computing, the modeling idea for the non-volatile memory HIMC for RRAM

and SRAM, and also the progress of one tape-out HIMC chips under 65nm with RRAM and SRAM.

- **Chapter 7** presents the progress on ferroelectric capacitive devices for next-generation AI hardware.
- **Chapter 8** concludes the dissertation.

Chapter 2

MONOLITHICALLY INTEGRATED RRAM AND CMOS BASED IN-MEMORY COMPUTING FOR EFFICIENT DEEP LEARNING

Resistive RAM (RRAM) has been presented as a promising memory technology towards deep neural network (DNN) hardware design, with non-volatility, high density, high on-off ratio, and compatibility with logic process. However, prior RRAM works for DNNs have shown limitations on parallelism for in-memory computing, array efficiency with large peripheral circuits, multi-level analog operation, and demonstration of monolithic integration. In this work, we propose circuit-/device-level optimizations to improve the energy and density of RRAM-based in-memory computing architectures. We report experimental results based on prototype chip design of 128×64 RRAM arrays and CMOS peripheral circuits, where RRAM devices are monolithically integrated in a commercial 90nm CMOS technology. We demonstrate CMOS peripheral circuit optimization using input-splitting scheme and investigate the implication of higher low resistance state on energy-efficiency and robustness. Employing the proposed techniques, we demonstrate RRAM based in-memory computing with up to 78.3 TOPS/W energy-efficiency and 84.2% CIFAR-10 accuracy. Furthermore, we investigate four-level programming with single RRAM device, and report the system-level performance and DNN accuracy results using circuit-level benchmark simulator NeuroSim.

2.1 Introduction

Deep learning algorithms have shown tremendous success in recent years LeCun *et al.* (2015) for various applications including computer vision, speech recognition, language translation, etc. However, an increasing gap exists between the exponential network size growth of state-of-the-art DNNs (e.g. tens of millions of parameters) and the incremental energy-efficiency improvement of conventional memory technologies (e.g. CMOS scaling) for hardware accelerator designs Xu *et al.* (2018).

To bridge this gap and largely improve the memory energy-efficiency, in-memory computing (IMC) has been proposed in recent years across different memory technologies Yin *et al.* (2020a); Valavi *et al.* (2019); Si *et al.* (2019); Seshadri *et al.* (2017); Li *et al.* (2017); Mochida *et al.* (2018); Xue *et al.* (2019). IMC typically asserts multiple or all rows simultaneously to perform multiply-and-accumulate (MAC) computations of DNNs inside the memory, e.g. along the bitlines with analog current/voltage.

SRAM based IMC works Yin *et al.* (2020a); Valavi *et al.* (2019); Si *et al.* (2019) demonstrate high energy-efficiency, however typically such IMC SRAM bitcells include a few additional transistors, which degrades density and leakage. In addition, custom peripheral circuits such as analog-to-digital converters (ADC) incur lower array efficiency. Since one SRAM cell occupies 150-300 F^2 (F is the feature size of a technology node), on-chip SRAMs cannot hold all weights of DNNs. Therefore, CMOS hardware accelerators inevitably involve off-chip DRAMs at the system level, which results in high energy consumption.

Consequently, a number of works proposed to bring computation closer to the DRAM. DRAM based near-memory computing proposes to add logic in the DRAM die, however logic capability in the optimized DRAM process is relatively limited. On

the other hand, DRAM based in-memory computing is more challenging, because the conventional 1T1C DRAM read is destructive, and thus requires additional overheads such as data copy and write back Seshadri *et al.* (2017). DRAM cell designs with non-destructive read have been proposed (e.g. 2T1C, 3T1C) Li *et al.* (2017), but they directly degrade density, which is especially disadvantageous for area-efficient DRAMs.

In addition, both SRAM and DRAM are volatile and have increasing concerns on leakage power in scaled CMOS nodes. To that end, resistive non-volatile memory (NVM) has emerged as a good alternative due to high density, non-volatility, and non-destructive read. Among several well-known candidates including phase change memory (PCM), resistive RAM (RRAM), and magnetic RAM (MRAM), this work focuses on RRAM owing to its high on/off ratio, multi-level programmability, and monolithic integration capability.

There has been only a few works that have demonstrated monolithically integrated RRAM and CMOS for DNN hardware design Mochida *et al.* (2018); Xue *et al.* (2019); Shulaker *et al.* (2017). The authors of Mochida *et al.* (2018) designed 180nm and 40nm prototype chips with embedded RRAM arrays. However, only simple multi-layer percepton (MLP) has been demonstrated that resulted in low inference accuracy of 90.8% for MNIST dataset. An RRAM macro integrated with multi-level sense amplifiers in 55nm CMOS logic process was recently reported in Xue *et al.* (2019), targeting convolutional neural networks (CNNs). However, a relatively low CNN accuracy of 81.83% accuracy for CIFAR-10 dataset was achieved with binary/ternary precision. Moreover, only 9 WLs are asserted simultaneously in the 256×512 sub-array, which limits further parallelism, and a relatively complex 4-bit ADC was employed at the RRAM array periphery, degrading array efficiency and energy consumption.

In Shulaker *et al.* (2017), a monolithically integrated 3D nanosystem has been presented, which connects CMOS transistors, carbon nanotube transistors (CNFET), and RRAM devices in different layers with inter-layer vias (ILVs). A small-scale support vector machine accelerator has been demonstrated, but applicability for larger DNNs has not been shown. While there has been considerable improvement in CNFET integration with CMOS or RRAM, in terms of manufacturability and yield, integration of RRAM with CMOS in commercial technology is much superior Ho *et al.* (2017).

In this work, we address such limitations in RRAM based in-memory computing towards energy-/area-efficient and accurate DNN hardware design, using monolithic integration of RRAM and CMOS. In particular, we investigate three different device/circuit techniques: (1) modulating resistance values for binary RRAM devices, (2) peripheral circuit minimization with input-splitting technique, and (3) multi-level RRAM programming. We report measurement results of 90nm CMOS prototype chip that monolithically integrated RRAM arrays, which executes in-memory computing operations of CNNs for CIFAR-10 dataset.

In our in-memory computing architecture, monolithic integration of RRAM and CMOS is crucial, since we need dense connections to all wordlines and bitlines of the RRAM array. If RRAM and CMOS are not monolithically integrated (e.g. using through-silicon-vias or silicon interposers), the bitline and wordline delays will be excessive and the integration density will be too low. Furthermore, monolithic integration of RRAM with CMOS is simpler and less expensive than that with CNT Shulaker *et al.* (2017). RRAM process is CMOS fabrication compatible, with just a few layers of oxide deposition at the contact via at back-end-of-line (BEOL) compatible temperature. Typically only one additional mask/lithography is required, allowing RRAM integration to be low-cost.

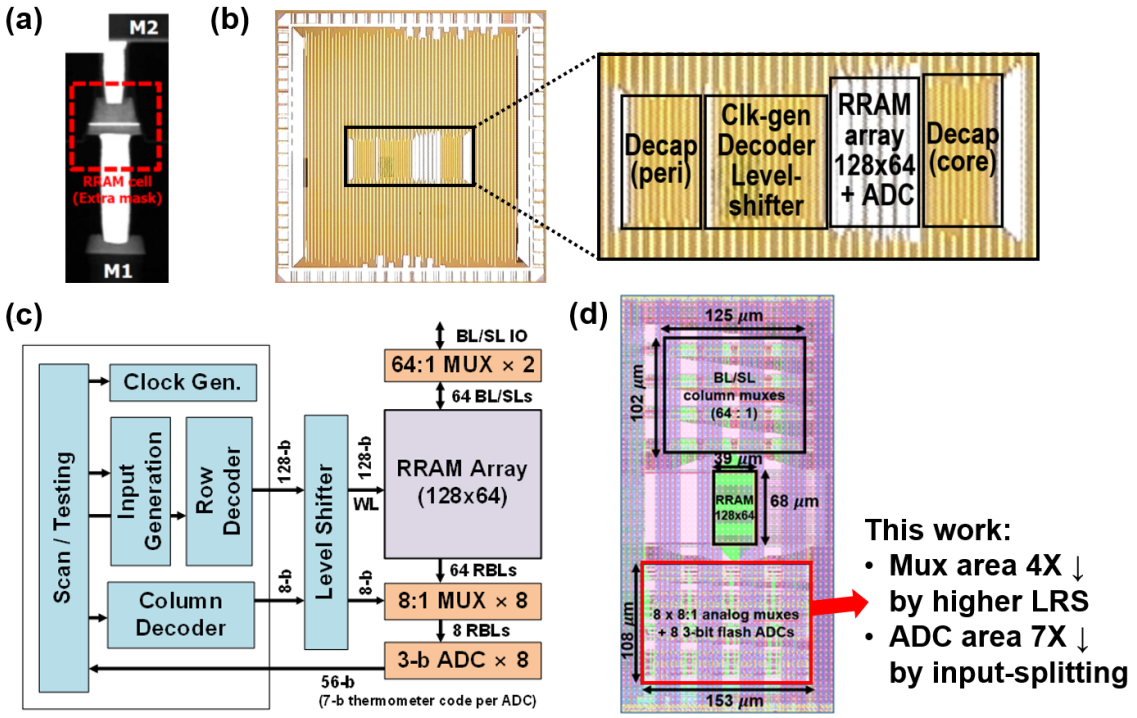
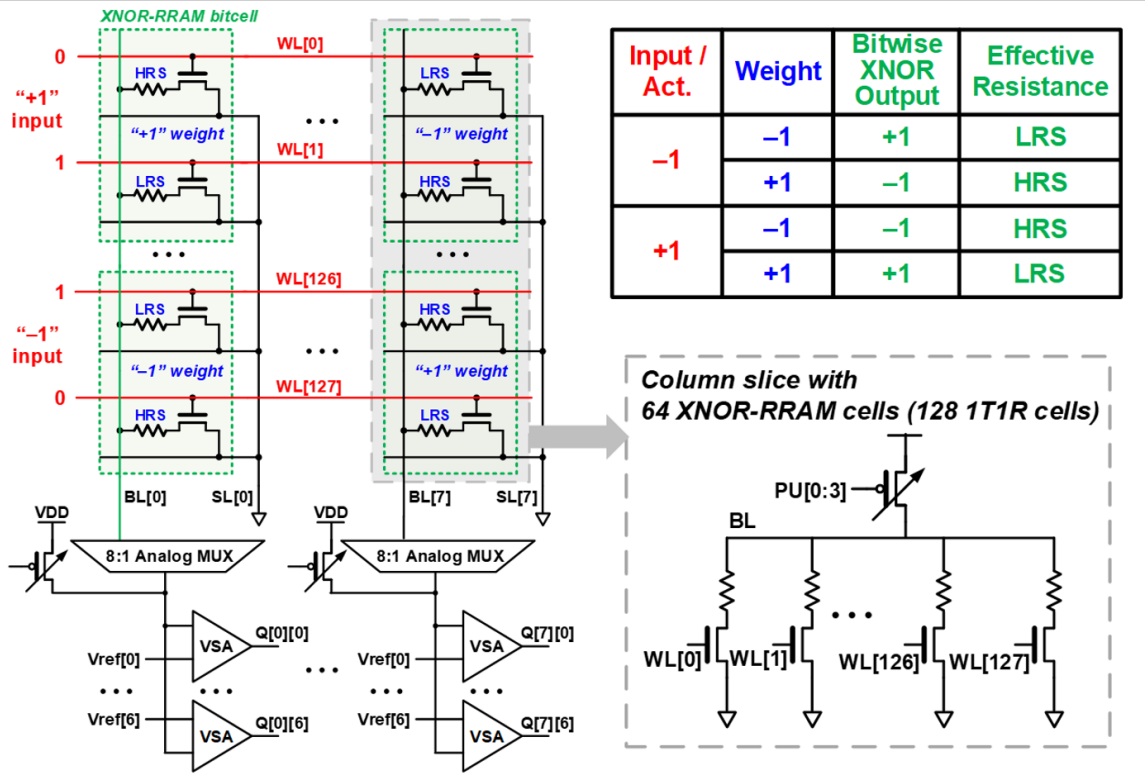


Figure 1. Prototype chip design with monolithically integrated RRAM and 90nm CMOS technology (adapted from Yin *et al.* (2020b), with permission). This work presents further energy/area optimization.

2.2 RRAM Prototype Chip Design

We designed a prototype chip for RRAM-based robust in-memory computing with Winbond's embedded RRAM technology Ho *et al.* (2017), which monolithically integrates 90nm CMOS and RRAM between M1 and M2 (**Figure 1**(a)). Figure 1(b) shows the pad-limited chip micrograph and the core area of the chip. As shown in the top-level block diagram in Figure 1(c), the chip design includes a 128×64 1T1R array, row decoder, level shifter, eight 8-to-1 column multiplexers, eight 3-bit flash ADCs based on seven voltage-mode sense amplifiers (SAs), and two 64-to-1 column decoders for RRAM cell-level programming. The row decoder has two modes of operation: (1) it



Input / Act.	Weight	Bitwise XNOR Output	Effective Resistance
–1	–1	+1	LRS
	+1	–1	HRS
+1	–1	–1	HRS
	+1	+1	LRS

Figure 2. In-memory computing operation of XNOR-RRAM (adapted from Yin *et al.* (2020b), with permission).

asserts all differential wordline (WL) signals simultaneously for binary or low-precision multiply-and-accumulate (MAC) operation, or (2) generates one-hot WL signals for cell-level programming. Eight ADCs (shared among 64 columns) and eight column multiplexers occupy 20% and 12% of the core area, respectively (Figure 1(d)). In this work, further energy/area optimization is investigated including peripheral circuit minimization by using higher LRS and input-splitting scheme.

Conventional binary RRAMs cannot effectively represent the positive and negative weight values (+1 and -1) in binarized neural networks (BNNs) Hubara *et al.* (2016), because the high resistance state (HRS) and low resistance state (LRS) values of binary RRAM devices are both positive. In addition, as shown in **Figure 2**, the

activation/weight value combinations of +1/+1 and -1/-1 should result in the same effective resistance. To that end, we proposed to use a "XNOR-RRAM" bitcell design Sun *et al.* (2018); Yin *et al.* (2020b) for BNNs. As shown in Figure 2, the XNOR-RRAM bitcell involves differential RRAM cells and differential wordlines. The binary activations are mapped onto the differential wordlines, and the binary weights are mapped onto the HRS/LRS values of XNOR-RRAM bitcells. By asserting all differential WLs of the RRAM array simultaneously, all cells in the same column are computed in parallel, which implements the binary MAC computations. The 128×64 1T1R array effectively represents 64×64 XNOR-RRAM bitcells, since one XNOR-RRAM bitcell consists of two 1T1R baseline bitcells to represent positive/negative weights and to perform embedded XNOR computation inside the XNOR-RRAM bitcell.

Both the preliminary simulation results Sun *et al.* (2018) and initial measurement results Yin *et al.* (2020b) of the XNOR-RRAM design only considered the default LRS and HRS values for the binary RRAM devices, and employed a 3-bit ADC at the periphery for digitizing the analog partial MAC value. In this work, we investigate three further optimizations in monolithically integrated RRAM devices and peripheral circuits, towards enhancing the energy-efficiency and density of the RRAM-based IMC systems.

First, since the default LRS value (∼6kΩ) consumes large current and the on/off ratio is relatively high (∼150), we explore using higher LRS values (e.g. ∼12kΩ and ∼24kΩ) to evaluate the trade-off between current reduction, on/off ratio, and CNN accuracy.

Second, although a 3-bit ADC is relatively simple, it still consumes a large area compared to the RRAM array itself, resulting in low array efficiency. We present further

algorithm/hardware improvements beyond the previous input-splitting techniques Kim *et al.* (2018a), and employ binary sense amplifiers with an unified reference voltage across all columns, instead of ADCs at the RRAM array periphery, for digitizing the analog partial MAC values. Considering that tightly-spaced reference voltages make flash ADCs more susceptible to variability at low voltages, we show that the proposed input-splitting scheme actually results in much improved accuracy at lower supplies.

Finally, beyond binary RRAM devices, we investigate four-level programming with the same RRAM devices in our prototype chip, and experimentally validate the density, energy and performance gains by benchmarking a CNN for CIFAR-10 dataset.

2.3 Higher Resistance For LRS Devices

In binary RRAM devices, only two states per device exist, namely LRS (high conductance) and HRS (low conductance). In commercial RRAM technologies that are typically used for storage applications, on-off ratio of higher than 100 has been reported. Having a large on-off ratio is certainly good, but on the other hand, having high conductance value for the LRS leads to high current consumption.

To that end, for a given HRS value is fixed, and if we have higher LRS values in binary RRAM devices, then the current and energy consumption could be largely reduced. On the other hand, compared to the default LRS, targeting LRS to have a higher resistance value can result in wider distribution after programming or more susceptible to non-ideal effects such as read disturb. In addition, and if the LRS and HRS ranges become relatively close, it will adversely affect the DNN accuracy for the RRAM-based IMC hardware.

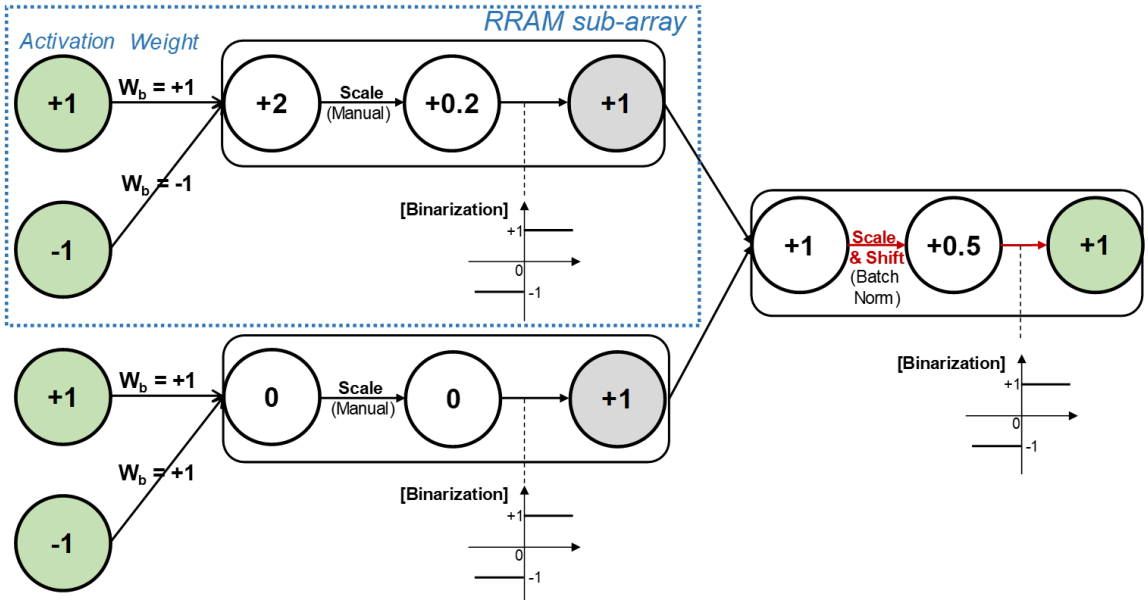


Figure 3. New input-splitting scheme that allows unified reference voltage for all sense amplifiers in the RRAM array periphery.

2.4 Peripheral Circuit Minimization with Input-splitting Scheme

Input-splitting is a method of BNN architecture design for ADC-free in-memory computing Kim *et al.* (2018a). Input-splitting reconstructs a large BNN layer with a network of small layers. It splits input of a large layer so that the number of inputs per split group is less than or equal to row count of the given RRAM array. Each split group constructs a new small layer, and the binary output generated from small layers are accumulated and subsequently binarized with a threshold value of zero. Then, each layer of input-split BNN can fit on RRAM array so that the array can generate binary neuron values as output values. However, batch normalization governs that each neuron has its own threshold value, which necessitates each column to have a digital-to-analog converter (DAC) Valavi *et al.* (2019), adding a large overhead.

In this work, we modified the conventional input-splitting method Kim *et al.*

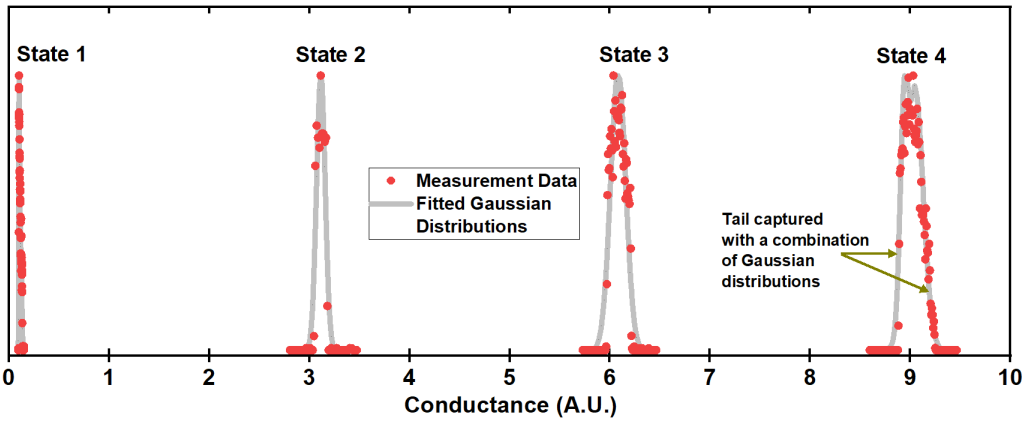


Figure 4. Conductance distribution is shown for four levels of RRAM device programming. Both measurement data from prototype chip and fitted Gaussian distribution curves are shown.

(2018a) to eliminate column-wise threshold values. Batch normalization conducts scaling and shifting operation, and the shifting operation generates threshold values. Therefore, as illustrated in **Figure 3**, we removed batch normalization before output binarization of small layers. Instead, we experimentally found a proper scaling factor for pre-binarization values of small layers. For the RRAM array with 64 rows, we found that, by scaling pre-binarization value with 1/20, most of scaled values lie in the range of [-1, 1]. As there is no shifting operation on pre-binarization value of small layers, the column-wise threshold is fixed to 0. Then, we added batch normalization after the merge to compensate for the loss of batch normalization on small layers.

We tested a VGG-like CNN for CIFAR-10 dataset, which has the network structure of input-128C3-128C3-MP2-256C3-256C3-MP2-512C3-512C3-MP2-1024FC-1024FC-10FC Hubara *et al.* (2016). Here, 128C3-128C3 refers to the convolution layer with 128 input feature maps, 3×3 kernels and 128 output feature maps, MP2 refers to 2x2 max-pooling, and 1024FC refers to the fully-connected layer with 1024 hidden neurons.

As we used RRAM arrays with 64 effective rows, the input counts per input-split

BNN layer was set to 63 for convolution layers and 64 for fully-connected layers. We used 63 for convolution layer because we use 3×3 kernel for convolution, and 63 is the closest value less than equal to 64. In addition, to make the input of convolution layer be divided by 63, we changed the number of channels to be an integer multiple of 7. Using Torch, we trained the input-split BNN with the same training condition used in conventional input splitting Kim *et al.* (2018a). For comparison, we trained baseline BNN (non-split BNN), input-split BNN with column-wise threshold, input-split BNN without column-wise threshold. The algorithm simulation results showed that the input-split BNN without column-wise threshold model has compatible accuracy (86.64%) with the baseline BNN (88.46%) and input-split BNN with column-wise threshold (88.24%).

2.5 Multi-level RRAM Devices

2.5.1 Multi-level Programming scheme

To achieve 2-bit RRAM, two more conductance states are inserted between minimum and maximum conductance levels so that the conductance interval is equal between adjacent states. A write-verify programming scheme is iterated until less than 2% of RRAM cells are outside the target conductance range for each of the four levels. The maximum number of write-verify iterations to program one RRAM cell is specified as N_{max}. For each conductance state, 4,096 RRAM cells in the prototype chip are programmed and measured. It is observed that the conductance distribution becomes more concentrated as N_{max} increases. The N_{max} to achieve the target conductance range are 15, 30, 15 and 10 for the four conductance states, respectively.

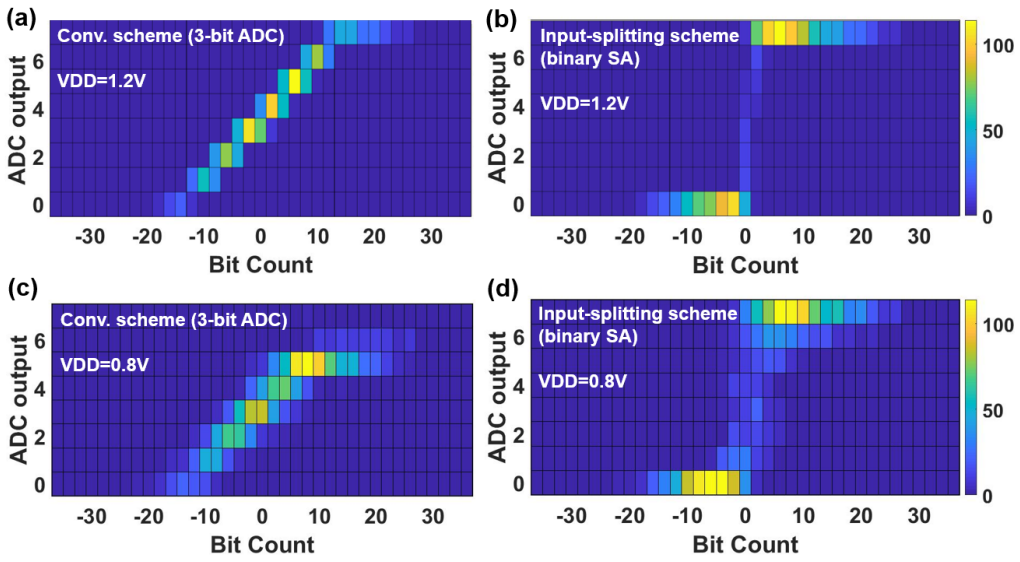


Figure 5. Measured ADC output results compared with bitcount values from BNN algorithm.

After programming, the percentage of the RRAM cells that are outside the target conductance ranges were 0.32%, 1.32%, 0.92% and 0.44%, respectively.

2.5.2 Inference Accuracy Simulation

The inference accuracy for a CNN is simulated with the measured 2-bit RRAM data. However, considering the limited measurement data (4,096 data points for each state) compared to the total number of parameters in a CNN, we first fitted the probability density function (PDF) of the measured conductance data with a linear combination of multiple Gaussians as the fitted PDF. Then, the conductance values are generated with the fitted PDF for a large CNN. **Figure 4** shows the PDF of the measured conductance and the conductance values generated with fitted PDF. The distribution tails of the experiment data are captured with the fitted PDF.

Using 2-bit weights and 4-bit activations, we benchmarked the same VGG-like

CNN for CIFAR-10. It is assumed that each 2-bit weight is stored into one RRAM cell. We first trained the CNN with the quantized training method proposed in Wu *et al.* (2018), and obtained the software baseline accuracy of 91.7%. The 2-bit weights are then mapped to conductance states, where the conductance values of each RRAM cell are generated with the fitted PDFs of the corresponding states. The inference accuracy is simulated for three different array size 64×64, 128×128 and 256×256, where we employed flash ADCs with 5-bit precision using non-linear quantization Sun *et al.* (2018).

2.6 Measurement and Simulation Results

2.6.1 Binary RRAM based IMC Energy and Accuracy Characterization with Higher LRS and Input-splitting

We envision that large binary CNNs are mapped onto multiple RRAM arrays, where weights for different input channels are stored on different rows, weights for different output channels are stored on different columns, and weights within each convolution kernel (e.g. 9=3×3) are stored in different RRAM macros Yin *et al.* (2020a,b). Subsequently, the partial MAC results from different RRAM macros are accumulated via digital simulation. In Yin *et al.* (2020b), 3-bit ADC was used to digitize the analog partial MAC values, where seven reference voltages for each flash ADC required offset cancellation in order to achieve >83% CIFAR-10 accuracy. The new input-splitting scheme presented in this work substantially reduces such calibration overhead, since we only need binary sense amplifiers to digitize the analog

partial sum, and the same reference voltages are used for all 64 columns of the RRAM array.

Another important challenge for the flash ADC is that, the adjacent reference voltages are very close to each other, especially since the partial sum data distribution is concentrated near zero Sun *et al.* (2018). If we lower the supply voltage, the reference voltages actually become even closer to each other, which makes it more susceptible to process variation. On the other hand, since the input-splitting scheme allows to have only one reference voltage for the sense amplifiers, the digitization is inherently more robust to variability and noise.

We performed chip measurements for the experiments of higher LRS values and the input-splitting scheme. For the higher LRS experiment, we programmed RRAM devices with different target LRS values of 6kΩ, 12kΩ and 24kΩ. For the input-splitting scheme experiment, with the same XNOR-RRAM prototype chip, we only use one SA out of the seven SAs that are present in the flash ADC. This means that, when we employ the input-splitting scheme, the overall ADC area in the RRAM macro is effectively reduced by 7X.

In **Figure 5**, we show the comparison of the bitcount values from the BNN algorithm (i.e. ideal partial sum values) and the measured ADC output values using 12kΩ LRS target, for both the conventional scheme with 3-bit ADCs and the input-splitting scheme with binary SAs. As we compare the 1.2V and 0.8V supply results, it can be seen that the ADC output values become less accurate at 0.8V. However, with a single reference level, the input-splitting scheme still maintains more robust operation even at lower voltages.

As shown in **Figure 6**, the energy-efficiency (TOPS/W) of RRAM-based IMC

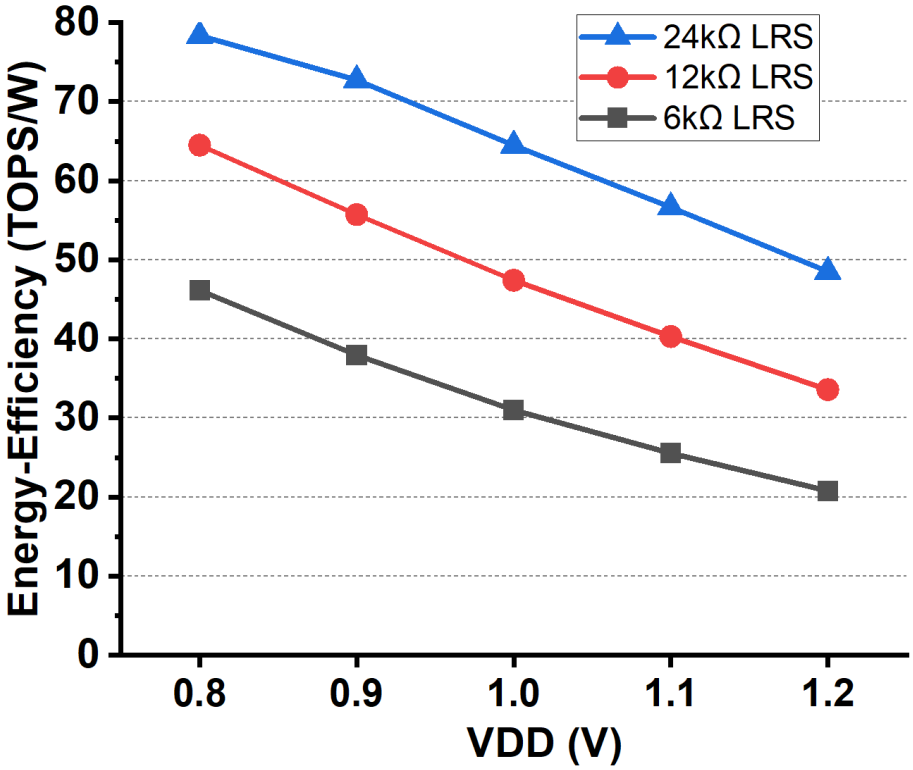


Figure 6. Energy-efficiency with voltage scaling for RRAM IMC with different LRS values.

increases with higher LRS values and with lower supply voltages. The CIFAR-10 accuracy values for the VGG-like CNN with voltage scaling are reported in **Figure 7** with different LRS values for both the input-splitting scheme with binary SAs and the conventional scheme with 3-bit ADCs. For the conventional scheme with ADCs, it can be seen that the CIFAR-10 accuracy degrades by a large amount when the supply voltage scales below the nominal 1.2V. This is due to the fact that that the seven reference voltages for the flash ADC are separated only by a small voltage value, which aggravates with lower supply voltages, incurs more ADC output errors, and adversely affects the DNN accuracy.

For the input-splitting scheme, there is only one reference voltage used by all eight sense amplifiers for 64 columns, the SA operation is more robust against voltage

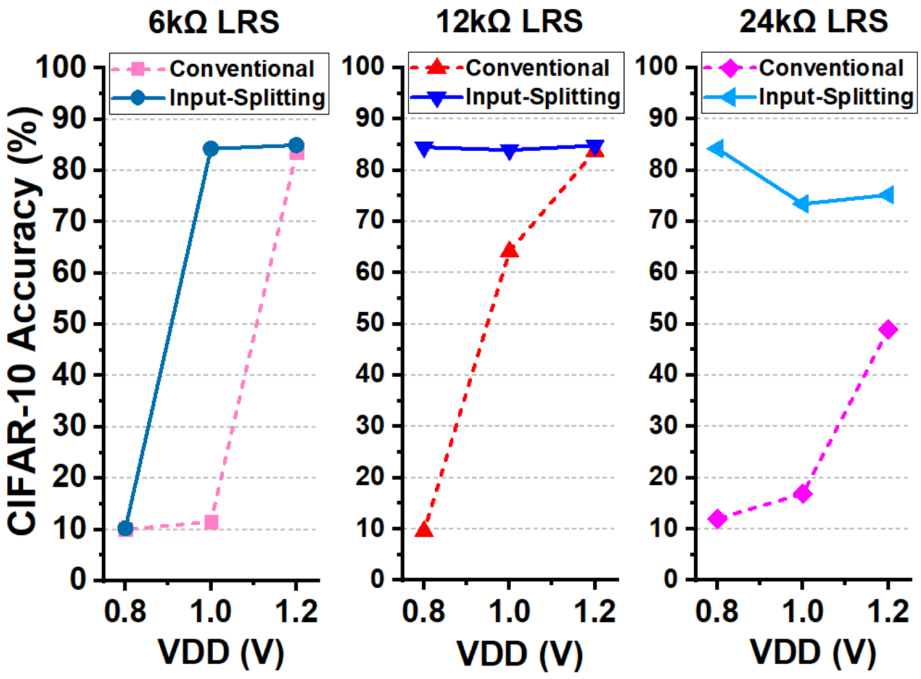


Figure 7. CIFAR-10 accuracy with voltage scaling for RRAM IMC with different LRS values and input-splitting scheme.

scaling, noise and variability. As a result, Figure 7 shows that high CNN accuracy is maintained for the input-splitting scheme for 12kΩ/24kΩ LRS values, down to 0.8V supply. The input-splitting scheme also shows higher accuracy for cases when RBL voltage is around 0.6-0.7V (high gain region for SA with differential NMOS input) for bitcount values near zero, e.g. higher supply with 6kΩ LRS and lower supply with 24kΩ LRS value.

The conventional scheme Yin *et al.* (2020b) achieves energy-efficiency of 20.8 TOPS/W at 1.2V supply (Figure 6), while achieving 83.5% accuracy with binary CNN. Jointly optimizing the use of higher LRS value of 12kΩ (24kΩ) and the proposed input-splitting scheme effectively enabled voltage scaling down to 0.8V without any additional accuracy loss, improving the energy-efficiency by 3.1X (3.8X), achieving 64.5 (78.3) TOPS/W.

Table 1. CNN simulation results with 2-bit RRAM for different RRAM array sizes.

RRAM array size	64×64	128×128	256×256
CIFAR-10 accuracy	91.2% [1] (91.4%) [2]	89.2% [1] (89.3%) [2]	79.0% [1] (81.2%) [2]
Chip area (mm^2)	19.64	19.21	14.71
Read dynamic energy (layer-by-layer, μJ)	33.73	32.50	55.46
Leakage energy (μJ)	1.78	0.77	0.81
Latency (ms)	8.90	6.11	12.93
Energy efficiency (TOPS/W)	17.35	18.52	10.95
Throughput (FPS)	112.38	163.72	77.36

[1] accuracy with ADC quantization and RRAM conductance variation
[2] accuracy with ADC quantization (without RRAM conductance variation)

2.6.2 2-bit RRAM Based CNN Accelerator Performance Benchmarking with NeuroSim

2-bit RRAM could further increase the integration density for the CNN accelerator. The performance benchmarking for 2-bit RRAM based CNN accelerator is conducted in NeuroSim Chen *et al.* (2018), where the aforementioned VGG-like CNN is utilized. We assume that eight columns share one ADC in the RRAM array, and there are a total of eight ADCs in the RRAM array periphery. The inference computation is processed layer by layer. **Table 1** presents the benchmarking results with different RRAM array sizes.

First, the inference accuracy drops as the array size is increased, since the ADC

precision is fixed at 5-bit. This is attributed to the fact that the partial sum distribution becomes broader with larger array size, and therefore quantization loss is increased. It should be noted that the conductance variation of 2-bit RRAM only leads to small accuracy drop when comparing the accuracy with ADC quantization only and with both ADC quantization and RRAM conductance variation.

In terms of chip area, 256×256 array shows smaller chip area compared with 128×128 array due to the increased array efficiency. However, only small chip area increase is observed when array size is reduced to 64×64. Comparing with 128×128 array, for 256×256 array, chip area is reduced as less subarrays are needed. It can be explained by the fact that in 64×64 array, the periphery circuit size is reduced due to lower maximum column partial sum current, therefore, the array efficiency does not drop significantly compared with 128×128 array.

For the read latency and dynamic energy consumption, comparing with 128×128 array, 64×64 array needs more partial sum accumulations between subarrays, which leads to higher latency and energy consumption. For 256×256 subarray, the large column current leads to significantly higher ADC energy consumption and therefore the overall energy consumption is increased. Besides, the larger column partial sum current leads to larger transmission gate (TG) size in the multiplexer, which induces higher latency for the decoder to drive the TG gate capacitor.

2.7 Conclusion

In this work, we demonstrated RRAM based in-memory computing with 90nm CMOS prototype chips that monolithically integrated RRAM and CMOS in different vertical layers. Using device-/circuit-/algorithm-level techniques, both the energy-

efficiency and density of binary RRAM based IMC hardware improved substantially, achieving up to 78.3 TOPS/W and 84.2% accuracy for CIFAR-10 dataset. Experiments with 2-bit RRAM demonstrate sufficient separation between four conductance levels, and show higher CNN accuracy up to 128×128 RRAM array size.

Chapter 3

2-BIT-PER-CELL RRAM BASED IN-MEMORY COMPUTING FOR AREA-/ENERGY-EFFICIENT DEEP LEARNING

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.

3.1 Introduction

With exponential growth in the sizes of deep/convolutional neural networks (DNN/CNN), demands for highly dense and energy-efficient memory devices have skyrocketedXu *et al.* (2018). Compared to CMOS memory technologies such 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 Verma *et al.* (2019). By asserting multiple or all rows simultaneously, analog computations of multiply-and-accumulate (MAC) operations are performed inside the memory (e.g. along the bitline), substantially reducing memory access energy and latency of row-by-row operation.

Several RRAM based in-memory computing prototypes have been presented, but most of them only employed single-level cell design Xue *et al.* (2019); Yan *et al.* (2019); Yin *et al.* (2019). The device-level programming of 2-bit/3-bit per RRAM cell has been reported but was limited to row-by-row read-out Le *et al.* (2018); Hsieh *et al.* (2019) or only simulation of multi-row read-out Yin *et al.* (2019). Recently, Liu *et al.* (2020) reported IMC with four-level RRAM, 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 87% CNN accuracy.

3.2 In-memory Computing RRAM Macro Design

3.2.1 In-Memory Computing Design with Four-Level RRAM Devices

Our current 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. 8(a), we use two vertically-adjacent cells and differential wordlines 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), $G_{HIGH}\times 1/3$, $G_{HIGH}\times 2/3$, and G_{HIGH} (lowest resistance state), and we program the two 1T1R cells differentially as [G_{LOW} and G_{HIGH}] or [$G_{HIGH}\times 1/3$ and $G_{HIGH}\times 2/3$], as shown in Fig. 8(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}, $G_{HIGH}\times 1/3$, $G_{HIGH}\times 2/3$, and G_{HIGH} conductance, respectively (Fig. 8(b)).

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. 9. The PMOS header is digitally configurable in 16 different strength values. Through 8:1 column multiplexers, RBL is connected to a group of SAs, which can be served collectively as a flash ADC or as individual SAs (Fig. 8(c)). One group of SAs is shared

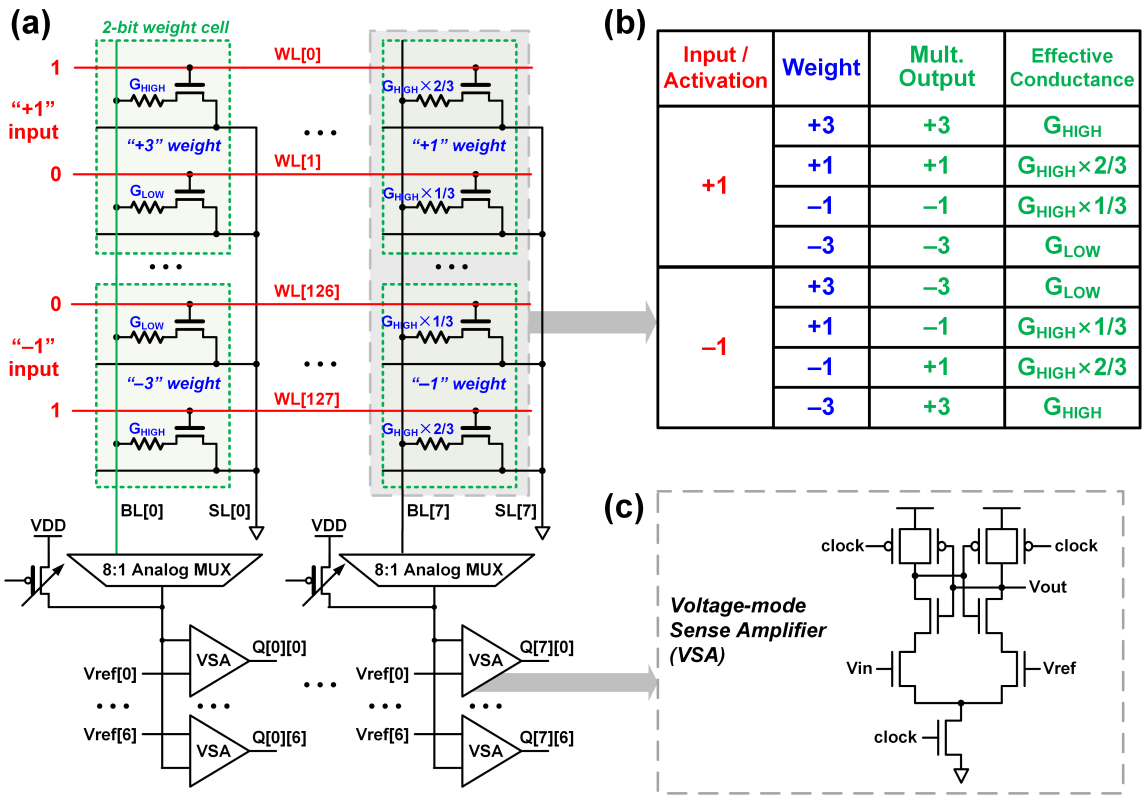


Figure 8. In-memory computing operation with four-level RRAM devices.

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.

3.2.2 Four-Level RRAM Programming

To achieve 2-bit RRAM, two intermediate conductance levels are inserted between 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 (VG) 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 VG of 4.0 V is applied, followed with

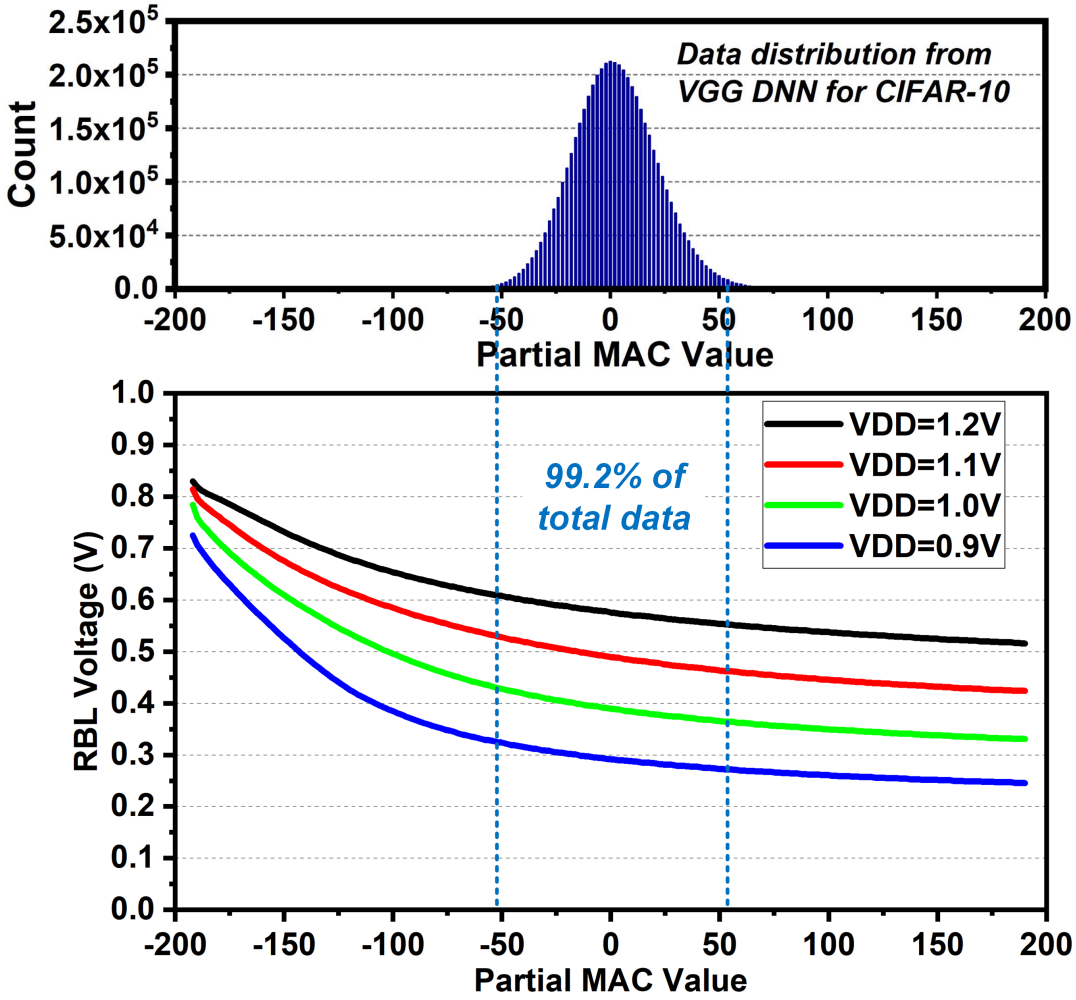


Figure 9. (Top) Partial MAC data distribution. (Bottom) Simulated transfer curve of the RBL voltage.

a SET pulse with a new VG lower by a 'lower voltage step' (lower ΔV in Table 2). If the resistance after SET is higher than the upper bound, a RESET pulse is applied, followed with a SET pulse with a new VG higher by an 'upper voltage step' (upper ΔV in Table I). After 15 write-verify iterations, if the resistance is still outside of the lower/upper bound, we set the lower/upper voltage step to 0.01V for finer adjustment.

Fig. 10 shows the initial four-level programming results and distribution of RRAM cells over time. While minimum and maximum conductance levels maintain tight distributions over 108 hours, the two intermediate conductance levels show moderate

Table 2. RRAM Programming Setup for Different Conductance Levels.

Target Conductance	Initial V0	Lower ΔV	Upper ΔV
G_{HIGH}	3.25V	0.10V	0.01V
$G_{HIGH} \times 2/3$	1.65V	0.07V	0.03V
$G_{HIGH} \times 1/3$	1.4V	0.05V	0.03V

relaxation over time. In particular, $G_{HIGH} \times 1/3$ 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. 10. We also performed transistor-level simulation of eight columns with randomly selected resistances from Fig. 10 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 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 3.2.

3.2.3 Column Sensing Optimization with Input-Splitting

In previous in-RRAM computing works Xue *et al.* (2019), it has been shown that ADCs pose critical challenges for area and energy. Input-splitting was proposed originally as a method to reduce the overhead of ADCs in in-memory computing DNN

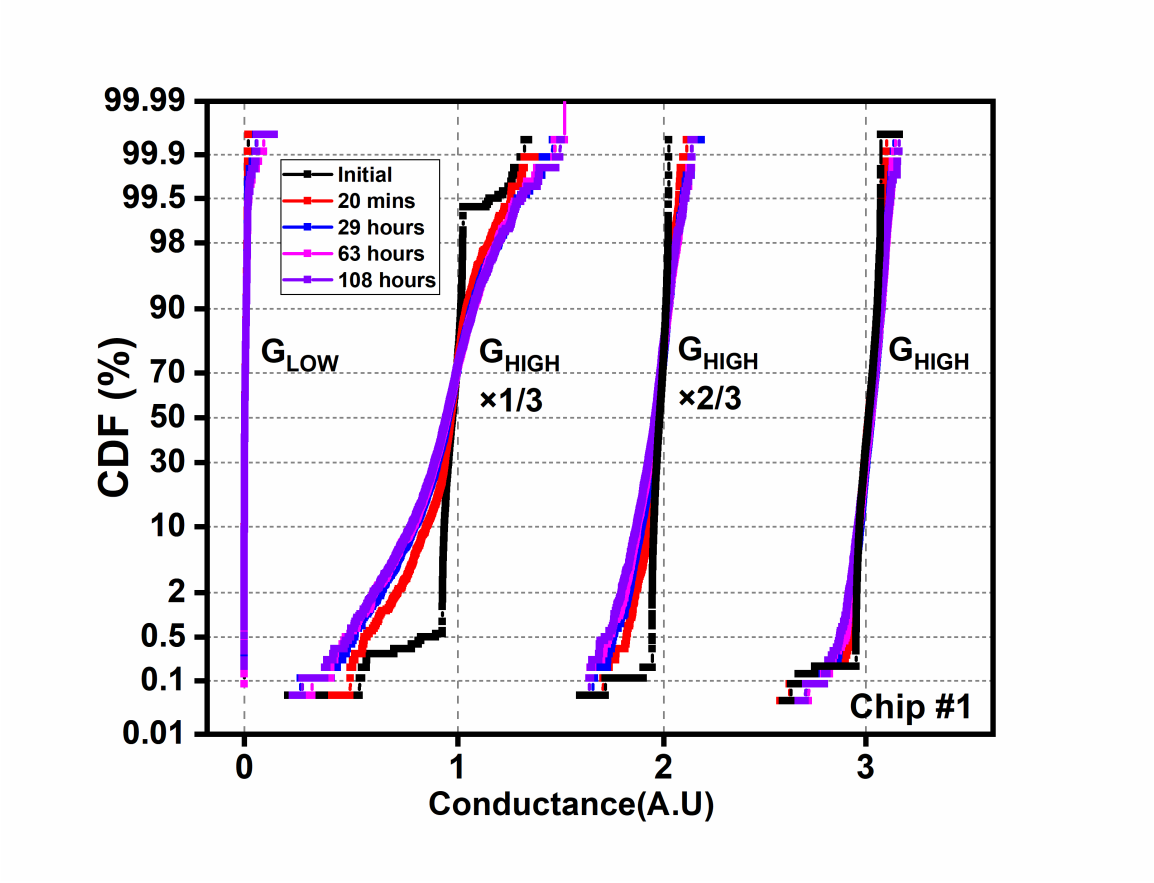


Figure 10. Measured four-level conductance distribution over 108 hours.

accelerators Kim *et al.* (2018b), and has been applied to binary RRAM arrays in Yin *et al.* (2019).

In this work, we re-designed the input-splitting algorithm to support 2-bit weights and 1-bit activations, as illustrated in Fig. 12. It splits input of a large layer so that the number of inputs per split group is less than or equal to row count of the RRAM array. Each split group constructs a new small layer, and the binary output generated from small layers (e.g. RRAM arrays) are accumulated and subsequently binarized to +1 or -1 with a threshold value of zero. Our new input- splitting algorithm trains DNNs so that inter-neuron binarization is achieved without batch normalization, which allows using identical reference voltages (V_{ref}) for all SAs for 64 columns.

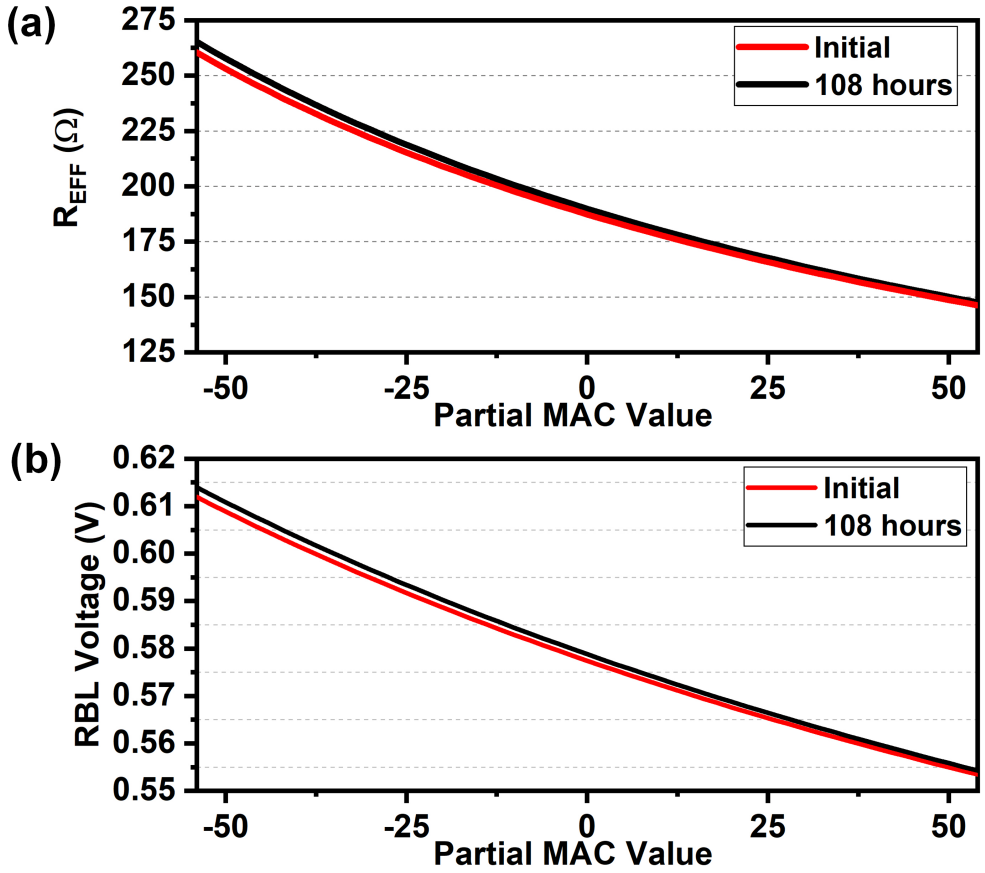


Figure 11. (a) Effective resistance (R_{EFF}) and (b) V_{RBL} change between initial programming and 108 hours.

3.3 Measurement Results and DNN Evaluation

The prototype chip (Fig. 13(a)) was fabricated in 90nm CMOS that monolithically integrates RRAM between M1 and M2 [10]. 128×64 RRAM array is integrated with the peripheral circuits including eight groups of SAs (one group of SAs shared among eight columns), 8:1 column multiplexers, and row decoders/drivers that include level-shifters (for high-voltage programming). We performed measurement of two chips at room temperature. Fig. 13(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

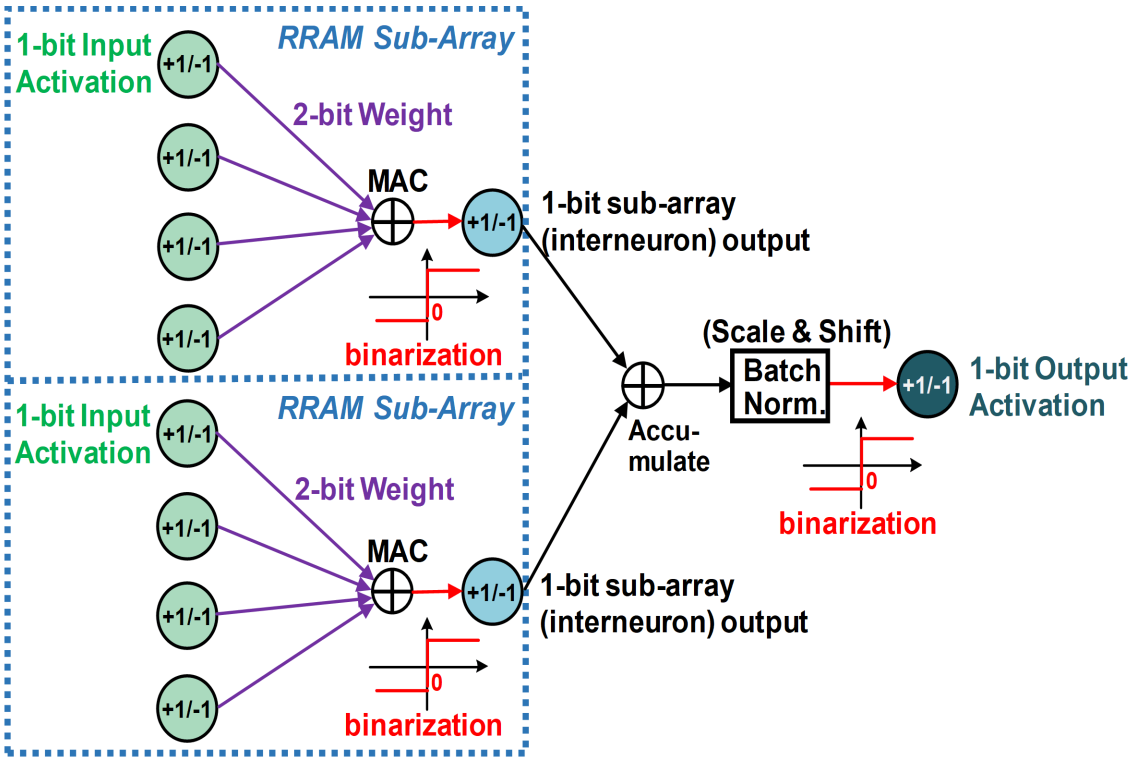


Figure 12. Input-splitting scheme with 2-bit weights. DNNs are trained so that RRAM array outputs are binarized.

from the chip with separate power supplies. With the resistive divider by PMOS header and RRAM pull-down paths dissipating static current, RRAM array dominates the power consumption.

3.3.1 IMC Measurement Results from RRAM Array

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 SAs with identical Vref to vote majority and obtain the binary output for the interneuron. We first programmed the RRAM with a 64×64 weight submatrix from the trained DNNs with 2-bit weights using

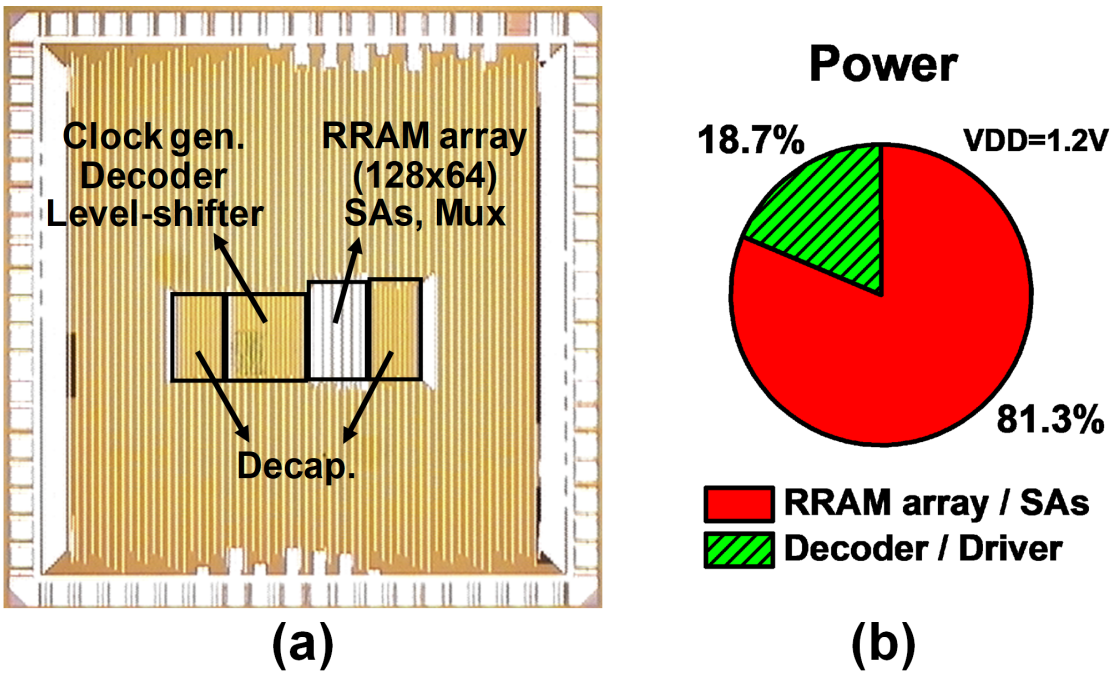


Figure 13. (a) Die photo of prototype chip. (b) Power breakdown of chip 1.

write-verify scheme described in Section 3.2.2. 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. 14(a). The sum of seven SAs' output needs to be binarized to either +1 or -1 as the interneuron output. From the results in Fig. 14(a), we obtain the conditional probability for each MAC value being binarized to +1, as shown in Fig. 14(b).

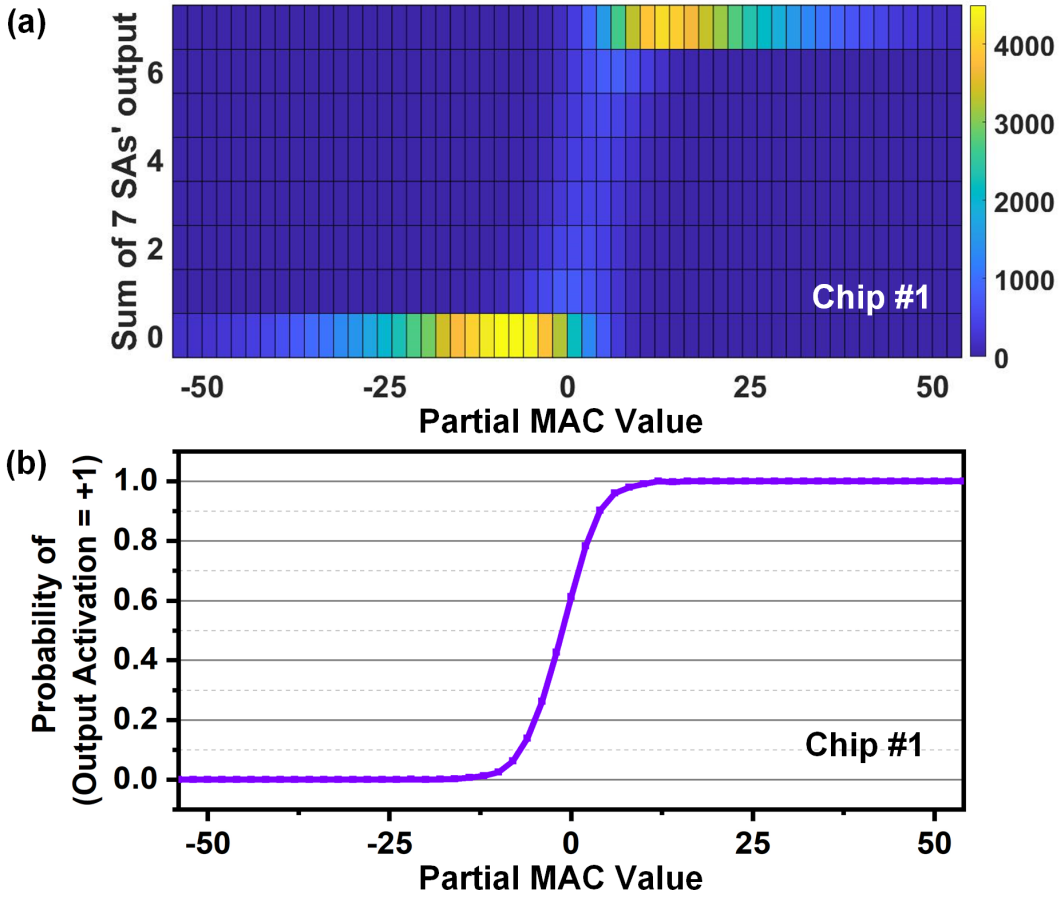


Figure 14. (a) 2-D histogram of the partial sum and the measured SA output at time=108 hours. (b) Probability of interneuron output for partial MAC values.

3.3.2 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. 14(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

Table 3. 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
** nCm: convolutional layer with n channels and m×m kernel, B: batch normalization layer, M: max-pooling (2×2), FC: fully-connected layer*	

DNNs (heavy-VGG, light-VGG, AlexNet-like CNN) for CIFAR-10 dataset (Table 3). 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.

3.3.3 Energy, Performance, and Accuracy Characterization

Fig. 15 shows that the input-splitting algorithm and corresponding measurements incur minimal accuracy degradation for all three DNNs, and also the two chips that we measured show 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. 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 SAs with identical Vref to vote majority and obtain the binary output for the interneuron.

While the results in Fig. 15 used all seven SAs in the prototype chip, we experimented using small number of SAs. Fig. 16 shows that, the best SA combination outputs show similar CIFAR-10 accuracy compared to the voting results of seven SAs.

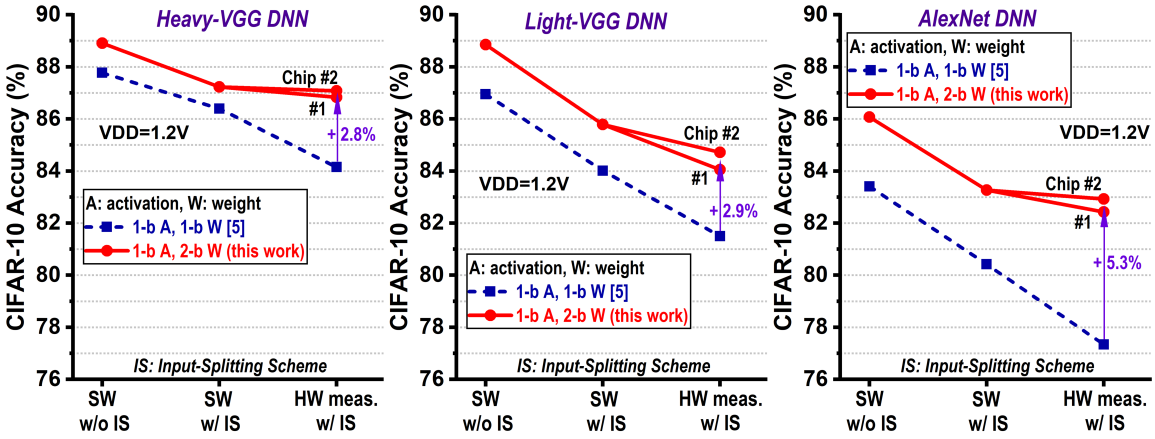


Figure 15. Accuracy (software vs. measurements) of three DNNs for 1-bit/2-bit weights and without/with input-splitting.

On average, using more SAs with voting results in improved CIFAR-10 accuracy, due to less variability from hardware.

With dynamic voltage scaling (Fig. 17(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. 17(b).

To assess the robustness of IMC over time amidst RRAM relaxation (Fig. 10), we characterized the Heavy-VGG CNN accuracy over 108 hours, as shown in Table 4. Similar relaxation in conductance has been reported in prior works Wang *et al.* (2015). 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 4). Table 5 shows the comparison with prior in-RRAM computing works. Our work is the first to demonstrate 2-bit-per-

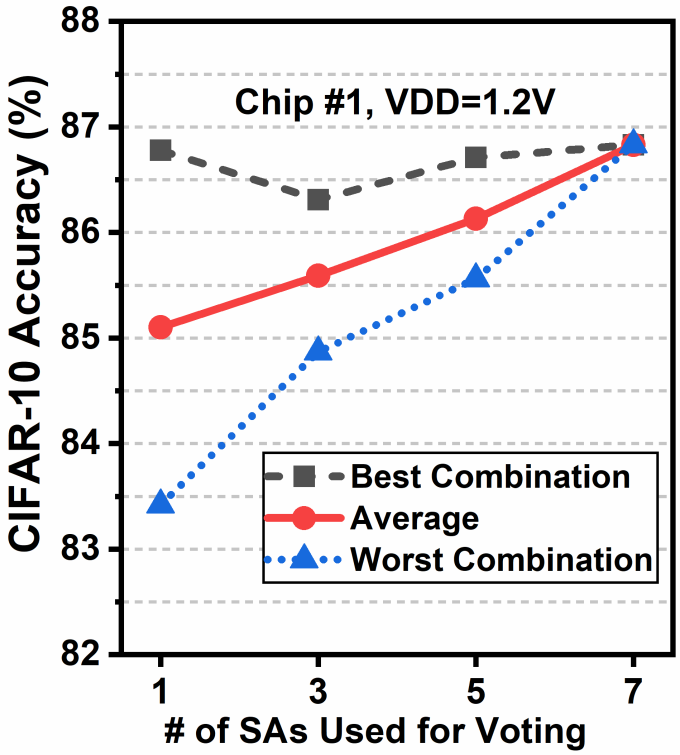


Figure 16. Number of SAs vs. accuracy.

Table 4. 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

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 Xue *et al.* (2019).

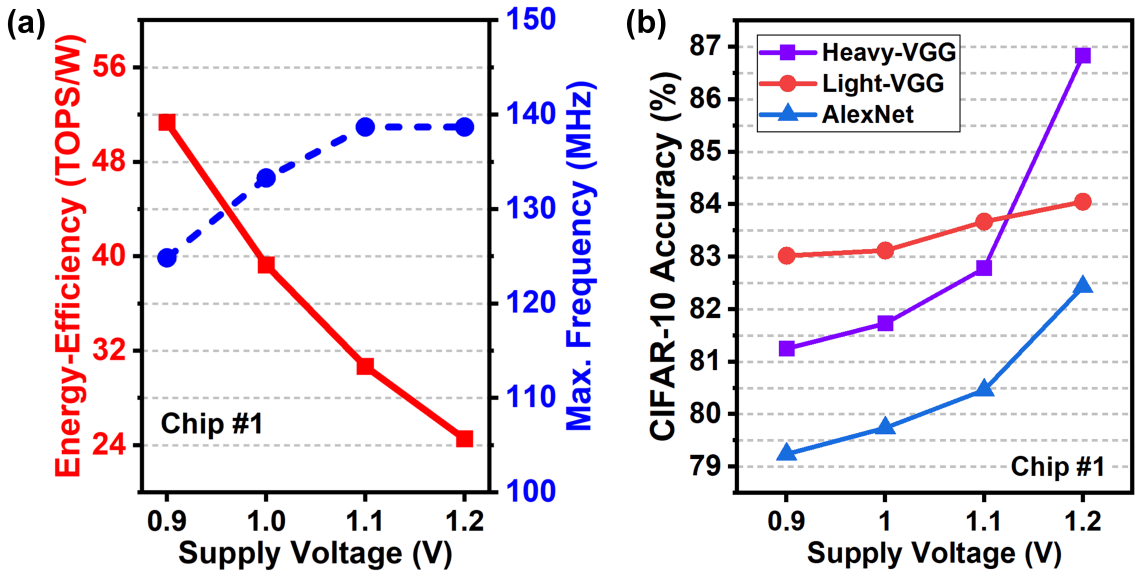


Figure 17. (a) Measured energy/frequency results with voltage scaling. (b) Accuracy of three DNNs with voltage scaling.

Table 5. Comparison with prior works on RRAM-based in-memory computing demonstrated on CNNs for CIFAR-10.

	Xue *et al.* (2019)	Yan *et al.* (2019)	Hsieh *et al.* (2019)	**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 turn 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	**6579(14X↑)**
CIFAR-10 Accuracy	81.8-88.5%	~80%	83.0%	**83.0-87.1%**

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

3.4 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.

Chapter 4

CHARACTERIZATION AND MITIGATION OF RELAXATION EFFECTS ON MULTI-LEVEL RRAM BASED IN-MEMORY COMPUTING

In this chapter, we investigate the relaxation effects on multi-level resistive random access memory (RRAM) based in-memory computing (IMC) for deep neural network (DNN) inference. We characterized 2-bit-per-cell RRAM IMC prototypes and measured the relaxation effects over 100 hours on multiple 8 kb test chips, where the relaxation is found to be most severe in the two intermediate states. We incorporated the experimental data into SPICE simulation and software DNN inference, showing DNN accuracy for CIFAR-10 dataset could degrade from 87.35% to 11.58% after 144 hours. To recover the largely degraded accuracy, mitigation schemes are proposed: 1) at the circuit level, the reference voltage for RRAM IMC could be calibrated after 80 hours when the relaxation is saturated. 2) At the algorithm level, the weights are trained with lower percentages to be quantized to the two intermediate states. With both schemes applied, the accuracy could be recovered to 87.32% for long-term stability.

4.1 Introduction

DNNs have been successful in many computer vision and speech recognition applications. While state-of-the-art DNN algorithms continue to achieve higher accuracy with less number of parameters, the most compact models still require >3 million weights to achieve >70% top-1 ImageNet accuracy Deng *et al.* (2020). This leads to an insatiable demand for high-density memories such as multi-level RRAM.

On the other hand, the DNN computations are dominated by multiply-and-accumulate (MAC) operations, but the overall energy consumption of DNN inference hardware has been dominated by memory access and data communication Zimmer *et al.* (2020), due to the separation of conventional memories with row-by-row access and dedicated MAC engines. To improve the energy-efficiency of DNN inference, in-memory computing (IMC) has emerged as a promising technique, which turns on multiple rows and performs analog MAC computations along the bitline inside the memory.

Recent array-level demonstrations have presented RRAM's potential towards IMC for area-/energy-efficient DNN inference Yin *et al.* (2020c); Xue *et al.* (2020); Wan *et al.* (2020); Hsieh *et al.* (2019); Liu *et al.* (2020), but most RRAM based IMC prototypes today feature only single-level cell design Yin *et al.* (2020c); Xue *et al.* (2020); Wan *et al.* (2020). Device-level programming of multi-level RRAM has been reported but was limited to row-by-row read-out Hsieh *et al.* (2019). Only a few works reported IMC with four-level RRAM devices Liu *et al.* (2020); He *et al.* (2020), while Liu *et al.* (2020) only demonstrated a simple two-layer multi-layer perceptron for a low 94.4% accuracy for MNIST dataset. More importantly, most of the prior prototype designs just reported the basic functionality of IMC, while the reliability aspect of RRAM at array-level and during actual IMC operations is largely unexplored, although it can considerably affect the DNN inference accuracy.

Relaxation occurs as a rapid drift of conductance right after initial programming but gradually saturates in the long term. For HfO_2 RRAM, its relaxation effects at device-level were reported in Wan *et al.* (2020); Wang *et al.* (2015), and read disturb induced RRAM conductance drift behavior was investigated in Shim *et al.* (2020a). In our prior work He *et al.* (2020), to maintain relatively stable DNN inference accuracy

with RRAM relaxation effects over time, the peripheral circuits (e.g. reference voltages to the sense amplifiers) needed to be recalibrated every once in a while.

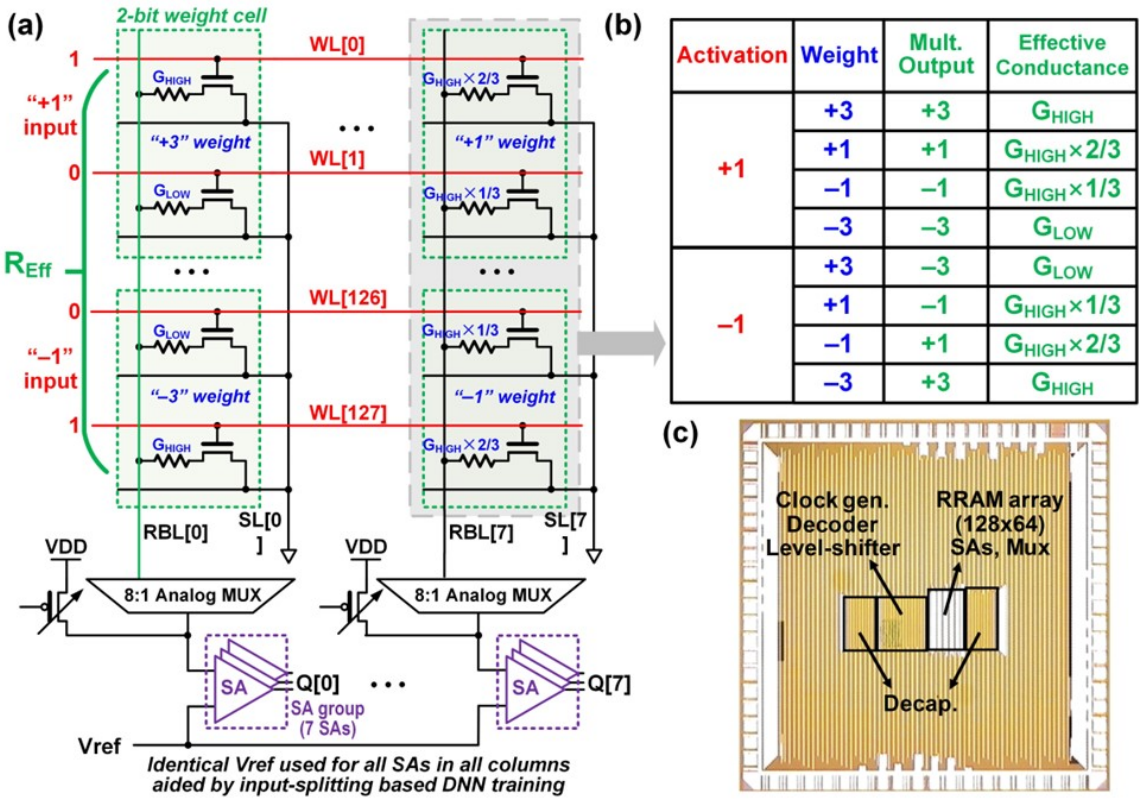
In this work, we comprehensively characterized the relaxation behavior with and without IMC operations on array-level with multiple 8 kb test chips He *et al.* (2020), and analyzed its impact on DNN inference accuracy over time. The experiments are based on relaxation measurements of 2-bit-per-cell HfO_2 RRAM cells over 1,047 hours (over one month) accumulatively collected from three test chips, which were designed for RRAM based IMC with CMOS peripheral circuits. We present two mitigation schemes to recover the degraded DNN accuracy due to the relaxation effects. First, at the circuit-level, we calibrate the reference voltages after the relaxation saturates. Second, at the algorithm-level, we present a relaxation-aware DNN training technique to maintain high accuracy over time without frequent peripheral circuit calibration.

4.2 RRAM IMC Bitcell and Chip Design

4.2.1 2-bit-per-cell RRAM IMC

As shown in Fig. 18(a), we use two vertically-adjacent cells and differential wordlines (WLs) to represent one 2-bit weight [8]. Four conductance values with equidistant conductance levels from G_{HIGH} to G_{LOW} correspond to +3, +1, -1 and -3 weight values. As shown in Fig. 18(b), the element-wise multiplication between binary activation (+1, -1) and four-level weights results in four different pull-down strengths governed by the effective conductance, corresponding to four MAC partial results of +3, +1, -1, and -3.

Our RRAM macro design exhibits a 128×64 array, and with the vertically differ-



Activation	Weight	Mult. Output	Effective Conductance
+1	+3	+3	G_{HIGH}
	+1	+1	$G_{HIGH}\times 2/3$
	–1	–1	$G_{HIGH}\times 1/3$
	–3	–3	G_{LOW}
–1	+3	–3	G_{LOW}
	+1	–1	$G_{HIGH}\times 1/3$
	–1	+1	$G_{HIGH}\times 2/3$
	–3	+3	G_{HIGH}

Figure 18. (a) 2-bit-per-cell RRAM schematic; (b) Conductance representation of multiplication with 2-bit weights; (c) Chip micrograph. Adapted from He *et al.* (2020).

ential cell structure, each column stores 64 distinct weights. With all cells in the same column conducting in parallel, the sum of multiplication or MAC computation of 64 inputs will be between -192 to +192. Each possible MAC value is represented by the RBL voltage (V_{RBL}), which is formed by the resistive divider between a controllable PMOS header and the parallel RRAM cells plus the analog multiplexer.

4.2.2 RRAM IMC Macro Periphery and Chip Design

V_{RBL} is compared with a reference voltage (V_{ref}) using voltage-mode sense amplifiers (SAs). One SA group consists of seven SAs, which can work in two different

modes. With seven different V_{ref} voltages, the seven SAs can operate as a 3-bit (8-level) flash ADC Yin *et al.* (2020c). Alternatively, we can use the seven SAs with identical V_{ref} to vote majority and obtain the binary output for the input-splitting algorithm He *et al.* (2020). For higher array-efficiency and density, each SA group is shared among every eight columns of RRAM array. The SAs convert the analog V_{RBL} voltage that represents the partial MAC results into digital values, which will be further analyzed for the DNN inference accuracy.

The RRAM macro can operate in two modes. First, the row decoder generates one-hot WL signals for cell-level RRAM programming and resistance read-out. Second, the row decoder asserts all differential wordline (WL) signals of the 128×64 simultaneously for IMC operation and performs the partial MAC computation.

The prototype chips (Fig. 18(c)) were fabricated in 90nm CMOS technology Ho *et al.* (2017) that monolithically integrates 128×64 HfO_2 RRAM array (between M1 and M2) with SAs, column multiplexers, clock generator, row/column decoder, level-shifters, decoupling capacitors, etc.

4.3 Experiment Results

4.3.1 Relaxation and Experiment Setup

We measured the relaxation effects of four-level RRAM device/array over time for different operating conditions across three different chips. During RRAM programming, we tighten the conductance distribution using a write-verify programming protocol He *et al.* (2020) so that the initial conductance is within 5% of the target state that ideally maps the four weight values.

Table 6. Relaxation setup information for eight experiments.

Experiment #	**Chip #**	**Total Hours**	**# of array-level IMC executed during total hour**
A1	**#1**	**94**	**0**
A2	**#2**	**94**	**0**
B1	**#3**	**108**	**7**
B2	**#1**	**144**	**10**
B3	**#1**	**156**	**9**
B4	**#2**	**144**	**9**
B5	**#1**	**154**	**49**
B6	**#1**	**153**	**12**

Table 6 describes the eight different relaxation experiments that we conducted on three test chips (#1-#3) to monitor the effect of RRAM relaxation. For A1 and A2 experiments, we focus on the RRAM array resistance change without IMC for 94 hours. On the other hand, for B1, B2, B3, and B4 experiments, we performed array-level IMC in hardware a different number of times during the total experiment hours (up to 156 hours). During B5 and B6 experiments, more IMC operations are executed to verify the mitigation between the read-disturb induced RRAM drift effect Shim *et al.* (2020a) and the RRAM relaxation effect.

For six experiments from Table 6, Fig. 19 shows the measured conductance results of the four-level RRAM devices starting from the programming time up to 156 hours. The conductance values for the four-level RRAMs shown in Fig. 19 are in the range of a few μS to a few hundreds of μS.

The overall workflow of the relaxation measurement, simulation and DNN accuracy evaluation is shown in Fig. 20. During the chip measurement process, the 2-bit

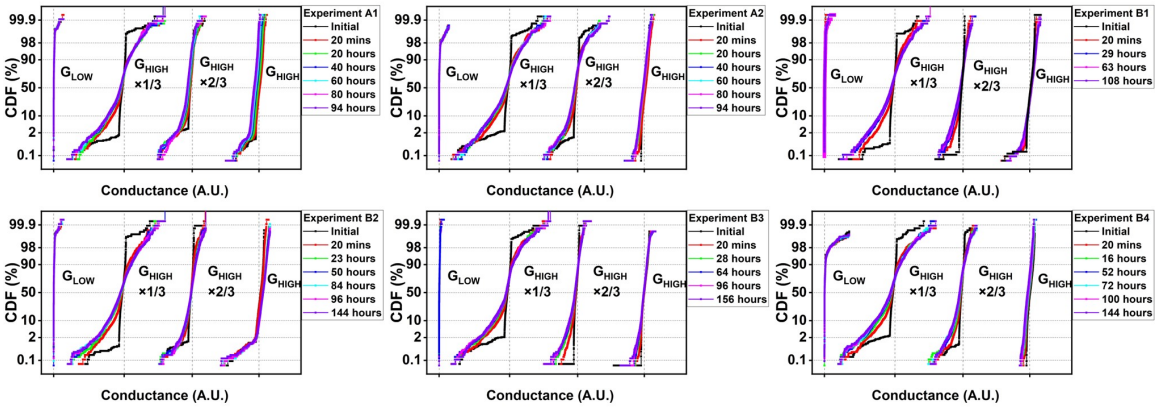


Figure 19. Four-level RRAM relaxation effect over time for six experiments A1/A2 (without IMC) and B1/B2/B3/B4 (with IMC). For the six experiments, most relaxation occurs right after the initial programming, and subsequently saturates over time.

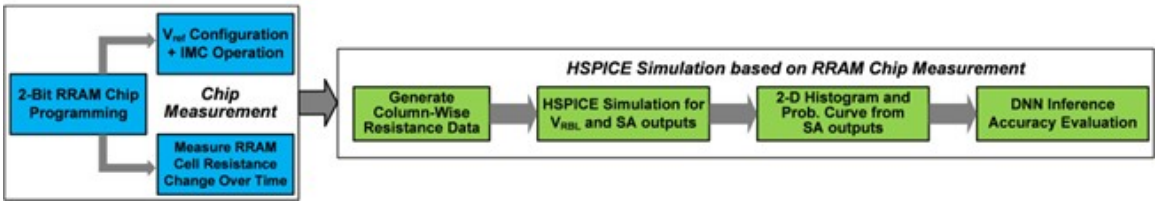


Figure 20. RRAM chip measurement and simulation framework of this work.

RRAM chips are programmed with the subsets of the DNN with the input-splitting algorithm He *et al.* (2020). With reference voltage configuration, IMC operation, and relaxation effect, the RRAM array cell resistance changes are monitored over time. To better characterize the relaxation impact on effective resistance of the column (R_{eff}), V_{RBL} and DNN accuracy, we performed HSPICE simulation on the RRAM array column with the peripheral circuits, where we used the individual RRAM resistance measurements from the eight experiments in Table 6. With the HSPICE simulation results, further data processing (2-D histogram and probability table generation) similar to hardware data processing is available, and we can generate DNN inference accuracy for RRAM array performance analysis, under different relaxation effect and operation stress condition.

While we can measure IMC results directly from the RRAM chips, we performed HSPICE simulation with RRAM device measurement results, to separate the relaxation effect over time with the IMC operation. In fact, as we discuss in Section 4.3.4, read disturb drift effects by IMC operations can partially mitigate the relaxation effects of RRAM devices.

4.3.2 Relaxation Measurements and DNN Inference Accuracy

For the relaxation measurements, we monitored the cell-level resistance changes over time for the 128×64 RRAM array across eight experiments, and Fig. 21 shows the results. The two intermediate states of $G_{HIGH}\times 1/3$ and $G_{HIGH}\times 2/3$ experienced more decrease in average conductance and more increase in standard deviation over time (Fig. 21), indicating that they are less stable than G_{HIGH} and G_{LOW}.

As shown in the cumulative distribution function (CDF) of Fig. 19 and four-level conductance color map in Fig. 22, we observed a noticeable relaxation effect for six experiments (A1-A2, B1-B4) obtained from the measurement of three test chips. It can be seen that a large portion of relaxation occurs right after the programming (Fig. 19), and a similar behavior has also been reported in other prior HfO_2 RRAM works Degraeve *et al.* (2016). Low resistance states of the cells tend to reduce their conductance over time, which agrees with what was reported in Wang *et al.* (2015), while high resistance states tend to fluctuate over time (Fig. 23) but they contribute negligible current to the bitline for MAC operation.

During the RRAM measurement based HSPICE simulation, to reflect the time-

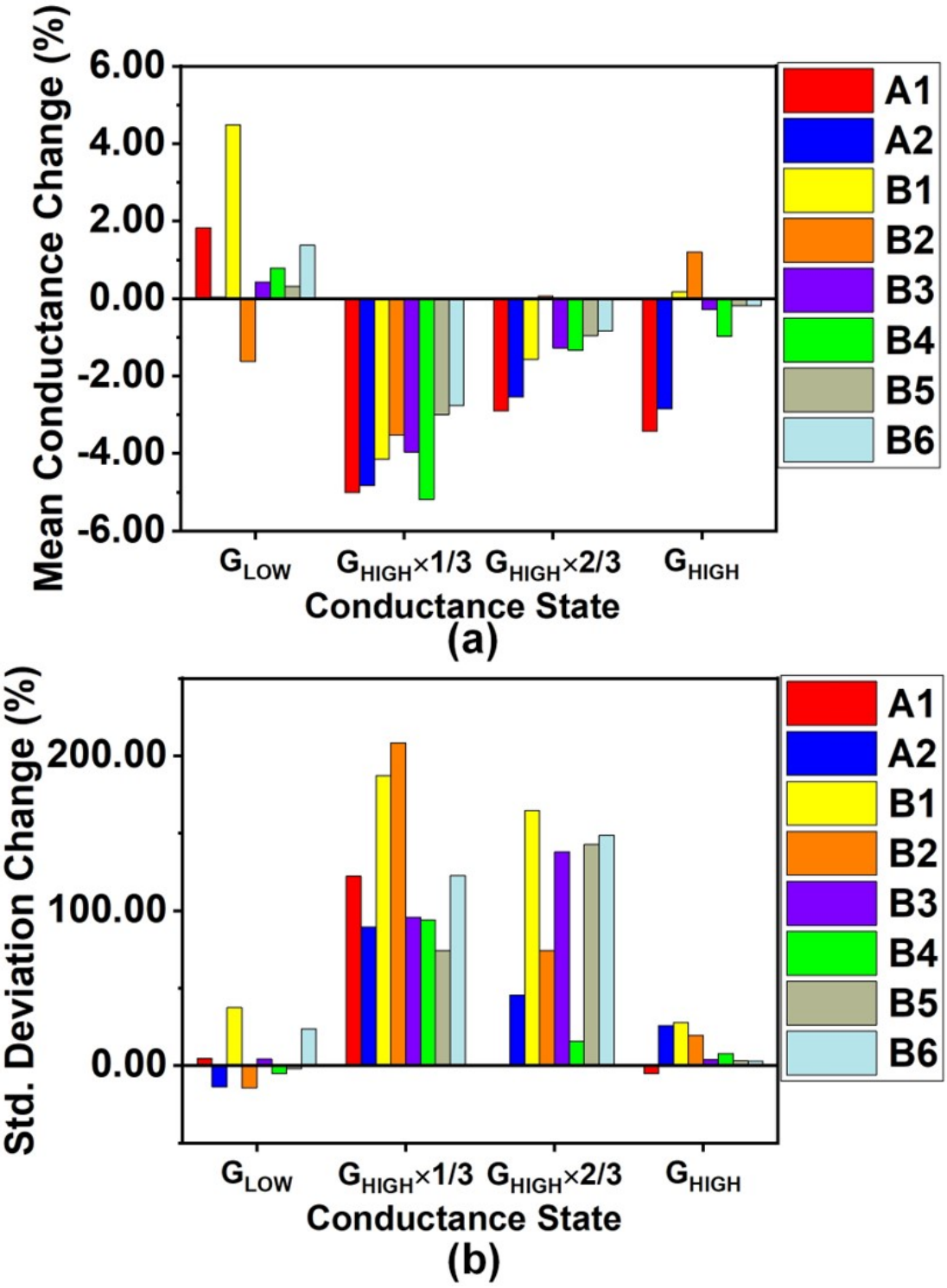


Figure 21. (a) Mean of conductance and (b) standard deviation of conductance changes in 100 hours are shown for the eight experiments.

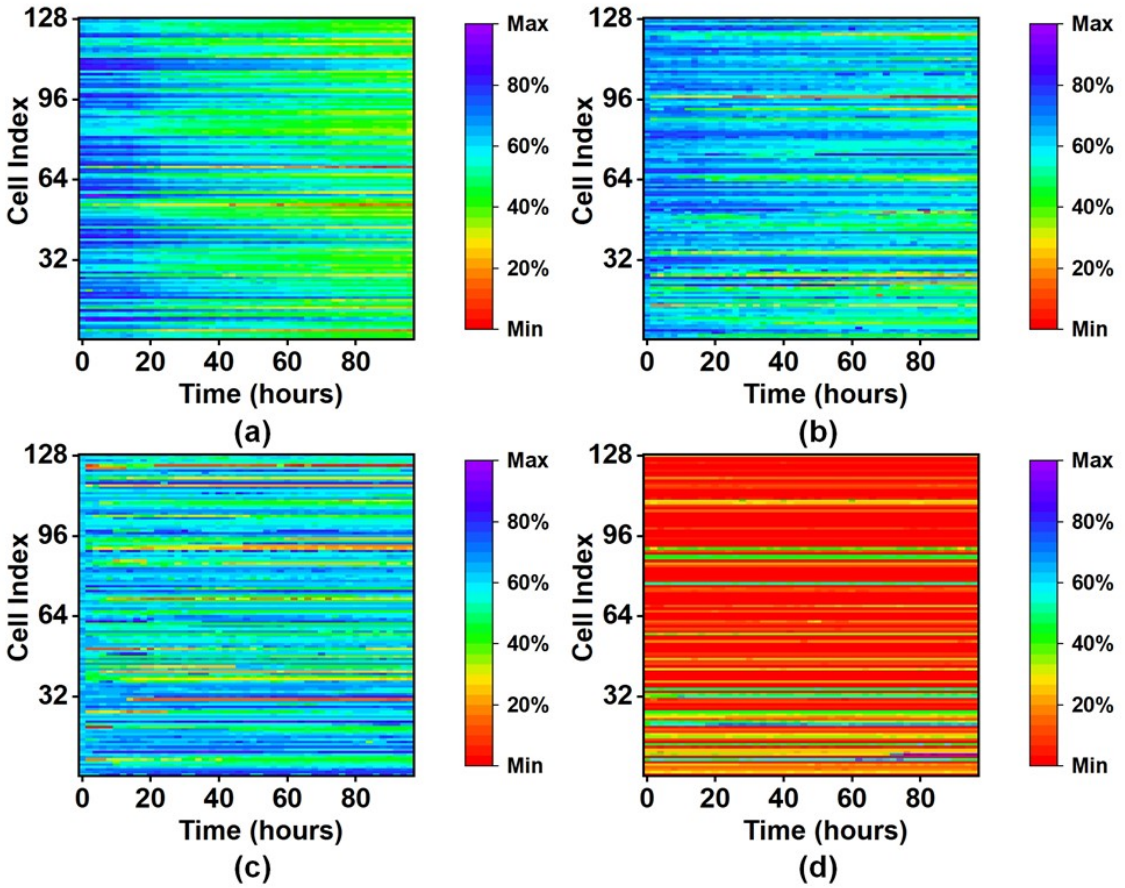


Figure 22. Four-level conductance color-map. (a) G_{HIGH}; (b) G_{HIGH} $\times 2/3$; (c) G_{HIGH} $\times 1/3$; (d) G_{LOW}. For each level, 128 cells' conductance values are from A1 experiment. Warmer color represents a lower conductance value.

induced noise or variation of SAs, we added a random variation value for each V_{ref} in the SA group. One input vector applied on 128 rows will perform IMC with 2-bit weights in the 128×64 array, and the output from the SA group indicates the computed results for the partial MAC value within one column. For each partial MAC value, Fig. 24(a) and Fig. 24(b) show the 2-D histograms of IMC measurements from the RRAM chips at 0 hours and 144 hours, respectively. Fig. 24(c) and Fig. 24(d) represent the 2-D histograms of the HSPICE simulations based on individual RRAM device measurements at 0 hours and 144 hours, respectively, which show similar

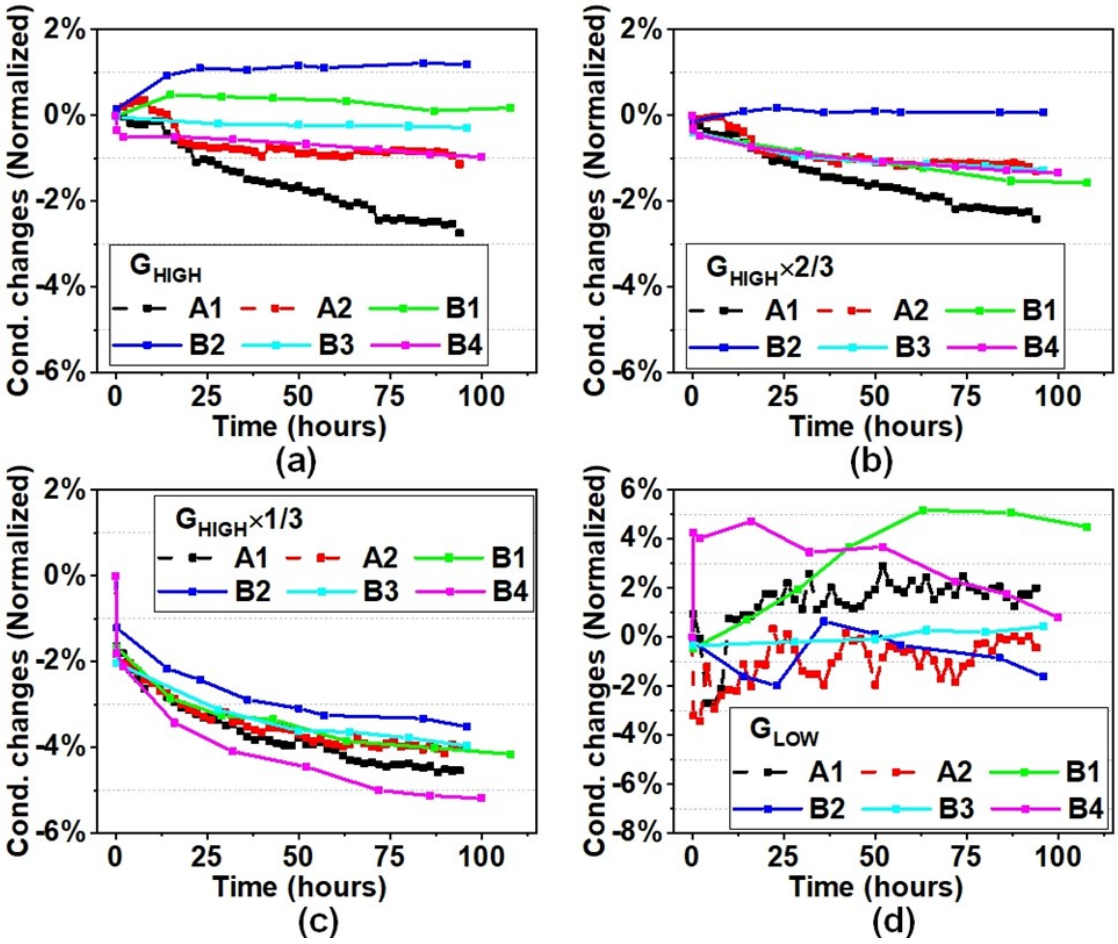


Figure 23. Average conductance change of four-level RRAM devices for six experiments. G_{HIGH} $\times 1/3$ is affected the most by relaxation. While G_{LOW} tends to fluctuate over time, they contribute negligible current for IMC.

behavior as the IMC measurement results. The 2-D histogram is used to generate a probability curve (Fig. 26), which states the probability for each MAC value output to be quantized to "+1" for the binary output ("+1" or "-1") of the RRAM array, based on the input-splitting scheme He *et al.* (2020).

Based on the RRAM conductance changes measured from six experiments, we compared the V_{RBL}, R_{eff}, and DNN accuracy change over time. First, we discovered a strong correlation in R_{eff} and V_{ref} changes over time. We simulated the average

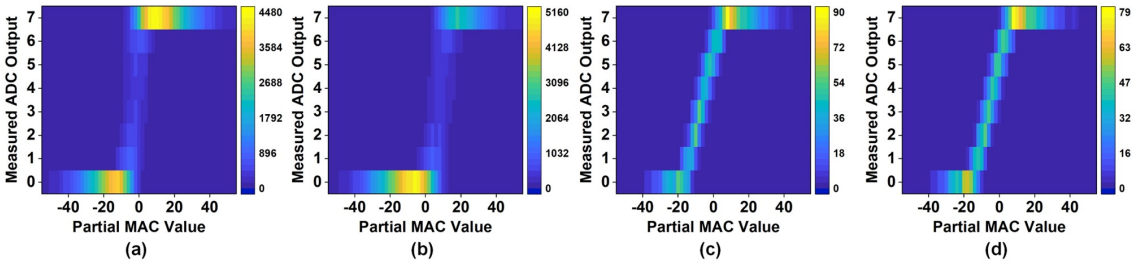


Figure 24. (a)-(b) 2-D histogram from IMC measurements from RRAM chips at 0 and 144 hours for B2 experiment. (c)-(d) 2-D histogram from HSPICE simulation using individual RRAM device measurements from 0 to 144 hours for B2 experiment.

R_{eff} and V_{ref} changes between 0 and ∼100 hours for six experiments (Fig. 25). A larger increase in R_{eff} corresponds to a larger V_{RBL} increase and leads to SA output difference over time. Second, we simulated the MAC operation starting from the time of 0 hours to ∼100 hours and use the V_{ref} setting at 0 hours throughout the entire experiment duration. The results indicate that a worse relaxation will lead to a larger change in average R_{eff}, and correspondingly results in a larger V_{RBL} change. These V_{RBL} changes cause the difference in SA group output, 2-D histogram, and shifting behavior in probability curves (Fig. 26), and lead to a considerable DNN inference accuracy loss, e.g. from 87.35% to 11.58% for B4 (Fig. 27(a)).

As we observe that a large portion of the RRAM relaxation effect occurs soon after the RRAM programming, one approach to avoid huge loss from it is to wait until most relaxation saturates, then calibrate the V_{ref} for SAs, and use the calibrated V_{ref} for the ensuing time. We selected ∼80 hours as the time to perform V_{ref} re-calibration for two reasons. First, while large conductance changes are observed right after programming (in 20 minutes from Fig. 19), non-negligible relaxation further occurs from 20 minutes to ∼80 hours, and the relaxation largely saturates after ∼80 hours. Second, our aim is that the V_{ref} re-calibration method can mitigate the long-term relaxation effect

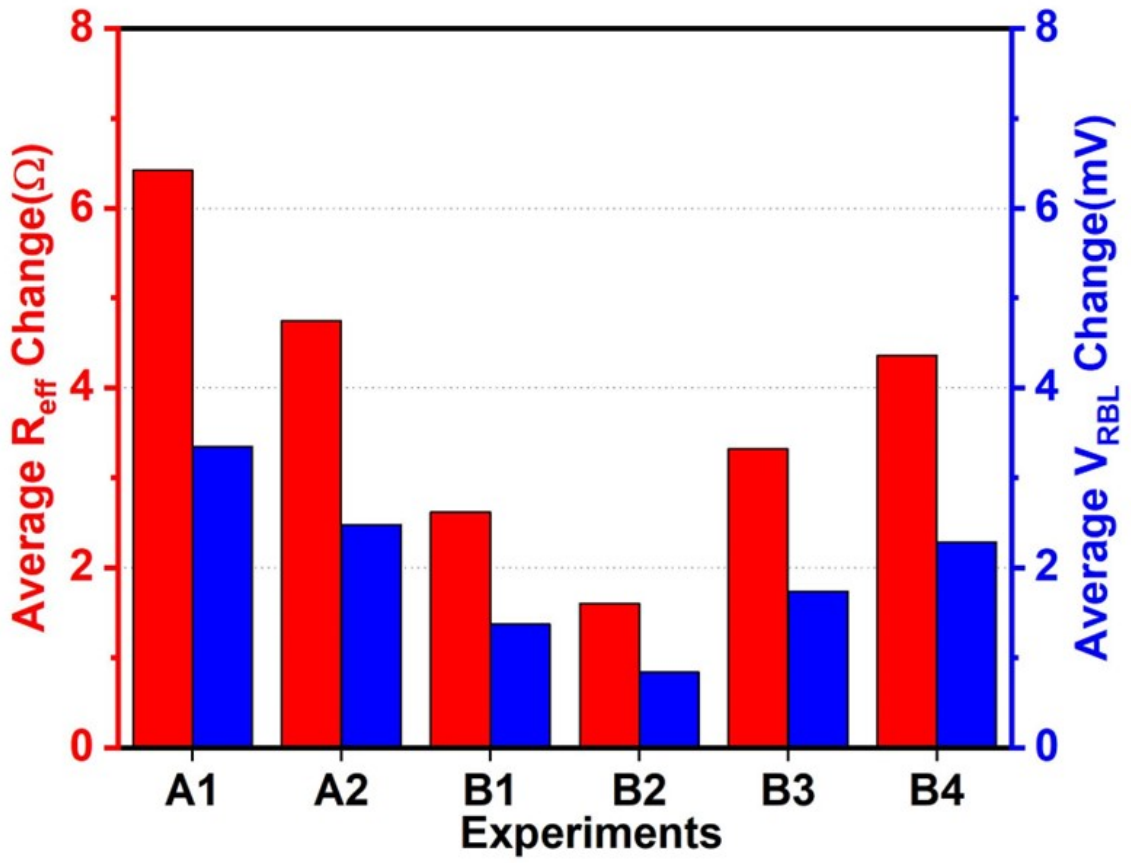


Figure 25. R_{eff} and V_{RBL} correlation. V_{RBL} will increase with R¬eff from the relaxation effect.

without additional write operations (e.g. throughout hundreds of hours, while our experiments in this paper are up to 150 hours due to the time limits). Compared to re-calibrating V_{ref} in 20 minutes or a few hours after programming, performing V_{ref} re-calibration in ∼80 hours could enable a longer operational time for the RRAM chips.

Based on the B2/B3/B4 experiments, we characterized the MAC operation starting from the time of around 80 hours to 144/156 hours and used the V_{ref} setting obtained at time of around 80 hours through the remaining time of experiments (Fig. 27(b)).

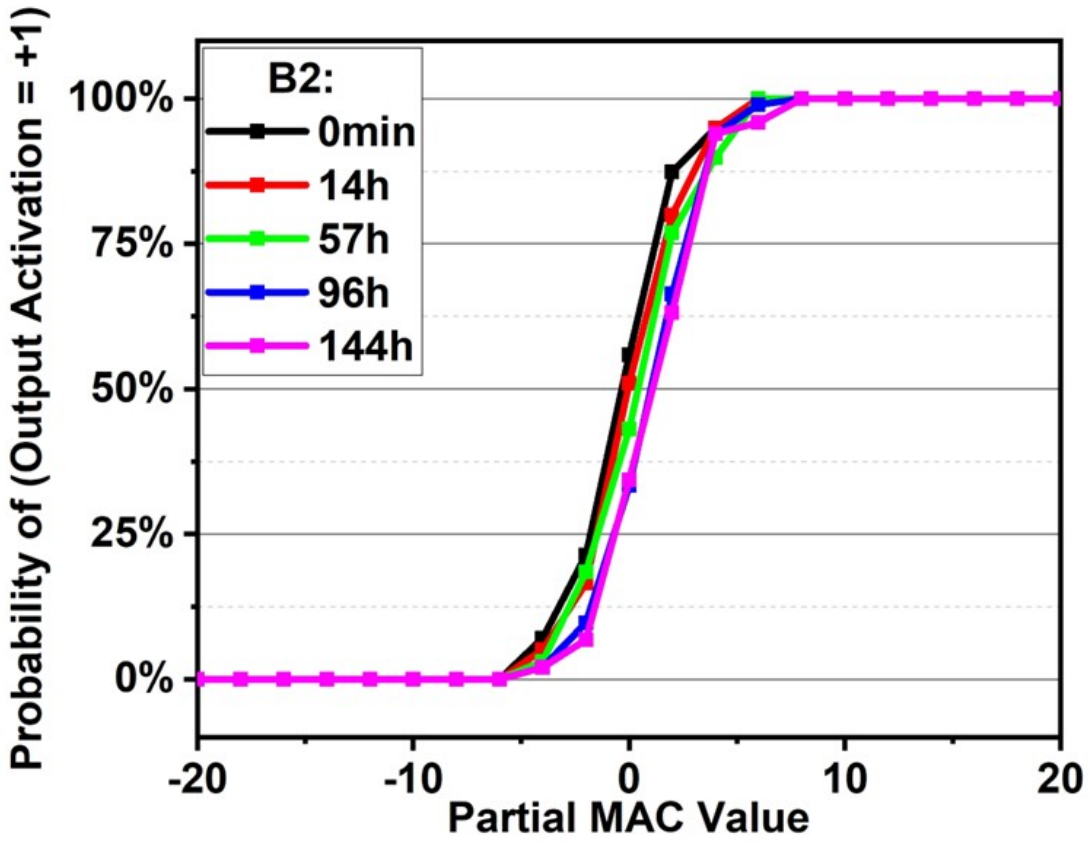


Figure 26. Probability curve shifting is observed over time from B2 experiment.

Compared to Fig. 27(a), a relatively smaller accuracy drop occurs over time in Fig. 27(b), due to the saturated RRAM relaxation behavior. This circuit-level technique improves the long-term stability for IMC, although B4 experiment needs further improvement.

4.3.3 Relaxation-aware DNN Training and Improvement

As seen in Fig. 20, among the four levels of 2-bit-per-cell RRAM, two intermediate states suffer more relaxation, due to the weak filament in RRAM cell. If the DNN is aware of this drawback on the two intermediate states and reduces the overall

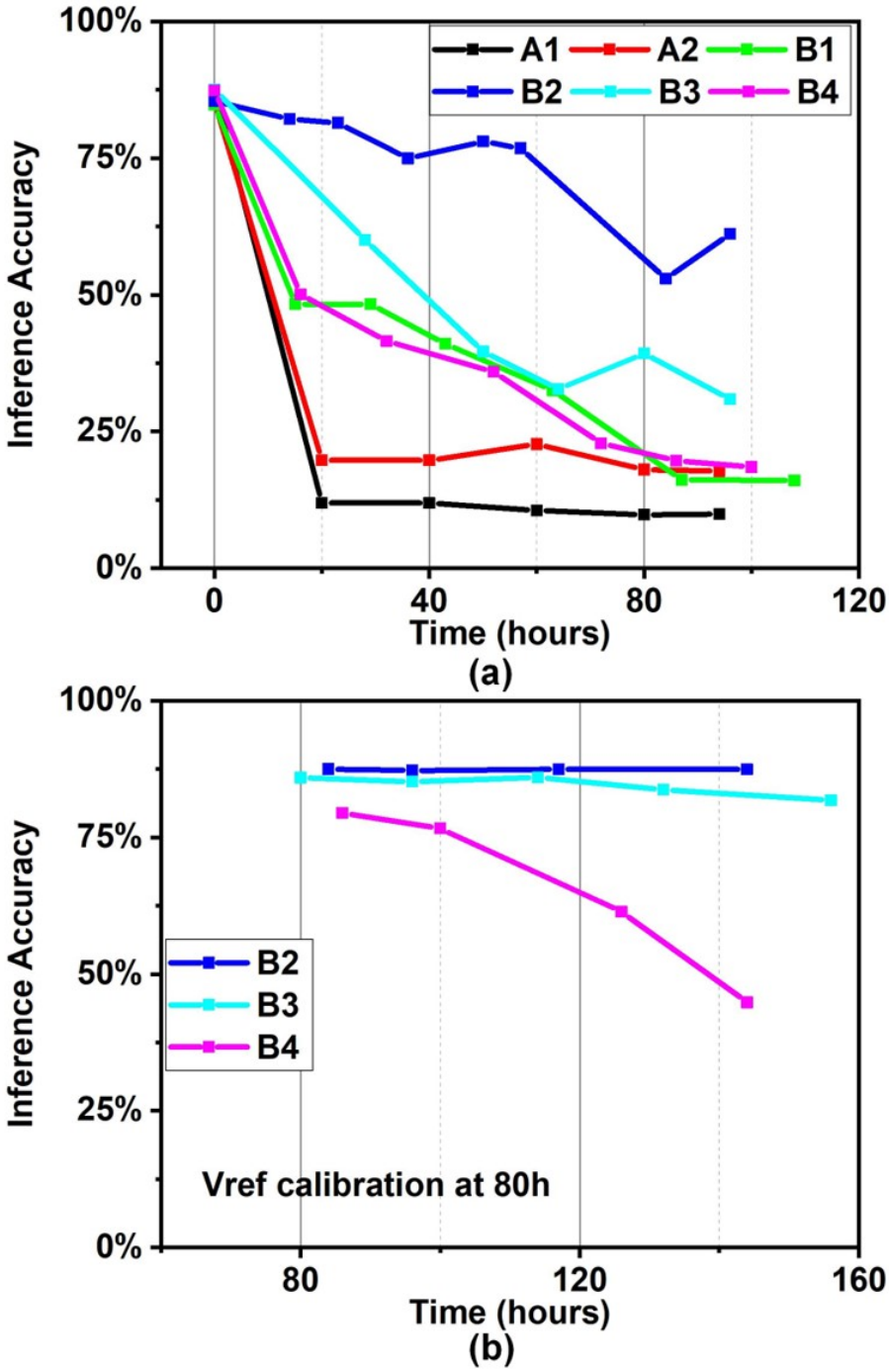


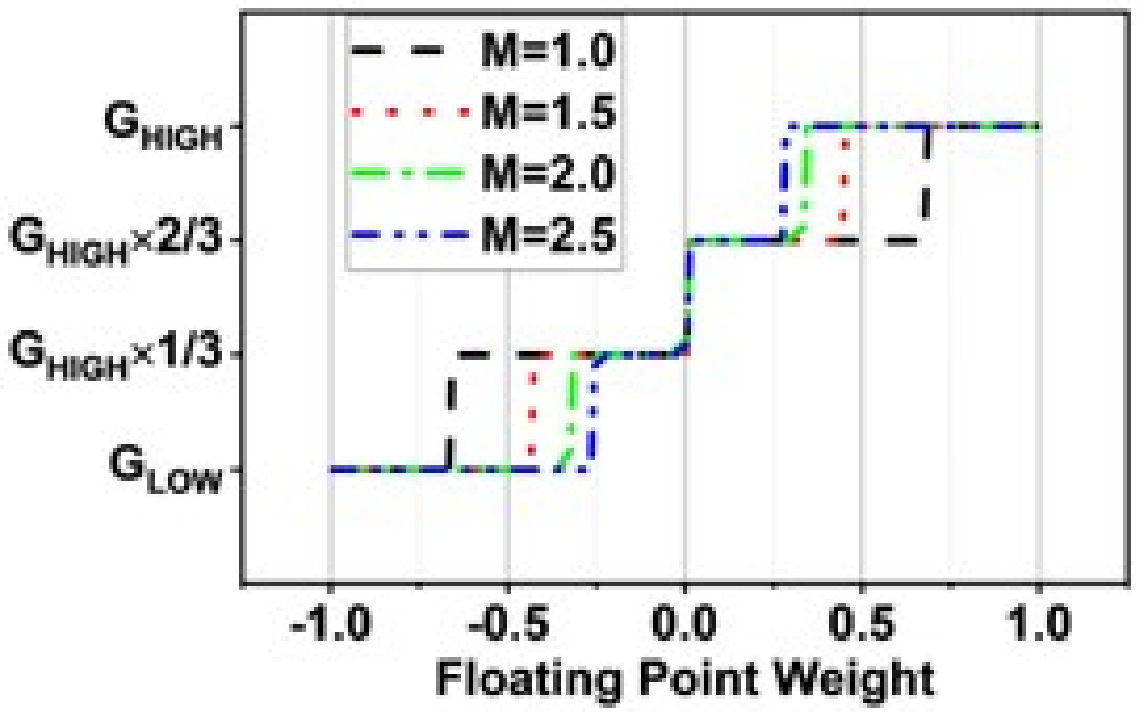
Figure 27. (a) DNN accuracy drops from 0 to ∼100 hours for 6 experiments. (b) DNN accuracy drops with V_{ref} calibrated at ∼80 hours for B2/B3/B4.

Table 7. Weight distribution change for DNN training with different magnification factor (M) for VGG-like CNN for CIFAR-10 dataset.

Mag. Factor for Training	G_{HIGH} (%)	$G_{HIGH} \times 2/3$ (%)	$G_{HIGH} \times 1/3$ (%)	G_{LOW} (%)	**Baseline DNN Accuracy**
1.0	**24.603**	**25.578**	**25.482**	**24.338**	**87.60%**
1.5	**32.948**	**17.157**	**17.128**	**32.767**	**87.39%**
2.0	**36.766**	**13.189**	**13.198**	**36.847**	**87.24%**
2.5	**39.389**	**10.755**	**10.716**	**39.141**	**87.60%**

percentage of these states during the training procedure, the inference accuracy should have less impact from the relaxation effect. Therefore, we introduce a magnification factor (M) during the training of 2-bit-weight VGG-like DNN for CIFAR-10 dataset (Fig. 28). By increasing the M during training, it pushes more weights within each layer to the highest and lowest resistance stages, with almost no initial accuracy change compared to the baseline DNN (Table 7).

Subsequently, using the measured RRAM conductance values from B2-B4 experiments, we evaluated the DNN accuracy over time for different M values from 1.0 to 2.5 (Table 7), using V_{ref} settings from the time at 0 hours and around 80 hours (Fig. 29). Compared with the results in Fig. 27(a), the re-trained DNNs with higher M achieves much higher DNN accuracy over time for both V_{ref} settings. By combining both schemes, DNN accuracy of >87.2% for CIFAR-10 is achieved for B4 over 144 hours with a single V_{ref} re-calibration at 86 hours. This indicates that the relaxation-aware training scheme largely alleviated the impact of RRAM relaxation.



$$W = 3 \times [(round(clip(weightarray \times M, -1, +1) \times 1.5 + 1.5) - 1.5)/1.5] \quad (4.1)$$

Figure 28. DNN training with higher magnification factor (M) pushes more weights to +3 (G_{HIGH}) and -3 (G_{LOW}).

4.3.4 RRAM Read Disturb on Relaxation Effect Mitigation

As discussed in Shim *et al.* (2020a), when the read voltage is higher than a certain level, read disturb will occur as conductance drift for both LRS and HRS states. High RBL voltage will tend to increase the conductance of LRS, and reduce that of HRS, which is the opposite trend of RRAM relaxation effect. In Fig. 19(a), compared to the no IMC operation experiments A1/A2, we observed that B1-B4 experiments have less mean conductance change. When IMC operation is applied on the chip, the voltage between the RRAM cells increases from normal read voltage (0.2V) to a relatively

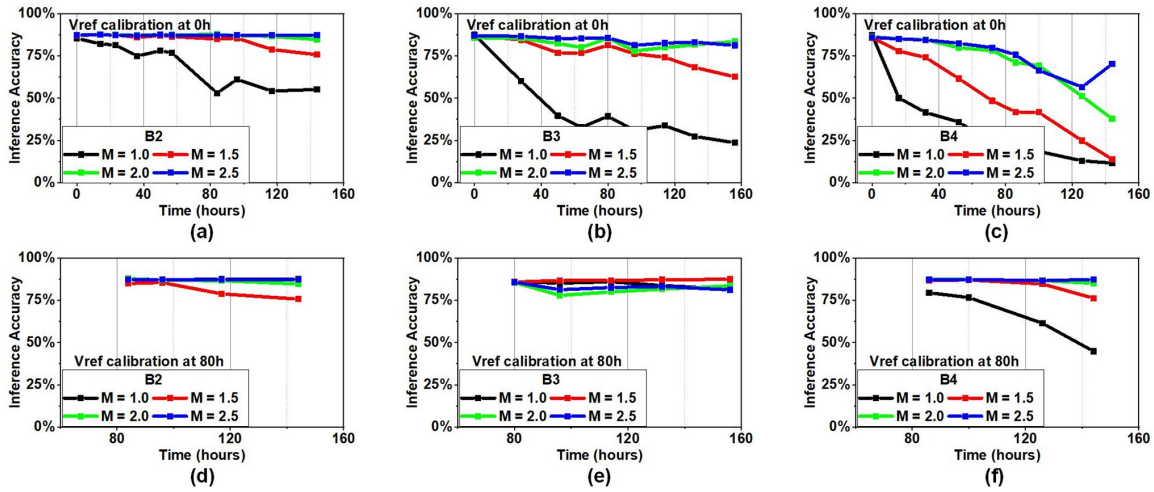


Figure 29. (a)-(c) For experiment B2/B3/B4, we tested the DNN inference accuracy from 0 to 144/156 hours, for RRAM arrays with weight distribution using different magnification factor (M). Overall, higher percentage of "+3/-3" weights offers improved robustness against relaxation. (d)-(f) DNN accuracy trends with V_{ref} calibrated at 80 hours for B2/B3/B4 experiments.

high value (0.4-0.6V), thus the read disturb induced conductance drift takes place. This drift mitigates the relaxation effect on LRS, and results in fewer conductance changes over time.

To verify this hypothesis, B5 and B6 experiments are conducted with a greater number of IMC operations and measured for inference accuracy performance comparison. B5 executes IMC operation every three hours, and B6 executes IMC operation every twelve hours, both having a higher IMC operation frequency than B1-B4 experiments. Overall, B5 and B6 experiments have less conductance reduction than B1-B4 at three LRS states (Fig. 20).

Then, similar relaxation simulations are reproduced with HSPICE simulations and both circuit-level and algorithm-level optimization methods are applied for B5 and B6 experiments. Fig. 30 shows the DNN inference accuracy changes over time under two

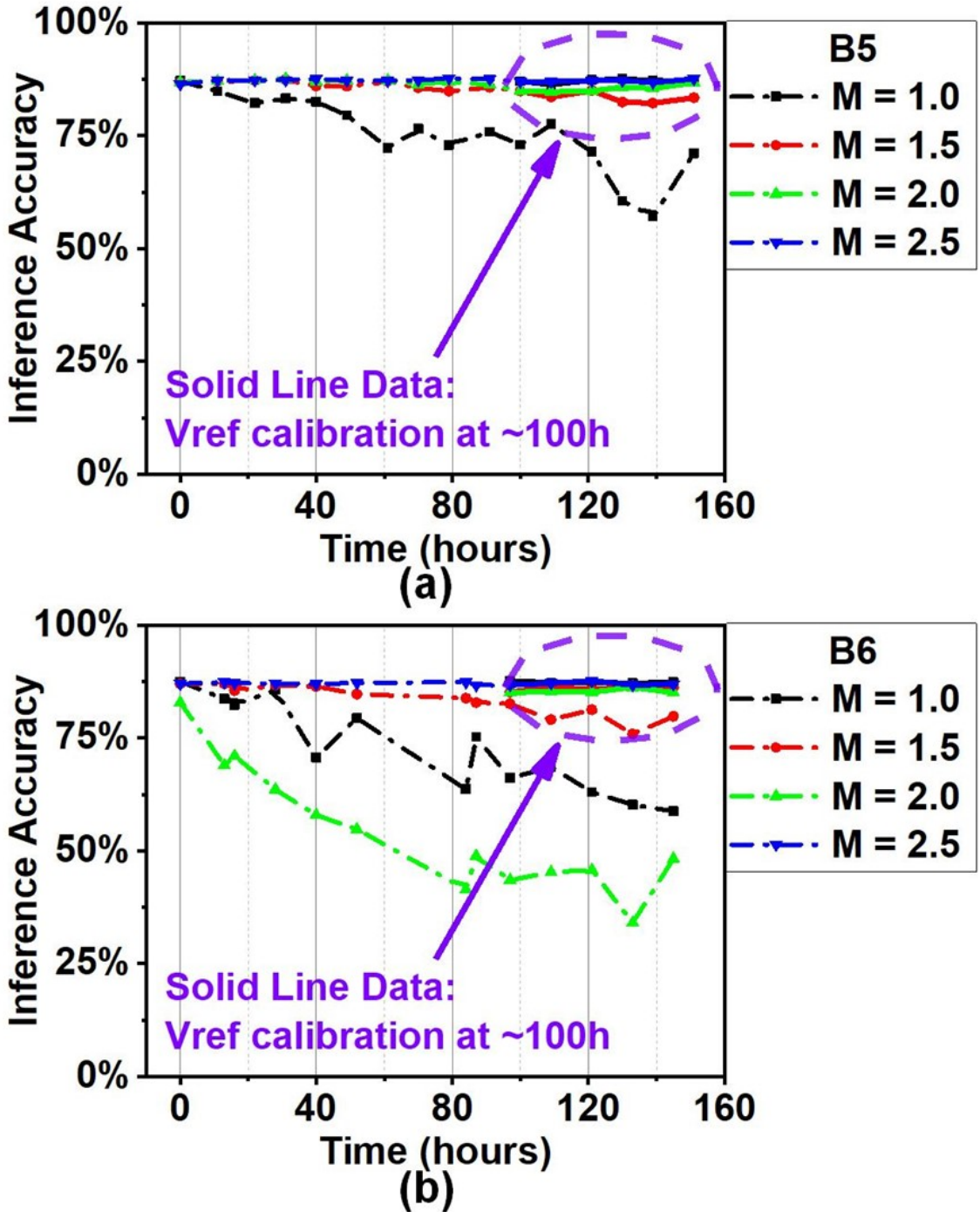


Figure 30. Simulation results of B5/B6 experiments under high frequency IMC operation stress. B5 and B6 show better accuracy retention after 150 hours.

mitigation methods. Compared to B2-B4 results in Fig 29, the accuracy degradation without any optimization scheme (M=1.0 case) is lower after 150 hours (87.18% to 71.12% for B5, 87.49% to 58.81% for B6). Regarding the two proposed mitigation schemes together with the purposely induced read-disturb, both methods result in similar or better accuracy retention over time compared with B2-B4 experiments. For B5 and B6 experiments, applying a single scheme is sufficient for DNN accuracy retention against the relaxation effect. V_{ref} recalibration alone (in solid lines in Fig. 30) can recover the accuracy of original DNN cases (M=1.0) to be above 83% after 150 hours. Similarly, relaxation-aware DNN training scheme alone serves similar improvement results as B2-B4 experiments. M=2.5 case in B5 and B6 (blue dash lines) can maintain the accuracy above 83% over 150 hours, without the help of V_{ref} calibration method. The results above indicate that the read disturb drift effects can also partially cancel out and mitigate the non-ideality from relaxation effect, overall improving the DNN inference accuracy over time.

4.4 Conclusion

In this chapter, we comprehensively characterized the relaxation effects of multi-level HfO2 RRAM at array-level for in-memory computing hardware targeting DNN inference applications. Relaxation effects are noticeable at intermediate states of multi-level RRAM, but can be compensated using circuit-level (e.g. V_{ref} calibration after relaxation saturation) and algorithm-level (e.g. relaxation-aware DNN training for weight re-distribution) techniques that we proposed and demonstrated. Also, the non-ideality from the read disturb induced drift effect can be utilized to mitigate the

relaxation effect and could potentially enhance the DNN inference accuracy retention over time.

Chapter 5

PRIVE: EFFICIENT RRAM PROGRAMMING WITH CHIP VERIFICATION FOR RRAM-BASED IN-MEMORY COMPUTING ACCELERATION

As deep neural networks (DNNs) have been successfully developed in many applications with continuously increasing complexity, the number of weights in DNNs surges, leading to consistent demands for denser memories than SRAMs. RRAM-based in-memory computing (IMC) achieves high density and energy-efficiency for DNN inference, but RRAM programming remains to be a bottleneck due to high write latency and energy consumption. In this work, we present the Progressive-wRite In-memory program-VErify (PRIVE) scheme, which we verify with an RRAM testchip for IMC-based hardware acceleration for DNNs. We optimize the progressive write operations on different bit positions of RRAM weights to enable error compensation and reduce programming latency/energy, while achieving high DNN accuracy. For 5-bit precision DNNs, PRIVE reduces the RRAM programming energy by 1.82×, while maintaining high accuracy of 91.91% (VGG-7) and 71.47% (ResNet-18) on CIFAR-10 and CIFAR-100 datasets, respectively.

5.1 Introduction

Deep neural networks (DNNs) have been successfully developed in many applications including computer vision, speech recognition, and others. As the complexity of DNN tasks increases, the number of weights or parameters in DNNs surges as well, leading to consistent demands for denser memories than SRAMs. Conventional DNN

accelerator systems have used DRAM to store a large number of DNN weights, but DRAM requires cumbersome refresh operations and off-chip memory access consumes very high energy consumption Sze *et al.* (2017). Instead of using off-chip memory, several recent accelerators employed embedded non-volatile memory (NVM) such as resistive RAM (RRAM) Li *et al.* (2021); Giordano *et al.* (2021) and magnetic RAM (MRAM) Rossi *et al.* (2021), to store a large amount of weights fully on-chip and reduce the energy consumption for overall memory access.

While these works Li *et al.* (2021); Giordano *et al.* (2021); Rossi *et al.* (2021) demonstrated on-chip integration of embedded NVMs, the NVMs only served the purpose of storage, and physically-separate processing engines (PEs) performed the computation. In this case, the DNN weights are accessed row-by-row from the NVM array and communicated to the PEs. To further improve this bottleneck, in-memory computing (IMC) Kang *et al.* (2014) has emerged as a promising scheme to embed computation inside the memory, thereby largely reducing the data transfer. Several different memory technologies, such as RRAM Liu *et al.* (2020); Yoon *et al.* (2022), SRAM Yin *et al.* (2021), and phase change memory (PCM) Khaddam-Aljameh *et al.* (2022) have been investigated for IMC. Non-volatile resistive devices such as RRAM can naturally support IMC operations with multiple rows turned on, where the weighted sum current between the wordline voltage (representing DNN activations) and RRAM conductance (representing DNN weights) represents the dot-product result.

Most RRAM-based IMC works employ a weight stationary scheme, but still the RRAM devices need to be programmed often, to execute inference of different DNN models over time, or to run workloads that require frequent weight updates (e.g. DNN training). Efficient programming is one of the critical bottlenecks for RRAM, since

RRAM write operation requires high voltage and latency. More importantly, a single write operation is insufficient to program a target conductance value to the RRAM device due to device and voltage variation.

A popular method to address this challenge is to iterate the process of writing the targeted value to an RRAM cell and reading the RRAM cell conductance for verification, which is referred to as write-verify or program-verify method Cheng *et al.* (2018); Gao *et al.* (2015). Besides RRAM, the conventional write-verify (CWV) method has been also employed for other devices to compute analog matrix-vector multiplications (MVMs) such as PCM Khaddam-Aljameh *et al.* (2022) and MRAM Noguchi *et al.* (2016). The CWV method can minimize the write uncertainty and RRAM conductance variation, but multiple iterations are required for each cell, which incurs large latency and energy overheads. Furthermore, frequent writing operations can hurt the cell endurance and possibly limit the lifetime of the memory device Zhao *et al.* (2018).

Several prior works investigated reducing the RRAM programming energy for IMC macros/systems, by reducing the number of write iterations and compensating the induced error (due to the smaller number of write iterations) by different methods. SWIPE Gonugondla *et al.* (2020) proposed to perform programming from MSB to LSB, and compensate for the write error from MSBs by adjusting the programming of LSBs. Probabilistic early termination was proposed in Meng *et al.* (2022), and the programming process was partitioned into the coarse predictive phase (with fewer write iterations) and the fine-tuning phase in Zhang *et al.* (2021). However, Gonugondla *et al.* (2020); Meng *et al.* (2022); Zhang *et al.* (2021) all reported only simulation results with behavioral RRAM models, and included unrealistic assumptions such as single-pulse programming for RRAM devices Gonugondla *et al.* (2020); Zhang *et al.*

(2021). Therefore, the claimed programming energy reduction of ∼10× likely will not be possible for actual RRAM hardware.

Le *et al.* (2021) is one of the few works that reported programming energy reduction with measurements of multi-level RRAM devices, but only row-by-row RRAM device readout was reported without turning on multiple rows as done for IMC and did not report accuracy results for DNNs.

In this work, we propose the Progressive-wRite In-memory program-VErify (PRIVE) scheme that is compatible with RRAM-based IMC hardware that has been prototyped in a commercial RRAM process. We performed RRAM programming with the PRIVE scheme and obtained measurements of the RRAM prototype chip, which was evaluated for VGG-7 and ResNet-18 DNNs on CIFAR-10 and CIFAR-100 datasets. To the best of our knowledge, this is the only work to date that reports RRAM programming energy reduction with an actual RRAM chip for IMC operation targeting DNN workloads. The main contributions of this work are:

- A new progressive write-verify scheme (PRIVE) is proposed for multi-bit weight programming using 1-bit-per-cell RRAM hardware. While inspired by prior work on RRAM programming energy reduction Gonugondla *et al.* (2020), we identify impractical aspects of Gonugondla *et al.* (2020) such as single-pulse programming and present a practical progressive programming scheme from MSB to LSB that is verified by RRAM chips.
- PRIVE implementation does not exhibit any overhead, e.g. DNN model re-training on the algorithm side Meng *et al.* (2022) or any additional circuits on the hardware side, as only the RRAM programming method has changed compared to the CWV scheme.
- We determine optimal configurations to apply PRIVE by investigating two

different low resistance state (LRS) values for RRAM programming and evaluate the trade-offs and corresponding DNN accuracy values.

- 1.82× energy reduction is achieved for the overall RRAM programming compared to the CWV scheme, while maintaining high DNN accuracy for VGG-7/ResNet-18 models on CIFAR-10 and CIFAR-100 datasets, based on RRAM chip measurements.

5.2 Background and Related Works

5.2.1 RRAM Based In-memory Computing

Recently, non-volatile memory (NVM) based IMC prototype chips have been reported in Liu *et al.* (2020); Yoon *et al.* (2022) and SRAM-based IMC prototype chips have been reported in Yin *et al.* (2021). Although the NVM technology is less mature than CMOS, they have advantages such as non-volatility, low power consumption, CMOS compatibility, and exhibit higher density compared to SRAMs at the same technology node. To that end, we focus on RRAM-based IMC design in this work.

Fig. 31 shows the high-level overview of the RRAM prototype chip reported in Yin *et al.* (2020c); He *et al.* (2020). Each 1T1R cell can be programmed to a low resistance state (LRS) or a high resistance state (HRS). The RRAM conductance represents DNN weights and the wordline (WL) voltage represents the activations. When multiple WLs are activated, the number of RRAM cells driven by LRS versus HRS determines the analog bitline (BL) voltage, which is digitized by the analog-to-digital converter (ADC).

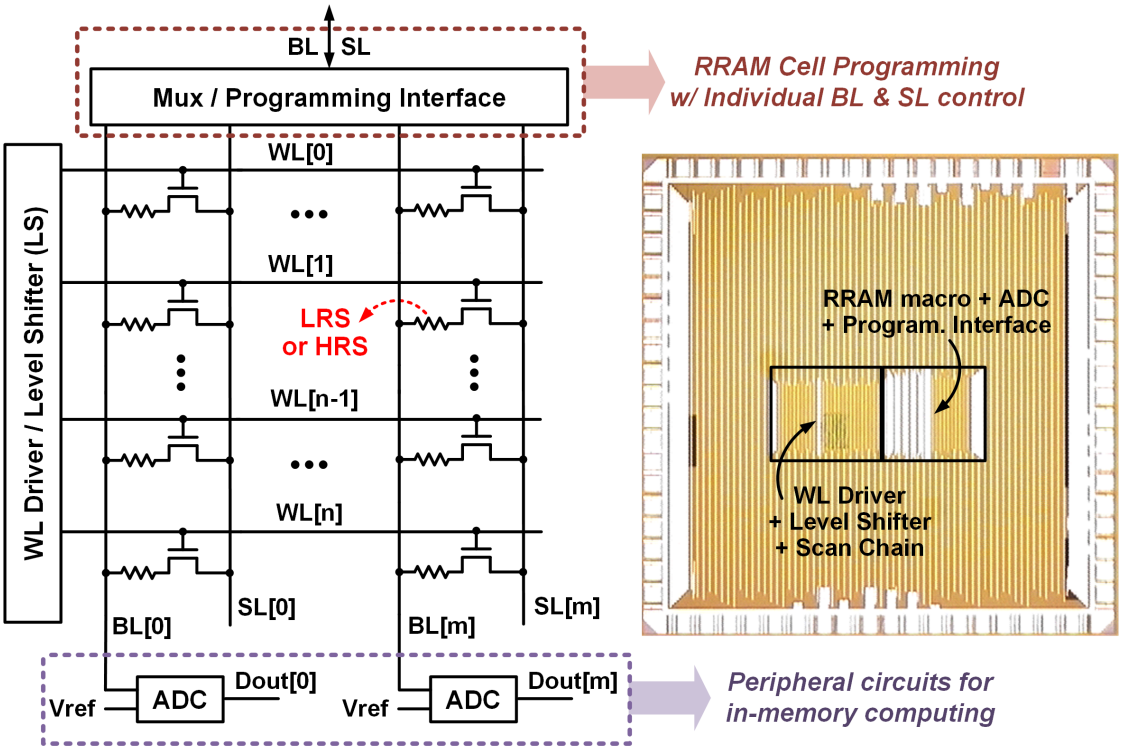


Figure 31. High-level overview of the RRAM prototype chip Yin *et al.* (2020c).

5.2.2 Prior Works on Efficient RRAM Programming

A number of prior works pursued efficient and accurate programming with the write-verify schemes on RRAM devices and arrays. Early research Cheng *et al.* (2018); Gao *et al.* (2015) proposed write-verify method for RRAM programming. Zhao *et al.* (2018); Chen *et al.* (2020) proposed write-verify method for multi-level RRAM devices, by tuning multiple parameters (e.g. bitline voltage, gate voltage, etc.) for set/reset processes. Building upon Zhao *et al.* (2018); Chen *et al.* (2020), a two-step 2-bit-per-cell write-verify scheme was presented in Shim *et al.* (2020b). Similar programming method such as POST Wang *et al.* (2020) was proposed, which splits the write pluses into several small pulses for single-cell programming in the RRAM array. However, these works only focused on the accurate programming of

a single RRAM cell, while multi-bit weight programming involving multiple RRAM cells has not been investigated.

For multi-bit weight programming, instead of iterating the programming of single RRAM cells excessively, SWIPE Gonugondla *et al.* (2020) proposed to compensate for the programming errors of more significant bits by adjusting the programming values of less significant bits. RRAM write energy reduction of up to 10× was reported, but only simulation results with behavioral device models were reported and incorrect assumptions such as single-pulse programming were made. Meng *et al.* (2022) presented probabilistic early termination on programming single devices and optimized re-programming a subset of multiple RRAMs that constitute the multi-bit weight, reporting >3× programming cost reduction based on simulation results. Zhang *et al.* (2021) partitioned the programming process into predictive phase and the fine-tuning phase, reducing the RRAM programming energy by ∼90% based on simulation. Simultaneously programming multiple RRAM cells was assumed in the simulation, while this has not been verified with actual RRAM hardware. RADAR Le *et al.* (2021) performed multi-level RRAM programming with a coarse resistance control phase and a fine-tuning control phase and reported 2.4× programming energy reduction with actual RRAM hardware, but it did not perform IMC or evaluate the accuracy of DNN workloads.

5.3 Proposed PRIVE Scheme

5.3.1 Limitations of Prior Works for RRAM Hardware

Several important discrepancies exist between actual RRAM hardware and the assumptions used in simulation-based prior works to reduce RRAM programming energy Gonugondla *et al.* (2020); Meng *et al.* (2022); Zhang *et al.* (2021). The main discrepancy is in the number of pulses needed for the RRAM crossbar programming. In Gonugondla *et al.* (2020); Zhang *et al.* (2021), it is assumed that a single write pulse could be sufficient to program the RRAM devices for a target conductance. However, in practice, programming each RRAM device with an acceptable error rate requires multiple programming pulses, as illustrated from the RRAM chip measurement results in Fig. 32. The RRAM programming results in Fig. 32 are based on multiple pulses repeatedly writing RRAM cells to LRS and HRS on chips from Yin *et al.* (2020c). Programming with multiple pulses is necessary for real RRAM hardware for three reasons:

1. The conductance error after the single-write pulse is too high, and this can be largely reduced and fine-tuned by programming using multiple pulses.
2. Due to the cell-to-cell variation, the single-write programming can result in a high variation of programming value. With adaptive multiple pulses of programming, cell-to-cell conductance variation can be reduced to a smaller value during multiple write-verify pulses, improving the stability against device non-ideality.
3. Aggressive high-voltage or wide pulses for single-write operations can potentially hurt the cell endurance, and increase the chances of breaking down the RRAM cell.

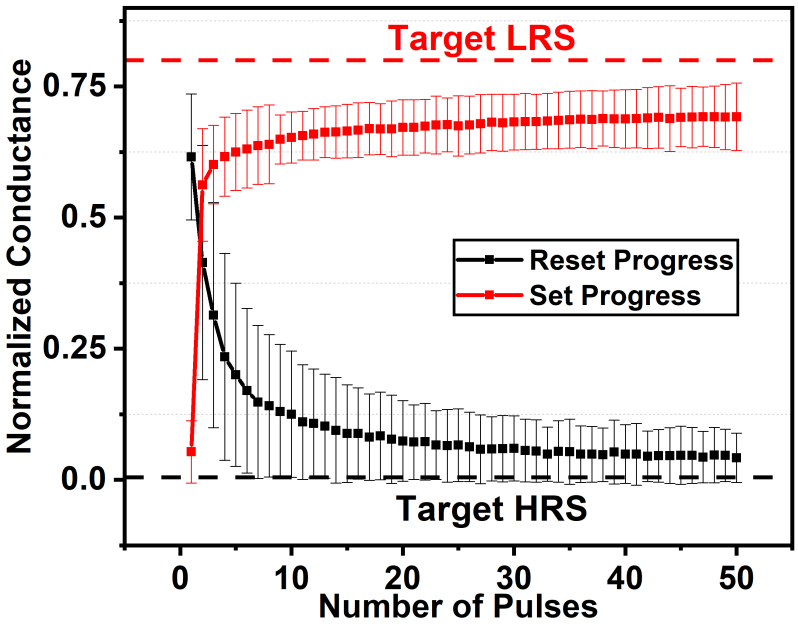


Figure 32. RRAM chip measurements on the RRAM conductance with the number of pulses for set and reset processes.

In addition, SWIPE assumes that the RRAM conductance programmed with single write pulses could either overshoot or undershoot the target LRS value. The existence of both positive and negative conductance errors at more significant bits could enable more error compensation capabilities at lower significant bits. However, especially with target LRS values that are configured to achieve a high on/off ratio, Fig. 32 shows that the intermediate conductance values during programming for LRS do not overshoot the target LRS value, which can limit the error compensation capabilities of SWIPE for certain weight values of DNNs.

5.3.2 Proposed Progressive Write-verify Algorithm

To address the limitations described in Section 5.3.1, we propose a progressive write-verify algorithm called PRIVE, as shown in Fig. 33. The PRIVE scheme is

based on 1-bit-per-cell RRAM devices, where a low resistance state (LRS) represents "1", and a high resistance state (HRS) represents "0" as binary storage. For a positive N-bit weight, we employ N RRAM cells and program each RRAM cell with LRS or HRS.

In every write epoch, the error of each RRAM cell can be presented as $E_i = G_i - G_{iW}$, where G_i is the programmed conductance for the i-th cell. Suppose $D_i \epsilon \{0, 1\}$ is the desired state of i-th bit, and $W_i \epsilon \{0, 1\}$ is the actual state that RRAM is programmed with. By default, W_i is identical to D_i. Depending on the value of W_i, G_{iW} would be G_{HRS} or G_{LRS}, the typical conductance value of HRS or LRS value, respectively. The write iteration will stop when the $|E_i|$ is smaller than the maximum error threshold, or when the epoch loop number exceeds the maximum epoch limit. Then, for a n-bit weight programming, the error of the i-th bit is accumulated as:

$$E \leftarrow E + E_i \times 2^{(n-i)} + (G_{LRS} - G_{HRS}) \times (W_i - D_i)^{(n-i)} \tag{5.1}$$

The accumulated error E is used to determine if the error in the current bit requires compensation in the next bit programming. If the error is larger than $(G_{LRS} - G_{HRS})$ conductance gap, the binary value for the next bit position is available for a bit-flipping compensation. This means that W_{i+1} will be effectively programmed as $W_{i+1} = 1 - D_{i+1}$, and this change will be reflected in the error accumulation for the $(i + 1)$-th bit in Eq. (5.1).

In this work, we mapped the 4-bit weights with four 1-bit-per-cell RRAM devices. To analyze the effectiveness of the PRIVE scheme, we quantify the equivalent weight (W_{eq}) for each 4-bit weight as formulated in Eq. (5.2), and compare it with the ideal value.

$$W_{eq} = [\sum_{i=0}^{3} (G_i \times 2^i) - G_{HRS} \times 15]/G_{LRS} \tag{5.2}$$

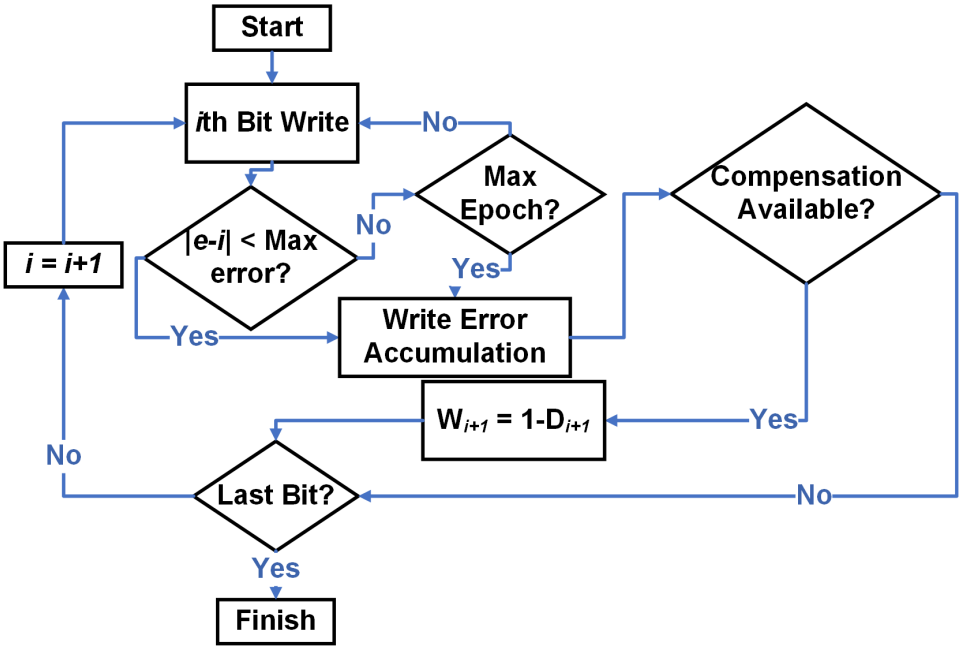
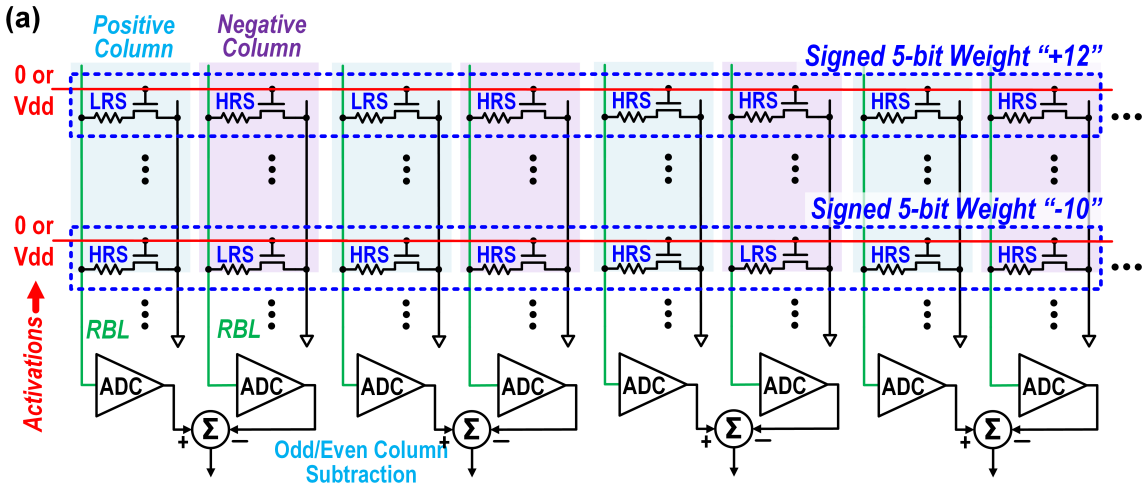


Figure 33. The algorithm and RRAM programming flow of the proposed progressive write-verify algorithm (PRIVE). The less significant bits are employed to compensate for the RRAM programming error in more significant bits.

For an ideal RRAM device with $G_{HRS} = 0$, the equivalent weight will be identical to the 4-bit weight value.

With the PRIVE scheme, the number of pulses required for write-verify iterations on each RRAM cell can be progressively reduced to an appropriate number, in order to balance the trade-off between precise RRAM programming and the energy reduction of RRAM programming. Note that PRIVE still follows the traditional write-verify at each pulse, and the proposed changes are made on the decision levels during each programming stage. Therefore, compared to the conventional write-verify progress, there does not exist any additional overhead or extra energy consumption to implement the PRIVE scheme on RRAM chips.

In this work, we set the default number of write-verify pulses to be 25 for the programming of each RRAM cell, since the error saturates after ~25 iterations of CWV



(b)

5-bit Weight	RRAM cells in 4 pos. columns	RRAM cells in 4 neg. columns
> 0	1 – 15 ('1': LRS, '0': HRS)	All HRS
< 0	All HRS	1 – 15 ('1': LRS, '0': HRS)
= 0	All HRS	All HRS

Activation Bit	Weight Bit	Read Bitline (RBL)
Vdd ('1')	LRS ('1')	Pull down
	HRS ('0')	–
0 ('0')	LRS ('1')	–
	HRS ('0')	–

(c)

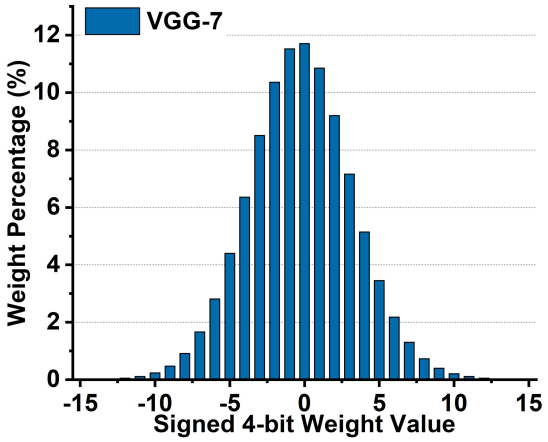
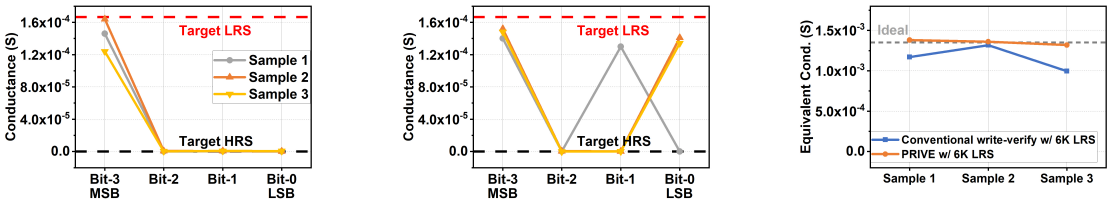


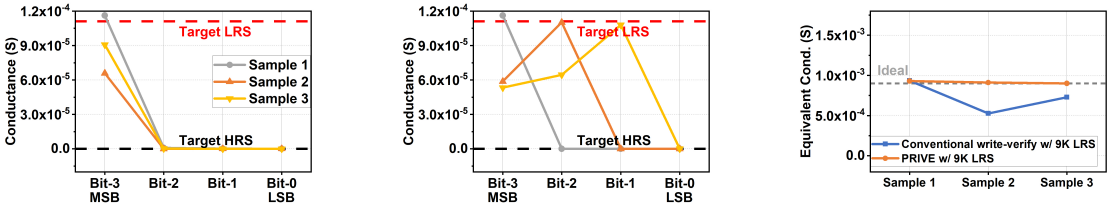
Figure 34. (a) Hardware implementation of PRIVE for the 5-bit weights. (b) Signed programming method truth table applied in this work with PRIVE. (c) Software quantization model mapping of PRIVE.

(Fig. 32) for the RRAM chips Yin *et al.* (2020c). For multi-bit programming, MSB contributes the highest amount of write error, so we keep the number of programming pulses for MSB at 25. Then, we progressively reduce the number of pulses from MSB to LSB for the 4-bit weight programming in the PRIVE scheme, to balance the programming energy reduction and the programming accuracy. In particular, we use 25 pulses for bit 3 (MSB), 15 pulses for bit 2, 10 pulses for bit 1, and 5 pulses for bit 0 (LSB). These number of pulses (25-15-10-5) were chosen as they provided a good



(a) (b) (c)

Figure 35. RRAM programming for three samples of 4 RRAM cells for weight value of 8 ("1000") with $6k\Omega$ LRS: (a) conventional write-verify scheme, (b) the PRIVE scheme, and (c) effective 4-bit weight conductance comparison.



(a) (b) (c)

Figure 36. RRAM programming for three samples of 4 RRAM cells for weight value of 8 ("1000") with $9k\Omega$ LRS: (a) conventional write-verify scheme, (b) the PRIVE scheme, and (c) effective 4-bit weight conductance comparison.

energy/accuracy balance, based on a coarse sweep of different pulse configurations (e.g. 25-10-5-2). Compared to the CWV scheme of using 25-25-25-25 pulses for 4-bit weights, the PRIVE scheme with 25-15-10-5 pulses reduces the total number of programming pulses by 45%, and this translates into proportional write latency and energy savings.

The main trade-off of PRIVE is the correctness of the single RRAM cell. As an example in Fig. 36(b), the errors on less significant bits can increase due to the reduced programming iterations, which may not get properly compensated (e.g. no LSBs available). However, this will only occur to a small portion of the weight levels, such as "7" and "15", and the overall DNN-level effect will be further discussed in Section 5.4.

5.4 Experiment Results and Analysis

The PRIVE scheme is evaluated with RRAM prototype chips Yin *et al.* (2020c); He *et al.* (2020) (Fig. 31), which were fabricated in a commercial CMOS process that monolithically integrates HfO_2 RRAM. For RRAM programming, we use SET voltage of 2.1V, SET pulse width of 100ns, SET gate voltage of 2.6V/2.3V for $6k\Omega/9k\Omega$ LRS, RESET voltage of 3.8V, RESET pulse width of 250ns, and RESET gate voltage of 4.0V.

We programmed 5-bit weights onto eight 1T1R RRAM cells with positive and negative columns Liu *et al.* (2020) as shown in Fig. 34(a), throughout the entire RRAM array for both conventional and PRIVE schemes. The truth table of each 1-bit RRAM is shown in Fig. 34(b). The programmed value on even or odd columns will determine the sign of the weight, and all RRAM cells in the other columns will be programmed with HRS for analog subtraction according to Eq. (5.2). As a result, the RRAM-based IMC inference is performed with the signed weight values without introducing auxiliary offset columns Peng *et al.* (2019). Furthermore, the weights of the IMC inference are directly represented by the measured conductance values, which fully incorporate the non-idealities of the actual RRAM devices. The column-wise partial sum will be scaled by the difference between the typical HRS and LRS values, so arithmetic correctness will be preserved in the algorithm-level simulation.

Based on the RRAM chip measurement data, we evaluate the DNN inference accuracy of RRAM IMC hardware with the pre-trained quantized DNN models on VGG-7 and ResNet-18 frames with CIFAR-10 and CIFAR-100 datasets. We program the quantized weights on the prototype chip and the measured conductance weight

values are used to evaluate the overall DNN accuracy. We performed the overall experiments in the following steps:

1. Map the DNN weights onto existing RRAM chips with a given LRS target for both CWV and PRIVE schemes
2. Retrieve programmed conductance data from RRAM chips
3. Validate inference accuracy on pre-trained quantization models of VGG-7 and ResNet-18 DNNs for CIFAR-10 and CIFAR-100 datasets.

5.4.1 Effectiveness of PRIVE programming

We experimented with two LRS values for RRAM programming: (1) the default $6k\Omega$ ($1.67\times10^{-4}S$) that achieves the highest on/off ratio of $\sim$100, and (2) a higher LRS value of $9k\Omega$ ($1.11\times10^{-4}S$) that achieves a lower on/off ratio but consumes a lower current and could exhibit better error compensation capability by PRIVE. $9k\Omega$ LRS can have better error compensation with PRIVE, because both overshoot and undershoot can occur during target conductance programming. As shown in Fig. 32, programming for $6k\Omega$ LRS will only exhibit undershoot with the write pulses.

The effectiveness of the PRIVE scheme is depicted in Fig. 35 (with $6k\Omega$ LRS) and Fig. 36 (with $9k\Omega$ LRS), where we selected three samples of four RRAM pairs that are programmed to represent the 5-bit weight value of "+8" ("1000").

The conventional write-verify scheme will always program LRS for a '1' binary bit and HRS for a '0' binary bit with many iterations, and could achieve relatively more accurate programming on the RRAM devices. However, even if the programmed LRS conductance after write-verify iterations cannot reach the target LRS conductance as

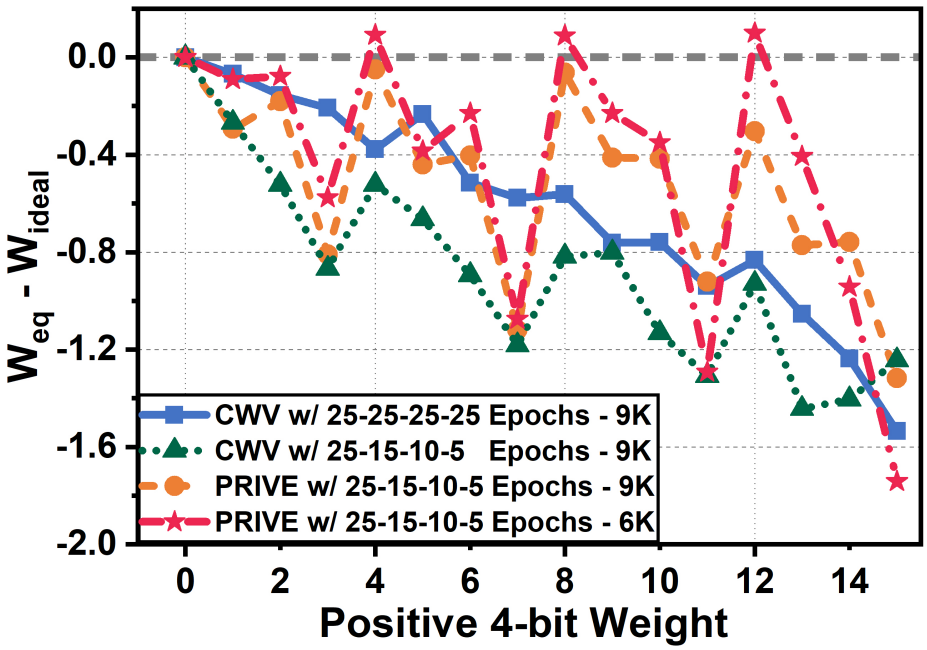


Figure 37. The deviation of measured W_{eq} from ideal positive 4-bit weight for CWV and PRIVE schemes with different programming epochs.

shown in Fig. 35(a) and Fig. 36(a), no further improvement or compensation can be made.

In contrast, the PRIVE scheme is aware of the errors during the programming from MSB to LSB, and is able to compensate for the programming error or conductance deviation in more significant bits by rearranging the programming states of less significant bits, as shown in Fig. 35(b) and Fig. 36(b). Fig. 35(c) and Fig. 36(c) show that the equivalent conductance of the PRIVE scheme is maintained very close to the ideal value for both $6k\Omega$ and $9k\Omega$ LRS cases, while noticeable deviations exist for the conventional write-verify scheme.

In Fig. 37, the differences between the equivalent weight W_{eq} (Eq. (5.2)) measured from the RRAM chip and the ideal positive 4-bit weight are compared for (1) CWV scheme with the default 25-25-25-25 programming, (2) CWV scheme with progressive

25-15-10-5 programming, and (3) PRIVE scheme with 25-15-10-5 programming, with target LRS of $6k\Omega$ and $9k\Omega$. The result of each datapoint is averaged from >100 measurements of 4 RRAM pairs. Larger W_{eq} errors are shown when we perform CWV with 25-15-10-5 epochs, but such errors are largely reduced in the PRIVE scheme owing to error compensation. While Fig. 36 shows that overshoot possibilities can provide better error compensation for $9k\Omega$ LRS in certain cases, Fig. 37 shows that the average programming error of many RRAM devices is still lower for the $6k\Omega$ LRS.

For a few weight values, the PRIVE scheme cannot improve the W_{eq} error of the 25-15-10-5 conventional scheme, and notably, these weight values, such as "7" ("0111"), have more '1' bits in the binary representation. Although $9k\Omega$ LRS allows some amount of conductance overshoot during programming, as shown in Fig. 36(b), conductance undershoot is still more dominant during programming. Conductance undershoot needs to get compensated by flipping less significant bits from '0' to '1', but if the less significant bits are already filled with '1' (e.g. "0111", "0011"), the compensation capability of the PRIVE scheme is limited within the 5-bit weight programming.

On the other hand, for weight values that have more '0' bits, e.g. "8" ("1000"), since fewer errors are produced at HRS, the PRIVE algorithm will have more capability to compensate the conductance undershoot error of programming "1" or LRS at more significant bits, by flipping the '0' bits at less significant bits to '1'. This leads to the result that, the PRIVE compensation works better in levels with more '0' bits (e.g. "8") than the levels with more '1' bits (e.g. "7"), as shown in Fig. 37.

5.4.2 DNN Accuracy Evaluation and Comparison to Prior Works

Based on the RRAM chip measurements, we tested the inference accuracy of VGG-7 and ResNet-18 models for CIFAR-10 and CIFAR-100 datasets, and the results are shown in Table 8. PRIVE provides 45% (1.82×) improvement in programming energy and latency reduction. Compared to the software baseline accuracy with the same precision, 1-2% accuracy degradation exists for CIFAR-10/100 datasets but the accuracy values are still higher than those of the CWV scheme for identical models/datasets.

We also compared the proposed PRIVE scheme and other works on RRAM programming energy reduction in Table 9. Due to the reasons mentioned in Section 5.3.1, the claimed energy reduction in Gonugondla *et al.* (2020); Meng *et al.* (2022) likely will not be possible with real RRAM hardware. PRIVE employs RRAM chip measurement data to evaluate the effectiveness of realistic programming techniques for IMC designs. The universal optimization results shown in Table 8 and the flexibility of the PRIVE algorithm during the programming stage indicate the potential of PRIVE on more practical and efficient programming for the multi-bit RRAM array, which can be used on top of other multi-bit-per-cell approaches such as Le *et al.* (2021).

5.5 Conclusion

RRAM-based IMC accelerators can achieve high density and high energy-efficiency for DNN workloads, but programming RRAM devices consume high energy. To effectively reduce the RRAM programming energy for IMC accelerators, this work

Table 8. The inference accuracy of the PRIVE scheme across different DNN models and datasets.

Model	Dataset	LRS Value	Baseline Accuracy	CWV Accuracy	**PRIVE Accuracy**
VGG-7	CIFAR-10	$6k\Omega$	92.53%	90.78%	**91.91%**
		$9k\Omega$		87.20%	**91.44%**
VGG-7	CIFAR-100	$6k\Omega$	69.90%	66.92%	**69.17%**
		$9k\Omega$		56.02%	**67.65%**
ResNet-18	CIFAR-10	$6k\Omega$	93.16%	91.56%	**92.61%**
		$9k\Omega$		86.00%	**91.97%**
ResNet-18	CIFAR-100	$6k\Omega$	72.56%	67.26%	**71.47%**
		$9k\Omega$		56.60%	**70.17%**

Table 9. Comparison to prior works.

	Gonugondla *et al.* (2020)	Zhang *et al.* (2021)	Meng *et al.* (2022)	Le *et al.* (2021)	**This work**
Hardware Verfication?	No	No	No	Yes	Yes
Weight Precision	2-to-9 bit	N/A	8-bit	3-bit-per-cell	5-bit
DNN Dataset	MNIST, CIFAR-10	MNIST, CIFAR-10	CIFAR-10, CIFAR-100	N/A	CIFAR-10, CIFAR-100
Network Type	LeNet-300-10, 8-Layer CNN	FC-NN, LeNet5, ResNet-20	ResNet-18, ResNet-34	N/A	VGG-7, ResNet-18
Accuracy Change	− <1%	−2-4%	+0.23%	N/A	**−1-2%**
Write Energy Reduction	5-10×	up to 19×	∼3×	2.4×	1.82×

demonstrated a progressive write-verify algorithm called PRIVE, which was verified with RRAM chip measurements. We measured two different LRS values during the RRAM programming to verify the PRIVE error compensation effectiveness. We tested PRIVE hardware measurement data on several different inference models and achieved 1.82× write energy and latency improvement with minimal accuracy degradation.

Chapter 6

RRAM-BASED HYBRID IN-MEMORY COMPUTING (HIMC) FOR DNN TRAINING

6.1 Introduction

With the advent of artificial intelligence (AI), various deep neural networks (DNNs) such as convolutional neural networks (CNNs) and recurrent neural networks (RNNs) have emerged and achieved human-level performance in image/speech recognition tasks. Several emerging applications (e.g. robotics, drones, etc.) require on-chip and continual learning, necessitating the development of special-purpose devices and architectures that can implement learning algorithms with large model sizes in an energy-efficient manner. Conventional SRAM or DRAM-based solutions cannot satisfy necessary density and energy requirements due to the large area footprint and prohibitive off-chip memory access costs. Several approaches have been applied to fulfill the energy-efficiency requirements for AI applications, among those, in-memory computing (IMC) condenses the processing unit (PU) within the memory blocks to overcome the bottleneck for data transportation through the buffer. And non-volatile memory devices (NVM) such as resistive random access memory (RRAM) and phase change memory (PCM), etc., can minimize the area and power consumption during the design stage, with the ability to support analog matrix computation within the memory blocks.

During the training process of the deep neural networks, this work has observed the ΔW is usually much smaller than W. This work investigated the statistics of

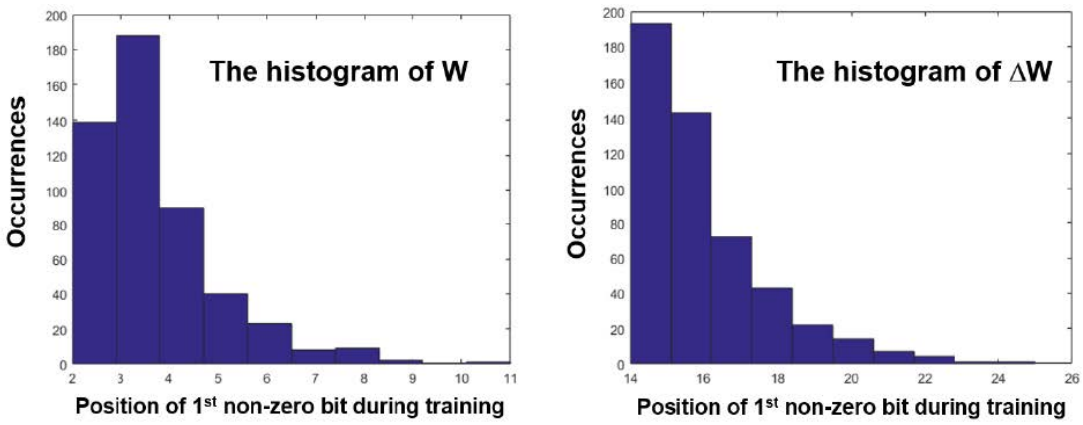


Figure 38. The position of the first non-zero bit in a LeNet-5 CNN example (for MNIST dataset), for both W and ΔW in the convolution layer. ΔW is much smaller than W during the training. Similar behavior is observed in other deep learning examples

W and ΔW throughout the training iterations of LeNet-5 CNNLeCun *et al.* (1998). Fig. 38 presents the preliminary data in which ΔW is >10 bits smaller than W. Such a behavior is not an empirical coincidence, but a fundamental property of statistical training, in which each iteration to update W should be a small contribution and the statistics of many ΔW will be accumulated to determine the classification. We leverage this property in the proposed hybrid in-memory computing design to minimize the high write energy for NVM devices without accuracy loss.

This work devises the architectural schemes that employ typical NVM devices in IMC techniques. A hybrid in-memory computing (HIMC) scheme is proposed that includes (1) an analog computation array with multi-bit RRAMs for MSBs of weights and (2) a digital weight update array with traditional SRAM for LSBs of weights. This hybrid design is motivated by the fact that only MSBs of weights are needed for forward/backward propagation (FP/BP) of DNN training that targets low-precision inference. In contrast, typically weight updates are small values that mostly modify only LSBs of weights.

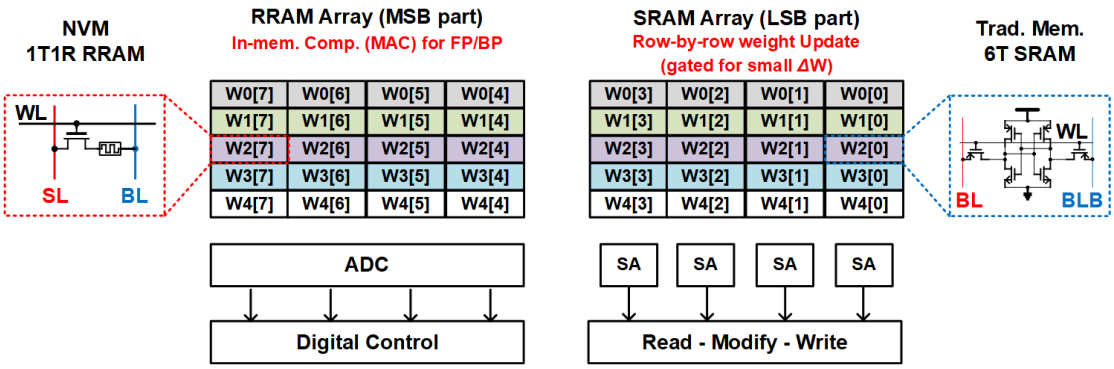


Figure 39. Hybrid in-memory computing architecture for accelerating DNN training: forward/backward propagation are performed by in-memory computing using RRAM array with analog storage (MSB part of weights), while weight update is implemented using traditional high-speed SRAM array (LSB part of weights).

The rest of the work will be delivered in the following order:

- Section 6.2 will propose the HIMC design of this work and illustrate the detailed functionalities of each section;
- Section 6.3 will discuss the experiment in software and hardware of this work.
- Section 6.4 will conclude the contribution of this work.

6.2 Proposed Design

As illustrated in Fig. 39, we propose a hybrid in-memory computing architecture that includes (1) an analog computation array with multi-bit RRAM for MSBs of weights and (2) a digital weight update array with traditional SRAM for LSBs of weights. This hybrid design is motivated by the fact that only MSBs of weights are needed for forward/backward propagation (FP/BP) of DNN training that targets low-precision inference [8], while typically weight updates are small values that mostly modify only LSBs of weights.

Fig. 40 (right) shows the logic diagram for the RRAM array for the MSB IMC process. The proposed design builds up based on the weight unit (WU) with 2T2R RRAM structure. This weight unit can store the signed information of the weight and the corresponding LRS/HRS will be connected in the positive column source line (SL) and negative column source line. During the programming stage, the bitline (BL) switch between BLs and ADC will be open and allow the SET/RESET/FORM process for RRAM cells, and during the MAC operation, these switches will be connected to enable ADC conversion. Every two adjacent columns will share one 4-bit SAR ADC to enable maximum parallelism during the computation. The truth table for the programming and MAC operation process of this RRAM array is shown in the left of Fig. 40. The activation is sent from the word line (WL), and the output of the IMC results is a ratio of the output BL voltage based on the connected LRS numbers in both positive SL and negative SL, followed by the equation as:

$$V_{BL} = V_{DD} \times \sum_{i=1}^{n} G_i \times A_i / \sum_{i=1}^{n} (G_{LRS}) \times A_i \tag{6.1}$$

Where G_i is the ith row conductance value on the positive column and A_i is the activation value at ith row.

While there have been many NVM-based in-memory computing works presented in the literature, the RRAM design proposed in Fig. 40 particularly focuses on the following advantages:

- Weight storage using analog programming of RRAM can substantially reduce the bitcell/ADC overhead for multi-bit precision in-memory computing;
- Enabling high parallel FP/BP computations by leveraging parallel activations of multiple rows with 2T2R RRAM cell structure;

Mode	INPUT (IN)				Weight (W)			Product
	BL	WL	SLO	SLE	DIG	POS	NEG	(IN X W)
PROG	gnd	V_{WL}	V_{WR}		N/A	LRS		N/A
	V_{WR}	V_{WL}	gnd		N/A	HRS		N/A
MAC	V_{ADC}	+1	vdd	gnd	+1	LRS	HRS	+1
		+1			-1	HRS	LRS	-1
		+1			0	HRS	HRS	0
		0			0	HRS	HRS	0

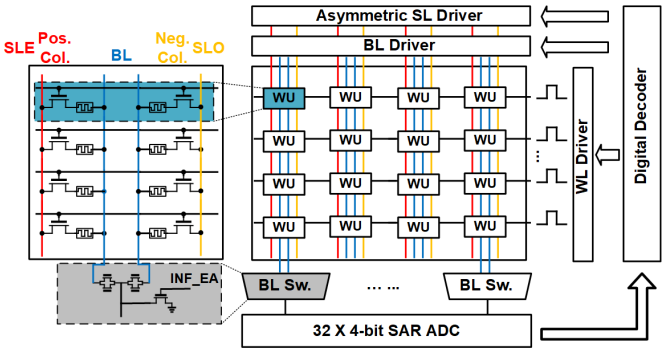


Figure 40. left: truth-table for the proposed 2T2R RRAM cell computation logic; right: Array-level diagram for the MSB RRAM block and detailed zoom-in for the RRAM array organization.

- Simplified signed computation of MAC without the requirement of dummy array, or complex subtraction after the individual column read-out, saving the cost of heavily needed ADCs in analog computing design;
- BL switches prevent the potential non-ideal disturbance during the programming from ADCs, and allow the symmetric sensing balance between two columns during MAC operations.

As shown in Equation. 6.1, the output setup reflects the ratio of the positive column conductance and one of the negative columns, which requires information about the activation from rows to obtain the precise value based on the ADC output. Therefore, this work introduces the Input-MAC-Voltage Tuning (IMVT) method to run post-processing inside the chip. The idea of the IMVT is introduced in Fig. 41. Besides the ADC output reflecting the voltage in BL, extra bits are calculated based on the numbers of "1" for the input vectors and the corresponding ADC output and sent out to the external system for further communication. The range of possible results is located with this extra information within the small blocks, to help expand

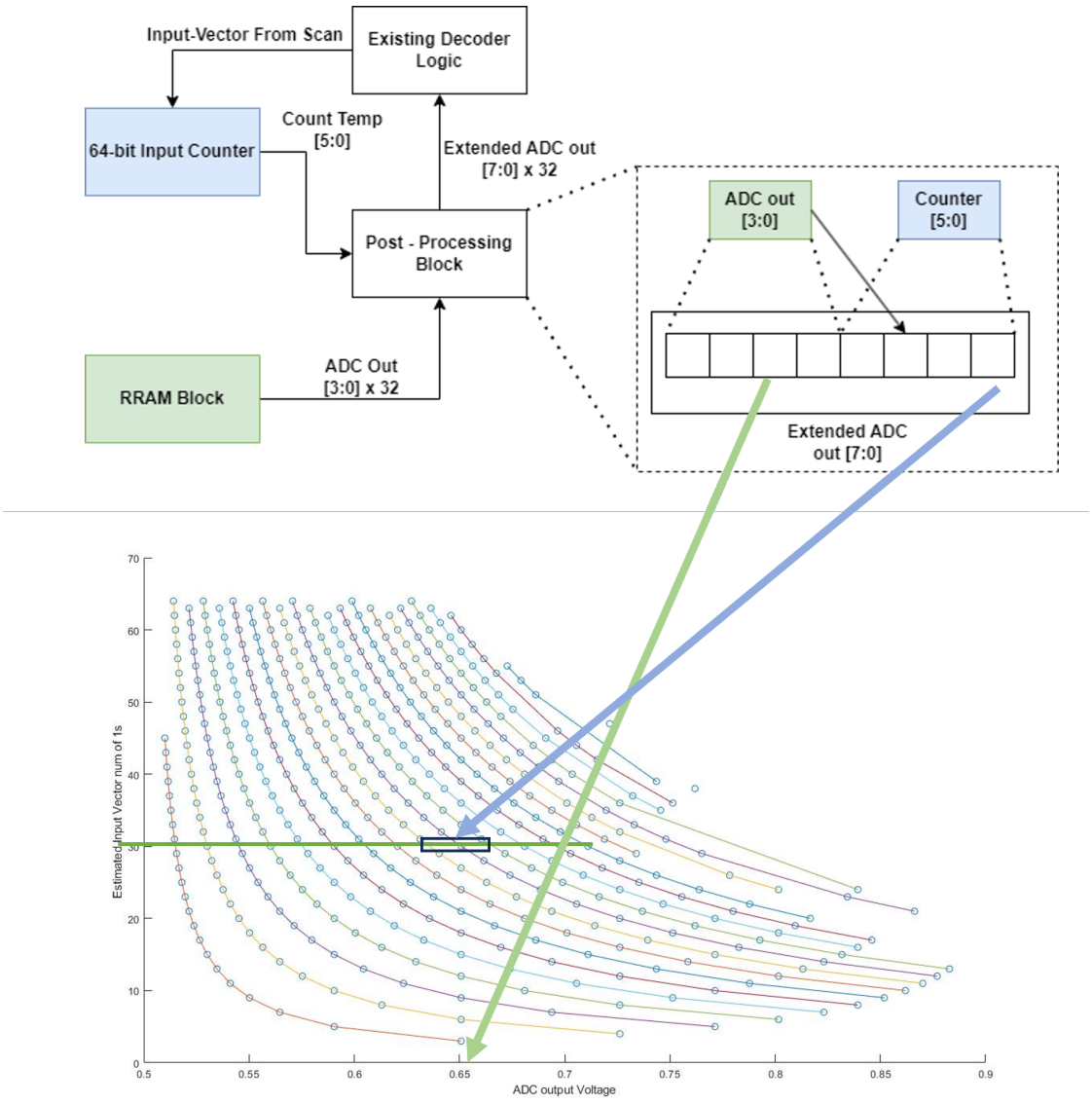


Figure 41. IMVT method for post-processing to obtain precise MAC results based on the ADC output.

the limited ADC precision for MAC operation, and determine the exact value for future processing.

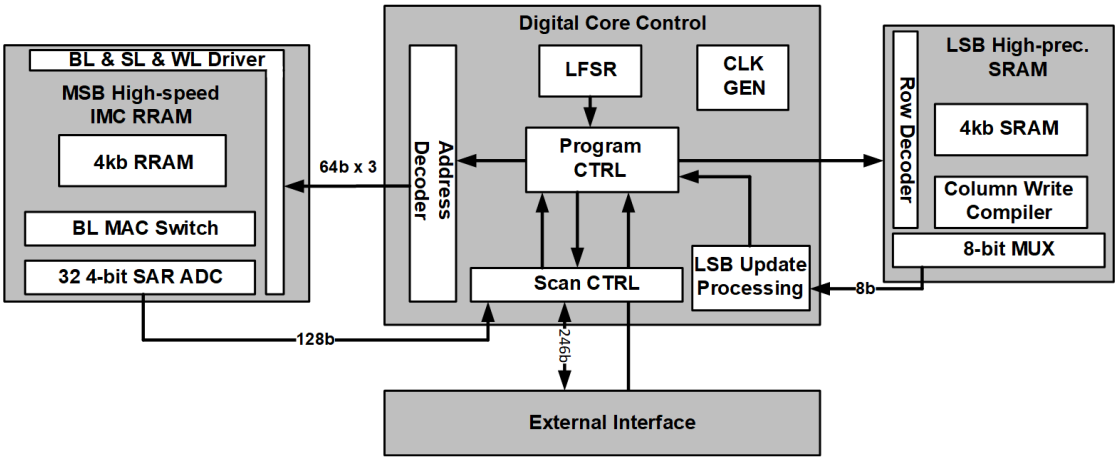


Figure 42. IMVT method for post-processing to obtain precise MAC results based on the ADC output.

6.3 Experiment Setup

The detailed block diagram for this HIMC chip design is shown in Fig. 42. The chip is taped out under a custom 65nm CMOS node, integrated with HfO_2 RRAM between M1 and M2. The tape-out process has cooperated with research and fabrication teams from SUNY Polytechnic Institute and the University of Tennessee, Knoxville. The chip is divided into three core sections: MSB blocks with high-speed IMC process under 4kb RRAM and 32 4-bit SAR ADC unit, digital core controller with data communication and post-processing IMVT session; and LSB high-precision 4kb SRAM for high precision weight updating. The updated tape-out design was completed in August 2023, and we are expecting to have some simulation-based paper submissions and some hardware measurement works in the future with other fellows. The tape-out die shot is shown in Fig. 43.

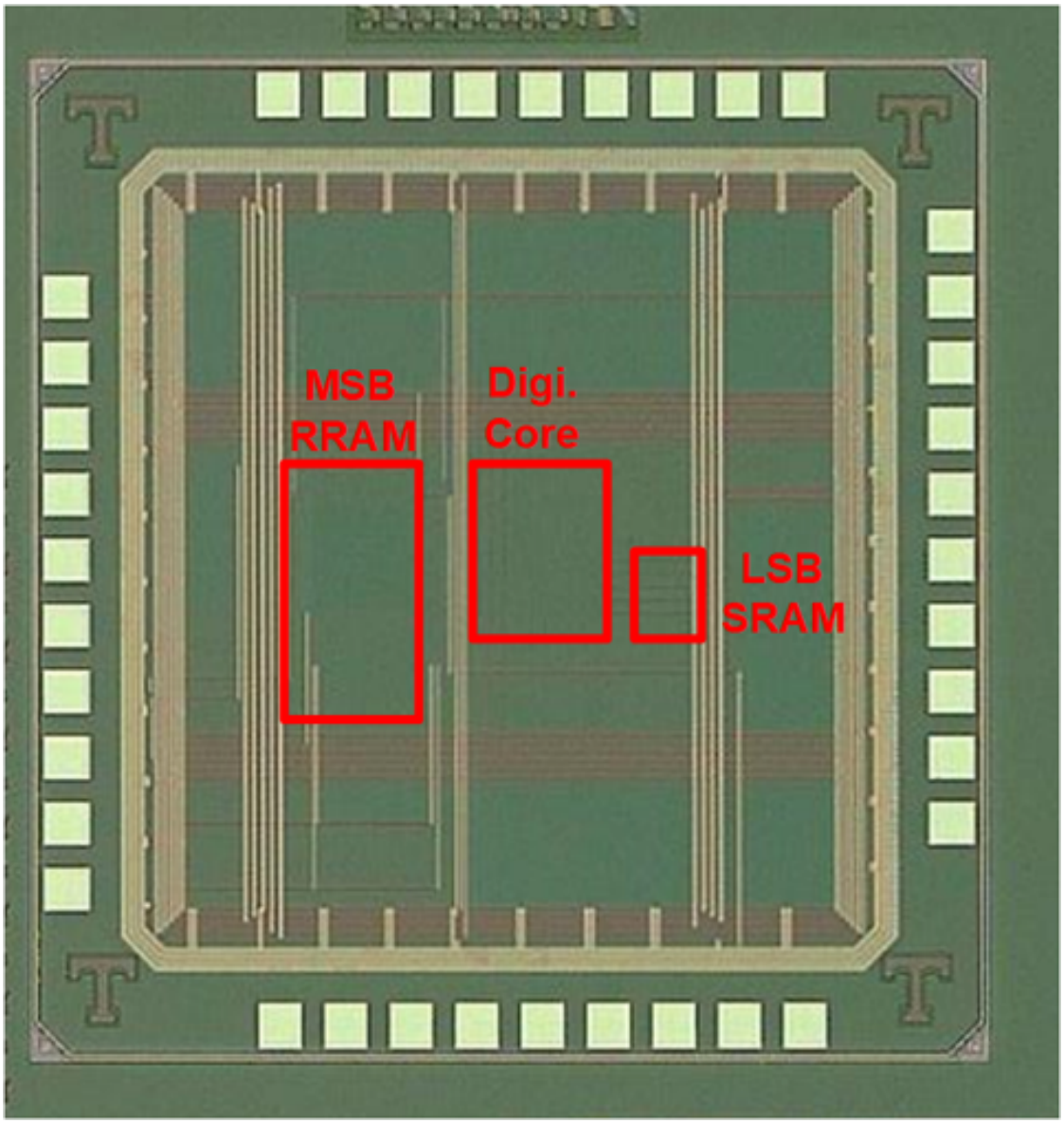


Figure 43. Tape-out die shots for the proposed 65nm HIMC chips.

6.4 Conclusion

This session introduces the framework for hybrid in-memory computing and the overall architecture for the HIMC process. RRAM array design with a 2T2R weight unit for the signed weight process is proposed, with the assistance of IMVT post-

processing in the digital controller blocks. The tape-out chip prototype has been sent out for fabrication, and we may expect future paper submission with further simulation or measurement-based analysis.

Chapter 7

FERROELECTRIC CAPACITIVE MATERIALS AND DEVICES FOR NEXT GENERATION AI HARDWARE

Analog computing-in-memory (CIM) with a emerging resistive nonvolatile memory(NVM) such as FeFET, RRAM Wan *et al.* (2022); Ye *et al.* (2022), PCRAM Khaddam-Aljameh *et al.* (2021), MRAM Deaville *et al.* (2021) have been developed and successfully improved energy-efficiency by reducing the data movement and performing the massive parallel matrix multiplication and accumulation (MAC) with $O(1)$ complexity. However, this approach encounter several challenges, which can prevent the achievement of energy efficiency compared to state-of-the-art SRAM based CIM. One of the main challenges is the static power consumption associated with the MAC operation based on Kirchhoff's Law. Also, large current flow due to the low resistance state in NVM induce IR drop along wires and column MUX in peripheral circuit, resulting in a degradation in computing accuracy. Additionally, the CIM's activation typically have a distribution that is centralized at mean value within a given operating range. Hence, an ADC having a linear relationship between the input and output can increase the computational cost in terms of power and area.

In this work, a ferroelectric $Hf_{0.5}Zr_{0.5}O_2$ (HZO)-based capacitor array (FCA) is employed to represent the programmable weight within CIM macro and perform the analog MAC operation according to the charge redistribution (CR). The usage of capacitor provide a solutions naturally for addressing issue such as static power consumption and IR drop. Our previous study Hur *et al.* (2022) demonstrated that the two-terminal FCA achieves an on/off ratio exceeding 110% at near-zero bias,

enabling read-disturbance-free readout. The distinctive memory states at zero bias, as shown in Figure 1, result from the asymmetric capacitance-voltage characteristics and can be attributed to the presence of oxygen vacancies at the bottom electrode of the FCA. These vacancies effectively pin nearby domains, induce varying domain-wall motions under different ferroelectric polarities Hur *et al.* (2022), and affects the capacitance values. Additionally, in Hur *et al.* (2022), experimental evidence showcases 8×8 array-level MAC operations with linear weighted sums and minimal read/write disturbance. Moreover, the investigation reveals a well-defined memory window even after subjecting the FCA to over 1000 strong 3V/1ms pulses and a projected retention to 10 years at 85°. However, it should be noted that the test setup employed discrete peripheral components. "On the other hand, the parasitic capacitor on the CR node (bit-line) can introduce an attenuation on the signal margin during the charge to voltage conversion. To minimize this effect, we employ a ring-amplifier (Ramp)-based Hung and Kuo (2016) switched capacitor (SC) integrator to enforce the CR node with virtual ground. Adapting Ramp offers several advantages over conventional operational amplifier. 1) Ramp operates as digital switch during slewing, resulting in dynamic power consumption. 2) Ramp's internal dynamic biasing eliminates the need for a global biasing circuit, further simplifying the design of column-parallel configuration. Lastly, we present an efficient non-linear SAR ADC scheme that incorporates power of two (PoT) quantization Przewlocka-Rus *et al.* (2022), specifically devised to match the distribution of CIM's activation. Compared to the conventional 5b SAR ADC, proposed PoT SAR ADC achieves 16× and 6.78× saving in terms of area and switching energy, respectively. Protype FCA-based CIM macro in 180nm CMOS perform the 16×8 analog MAC operation with an energy efficiency of 1.75 TOPS/W at 1.5-bit input (IN), 1.5-bit weight (W), and 5-bit output (O).

The proposed FCA-based CIM macro (Fig. 45) perform highly efficient and reliable analog MAC operation through a two-step process. During the multiplication phase, the word-line (WL) voltages corresponding to the INs are simultaneously sampled on the bottom plate of the FCA, while bit-line (BL) is connected to the common mode voltage (VCM). The stored charge on each cross-junction capacitor cell represents the product of the IN and W.

$$Q_{BL} = \sum_{i} (C_i^P - C_i^N) \times IN_i \times V_{read} + C_P \times V_{CM} \tag{7.1}$$

where CP denotes the total parasitic capacitance at the BL, which mainly arises from the long off-chip trace. And i is the index of WL. In the accumulation phase, the WLs are connected to VCM and the BL is set the bias by virtual ground, the voltage across the capacitor in FCA become zero. After the accumulated charge (QBL) is transferred into feedback capacitor (CF) in SC integrator. From the law of charge conservation, the output voltage of SC integrator is given by

$$V_{OUT} = V_{CM} + \sum_{i} \frac{(C_i^P - C_i^N)}{C_F} \times IN_i \times V_{READ} \tag{7.2}$$

Since the potential across CP changes from VCM to VCM, we can avoid the charge attenuation for the output voltage generation. When applying Ramp into SC feedback structure, the input DC level is not a key issue because the auto-zero CF in closed loop can hold the DC voltage difference during the multiplication phase. The schematic and performance of the Ramp are shown in Fig. 46. In WL decoder, the ternary input activation is converted into input voltages (VRP, VCM, VRN) through the integrated analog mux and are fed to the bottom of ferroelectric capacitor array, simultaneously. The proposed PoT SAR ADC (Fig. 46) compose of two identical capacitors, a dynamic comparator, and SAR logic. The output of SC integrator is

compared with the reference voltage capacitive DAC. Each reference voltage level is sequentially generated by pre-charge CPRE according to the output of comparator and then perform the charge sharing between CPRE and CDAC in order to monotonically add or subtract charge from CDAC. If MSB and MSB-1 are either 1,1 or 0,0 as shown in Fig. 47, the ADC no more A/D conversion for the remaining cycles. This is due to the detection of the given input-level as belong in the outer border of the conversion range. Otherwise, the MSB and MSB-1 has either 1,0 or 0,1, succeeding A/D conversion up to the LSB when the third and fourth comparison results are either 0,0 (for 1,0) or 1,1 (for 0,1). Through this process, the generated DOUTs exhibit fine precision at concentrated input levels, while sparse input levels are resulted in coarse conversion. While proposed SAR ADC introduce some minor computational errors, but utilizing a few number of capacitor and skipping of A/D conversion, we can achieve a substantial reduction in energy consumption. Moreover, because the proposed ADC realize the 10 quantization-levels with exploiting only two capacitors, it has higher robustness than conventional SAR ADC against the capacitor mismatch. As a result, the proposed technique improves the energy, area, and linearity of ADC by 6.78×, 16×, and 2.47×, respectively, compared to the conventional SAR ADC.

The FCA-based CIM macro was implemented in a 180nm process, and measured performance are shown in Fig. 48. In order to measure the transfer characteristics, a 16×8 FCA is projected into a 16×8 dummy capacitor array, which is realized inside the CMOS chip. The unit cell of the dummy capacitor had a fixed capacitance of 60 fF, and a total of 16 capacitors in a column were program-med with weight values of ±16, ±12, ±8, ±4. With the IN swept from -16 to 16, the output voltage of SC integrator was quantized using the proposed ADC with VRP=1.0V and VRN=0.8V. To evaluate the nonlinearity, the root mean square error (RMSE) was calculated in DOUT as

function of a column-wisely and IN-wisely. A testing of input sweeps is repeated in eight different chips, and results are extracted based on the 64 plots. Across all parameters, the worst case absolute nonlinearity in DOUT is 4.0 LSB. The impact of nonlinearity on the convolutional neural network verified by replacing the ideal transfer function with the transfer characteristic from measurements. The classification accuracy of 90.2% is obtained for CIFAR-10 dataset. During MAC operations with 12M Hz clock frequency, SC integrators, PoT SAR ADCs and WL decoder draw 2.6 mW, 0.96 mW, 0.79 mW, respectively from 1.8V power supply. The proposed ADC reduces the power consumption by 7.89× per CIFAR-10 classification, compared to conventional 5b SAR ADC. Fig. 49 show the comparison with state-of-the-art NVM CIM works. For a fair comparison, performance of prior works is scaled to a 22nm and to equivalent precision. The presented FCA-based CIM macro have the potential to achieves energy/area-efficiency when advanced technology. The die micrograph and each block deployment are shown in Fig. 50.

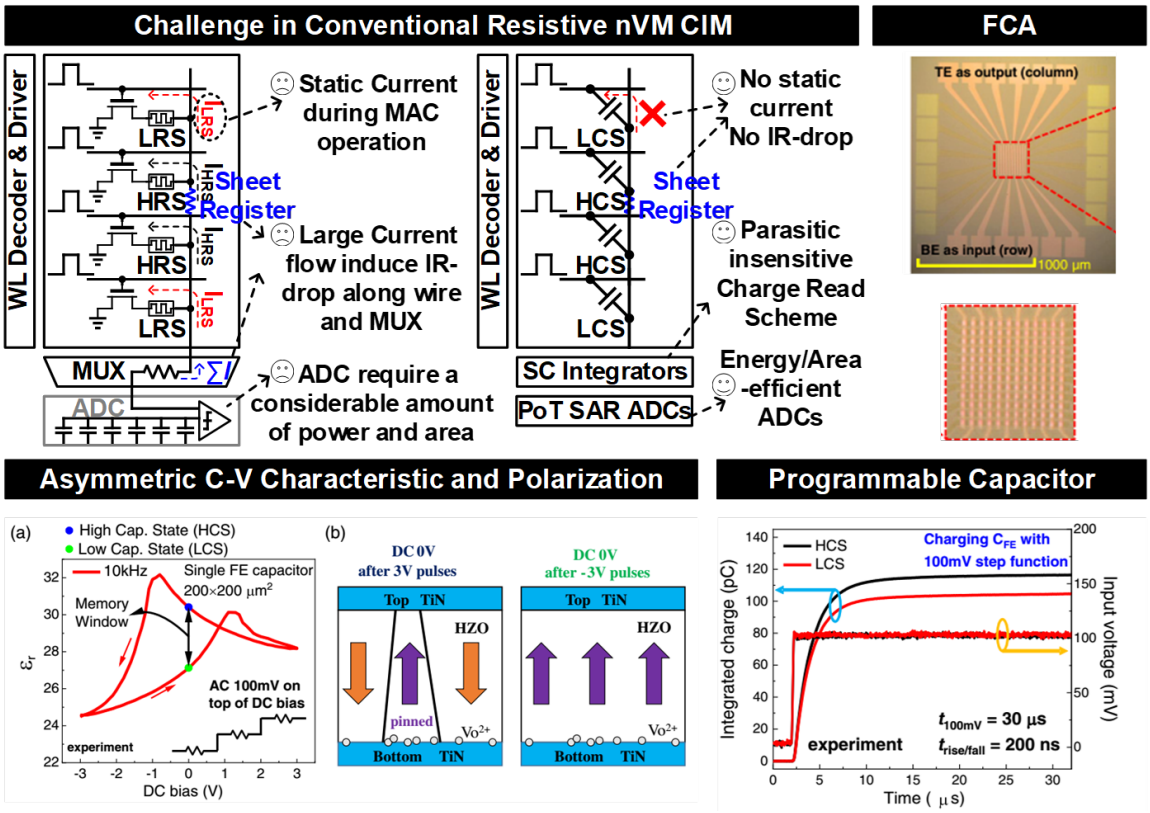
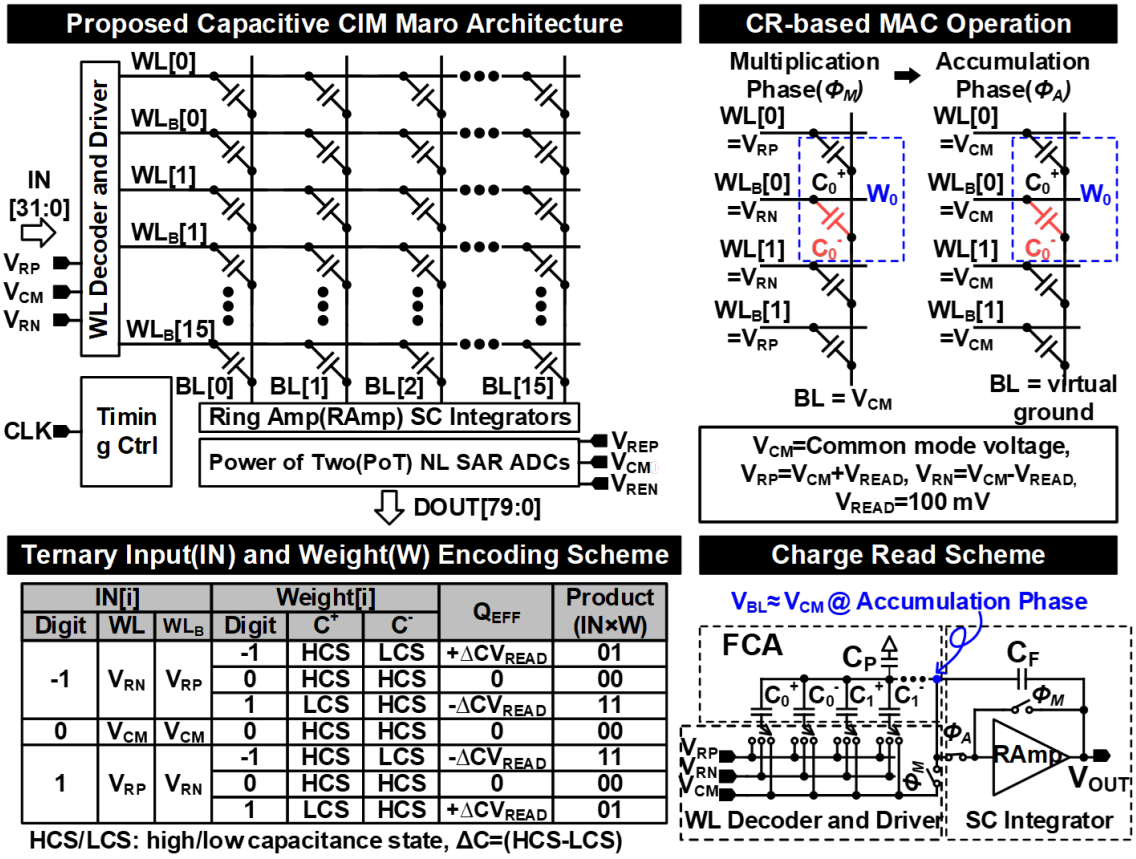


Figure 44. Challenges in resistive NVM-based CIM and motivations of proposed ferroelectric capacitor array (FCA) -based CIM are illustrated. Due to the polarization switching, FCA can be writable.



IN[i]			Weight[i]			Q_{EFF}	Product (IN×W)
Digit	WL	WL_B	Digit	C^+	C^-		
-1	V_{RN}	V_{RP}	-1	HCS	LCS	$+\Delta CV_{READ}$	01
			0	HCS	HCS	0	00
			1	LCS	HCS	$-\Delta CV_{READ}$	11
0	V_{CM}	V_{CM}	0	HCS	HCS	0	00
1	V_{RP}	V_{RN}	-1	HCS	LCS	$-\Delta CV_{READ}$	11
			0	HCS	HCS	0	00
			1	LCS	HCS	$+\Delta CV_{READ}$	01

Figure 45. (Top) Architecture of the proposed FCA-based CIM. Analog MAC operation is conducted in charge domain. (Bottom) Encoding of IN/W and parasitic-insensitive charge read schemes are shown.

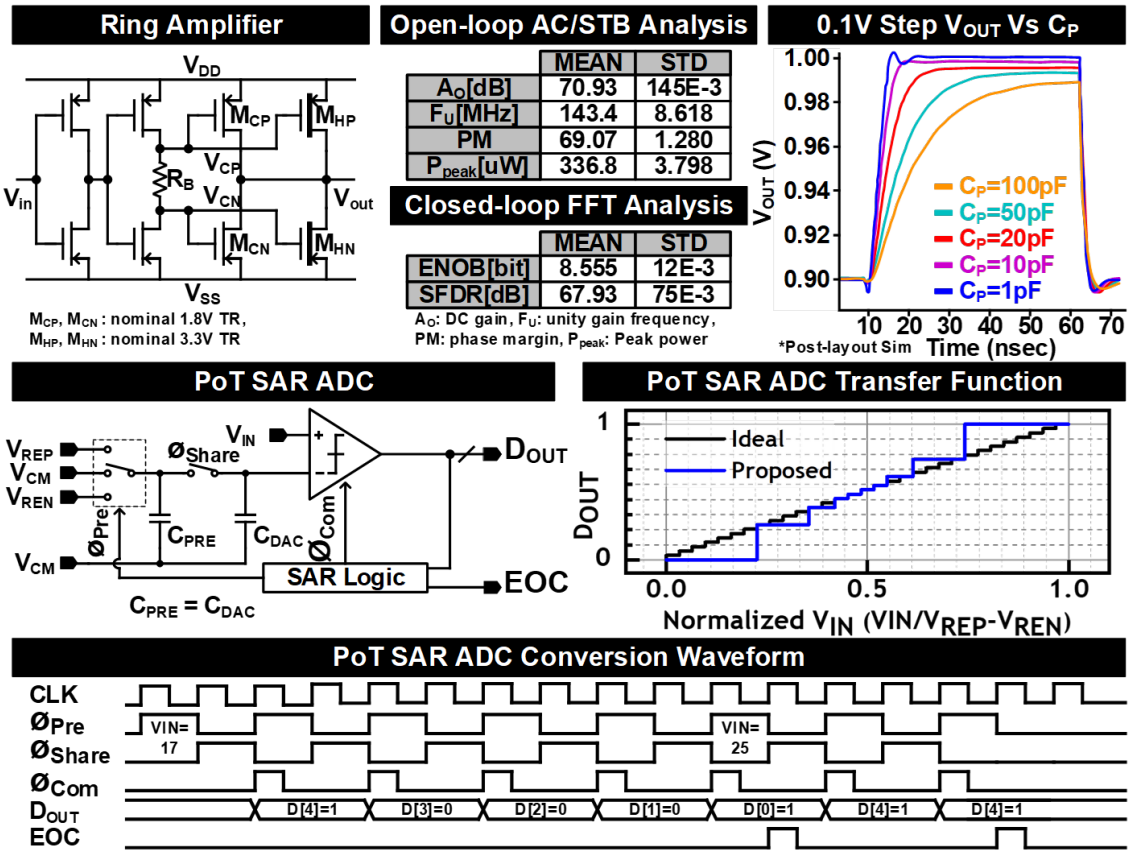
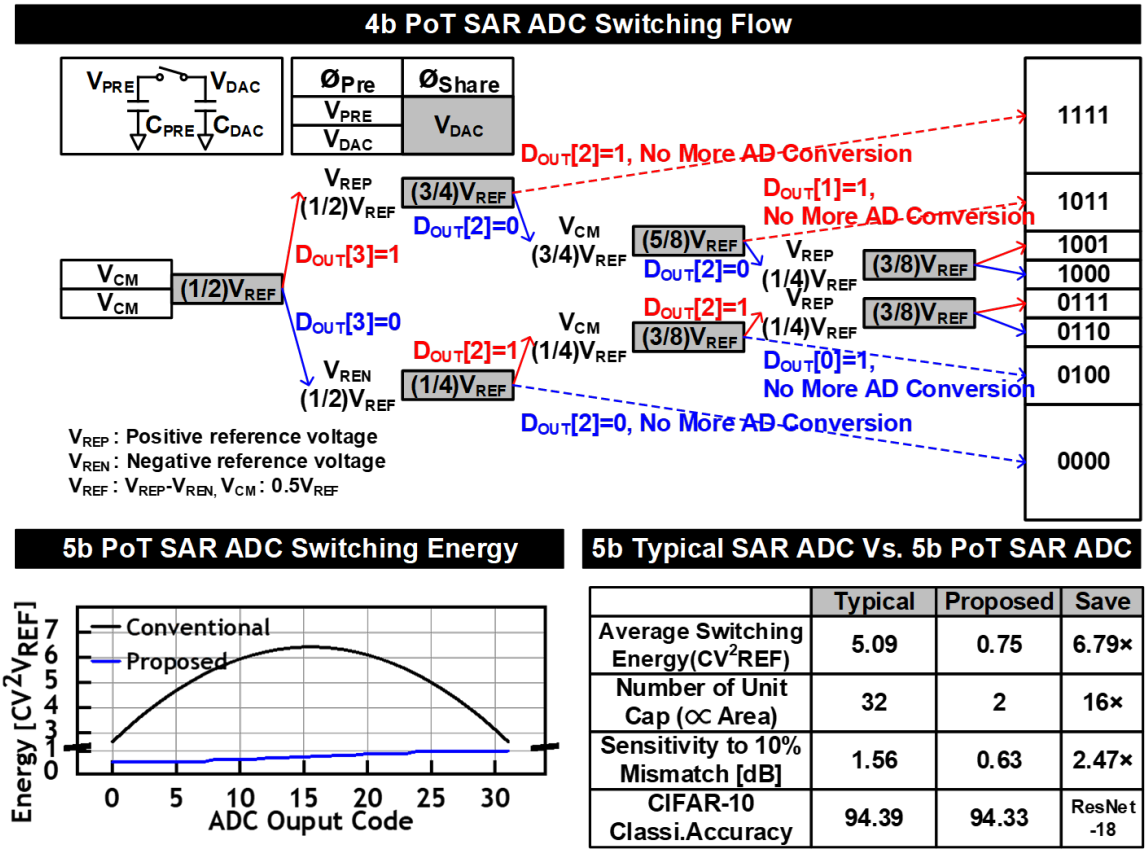


Figure 46. (Top) Schematic and post-layout simulation results for Ramp are represented. (Bottom) Proposed PoT SAR ADC has a non-linear behavior and well matched to the distribution of activation.



	Typical	Proposed	Save
Average Switching Energy(CV^2REF)	5.09	0.75	6.79×
Number of Unit Cap ($\propto$ Area)	32	2	16×
Sensitivity to 10% Mismatch [dB]	1.56	0.63	2.47×
CIFAR-10 Classi.Accuracy	94.39	94.33	ResNet -18

Figure 47. (Top) Example of switching behavior for 4bit PoT SAR. (Bottom) It improves the energy, area, and linearity of ADC to the conventional SAR ADC with minor computational error.

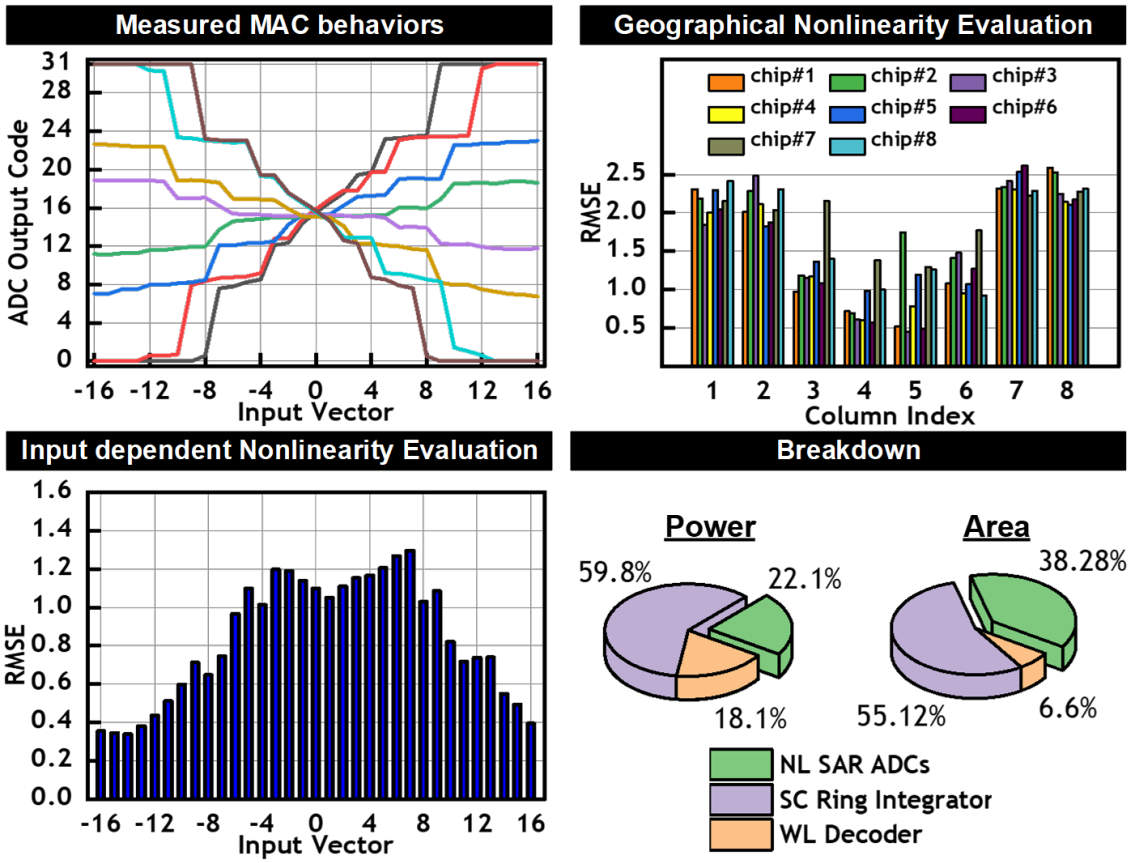


Figure 48. (Top) Measured transfer characteristics for a 16×8 FCA. Nonlinearity are quantified with RMSE as function of IN pattern and deployment. (Bottom)Power/Area breakdown of the prototype.

	[1] Nature'2021	[2] ASSCC'2022	[3] VLSI' 2021	[4] ESSCIRC'2021	This Work
Technology	130nm	28nm	14nm	22nm	180nm
Supply Voltage	1.8 V	0.9 V	0.8 V	1.0 V	1.8 V
Cell Type	RRAM	RRAM	PCRAM	MRAM	Ferroelectric Capacitor
Standard Bit Cell	NO	YES	YES	YES	NO
Input(I)/Weight(W) Precision (bit)	8I / 4W	1I / 3W	8I / 4W	1I / 1W	1.5I / 1.5W
ADC Resolution	10 bit	4 bit	8 bit	4 bit	5 bit
Energy-Efficiency(TOPS/W)	7	30.34	10.5	5.1	1.75
Area-Efficiency (GOPS/mm^2)	1.7	-	1587	758	597
Scaled Energy-Efficiency* (1b TOPS/W)	34.87	30.34	9.25	5.1	33.74
Classification Accuracy	CIFAR-10, 85.7%	CIFAR-10, 89.0%	CIFAR-10, 85.6%	CIFAR-10, 90.1%	CIFAR-10, 90.2%

*We predicted the energy-efficiency using 22nm technology by following the method presented in [1]

Figure 49. Comparison table with state-of-are NVM-based CIM.

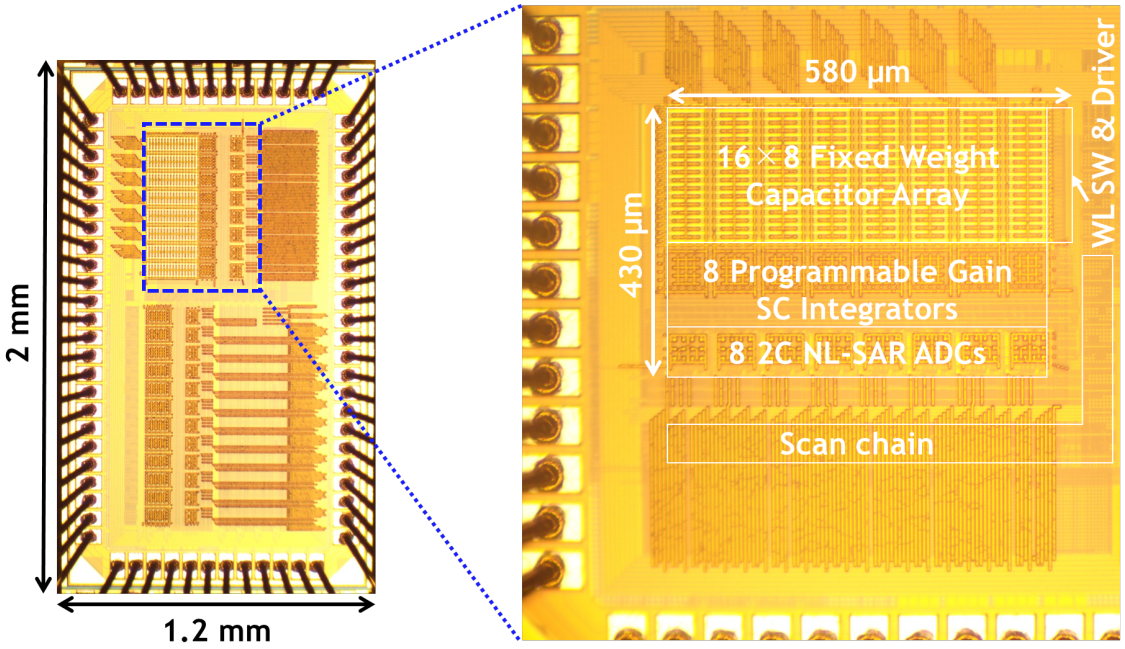


Figure 50. Die photo of prototype FCA-based CIM in CMOS 180nm technology.

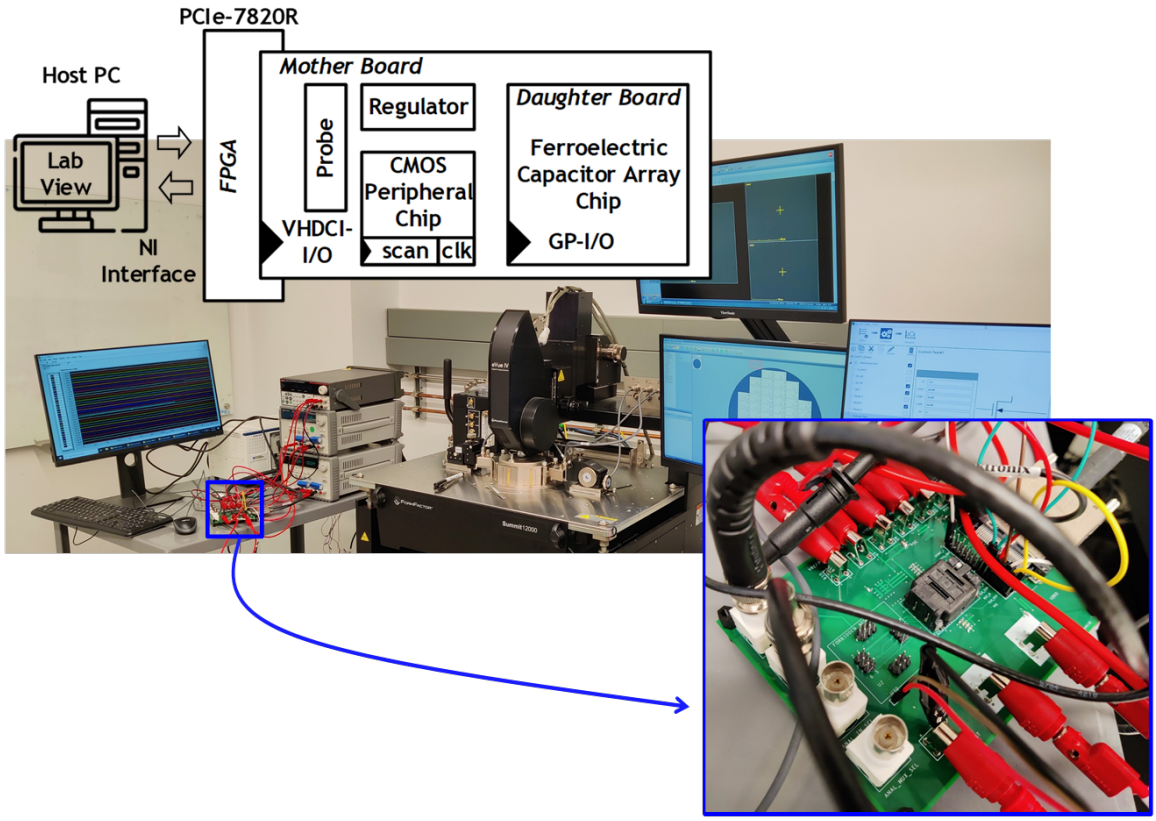


Figure 51. Test setup.

Chapter 8

CONCLUSION

This dissertation, first demonstrated RRAM based in-memory computing with 90nm CMOS prototype chips that monolithically integrated RRAM and CMOS in different vertical layers.Using device-/circuit-/algorithm-level techniques, both the energy-efficiency and density of binary RRAM based IMC hardware improved substantially, achieving up to 78.3 TOPS/W and 84.2% accuracy for CIFAR-10 dataset. Experiments with 2-bit RRAM demonstrate sufficient separation between four conductance levels, and show higher CNN accuracy up to 128×128 RRAM array size.

Then, this dissertation presents 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.

Moreover, this dissertation comprehensively characterized the relaxation effects of multi-level HfO_2 RRAM at array-level for in-memory computing hardware targeting DNN inference applications. Relaxation effects are noticeable at intermediate states of multi-level RRAM, but can be compensated using circuit-level (e.g. V_{ref} calibration after relaxation saturation) and algorithm-level (e.g. relaxation-aware DNN training for weight re-distribution) techniques that we proposed and demonstrated. Also, the non-ideality from the read disturb induced drift effect can be utilized to mitigate the

relaxation effect and could potentially enhance the DNN inference accuracy retention over time.

Next, this dissertation presents the progress on hybrid in-memory computing with non-volatile memory. This chapter includes the high-level idea for hybrid in-memory computing, modeling idea for the non-volatile memory HIMC for RRAM and SRAM, and also the progress of one tape-out HIMC chips under 65nm with RRAM and SRAM.

Finally, besides the RRAM based architecture, this dissertation presents the progress on ferroelectric capacitive devices for next-generation AI hardware.

In summary, this dissertation comprehensively discusses the novel applications and research development of the RRAM-based in-memory computing for area-/energy-efficient deep learning inference/training accelerators.

REFERENCES

Arikpo, I., F. Ogban and I. Eteng, "Von neumann architecture and modern computers", Global Journal of Mathematical Sciences **6**, 2, 97–103 (2007).

Chen, J. *et al.*, "A parallel multibit programing scheme with high precision for rram-based neuromorphic systems", IEEE Transactions on Electron Devices **67**, 5 (2020).

Chen, P.-Y., X. Peng and S. Yu, "Neurosim: A circuit-level macro model for benchmarking neuro-inspired architectures in online learning", IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems **37**, 12, 3067–3080 (2018).

Cheng, M. *et al.*, "Time: A training-in-memory architecture for rram-based deep neural networks", IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems **38**, 5 (2018).

Deaville, P., B. Zhang, L.-Y. Chen and N. Verma, "A maximally row-parallel mram in-memory-computing macro addressing readout circuit sensitivity and area", in "ESSCIRC 2021-IEEE 47th European Solid State Circuits Conference (ESSCIRC)", pp. 75–78 (IEEE, 2021).

Degraeve, R., A. Fantini, G. Gorine, P. Roussel, S. Clima, C. Chen, B. Govoreanu, L. Goux, D. Linten, M. Jurczak *et al.*, "Quantitative model for post-program instabilities in filamentary rram", in "2016 IEEE International Reliability Physics Symposium (IRPS)", pp. 6C–1 (IEEE, 2016).

Deng, L., G. Li, S. Han, L. Shi and Y. Xie, "Model Compression and Hardware Acceleration for Neural Networks: A Comprehensive Survey", Proc. of the IEEE **108**, 4 (2020).

Gao, L. *et al.*, "Programming protocol optimization for analog weight tuning in resistive memories", IEEE Electron Device Letters **36**, 11 (2015).

Giordano, M. *et al.*, "CHIMERA: A 0.92 TOPS, 2.2 TOPS/W Edge AI Accelerator with 2 MByte On-Chip Foundry Resistive RAM for Efficient Training and Inference", in "IEEE Symposium on VLSI Circuits", (2021).

Gonugondla, S. K. *et al.*, "SWIPE: Enhancing Robustness of ReRAM Crossbars for In-Memory Computing", in "IEEE/ACM ICCAD", (2020).

He, W., S. Yin, Y. Kim, X. Sun, J.-J. Kim, S. Yu and J. Seo, "2-bit-per-cell RRAM-based in-memory computing for area-/energy-efficient deep learning", IEEE Solid-State Circuits Letters **3**, 194–197 (2020).

Ho, C., S.-C. Chang, C.-Y. Huang, Y.-C. Chuang, S.-F. Lim, M.-H. Hsieh, S.-C. Chang and H.-H. Liao, "Integrated hfo 2-rram to achieve highly reliable, greener, faster, cost-effective, and scaled devices", in "2017 IEEE International Electron Devices Meeting (IEDM)", pp. 2–6 (IEEE, 2017).

Hsieh, E., M. Giordano, B. Hodson, A. Levy, S. Osekowsky, R. Radway, Y. Shih, W. Wan, T. Wu, X. Zheng *et al.*, "High-density multiple bits-per-cell 1t4r rram array with gradual set/reset and its effectiveness for deep learning", in "2019 IEEE International Electron Devices Meeting (IEDM)", pp. 35–6 (IEEE, 2019).

Hubara, I., M. Courbariaux, D. Soudry, R. El-Yaniv and Y. Bengio, "Binarized neural networks", Advances in neural information processing systems **29** (2016).

Hung, T.-C. and T.-H. Kuo, "A 4.86 mw 15-bit 22.5 ms/s pipelined adc with 74 db sndr in 90 nm cmos using averaging correlated level shifting technique", in "2016 IEEE Asian Solid-State Circuits Conference (A-SSCC)", pp. 161–164 (IEEE, 2016).

Hur, J., Y.-C. Luo, A. Lu, T.-H. Wang, S. Li, A. I. Khan and S. Yu, "Nonvolatile capacitive crossbar array for in-memory computing", Advanced Intelligent Systems **4**, 8, 2100258 (2022).

Kang, M. *et al.*, "An Energy-Efficient VLSI Architecture for Pattern Recognition via Deep Embedding of Computation in SRAM", in "IEEE ICASSP", (2014).

Khaddam-Aljameh, R., M. Stanisavljevic, J. F. Mas, G. Karunaratne, M. Braendli, F. Liu, A. Singh, S. M. Müller, U. Egger, A. Petropoulos *et al.*, "Hermes core–a 14nm cmos and pcm-based in-memory compute core using an array of 300ps/lsb linearized cco-based adcs and local digital processing", in "2021 Symposium on VLSI Circuits", pp. 1–2 (IEEE, 2021).

Khaddam-Aljameh, R. *et al.*, "HERMES-Core—A 1.59-TOPS/mm^2 PCM on 14-nm CMOS In-Memory Compute Core Using 300-ps/LSB Linearized CCO-Based ADCs", IEEE JSSC (2022).

Kim, Y., H. Kim, D. Ahn and J.-J. Kim, "Input-splitting of large neural networks for power-efficient accelerator with resistive crossbar memory array", in "Proceedings of the International Symposium on Low Power Electronics and Design", pp. 1–6 (2018a).

Kim, Y. *et al.*, "Input-Splitting of Large Neural Networks for Power-Efficient Accelerator with Resistive Crossbar Memory Array", in "ACM/IEEE ISLPED", (2018b).

Le, B. Q., A. Grossi, E. Vianello, T. Wu, G. Lama, E. Beigne, H.-S. P. Wong and S. Mitra, "Resistive ram with multiple bits per cell: Array-level demonstration of 3 bits per cell", IEEE Transactions on Electron Devices **66**, 1, 641–646 (2018).

Le, B. Q. *et al.*, "RADAR: A Fast and Energy-Efficient Programming Technique for Multiple Bits-Per-Cell RRAM Arrays", IEEE TED (2021).

LeCun, Y., Y. Bengio and G. Hinton, "Deep learning", nature **521**, 7553, 436–444 (2015).

LeCun, Y., L. Bottou, Y. Bengio and P. Haffner, "Gradient-based learning applied to document recognition", Proceedings of the IEEE **86**, 11, 2278–2324 (1998).

Li, S., D. Niu, K. T. Malladi, H. Zheng, B. Brennan and Y. Xie, "Drisa: A dram-based reconfigurable in-situ accelerator", in "Proceedings of the 50th Annual IEEE/ACM International Symposium on Microarchitecture", pp. 288–301 (2017).

Li, Z. *et al.*, "RRAM-DNN: An RRAM and Model-Compression Empowered All-Weights-On-Chip DNN Accelerator", IEEE Journal of Solid-State Circuits **56**, 4 (2021).

Liu, Q., 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)", pp. 500–502 (2020).

Meng, Z. *et al.*, "Write or not: Programming scheme optimization for rram-based neuromorphic computing", ACM/IEEE DAC (2022).

Mochida, R., 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", pp. 175–176 (IEEE, 2018).

Noguchi, H. *et al.*, "7.2 4mb stt-mram-based cache with memory-access-aware power optimization and write-verify-write / read-modify-write scheme", in "IEEE International Solid-State Circuits Conference (ISSCC)", (2016).

Peng, X., S. Huang, Y. Luo, X. Sun and S. Yu, "DNN+NeuroSim: An End-to-End Benchmarking Framework for Compute-in-Memory Accelerators with Versatile Device Technologies", in "IEEE International Electron Devices Meeting (IEDM)", pp. 32.5.1–32.5.4 (2019).

Przewlocka-Rus, D., S. S. Sarwar, H. E. Sumbul, Y. Li and B. De Salvo, "Power-of-two quantization for low bitwidth and hardware compliant neural networks", arXiv preprint arXiv:2203.05025 (2022).

Rossi, D. *et al.*, "4.4 A 1.3TOPS/W @ 32GOPS Fully Integrated 10-Core SoC for IoT End-Nodes with 1.7W Cognitive Wake-Up From MRAM-Based State-Retentive Sleep Mode", in "IEEE ISSCC", (2021).

Seshadri, V., D. Lee, T. Mullins, H. Hassan, A. Boroumand, J. Kim, M. A. Kozuch, O. Mutlu, P. B. Gibbons and T. C. Mowry, "Ambit: In-memory accelerator for bulk bitwise operations using commodity dram technology", in "Proceedings of the 50th Annual IEEE/ACM International Symposium on Microarchitecture", pp. 273–287 (2017).

Shim, W., Y. Luo, J.-S. Seo and S. Yu, "Investigation of read disturb and bipolar read scheme on multilevel rram-based deep learning inference engine", IEEE Transactions on Electron Devices **67**, 6, 2318–2323 (2020a).

Shim, W. *et al.*, "Two-step write–verify scheme and impact of the read noise in multilevel rram-based inference engine", Semiconductor Science and Technology **35**, 11, 115026 (2020b).

Shulaker, M. M., G. Hills, R. S. Park, R. T. Howe, K. Saraswat, H.-S. P. Wong and S. Mitra, "Three-dimensional integration of nanotechnologies for computing and data storage on a single chip", Nature **547**, 7661, 74–78 (2017).

Si, X., J.-J. Chen, Y.-N. Tu, W.-H. Huang, J.-H. Wang, Y.-C. Chiu, W.-C. Wei, S.-Y. Wu, X. Sun, R. Liu *et al.*, "24.5 a twin-8t sram computation-in-memory macro for multiple-bit cnn-based machine learning", in "2019 IEEE International Solid-State Circuits Conference-(ISSCC)", pp. 396–398 (IEEE, 2019).

Sun, X., S. Yin, X. Peng, R. Liu, J.-s. Seo and S. Yu, "Xnor-rram: A scalable and parallel resistive synaptic architecture for binary neural networks", in "2018 Design, Automation & Test in Europe Conference & Exhibition (DATE)", pp. 1423–1428 (IEEE, 2018).

Sze, V. *et al.*, "Efficient Processing of Deep Neural Networks: A Tutorial and Survey", Proceedings of the IEEE **105**, 12, 2295–2329 (2017).

Valavi, H., P. J. Ramadge, E. Nestler and N. Verma, "A 64-tile 2.4-mb in-memory-computing cnn accelerator employing charge-domain compute", IEEE Journal of Solid-State Circuits **54**, 6, 1789–1799 (2019).

Verma, N., H. Jia, H. Valavi, Y. Tang, M. Ozatay, L. Chen, B. Zhang and P. Deaville, "In-Memory Computing: Advances and Prospects", IEEE Solid-State Circuits Magazine **11**, 3, 43–55 (2019).

Wan, W., R. Kubendran, S. B. Eryilmaz, W. Zhang, Y. Liao, D. Wu, S. Deiss, B. Gao, P. Raina, S. Joshi *et al.*, "33.1 a 74 tmacs/w cmos-rram neurosynaptic

core with dynamically reconfigurable dataflow and in-situ transposable weights for probabilistic graphical models", in "2020 IEEE International Solid-State Circuits Conference-(ISSCC)", pp. 498–500 (IEEE, 2020).

Wan, W., R. Kubendran, C. Schaefer, S. B. Eryilmaz, W. Zhang, D. Wu, S. Deiss, P. Raina, H. Qian, B. Gao *et al.*, "A compute-in-memory chip based on resistive random-access memory", Nature **608**, 7923, 504–512 (2022).

Wang, C., H. Wu, B. Gao, L. Dai, N. Deng, D. Sekar, Z. Lu, M. Kellam, G. Bronner and H. Qian, "Relaxation effect in rram arrays: Demonstration and characteristics", IEEE Electron Device Letters **37**, 2, 182–185 (2015).

Wang, C. *et al.*, "Improving Multilevel Writes on Vertical 3-D Cross-point Resistive Memory", IEEE TCAD (2020).

Wu, S., G. Li, F. Chen and L. Shi, "Training and inference with integers in deep neural networks", arXiv preprint arXiv:1802.04680 (2018).

Xu, X., Y. Ding, S. X. Hu, M. Niemier, J. Cong, Y. Hu and Y. Shi, "Scaling for edge inference of deep neural networks", Nature Electronics **1**, 4, 216–222 (2018).

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

Xue, C.-X. *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", in "IEEE ISSCC", (2020).

Yan, B., Q. Yang, W.-H. Chen, K.-T. Chang, J.-W. Su, C.-H. Hsu, S.-H. Li, H.-Y. Lee, S.-S. Sheu, M.-S. Ho *et al.*, "Rram-based spiking nonvolatile computing-in-memory processing engine with precision-configurable in situ nonlinear activation", in "2019 Symposium on VLSI Technology", pp. T86–T87 (IEEE, 2019).

Yang, J. J., D. B. Strukov and D. R. Stewart, "Memristive devices for computing", Nature nanotechnology **8**, 1, 13–24 (2013).

Ye, W., C. Dou, L. Wang, Z. Zhou, J. An, W. Li, H. Gao, X. Xu, J. Yue, J. Yang *et al.*, "A 28nm hybrid 2t1r rram computing-in-memory macro for energy-efficient ai edge inference", in "2022 IEEE Asian Solid-State Circuits Conference (A-SSCC)", pp. 2–4 (IEEE, 2022).

Yin, S., Z. Jiang, J.-S. Seo and M. Seok, "Xnor-sram: In-memory computing sram macro for binary/ternary deep neural networks", IEEE Journal of Solid-State Circuits **55**, 6, 1733–1743 (2020a).

Yin, S., Y. Kim, X. Han, H. Barnaby, S. Yu, Y. Luo, W. He, X. Sun, J.-J. Kim and J. Seo, "Monolithically integrated RRAM-and CMOS-based in-memory computing optimizations for efficient deep learning", IEEE Micro **39**, 6, 54–63 (2019).

Yin, S., X. Sun, S. Yu and J.-s. Seo, "High-throughput in-memory computing for binary deep neural networks with monolithically integrated rram and 90-nm cmos", IEEE Transactions on Electron Devices **67**, 10, 4185–4192 (2020b).

Yin, S. *et al.*, "High-Throughput In-Memory Computing for Binary Deep Neural Networks With Monolithically Integrated RRAM and 90-nm CMOS", IEEE TED (2020c).

Yin, S. *et al.*, "PIMCA: A 3.4-Mb Programmable In-Memory Computing Accelerator in 28nm for On-Chip DNN Inference", in "Symp. VLSI", (2021).

Yoon, J.-H. *et al.*, "A 40-nm 118.44-TOPS/W Voltage-Sensing Compute-in-Memory RRAM Macro With Write Verification and Multi-Bit Encoding", IEEE JSSC (2022).

Zhang, G. L. *et al.*, "An Efficient Programming Framework for Memristor-based Neuromorphic Computing", in "DATE", (2021).

Zhao, M. *et al.*, "Characterizing Endurance Degradation of Incremental Switching in Analog RRAM for Neuromorphic Systems", in "IEEE IEDM", (2018).

Zimmer, B., R. Venkatesan, Y. S. Shao, J. Clemons, M. Fojtik, N. Jiang, B. Keller, A. Klinefelter, N. Pinckney, P. Raina, S. G. Tell, Y. Zhang, W. J. Dally, J. S. Emer, C. T. Gray, S. W. Keckler and B. Khailany, "A 0.32–128 TOPS, Scalable Multi-Chip-Module-Based Deep Neural Network Inference Accelerator With Ground-Referenced Signaling in 16 nm", IEEE Journal of Solid-State Circuits (JSSC) **55**, 4, 920–932 (2020).

CHAPTER 9

PREVIOUS PUBLISHED WORKS

Three chapters are based on the previously published works of the first author, listed below:

Chapter 3: W. He et al., "2-Bit-Per-Cell RRAM-Based In-Memory Computing for Area-/Energy-Efficient Deep Learning," in IEEE Solid-State Circuits Letters, vol. 3, pp. 194-197, 2020, doi: 10.1109/LSSC.2020.3010795.

Chapter 4: W. He et al., "Characterization and Mitigation of Relaxation Effects on Multi-level RRAM based In-Memory Computing," 2021 IEEE International Reliability Physics Symposium (IRPS), Monterey, CA, USA, 2021, pp. 1-7, doi: 10.1109/IRPS46558.2021.9405228.

Chapter 5: W. He, J. Meng, S. K. Gonugondla, S. Yu, N. R. Shanbhag and J. -s. Seo, "PRIVE: Efficient RRAM Programming with Chip Verification for RRAM-based In-Memory Computing Acceleration," 2023 Design, Automation, Test in Europe Conference and Exhibition (DATE), Antwerp, Belgium, 2023, pp. 1-6, doi: 10.23919/DATE56975.2023.10137266.

CHAPTER 10

PERMISSION STATEMENT FOR THE PREVIOUS PUBLISHED WORKS

The author of this dissertation thesis, Wangxin He, confirms that all included published works have been granted permission by all co-authors.