Hybrid Memristor-CMOS Computer for Artificial Intelligence: from Devices to Systems

by

Seung Hwan Lee

A dissertation submitted in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
(Electrical Engineering)
in the University of Michigan
2020

Doctoral Committee:

Professor Wei D. Lu, Chair
Professor Michael P. Flynn
Professor Cagliyan Kurdak
Associate Professor Zhengya Zhang

Seung Hwan Lee

seulee@umich.edu

ORCID iD: 0000-0002-5839-6533

© Seung Hwan Lee 2020

Acknowledgements

Firstly, I would like to express my sincere gratitude to my advisor, Prof. Wei D. Lu, who has always given me insightful suggestions and constant support throughout my Ph.D. study and related research, for his patient, motivation, and immerse knowledge. His guidance helped me in all the time of research and writing of this thesis. I could not have imagined having a better advisor and mentor for my Ph.D. study.

I would also like to extend my deepest appreciation to my committee members for their valuable discussions: Prof. Michael P. Flynn, Prof. Zhengya Zhang, and Prof. Cagliyan Kurdak. I had the great pleasure and honor of working with Prof. Flynn and Prof. Zhang in the integrated memristor-CMOS chip project, which would not have been possible without their supports. Prof. Cagliyan Kurdak has also provided many valuable insights into my research. All the professors have given useful suggestions for my thesis.

Furthermore, I am also grateful to my former and current group members for their helpful discussions and timely assistance. I would especially like to thank Dr. Fuxi Cai, Dr. Xiaojian Zhu, and Dr. Mohammed Zidan, who helped me a lot in completing my research projects. I want to thank the former members: Dr. Chao Du, Dr. Wen Ma, Dr. Jihang Lee, Dr. Yeonjoo Jeong, and Dr. Jonghoon Shin. I also want to thank all current group members, especially John Moon, Qiwen Wang, Fan-Hsuan Meng, and Xinxin Wang, for their helpful discussions and assistance in completing my research projects.

I would especially like to thank my research collaborators from other groups who provided great help with my Ph.D. researches. I would especially like to thank Justin M. Correll, Seongjong Lee, and Dr. Yong Lim from Prof. Flynn's group as well as Vishishtha Bothra and Jerry Zhu from Prof. Zhang's group, who designed the system architecture and CMOS circuit and have provided me great and constant assistance in the integrated chip project. I would also like to thank Dr. Xinyi Li and Prof. Huaqiang Wu from Tsinghua University, who have provided me great and constant assistance in the 1T1R project.

I would also like to thank the Lurie Nanofabrication Facility (LNF) staff: Sandrine Martin, Matthew Oonk, Vishva Ray, Gregory Allion, David Sebastian, Anthony Sebastian, for their technical support in my device integration.

And last but not least, I would like to express my deepest gratitude to my family, especially my wife Hyunjeong Lee and two sons, Do Hyeon Lee and Yujun Lee, as well as my mom and dad, for their relentless support and encouragement during my course of study. This accomplishment would not have been possible without them. Thank you very much.

Table of Contents

Acknowledgements ii

List of Figures ix

List of Tables xiii

Abstract xiv

Chapter 1 Introduction 1

1.1 The slowing down of Moore's law and the von-Neumann Bottleneck 1

1.2 Neuromorphic computing 2

1.3 Memristor device 3

1.3.1 Oxygen-ion based memristor devices (VCM) 5

1.3.2 Metal-ion based memristor devices (ECM) 6

1.4 Memristor structure: From single device to 3D integration 9

1.5 Memristor Crossbar Network for Neuromorphic (DNNs & SNNs) 10

1.6 Organization of Dissertation 14

Chapter 2 A Quantitative, Dynamic Memristor/RRAM Model 21

2.1 COMSOL Multiphysics and finite element method (FEM) ... 22

2.2 Device structure and dynamic model for simulation ... 25

2.2.1 Tantalum-oxide (TaO_x) based memristor ... 25

2.2.2 Integration of the tantalum-oxide (TaO_x) based memristor on-chip ... 26

2.2.3 Device structure and physical model ... 27

2.2.4 physical parameters and modeling compliance effect (*Icc* and transistor) ... 30

2.3 Forming process ... 33

2.4 Reset and Set cycling (1R with I_{CC}) ... 37

2.5 1T1R simulation ... 43

2.6 Conclusion ... 47

Chapter 3 A Fully Integrated Memristor–CMOS System for Neuromorphic Computing ... 53

3.1 System architecture and circuit design (designed by Prof. Flynn's group & Prof. Zhang's group) ... 54

3.2 Data path during training and inference on chip ... 58

3.3 Integration of memristor array on CMOS chip ... 61

3.4 Task 1: Training and classification using a single-layer perceptron ... 66

3.5 Task 2: Sparse coding algorithm implementation ... 69

3.5.1 Locally competitive algorithm (LCA) 70

3.5.2 Implementation of Sparse coding on chip 72

3.6 Task 3: A multi-layer neural network: PCA + Classification 75

3.6.1 Principal component analysis (PCA) 76

3.6.2 2nd layer for classification (SLP) 79

3.6.3 Evaluation of the PCA + classifier network 81

3.7 Performance & area analysis of the integrated memristor/CMOS chip 83

3.7.1 Power estimate 83

3.7.2 Area efficiency 86

3.7.3 Optimization of Power and Area 87

3.8 Non-ideal effect of memristor/CMOS chip and future implementations 91

3.8.1 Device variability on integrated chip 91

3.8.2 Expected other issues for complex model and large datasets 96

3.8.3 Tiled architecture 97

3.9 Conclusion 99

Chapter 4 Short Term Memory and Reservoir Computing System 107

4.1 Short term memory effect on WO_x memristor 107

4.1.1 Short term dynamics on WO_x memristor 107

4.1.2 Integration of WOx device for short term memory 110

4.2 Reservoir computing (RC) with memristor devices 111
4.2.1 Classification task with the RC system 113
4.2.2 Other complex temporal data processing with the RC system 115
4.3 Conclusion 117
Chapter 5 Optimization of Memristor Devices and Systems, and Future Works .121
5.1 Device non-ideality issues for neuromorphic computing 121
5.2 Device optimization for online training 123
5.2.1 High entropy oxide (HEO) memristor 123
5.2.2 High entropy oxide (HEO) materials 124
5.2.3 Characterization of HEO device and Investigation of systematic trend 125
5.2.4 Future Plans: 127
5.3 1T1R Array for DNN Inference Implementation 129
5.3.1 Issue of online training with 1T1R system 129
5.3.2 1T1R array implementation for inference applications 131
5.3.3 Future Plans: 132
5.4 Deep neural network (DNN) accelerator based on tiled architecture 133
5.4.1 Tiled architecture implementation 134
5.4.2 Future Plans: 139

5.5 Summary 140

List of Figures

Figure 1-1: von-Neumann bottleneck. 2

Figure 1-2: Memristor as the fourth electrical element. 3

Figure 1-3: Hysteresis loop characteristic of memristor devices. 4

Figure 1-4: Memristor response to programming pulses. 4

Figure 1-5: Oxide-based memristor devices. 6

Figure 1-6: Metal ion-based memristor device switching mechanism. 7

Figure 1-7: Memristor devices and integrated systems. 10

Figure 1-8: Memristor Crossbar Network. 11

Figure 2-1: Finite element method (FEM). 24

Figure 2-2: Solution with Finite element method (FEM). 25

Figure 2-3: Tantalum oxide memristor during Forming and Set/Reset cycling. 27

Figure 2-4: Device geometry used in the simulation. 29

Figure 2-5: Parameters used in the proposed model. 31

Figure 2-6: Current modulation layer *I-V* behavior, without memristor devices. 33

Figure 2-7: Evolution of V_O concentration (n_D) and temperature (T) during DC Forming. 35

Figure 2-8: Forming process. 36

Figure 2-9: Variation of the Forming voltage originated from different initial states. 37

Figure 2-10: Simulated Reset process. 38

Figure 2-11: Two different filament growth mechanisms during Set with different I_{CC}s. 39

Figure 2-12: Simulated RS behavior with different I_{CC}s.....41

Figure 2-13: 2-D maps of n_D profile after Set, for different Set I_{CC}s.....42

Figure 2-14: Simulated LTP behaviors with incremental I_{CC}, for different Forming conditions..42

Figure 2-15: 1T1R characteristics with different gate voltage (V_G).....44

Figure 2-16: Schematic of the 1T1R structure. 45

Figure 3-1: Chip System Architecture and mixed signal interface design.....55

Figure 3-2: Schematic of 13b current-integrating ADC.....56

Figure 3-3: Pulsed-mode DAC scheme.....57

Figure 3-4: Data path during training and inference.....59

Figure 3-5: Fully integrated memristor/CMOS chip.....62

Figure 3-6: Cross-section schematic of the integrated chip. 63

Figure 3-7: Measurement results of the memristor devices in the array.....64

Figure 3-8: Schematic of the single-layer perceptron.....66

Figure 3-9: Training and test data set.....68

Figure 3-10: Classification results with SLP implementation.....69

Figure 3-11: Schematic of the LCA algorithm.....72

Figure 3-12: Dictionary elements based on horizontal and vertical bars.....73

Figure 3-13: Sparse coding results.....74

Figure 3-14: Additional examples. 74

Figure 3-15: Schematic of the bilayer neural network.....75

Figure 3-16: The results of 1st layer (PCA) implementation.....78

Figure 3-17: Classification results in the bilayer network. 79

Figure 3-18: AUC-ROC curve and F_1 score of the breast cancer task.....82

Figure 3-19: Schreier FOM for 180nm and 40nm ADCs. 85

Figure 3-20: Error rates from different models as a function of weight and activation quantization effects. 87

Figure 3-21: Quantization effect for common CNN models. 87

Figure 3-22: ADC design choices. 89

Figure 3-23: Shared ADC options. 89

Figure 3-24: ADC area efficiency 90

Figure 3-25: Line resistance effects. 92

Figure 3-26: Voltage loss effects in the memristor array. 95

Figure 3-27: Improvements with monolithic integration. 96

Figure 3-28: Tiled architecture based on small crossbar arrays. 98

Figure 3-29: Tiled architecture with high area efficiency. 98

Figure 4-1: Conductance decay in a WOx memristor. 108

Figure 4-2: Memristor's temporal response to a pulse train. 110

Figure 4-3: SEM image of the 32×32 WO_x memristor array. 111

Figure 4-4: Schematic of an RC system. 112

Figure 4-5: Reservoir for simple digit recognition. 113

Figure 4-6: Test set and measured reservoir states. 114

Figure 4-7: Long-term forecasting of Mackey–Glass time series using a reservoir system 116

Figure 5-1: The periodic table and adjacent 6 transition metals for high entropy oxide (HEO) memristor. 124

Figure 5-2: HEO target synthesis and deposition results. 126

Figure 5-3: Structure of HEO devices. 127

Figure 5-4: DC and pulse characteristics of HEO and Ta_2O_5 devices. 127

Figure 5-5: Analog behavior of typical 1R and 1T1R. 131

Figure 5-6: Test set-up for 1T1R array. 132

Figure 5-7: Different types of layers in DNN...133

Figure 5-8: Full system overview. 135

Figure 5-9: Optimized tiled architecture. 135

Figure 5-10: Schematic of the CNN model for MNIST..136

Figure 5-11: Weight mapping of the CNN to the 4 tiles. 137

Figure 5-12: Weight mapping..138

Figure 5-13: Model accuracy with lower precision ADC. 138

Figure 5-14: Model accuracy vs. ADC and weight precision. 139

List of Tables

Table 2-1: Material parameters and constants used in the proposed model. 30

Table 3-1: The parameters used in the SPICE simulation..94

Abstract

Neuromorphic computing systems, which aim to mimic the function and structure of the human brain, is a promising approach to overcome the limitations of conventional computing systems such as the von-Neumann bottleneck. Recently, memristors and memristor crossbars have been extensively studied for neuromorphic system implementations due to the ability of memristor devices to emulate biological synapses, thus providing benefits such as co-located memory/logic operations and massive parallelism. A memristor is a two-terminal device whose resistance is modulated by the history of external stimulation. The principle of the resistance modulation, or resistance switching (RS), for a typical oxide-based memristor, is based on oxygen vacancy (V_O) migration in the oxide layer through ion drift and diffusion. When applied in computing systems, the memristor is often formed in a crossbar structure and used to perform vector-matrix multiplication (VMM) operations. Since the values in the matrix can be stored as the device conductance values of the crossbar array, when an input vector is applied as voltage pulses with different pulse amplitudes or different pulse widths to the rows of the crossbar, the currents or charges collected at the columns of the crossbar correspond to the resulting VMM outputs, following Ohm's law and Kirchhoff's current law. This approach makes it possible to use physics to execute direct computing of this data-intensive task, both in-memory and in parallel in a single step.

This dissertation presents my studies on the memristor device characteristics, integration, optimization, modeling, and directly integrated hybrid memristor/CMOS systems for neuromorphic computing applications.

First of all, I will present a comprehensive physical model of the TaO_x-based memristor device where the internal parameters including electric field (E), temperature (T), and V_O concentration (n_D) are self-consistently solved to accurately describe the device operation. Starting from the initial Forming process, the model quantitatively captures the dynamic RS behavior, and can reliably reproduce Set/Reset cycling in a self-consistent manner. Beyond clarifying the nature of the Forming and Set/Reset processes, a bulk-like doping effect was revealed by the model during Set and supported by experimental results. This phenomenon can lead to linear analog conductance modulation with a large dynamic range, which is very beneficial for low-power neuromorphic computing applications.

Second, an integrated memristor/CMOS system consisting of a 54×108 passive memristor crossbar array directly fabricated on a CMOS chip is presented. The system includes all necessary analog/digital circuitry (including analog-digital converters (ADCs) and digital-analog converters (DACs)), digital buses, and a programmable processor to control the digital and analog components to form a complete hardware system for neuromorphic computing applications. With the fully-integrated and reprogrammable chip, we experimentally demonstrated three popular models – a perceptron network, a sparse coding network, and a bilayer principal component analysis system with an unsupervised feature extraction layer and a supervised classification layer – all on the same chip.

Beyond VMM operations, the internal dynamics of memristors allow the system to natively process temporal features in the input data. Specifically, a WO_x-based memristor with short-term memory effect caused by spontaneous oxygen vacancy diffusion was utilized to implement a reservoir computing system to process temporal information. The spatial information of a digit image can be converted into streaming inputs fed into the memristor reservoir, leading to 100%

accuracy for simple 4×5 digit recognition and 88.1% accuracy for the MNIST data set. The system was also employed for solving other nonlinear tasks such as emulating a second-order nonlinear system.

Other attempts to improve the device characteristics, such as increasing the dynamic range and retention, and to scale up the system for larger model implementations are also investigated. Outlooks to future studies that can lead to practical, efficient neuromorphic hardware systems will be discussed in the end.

Chapter 1

Introduction

1.1 The slowing down of Moore's law and the von-Neumann Bottleneck

Machine learning (ML) has recently generated strong interest, with performances approaching or in limited circumstances exceeding human-level performance for specialized tasks such as object recognition[1,2], speech recognition[3], and complex strategic games[4]. These achievements have been made possible thanks to the advances in computer hardware, the availability of large amounts of labeled datasets, and algorithm developments. However, continued improvements in hardware performance face several challenges. Although the performance of processing units and the storage capacity of memory units have historically improved through successive scaling according to the Moore's Law, such scaling has slowed down significantly as the device reaches fundamental physical limitations. Additionally, current hardware implementations based on the von-Neumann architecture[5] suffers from severe throughput and energy penalties due to the frequent data movement between the main memory and the processor, especially in data-intensive applications such as machine learning tasks.

For instance, an ML program, AlphaGo[4], developed by DeepMind defeated one of the top-ranking professional players, Sedol Lee, in the board game Go in 2016. This event is widely considered a milestone in the progress of artificial intelligence (AI). However, the AlphaGo system used 1202 CPUs and 176 GPUs, corresponding to the power consumption of around 170 kW (1202 $\times$ 100 W + 176 $\times$ 300 W), much larger than that of the human brain, 20 W[6,7]. Examining the system

performance showed that for ML systems such as Deep Neural Networks (DNNs), between 80% and 90% of the execution time is spent in memory access, compared to the 10% to 20% of the execution time spent in computation. It is clear that to continue the progress in AI and other data-intensive tasks a radical change in the computer architecture that allows a significant reduction in memory access time and power is needed.

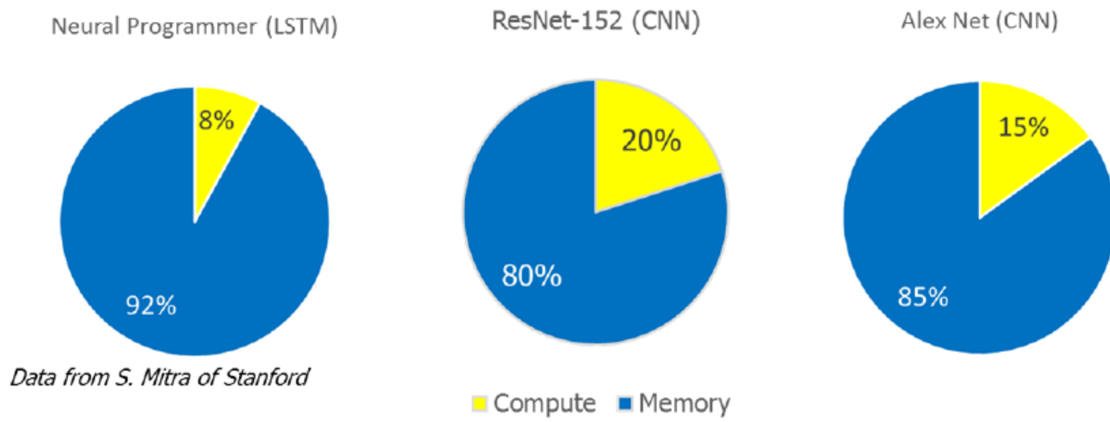


Figure 1-1: von-Neumann bottleneck.
Comparison of clock cycles spent in computation and in memory access for several Deep Neural Network algorithms as run on a Machine Learning accelerator executed in 7nm CMOS technology. *Picture Source: Defense Advanced Research Projects Agency (DARPA)*

1.2 Neuromorphic computing

The Inspiration for efficient computing can be obtained by examining biological systems. The human brain is more than five orders of magnitude more energy-efficient than all current Deep Neural Networks (DNNs)[8,9]. Unlike DNN systems, it also does not require separate neural network structures for different tasks. Part of the efficiency can be attributed to the use of analog physical basis functions in the brain system rather than accurate digital logic basis functions in the modern computer system where the brain system is already conducting in-memory computing in parallel with very low power synaptic operations.

The concept of neuromorphic computing was first introduced by Carver Mead[10]. It aims to emulate the neuro-biological architecture with analog, digital, mixed-mode analog/digital VLSI, and software systems. To implement a neuromorphic system in computing hardware, an appropriate electronic device with the capability of performing analog computing and crossbar array with the devices are essential where crossbar array performs vector-matrix multiplication (VMM) in parallel with very low power, which allows simultaneously storing the synaptic weight and modulating the transmitted signal to avoid the von-Neumann bottleneck.

1.3 Memristor device

Memristors are two-terminal 'memory resistors' that retain internal resistance state according to the history of external stimulations such as applied voltage and current. They are simple passive circuit elements that were theoretically conceived and mathematically formulated[11] in the 1970s, but their function cannot be replicated by any combination of fundamental resistors, capacitors, and inductors.

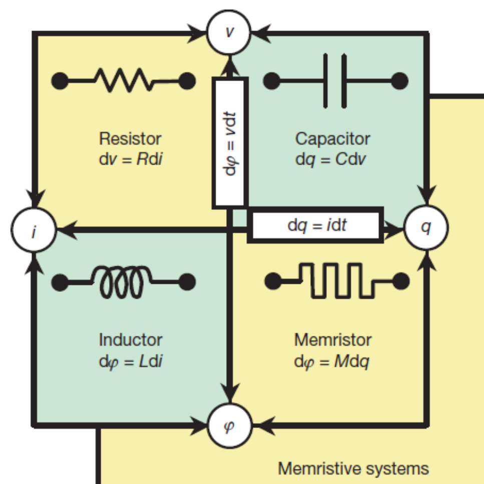


Figure 1-2: Memristor as the fourth electrical element.
The fourth element, the memristor with memristance M, defined as the rate of change of flux with charge. *Image adapted from Reference [12].* Image credit: Dr. Dmitri *B.* Strukov.

The states of experimentally implemented memristors are described by one or more internal state variables and are typically governed by dynamic ionic processes[12,13]. A key characteristic of a memristor device is the pinched hysteresis loop[14], as shown in Figure 1-3. It can be scaled down to less than 10 nm and offer large on/off ratio and fast, non-volatile, low-energy resistive switching, producing a competitive technology candidate for non-volatile memory[15–18] and in-memory logic[19,20] applications.

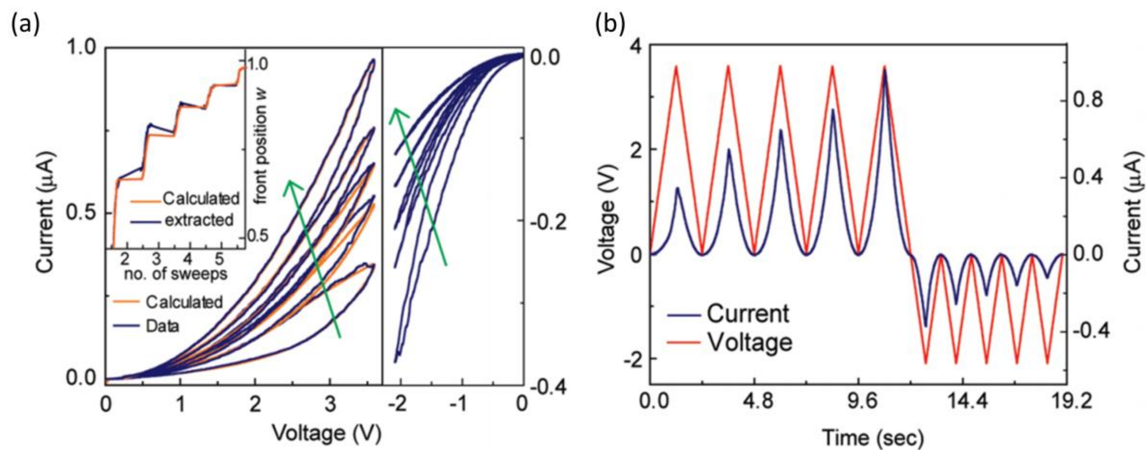


Figure 1-3: Hysteresis loop characteristic of memristor devices.
Image adapted from Reference [22]. Image credit: Dr. Sung Hyun Jo

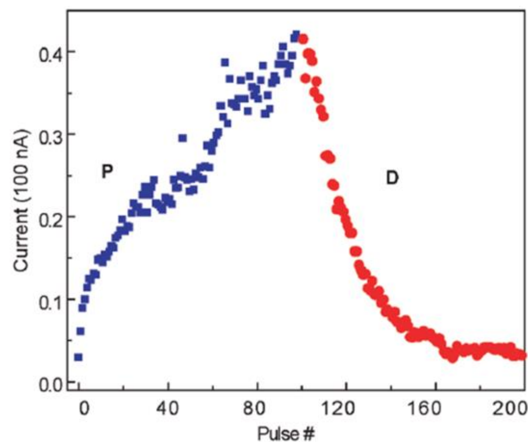


Figure 1-4: Memristor response to programming pulses.
The device conductance can be incrementally increased or decreased by consecutive potentiating or depressing pulses. *Image adapted from Reference [22].* Image credit: Dr. Sung Hyun Jo

The analog conductance modulation of memristors, as shown in Figure 1-4, in turn, allows such devices to implement online learning and make them attractive candidates for neuromorphic systems[21–24].

1.3.1 Oxygen-ion based memristor devices (VCM)

In an oxide-based memristor device, often called valence change memory (VCM) or oxide-based resistive random access memory (oxide-RRAM or ReRAM), resistive switching (RS) is caused by the redistribution of oxygen vacancies (V_Os) in the switching layer acting as a solid electrolyte[25,26]. In general, these devices may consist of two oxide layers, a V_O-rich layer and a V_O-poor layer, sandwiched by a pair of inert electrodes. The V_O-rich layer can be a deposited non-stoichiometric suboxide or formed at the interface between a reactive metal layer and the switching layer (the V_O-poor layer). The V_O-rich layer has high conductivity and acts like a V_O source during RS. The V_O-poor layer is usually stoichiometric and usually insulating when deposited. Under a high electric field and followed by increased local temperature due to Joule heating, V_Os can migrate from the V_O-rich layer to the V_O-poor layer, leading to the formation of local V_O-rich conduction channels (CFs) and switching the device to the low resistance state (LRS). The reverse process breaks the CFs and switches the device back to the high resistance state (HRS)

Figures 1-5(a) and 1-5(b) show results from high-angle annular dark-field (HAADF) scanning transmission electron microscopy (STEM) studies on a $Pt/SiO_2/Ta_2O_{5-x}$ memristor at HRS and LRS, respectively, using an *in-situ* experimental setup[27]. The conduction channel appears as a brighter region in the image, suggesting it contains more atoms with large atomic numbers (Ta in this case) since the intensity in the STEM mode strongly depends on Z (Z: atomic number

or proton number) and the atomic density. Horizontal Electron energy loss spectroscopy (EELS) line scan analysis also shows that the local oxygen concentration after the Set operation (LRS) is significantly reduced compared to the Reset operation (HRS), verifying the role of V_O migration during the operation of oxide-based memristors, as shown in Figure 1-5(c).

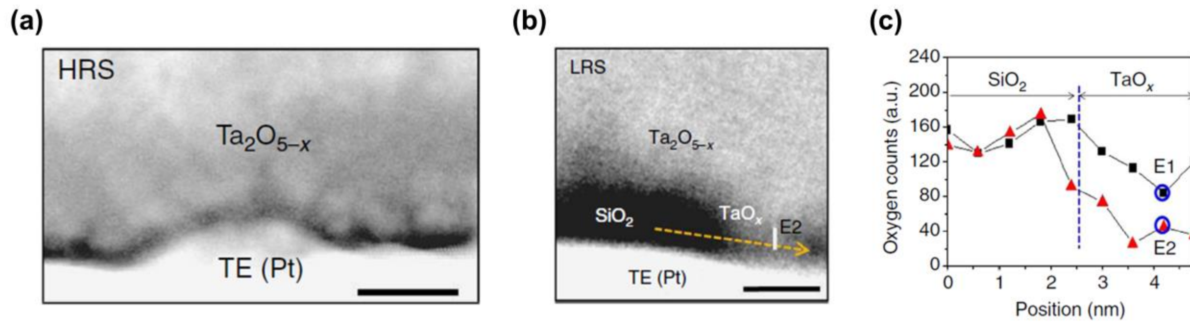


Figure 1-5: Oxide-based memristor devices.
HAADF-STEM image of a $Pt/SiO_2/Ta_2O_{5-x}$ device at (a) HRS state; (Scale bar: 5nm) and (b) LRS state (Scale bar: 2nm). (c) Horizontal EELS line scans with the corresponding oxygen profiles taken at LRS (red symbol) and HRS (black symbol) moving along the arrow shown in (b). *Image adapted from Reference [27].*

1.3.2 Metal-ion based memristor devices (ECM)

Resistive switching effects have also been widely observed in metal-ion based devices, also known as electrochemical metallization (ECM) or conductive bridge random-access memory (CBRAM) devices[15,28,29]. The RS effect originates from the electrochemical growth/dissolution of metal (e.g. Ag and Cu) filaments within the insulating layer that acts as the switching layer and plays the role of a solid electrolyte for Ag or Cu cation migration. Typical switching layers for ECM devices include SiO_2[30,31], Al_2O_3[32,33] or a-Si[34].

During RS, essentially nanoscale metal filaments are formed in the switching layer, through electrochemical processes and ion migration processes. For instance, Ag^+ ions will be first generated from Ag atoms at the anode side through an electrochemical oxidation process under

the high electric field. Then, the Ag^+ ion will drift along with the electric field and become reduced (from Ag^+ ion to Ag atom) when it reaches a cathode and captures an electron. The reduced Ag atoms form Ag nanoclusters after a nucleation process, eventually leading to a continuous Ag filament and increase of the device conductance.

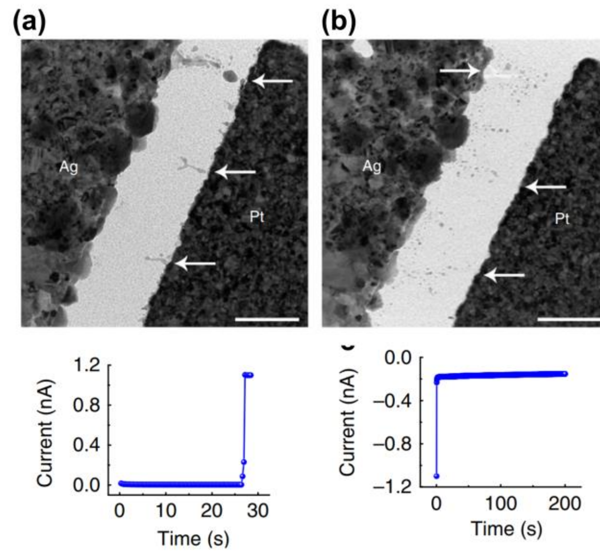


Figure 1-6: Metal ion-based memristor device switching mechanism.
(a) TEM image of an $Ag/SiO_2/Pt$ device after the forming process, showing conducting filaments. Scale bar: 200nm. Bottom inset: Corresponding I-t curve during the forming process. (b) TEM image of the same device after erasing. Scale bar: 200nm. Bottom inset: corresponding I-t curve during erasing process. *Image adapted from Reference [30].*

A key difference between ECM devices and VCM (oxide-RRAM) devices is that in VCM, RS leads to changes in the stoichiometry of the switching material itself, i.e. locally converting a stoichiometric oxide into a sub-oxide as the conducting channel. This mechanism, on one hand, suggests the devices will have long write/erase endurance since the electrochemical processes are reversible and the species involved in the processes are native to the switching material. On the other hand, the native defects created (V_Os) become very difficult to completely remove during the Reset process, leading to a leaky HRS state after Reset. On the contrary, in ECM, the

electrochemical processes involve only the foreign species (e.g. Ag atoms and Ag^+ ions), where the switching film (e.g. SiO_2) does not directly play an active role and chemical reactions may not occur between the filament and the switching layer. As a result, the Reset process of ECM devices can be very clean and allows the device to recover the very insulating HRS state. This leads to a very high on/off ratio and low switching energy. On the other hand, since extrinsic specifies are constantly moving in and out of the switching layer, this process may lead to physical damage to the switching film over time, e.g. in the form of plastic deformation, limiting the device's write/erase endurance when compared with VCM devices.

Ex-situ transmission electron microscopy (TEM) studies were first conducted to observe conducting filaments in Ag/SiO_2/Pt devices based on active Ag metal electrodes[30]. To switch the device from the initial high resistance state, a constant positive voltage was applied to the Ag electrode. The device current level was found to abruptly increase at t~27s, corresponding to the formation of a conducting (Figure 1-6(a)). After the forming process, a well-defined conducting filament with a typical cone shape is revealed in the switching layer, highlighted by the arrow in Figure 1-6(a). Once a single dominant filament is formed, further filament growth will be suppressed due to the reduced electric field. The partially formed filaments suggest in this device the filament growth starts from the inert electrode (cathode) side. After a Reset process by applying a negative voltage to the Ag electrode, it is found that all filaments break at the interface between the filament and the inert electrode, and the dissolved parts of the filaments migrated back towards the Ag side, as shown in Figure 1-6(b).

1.4 Memristor structure: From single device to 3D integration

A memristor has the simple form of a two-terminal structure that allows device integration in a crossbar form with high density and connectivity. For instance, a memristor can be formed at each crosspoint between the top (rows) and bottom (columns) electrodes, with an effective cell area of $4F^2$ (F: the smallest feature size), which guarantees the layout with maximum cell area efficiency and memory density in any planar design, as schematically shown in Figure 1-7(a). It should be noted that to operate a crossbar array properly, selector devices or memristors with intrinsic highly-nonlinear I-V behavior is required, in order to control total power dissipation through unselected memristors and to minimize parasitic effects caused by line resistance[35]. Recently, several types of two-terminal selector devices with high on/off ratios have been explored for large-scale crossbar array implementations[36–39]. However, adding the selector device to a memristor should be carefully analyzed, considering the voltage divider effect between the two devices in the presence of device variability. Otherwise, the read voltage window margin can be significantly affected and then allowed crossbar array size will be significantly reduced. Integration memristors with MOS transistors can be an alternative to achieve a large crossbar array. Connection with a transistor to the memristor in series (1T1R) effectively suppresses the sneak path current, leading to accurate reading and programming operations by controlling the third terminal, gate voltage, even in the very large crossbar array[40]. However, the use of transistors can increase the effective cell size and is also not compatible with 3D stacking which will limit the memory density the system can ultimately achieve.

To further increase the memory density, 3D stackable structures are essential. The 2D crossbar array can be stacked on top of each other in which the storage density increases as the number of stacked layers increases. For example, Figure 1-7(b) shows 2 stacks of crossbar arrays,

increasing the memory density by a factor of 2. Vertical side-wall structures can also be fabricated, achieving a similar increase in memory density, depending on how many cells can be fabricated along a vertical electrode[41–43]. The 3D stackable crossbar arrays or vertical arrays should be fabricated on the CMOS circuitry which can include decoders and sense amplifiers to form a complete memory system[44], or mixed-signal interface, SRAM, and logic circuitry for computing applications[24], as shown in Figure 1-7(c). In general, memristor fabrication can be compatible with the low-temperature back-end of line (BEOL) process. As a result, integration of memristor arrays can be achieved in the same fab after the front-end CMOS circuitry fabrication was completed, leading to high wafer yield and low cost.

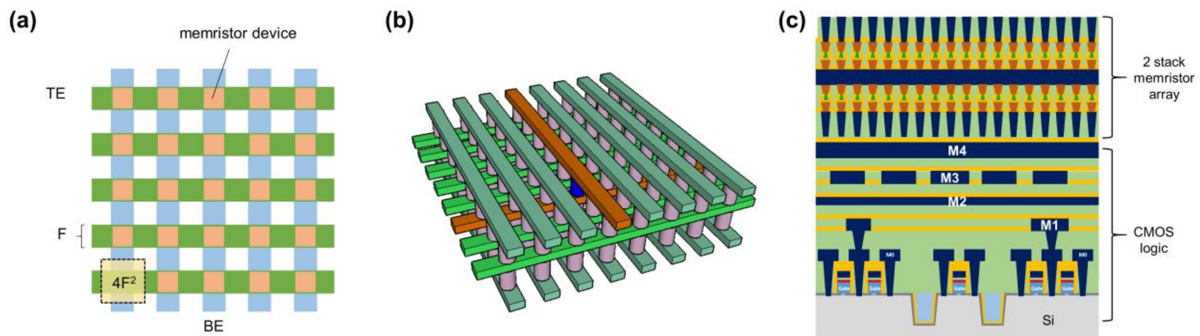


Figure 1-7: Memristor devices and integrated systems.
(a) A crossbar array structure, in which a device is formed at each crosspoint with a cell size of $4F^2$. (b) Schematic of a 2 stack crossbar architecture. The 2D crossbar arrays are stacked on top of each other. (c) Schematic of 3D integration of a 2 stack crossbar array with CMOS circuitry underneath.

1.5 Memristor Crossbar Network for Neuromorphic (DNNs & SNNs)

As discussed above, memristors can be readily fabricated by having the switching layer sandwiched between two crossing metal lines, thus forming the cell at the cross-point, as shown in Figure 1-8(a). This so-called “crossbar” structure has now been extensively studied in a variety

of studies, due to its high storage density and its capability to implement certain matrix operations in a natural and elegant fashion.

In deep learning algorithms, the vector-matrix multiplication (VMM) operation (or more basic multiply-accumulate (MAC) operation) is the core computing operation for training and inference but is very resource-expensive for conventional computing systems to implement. To accelerate VMM efficiently, the graphics processing unit (GPU) has been extensively used to improve parallelism by using 1000s of computing cores with high-throughput connections to the memory. Algorithm studies to more efficiently map the neural networks (NNs) onto the hardware have also been conducted[45]. New hardware "accelerators", such as the tensor processing unit (TPU), were also designed to improve the efficiency of matrix operations and have enjoyed success through optimizations of the digital circuit and architecture design for these relatively narrow types of operations[46].

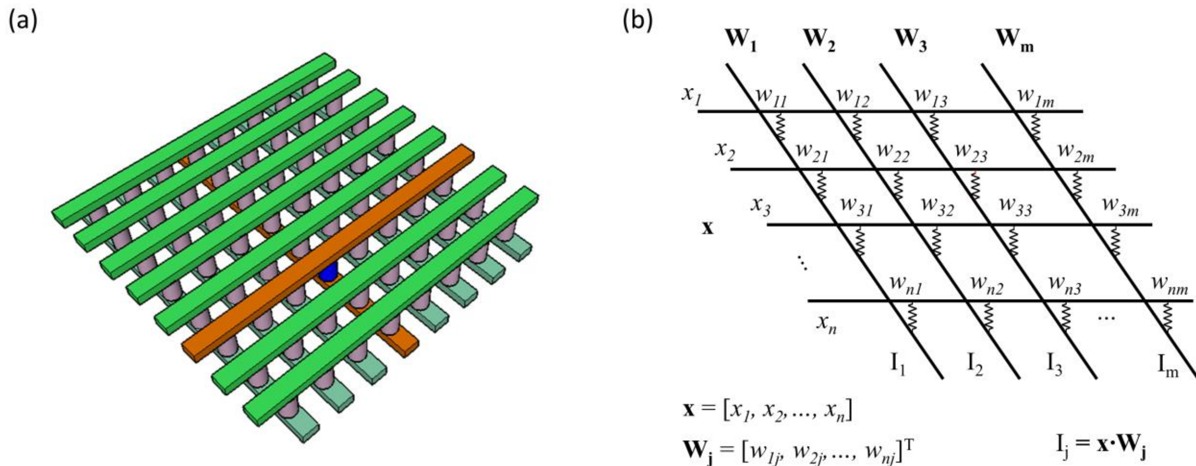


Figure 1-8: Memristor Crossbar Network.
(a) Memristor crossbar architecture. (b) Memristors are formed at the cross-points with weights w_{ij}.
Inputs are applied on the rows as x_i, while the charges or currents are collected on the columns that produce the VMM results.

As illustrated in Figure 1-8(b), if a vector **x** is input with each element x_i applied on the top electrode (the row metal lines on the left) while keeping the bottom electrode (the column metal lines at the bottom) grounded, the current flowing through each memristor at the cross-point *(i, j)* will be

$$I_{ij} = x_i w_{ij} \qquad (1-1)$$

where x_i represents the vector element which could be a pulse with a fixed width but amplitude modulated, or a pulse with a fixed amplitude but width modulated, or equivalently a number of pulses with fixed amplitude and width according to the input, and w_{ij} represents the conductance of memristor at each cross-point. Since all memristors on one column share the same bottom electrode, the current collection at the bottom electrode is the sum of all the currents flowing through all the memristors on this column

$$I_j = \sum_{i=1}^{n} x_i w_{ij} = \mathrm{x} \cdot \mathrm{W_j} \qquad (1-2)$$

where $\mathbf{W_j}$ is stored weight vector in column j of the crossbar.

By collecting currents in all the columns, the vector-matrix multiplication (VMM) output, $\mathbf{x} \cdot \mathbf{W}$, can then be obtained in a single "read" operation where $\mathbf{W} = [\mathbf{W_1}, \mathbf{W_2}, \ldots, \mathbf{W_m}]$ is the stored weight matrix of the crossbar. This operation best represents the benefits of computing in memristor crossbars – the ability to perform computing in the weight storage devices directly (VMM), as well as the high degree of parallelism where all devices in the crossbar operate in parallel and perform the multiply and add functions simultaneously, *i.e.* in-memory computing in parallel. Therefore, this compute-intensive operation can be easily implemented in a memristor crossbar network.

In practice, since memristor devices only store positive conductance values each synaptic weight (w) can be implemented with two memristor devices representing a positive and a negative weight, G_{ij}^{+} and G_{ij}^{-}, respectively[47]. That is, $w_{ij} = G_{ij}^{+} - G_{ij}^{-}$, where w_{ij} is the desired synaptic weight at row i and column j in the neural network.

It is generally believed that more bio-realistic implementations could lead to even higher energy efficiency. One such example is spiking neural networks (SNNs), which encode information in the timing and frequency of spikes. SNNs have been shown to offer extremely high energy efficiency[47,48]. Unlike DNNs in which signals are continuously collected and processed, neurons in SNNs fire only when the membrane potential reaches above a threshold value. When a neuron fires, the connection strength of the synapses associated with it may also be modulated accordingly. With this approach, data can be represented and processed with a small number of spikes, with the system consuming very little power in between.

With the rich internal ionic dynamic processes, memristor devices can natively emulate some of the key underlying physical and chemical processes in biological synapses and neurons. This allows SNN networks to be efficiently implemented using memristor devices. For example, different synaptic plasticity effects[22,49,50] can be natively implemented using so-called second-order memristor effects that can emulate the internal Ca^{2+} concentration dynamics[51,52]. Neuron functions such as integrate-and-fire function[53,54] have also been demonstrated using memristors.

Despite the great potential, SNNs have not been as widely implemented as DNNs. One of the reasons is the lack of efficient SNN algorithms, particularly for complex tasks. For instance, Spike-time-dependent plasticity (STDP)[55] is an efficient learning rule of synaptic plasticity, but the image classification accuracy of networks based on this rule is generally lower than those achieved using conventional DNNs[56]. In particular, efficient training algorithms for large SNN

networks are not well established, making it difficult for SNN systems to compete with conventional DNN implementations which have enjoyed great commercial success recently. Systematic developments of efficient algorithms, along with devices and hardware, are needed to bring the training of SNNs from the current academic research level to large scale commercial implementations.

1.6 Organization of Dissertation

In this chapter, we introduced the Neuromorphic computing concept, a fundamental background of memristor and its ability.

Chapter 2 discusses an accurate dynamic memristor switching model that can quantitatively explain Forming, sustained Set/Reset cycles, and multi-level storage. Two different filament growth processes are captured by the model and verified by experimental DC and pulse measurements. Doping-type mode with linear conductance updates and high on/off will be discussed.

Chapter 3 discusses a hybrid integrated system with the memristor crossbar array directly fabricated on a custom-designed CMOS chip to implement multiple neuromorphic applications on-chip in a functional, standalone system. System architecture, data path, and power and area efficiency will be discussed.

Chapter 4 discusses memristor-based reservoir computing, emphasizing the temporal information processing ability of memristors through its internal dynamics.

Chapter 5 discusses further device optimization and architecture for neuromorphic applications, such as high entropy oxide (HEO) and 1T1R structure, as well as tiled architecture.

References

1. Krizhevsky, A., Sutskever, I. & Hinton, G. E. ImageNet Classification with Deep Convolutional Neural Networks. *Proc. Adv. Neural Inf. Process. Syst. 25* 1090–1098 (2012).
2. He, K., Zhang, X., Ren, S. & Sun, J. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. *Proc. IEEE Int. Conf. Comput. Vis.* 1026–1034 (2015). doi:10.1109/ICCV.2015.123
3. Hinton, G. *et al.* Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups. *IEEE Signal Process. Mag.* **29**, 82–97 (2012).
4. Silver, D. *et al.* Mastering the game of Go with deep neural networks and tree search. *Nature* **529**, 484–489 (2016).
5. McKee, S. A. Reflections on the memory wall. *2004 Comput. Front. Conf.* 162–167 (2004). doi:10.1145/977091.977115
6. Merkle, R. C. Energy Limits to the Computational Power of the Human Brain The Brain as a Computer. *Foresight Updat.* 6–8 (1989).
7. Sarpeshkar, R. Ultra Low Power Bioelectronics: Fundamentals, Biomedical Applications, and Bio-Inspired Systems 697–752. *(Cambridge Univ. Press. 2010)* doi:10.1017/CBO9780511841446
8. Fischetti, M. Computers versus brains. Scientific American (1 November 2011). *https://www.scientificamerican.com/article/computers-vs-brains/*
9. Meier, K. The brain as computer: Bad at math, good at everything else. IEEE Spectrum (31 May 2017). *https://spectrum.ieee.org/computing/hardware/the-brain-as-computer-bad-at-math-good-at-everything-else*

10. Mead, C. Neuromorphic Electronic Systems. *Proc. IEEE* **78**, 1629–1636 (1990).

11. Chua, L. O. Memristor-The Missing Circuit Element. *IEEE Trans. Circuit Theory* **18**, 507–519 (1971).

12. Strukov, D. B., Snider, G. S., Stewart, D. R. & Williams, R. S. The missing memristor found. *Nature* **453**, 80–83 (2008).

13. Zidan, M. A., Strachan, J. P. & Lu, W. D. The future of electronics based on memristive systems. *Nat. Electron.* **1**, 22–29 (2018).

14. Chua, L. Resistance switching memories are memristors. *Appl. Phys. A Mater. Sci. Process.* **102**, 765–783 (2011).

15. Waser, R. & Aono, M. Nanoionics-based resistive switching memories. *Nanosci. Technol. A Collect. Rev. from Nat. Journals* 158–165 (2009). doi:10.1142/9789814287005_0016

16. Lee, M. J. *et al.* A fast, high-endurance and scalable non-volatile memory device made from asymmetric Ta2O5-x/TaO2-x bilayer structures. *Nat. Mater.* **10**, 625–630 (2011).

17. Wang, M. *et al.* Robust memristors based on layered two-dimensional materials. *Nat. Electron.* **1**, 130–136 (2018).

18. Yang, Y., Choi, S. & Lu, W. Oxide heterostructure resistive memory. *Nano Lett.* **13**, 2908–2915 (2013).

19. Borghetti, J. *et al.* Memristive switches enable stateful logic operations via material implication. *Nature* **464**, 873–876 (2010).

20. Linn, E., Rosezin, R., Tappertzhofen, S., Böttger, U. & Waser, R. Beyond von Neumann - Logic operations in passive crossbar arrays alongside memory operations. *Nanotechnology* **23**, (2012).

21. Prezioso, M. *et al.* Training and operation of an integrated neuromorphic network based on metal-oxide memristors. *Nature* **521**, 61–64 (2015).
22. Jo, S. H. *et al.* Nanoscale memristor device as synapse in neuromorphic systems. *Nano Lett.* **10**, 1297–1301 (2010).
23. Yao, P. *et al.* Face classification using electronic synapses. *Nat. Commun.* **8**, 15199 (2017).
24. Cai, F. *et al.* A fully integrated reprogrammable memristor–CMOS system for efficient multiply–accumulate operations. *Nat. Electron.* **2**, 290–299 (2019).
25. Yang, J. J., Strukov, D. B. & Stewart, D. R. Memristive devices for computing. *Nat. Nanotechnol.* **8**, 13–24 (2013).
26. Yang, J. J. *et al.* Memristive switching mechanism for metal/oxide/metal nanodevices. *Nat. Nanotechnol.* **3**, 429–433 (2008).
27. Park, G. S. *et al.* In situ observation of filamentary conducting channels in an asymmetric Ta2O5-x/TaO2-x bilayer structure. *Nat. Commun.* **4**, 1–9 (2013).
28. Waser, R., Dittmann, R., Staikov, C. & Szot, K. Redox-based resistive switching memories nanoionic mechanisms, prospects, and challenges. *Adv. Mater.* **21**, 2632–2663 (2009).
29. Valov, I., Waser, R., Jameson, J. R. & Kozicki, M. N. Electrochemical metallization memories - Fundamentals, applications, prospects. *Nanotechnology* **22**, 254003 (2011).
30. Yang, Y. *et al.* Observation of conducting filament growth in nanoscale resistive memories. *Nat. Commun.* **3**, (2012).
31. Tian, X. *et al.* Filament growth dynamics in solid electrolyte-based resistive memories revealed by in situ TEM. *Nano Res.* **7**, 1065–1072 (2014).

32. Gaba, S., Cai, F., Zhou, J. & Lu, W. D. Ultralow Sub-1-nA operating current resistive memory with intrinsic non-linear characteristics. *IEEE Electron Device Lett.* **35**, 1239–1241 (2014).

33. Belmonte, A. *et al.* Voltage-controlled reverse filament growth boosts resistive switching memory. *Nano Res.* **11**, 4017–4025 (2018).

34. Jo, S. H., Kim, K. H. & Lu, W. High-density crossbar arrays based on a Si memristive system. *Nano Lett.* **9**, 870–874 (2009).

35. Burr, G. W. *et al.* Access devices for 3D crosspoint memory. *J. Vac. Sci. Technol. B* **32**, 040802 (2014).

36. Kim, G. H. *et al.* 32 × 32 Crossbar Array Resistive Memory Composed of a Stacked Schottky Diode and Unipolar Resistive Memory. *Adv. Funct. Mater.* **23**, 1440–1449 (2013).

37. Choi, B. J. *et al.* Trilayer Tunnel Selectors for Memristor Memory Cells. *Adv. Mater.* **28**, 356–362 (2016).

38. Govoreanu, B. *et al.* High-performance metal-insulator-metal tunnel diode selectors. *IEEE Electron Device Lett.* **35**, 63–65 (2014).

39. Kau, D. *et al.* A stackable cross point phase change memory. *Tech. Dig. - Int. Electron Devices Meet. IEDM* 617–620 (2009). doi:10.1109/IEDM.2009.5424263

40. Zahurak, J. *et al.* Process integration of a 27nm, 16Gb Cu ReRAM. *Tech. Dig. - Int. Electron Devices Meet. IEDM* **2015-Febru**, 6.2.1-6.2.4 (2015).

41. Baek, I. G. *et al.* Realization of vertical resistive memory (VRRAM) using cost effective 3D process. *Tech. Dig. - Int. Electron Devices Meet. IEDM* 737–740 (2011). doi:10.1109/IEDM.2011.6131654

42. Hsu, C. W. *et al.* 3D vertical TaOx/TiO2RRAM with over 103self-rectifying ratio and sub-μA operating current. *Tech. Dig. - Int. Electron Devices Meet. IEDM* **2**, 10.4.1-10.4.4 (2013).

43. Bai, Y. *et al.* Study of multi-level characteristics for 3D vertical resistive switching memory. *Sci. Rep.* **4**, 5780 (2014).

44. Liu, T. Y. *et al.* A 130.7mm2 2-layer 32Gb ReRAM memory device in 24nm technology. *Dig. Tech. Pap. - IEEE Int. Solid-State Circuits Conf.* **56**, 210–211 (2013).

45. Chen, Y. H., Krishna, T., Emer, J. S. & Sze, V. Eyeriss: An Energy-Efficient Reconfigurable Accelerator for Deep Convolutional Neural Networks. *IEEE J. Solid-State Circuits* **52**, 127–138 (2017).

46. Jouppi, N. P. *et al.* In-datacenter performance analysis of a tensor processing unit. *Proc. - Int. Symp. Comput. Archit.* **Part F1286**, 1–12 (2017).

47. Agarwal, S. *et al.* Energy scaling advantages of resistive memory crossbar based computation and its application to sparse coding. *Front. Neurosci.* **9**, 1–9 (2016).

48. Wang, W. *et al.* Learning of spatiotemporal patterns in a spiking neural network with resistive switching synapses. *Sci. Adv.* **4**, 1–9 (2018).

49. Ohno, T. *et al.* Short-term plasticity and long-term potentiation mimicked in single inorganic synapses. *Nat. Mater.* **10**, 591–595 (2011).

50. Wang, Z. Q. *et al.* Synaptic learning and memory functions achieved using oxygen ion migration/diffusion in an amorphous InGaZnO memristor. *Adv. Funct. Mater.* **22**, 2759–2765 (2012).

51. Kim, S. *et al.* Experimental demonstration of a second-order memristor and its ability to biorealistically implement synaptic plasticity. *Nano Lett.* **15**, 2203–2211 (2015).

52. Zidan, M. A., Jeong, Y. J. & Lu, W. D. Temporal Learning Using Second-Order Memristors. *IEEE Trans. Nanotechnol.* **16**, 721–723 (2017).

53. Pickett, M. D., Medeiros-Ribeiro, G. & Williams, R. S. A scalable neuristor built with Mott memristors. *Nat. Mater.* **12**, 114–117 (2013).

54. Stoliar, P. *et al.* A Leaky-Integrate-and-Fire Neuron Analog Realized with a Mott Insulator. *Adv. Funct. Mater.* **27**, (2017).

55. Markram, H., Lübke, J., Frotscher, M. & Sakmann, B. Regulation of synaptic efficacy by coincidence of postsynaptic APs and EPSPs. *Science* **275**, 213–215 (1997).

56. Feng, S., Zhou, H. & Dong, H. Using deep neural network with small dataset to predict material defects. *Mater. Des.* **162**, 300–310 (2019).

Chapter 2

A Quantitative, Dynamic Memristor/RRAM Model

Nanoscale memristor devices have been widely considered as a promising candidate for neuromorphic and other memory-centric applications[1–3] due to their simple structure and superior performance metrics, including endurance, switching speed, and exceptional scalability[4–6]. Moreover, memristors can be used to both store data in their analog conductance states and process data at the same physical locations, allowing power-efficient computing both in-memory and in parallel[7–16]. The principle of resistive switching (RS) in oxide-based memristors has been explained by V_O migration in the oxide layer through ion drift and diffusion, where V_Os act as dopants and modulate local electrical and thermal conductivity[17–21]. After the Set process, high V_O concentration (n_D) regions can act as conducting filaments (CFs) and provide high conductance channels in the switching layer.

A number of device models have been proposed to describe the RS dynamics such as the formation/rupture of CFs[22–28]. However, these models typically start from assumptions that a filament is already (partly) formed, and cannot in general reliably reproduce multiple RS cycles. Additionally, most models do not provide a mechanism to control the programming current during filament formation, which is critical during practical device applications. These challenges can be largely attributed to the incomplete physics captured in the models.

In this chapter, we present a comprehensive physical model that can accurately capture practical device operations, starting from the initial state, by self-consistently solving[18,26] the

electric field (E), temperature (T), and V_O concentration (n_D) dynamic evolutions in realistic device structures. The maximum programming current during filament formation, i.e. the compliance current (I_{CC}), is also introduced in the model through a current modulation layer. Our model clearly revealed the Forming process is a result of the initial non-uniform oxygen vacancy (V_O) defect distribution and initiated by electric field focusing and localized thermal effects. Additionally, we observed different RS mechanisms during the Set process, depending on the programming current level. With a low programming current, a bulk-type doping phenomenon was observed. This effect can lead to linear analog conductance modulation with a large dynamic range, which is beneficial for low-power neuromorphic computing applications. On the other hand, filament width growth becomes dominant at high programming current levels, resulting in high conductance values and a leaky high-resistance state (HRS) and a small dynamic range. These results were further verified by experimental studies using a 1T1R device structure, where the maximum programming current was controlled by the gate voltage (V_G) to allow systematical tuning of the conductance level and the V_O configuration.

2.1 COMSOL Multiphysics and finite element method (FEM)

COMSOL Multiphysics is a general-purpose simulation software to simulate designs, devices, and processes in all fields of engineering, manufacturing, and scientific research[29]. Especially, it contains built-in physics interfaces to simulate a wide range of physics phenomena, including many common multiphysics systems such as solid mechanics, acoustics, fluid flow, heat transfer, chemical species transport, and electromagnetics. In addition, it is able to create uses' own model definitions based on mathematical equations with custom partial differential equations

(PDEs) and directly input them into the software's graphical user interface (GUI). It is also possible to combine both the predefined interfaces and custom PDEs.

PDEs are differential equations that contain unknown multi-variable functions and their partial derivatives, and are used to describe the laws of physics for space- and time-dependent problems. Unfortunately, for most geometries and problems these PDEs cannot be solved with analytical methods. Instead, an approximation of the equations can be used, typically based upon different types of discretization. These discretization methods approximate the PDEs with numerical model equations, which can be solved using numerical methods. The solution to the numerical model equations are, in turn, an approximation of the real solution to the PDEs. The finite element method (FEM) is used to compute such approximations.

As a toy example, we can solve Laplace equation, ${\partial^2 u}/{\partial x^2} + {\partial^2 u}/{\partial y^2} = 0$, with the discretization, for $\mathrm{u(x,y)}$ defined on $x \in [0,1], y \in [0,1]$ with the boundary conditions

$$(x,0) = 1, \qquad u(x,1) = 2, \qquad u(0,y) = 1, \qquad u(1,y) = 2 \qquad (2-1)$$

The second partial derivatives can be approximated and described by

$$\left(\frac{\partial^2 u}{\partial x^2}\right)_{i,j} \approx \frac{u_{i-1,j} - 2u_{i,j} + u_{i+1,j}}{(\Delta x)^2} \qquad (2-2)$$

$$\left(\frac{\partial^2 u}{\partial y^2}\right)_{i,j} \approx \frac{u_{i,j-1} - 2u_{i,j} + u_{i,j+1}}{(\Delta y)^2} \qquad (2-3)$$

Plugging equations (2-4) and (2-5) into original Laplace equation, then we obtain

$$\frac{u_{i-1,j} - 2u_{i,j} + u_{i+1,j}}{(\Delta x)^2} + \frac{u_{i,j-1} - 2u_{i,j} + u_{i,j+1}}{(\Delta y)^2} = 0 \qquad (2-4)$$

at grid point (*i,j*). For $\Delta x = \Delta y$, equation (2-4) can be summarized into

$$-4u_{i,j} + u_{i-1,j} + u_{i+1,j} + u_{i,j-1} + u_{i,j+1} = 0 \qquad (2-5)$$

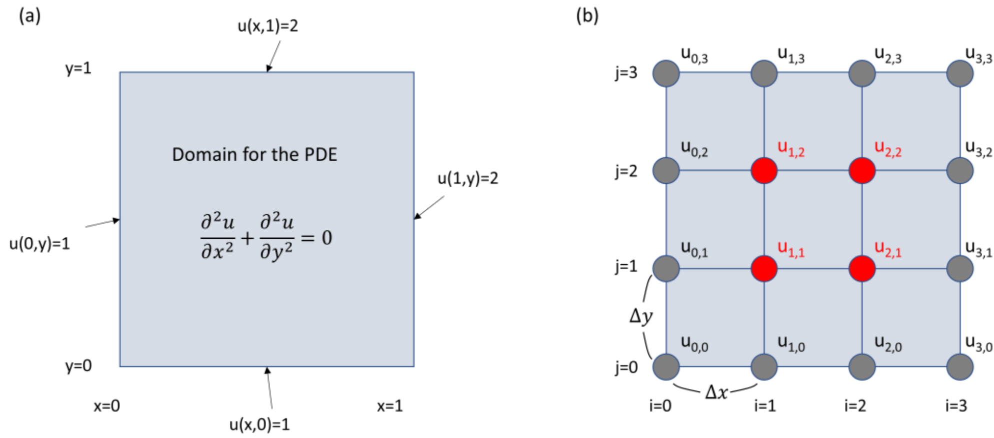


Figure 2-1: Finite element method (FEM).
(a) Laplace equation with boundary conditions. (b) a few grid points with the discretization to solve Laplace equation.

From the Figure 2-1, we can re-write the equation (2-5) for each point of $u_{i,j}$,

$$-4u_{1,1} + u_{1,2} + u_{2,1} + u_{0,1} + u_{1,0} = 0 \qquad (2-6a)$$

$$u_{1,1} - 4u_{1,2} + u_{2,2} + u_{0,2} + u_{1,3} = 0 \qquad (2-6b)$$

$$u_{1,2} - 4u_{2,2} + u_{2,1} + u_{2,3} + u_{3,2} = 0 \qquad (2-6c)$$

$$u_{1,1} + u_{2,2} - 4u_{2,1} + u_{2,0} + u_{3,1} = 0 \qquad (2-6d)$$

and with the boundary conditions, equation (2-1), we obtain,

$$\begin{pmatrix} -4 & 1 & 0 & 1 \\ 1 & -4 & 1 & 0 \\ 0 & 1 & -4 & 1 \\ 1 & 0 & 1 & -4 \end{pmatrix} \begin{pmatrix} u_{1,1} \\ u_{1,2} \\ u_{2,2} \\ u_{2,1} \end{pmatrix} = \begin{pmatrix} -2 \\ -3 \\ -4 \\ -3 \end{pmatrix} \qquad (2-7)$$

which can be readily solved to obtain the final solution, $(u_{1,1}, u_{1,2}, u_{2,2}, u_{2,1})$ = (1.25, 1.5, 1.75, 1.5). The solution is illustrated in the following Figure 2-2.

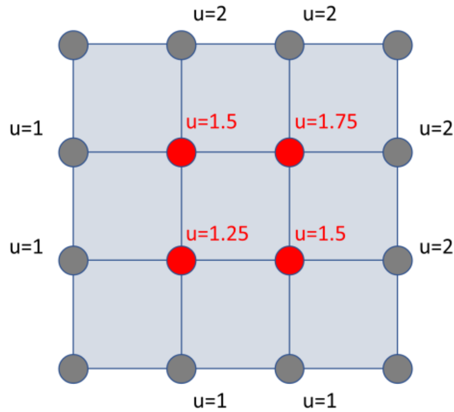


Figure 2-2: Solution with Finite element method (FEM).
The solution of the Laplace equation with a given boundary condition and a few grid points.

One of the benefits of using the finite element method (FEM) is that it offers great freedom in the selection of discretization, both in the elements that may be used to discretize space and the basis functions. The elements can be uniformly distributed over the axis, and smaller elements in a region where the gradient of *u* is large could also be available.

In this device simulation, we used coupled systems with the heat transfer module (predefined interfaces) and a custom coefficient form PDE module to describe the diffusion and drift of oxygen vacancies (V_Os).

2.2 Device structure and dynamic model for simulation

2.2.1 Tantalum-oxide (TaO_x) based memristor

The tantalum-oxide-based memristor has become a leading material for both memory applications and neuromorphic applications[1–3] due to superior performance metrics in many aspects including endurance up to 10^{12}, switching speed of < 1 ns, and exceptional scalability[4–6]. A standard bilayer Ta_2O_5/TaO_x device structure is chosen as the model system. The device consists

of a high resistance Ta_2O_5 layer on top of a more conductive TaO_x layer[30], sandwiched by a Pd-based top electrode (TE) and bottom electrode (BE). Resistive switching in the Ta_2O_5/TaO_x device has been generally explained by the V_O re-distribution and V_O exchange between the high resistance Ta_2O_5 layer and the low resistance TaO_x layer[25,31]. Specifically, a negative voltage applied to the TE attracts V_Os from the TaO_x layer, which acts as a V_O reservoir, into the Ta_2O_5 layer and forms a conduction filament connecting the TE and the conductive TaO_x layer. This process switches the device to the low resistance state (LRS) and is termed the Set process. When a positive voltage is applied to the TE, it repels V_Os from the Ta_2O_5 layer and breaks the filaments, thus switching the device back to the high resistance state (HRS) in the so-called Reset process.

2.2.2 Integration of the tantalum-oxide (TaO_x) based memristor on-chip

The TaOx-based memristor used in this work was directly fabricated on top of the CMOS circuit. First, SiO_2 with 100 nm thickness was deposited on the CMOS chip, followed by a reactive ion etching process to open the CMOS landing pads (Gate, Drain, Source, Bulk). Then, Pd bottom electrodes with 40 nm thickness were defined on the SiO_2 by photo lithography and e-beam evaporation of the Pd metal, followed by a lift-off process. A 30 nm TaO_x layer was deposited by DC reactive sputtering of a Ta metal target in an Ar/O_2 environment at room temperature, followed by the deposition of the 5 nm Ta_2O_5 switching layer through sputtering a Ta_2O_5 ceramic target in the same chamber without O_2. Afterwards, the top electrode was fabricated, followed by a reactive ion etching process in SF_6/Ar to expose the contact regions of the bottom electrodes. Finally, metallization processes were performed by photolithography to connect the electrodes (BE and TE) with the CMOS landing pads, as shown in Figure 2-3(a). The TaO_x memristor device is located on the Drain side, and the BE becomes the Drain in the 1T1R system to control the transistor

saturation current along with V_G during Forming/Set. For flexibility, devices can be measured in either the 1T1R structure or 1R (without transistor) structure using an additional pad.

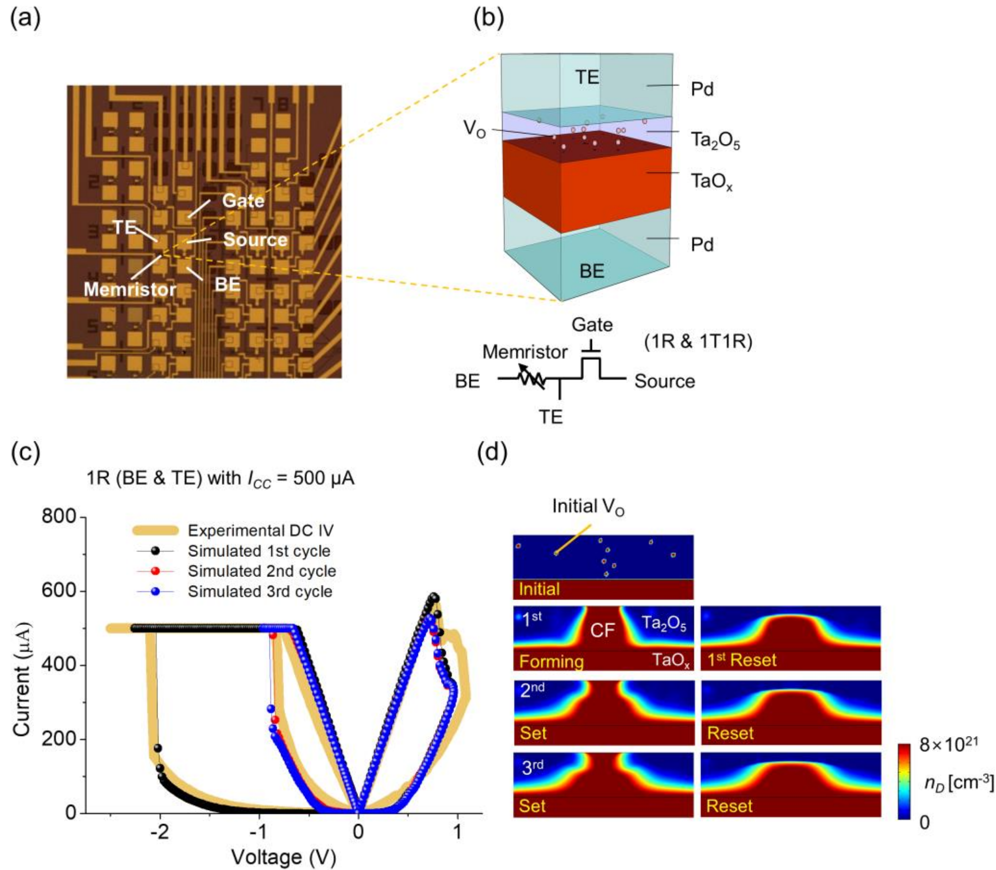


Figure 2-3: Tantalum oxide memristor during Forming and Set/Reset cycling.
(a) Top view of the integrated memristor (1R & 1T1R) on the CMOS chip. (b) Schematic of the Pd/Ta_2O_5/TaO_x/Pd bilayer device and the 1T1R circuit. (c) Measured and simulated DC *I-V* characteristics of the memristor device with 500 μA I_{CC}, showing the Forming and consecutive Set and Reset switching cycles. (d) 2-D maps of n_D obtained in the model for Initial, Forming, 1st Reset, and subsequent Set and Reset states. After Forming and first Reset, CF formation/rupture can be reliably repeated in subsequent cycles.

2.2.3 Device structure and physical model

Our model aims to achieve quantitative agreements with experimental results, starting from the initial state. To start with, we assume a uniform V_O concentration of $n_D = 1\times10^{22}$ cm^{-3} in the conductive TaO_x layer, based on results from density functional theory (DFT) calculations[32], and

only a small number of V_Os that are randomly distributed in the Ta_2O_5 layer (Figure 2-3(b) and (d)). The size of the V_O defects in the Ta_2O_5 layer is assumed to be 3 Å[32], and the V_O defect region is assumed to have a conductivity of 10^5 S/m, while the stoichiometric Ta_2O_5 film is assumed to have a very low n_D level (1×10^{16} cm^{-3}) and highly resistive. In the proposed model, the electrical conductivity (σ) in the oxide is assumed to depend on the V_O concentration (n_D), temperature (T), and electrical field (E), and is given by

$$\sigma_{Ta_2O_{5-x}} = \sigma_0(n_D)\exp(-E_{AC}(n_D)/kT) + \sigma_{PF}(E,T) \quad (2-8)$$

where σ_0 is a pre-factor, E_{AC} is the activation energy for electron conduction, and $\sigma_{PF}(E,T)$ represents the Poole-Frenkel conduction term ($\sigma_{PF}(E,T) = \exp\,(293/T\cdot(\alpha\sqrt{E}+\beta))$).

Considering ion drift/diffusion in the oxide layer, n_D can be determined by

$$\partial n_D/\partial t = \nabla\cdot(D\nabla n_D - v n_D + DSn_D\nabla T) \quad (2-9)$$

where $D\nabla n_D$ and $v n_D$ are Fick diffusion flux and drift flux terms, respectively[25]. The $DSn_D\nabla T$ term corresponds to Soret diffusion flux, where S is the Soret coefficient ($S = -E_a/kT^2$). The diffusion coefficient (D) is given by

$$D = 1/2\cdot a^2 f\exp(-E_a/kT) \quad (2-10)$$

and the drift velocity (v) is given by

$$v = af\exp(-E_a/kT)sinh(qaE/2kT) \quad (2-11)$$

where f is the escape-attempt frequency (10^{12} Hz)[25], a is the effective hopping distance (0.32 nm), and E_a is the activation energy for V_O migration (0.85 eV).

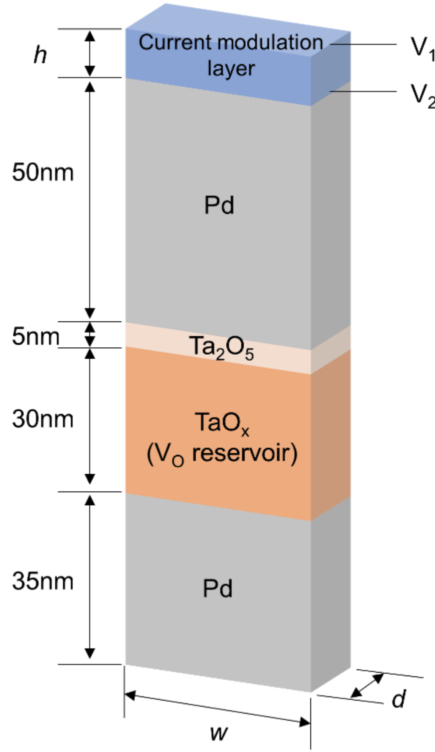


Figure 2-4: Device geometry used in the simulation.
A uniform doping concentration of $n_D = 1\times10^{22}$ cm^{-3} was assumed within the conducting TaO_x layer (V_O reservoir) at the initial state. w = 40 nm, h = 10 nm, d = 20 nm were used in the simulation. V_1 is the voltage applied to the current modulation layer top surface and V_2 represents the actual voltage applied to the memristor TE.

Equation (2-9) can be self-consistently solved along with equation (2-12), the continuity equation for electrical conduction, and equation (2-13), the Fourier equation for Joule heating, following the approach proposed by Ielmini *et al*[26].

$$\nabla \cdot \sigma \nabla \Psi = 0 \qquad (2-12)$$

$$\rho C_p \frac{\partial T}{\partial t} - \nabla \cdot k_{th} \nabla T = J \cdot E \qquad (2-13)$$

The equations are solved in a numerical solver (COMSOL) to calculate n_D, Ψ, and T. To simulate realistic devices, we also introduce a current modulation layer that can control the programming current, as shown in Figure 2-4. The definitions of the parameters used in the model

are summarized in Table 2-1 and Figure 2-5. The details for the proposed model are also discussed in Supporting Information.

2.2.4 physical parameters and modeling compliance effect (*Icc* and transistor)

Figure 2-4 shows the structure used in the model. The width (*w*) and depth (*d*) of the structure is 40 nm and 20 nm, respectively. The thicknesses of the Pd bottom electrode (BE), V_O reservoir (TaO_x) layer, switching (Ta_2O_5) layer, and the Pd top electrode (TE) layer are 35 nm, 30 nm, 5 nm, and 50 nm, respectively. The V_O density of 5×10^{21} cm^{-3} is chosen as a criterion for metallic CF formation[32], and the maximum V_O density is ~ 1×10^{22} cm^{-3}. A uniform concentration of n_D = 1×10^{22} cm^{-3} is assumed in the conductive TaO_x layer as an initial state. The current modulation layer (CML) height (*h*) is 10 nm, which is directly connected to TE.

Material properties	Pd	TaO_x & Ta_2O_5
Density (ρ) [kg/m^3]	11900	8200
Heat capacity (C_p) [J/(kg·K)]	50	26
Electrical conductivity (σ) [S/m]	1e7	Eq.(1)
Thermal conductivity (k) [W/(m·K)]	71.8	$k_{th0}(1+\lambda(T-T_0))$

Constants		value
a	Hopping distance	3.2 Å
f	Escape-attempt frequency	10^{12} Hz
λ	Linear thermal coefficient	0.1
T_0	Room temperature	293 K
E_a	V_O diffusion barrier	0.85 eV
α	PF coefficient1	5.48e-4
β	PF coefficient2	-5.70

Table 2-1: Material parameters and constants used in the proposed model.

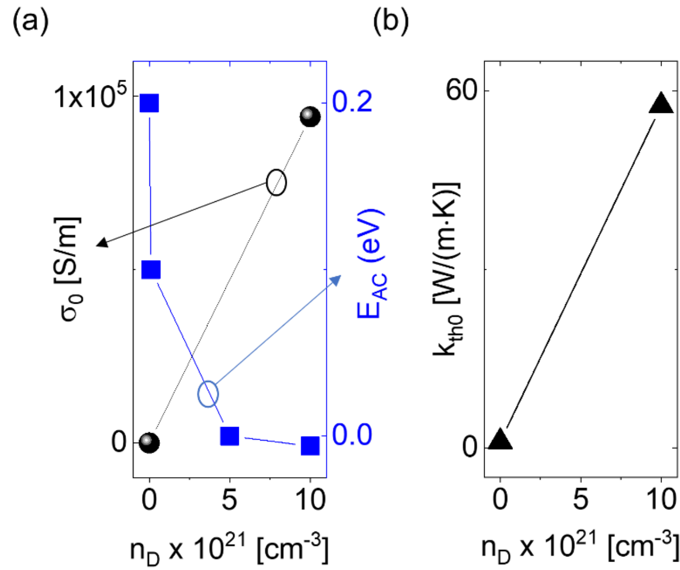


Figure 2-5: Parameters used in the proposed model.
(a) The electrical conductivity pre-factor σ_0 and the electron conduction activation energy E_{AC} as function of n_D. (b) The thermal conductivity pre-factor k_{th0} as function of n_D.

The following assumptions are used as boundary conditions: the temperature at BE, TE and CML surfaces are 300 K, BE is always grounded during simulation, while voltage is applied to the top of CML. The definition of the parameters and constant values used in the model are summarized in Table 2-1 and Figure 2-5(a). The electrical conductivity is given by the Arrhenius equation[33] where the pre-exponential factor σ_0 is assumed to linearly increase from 1 S/m to 10^5 S/m with increasing n_D. The electron conduction activation energy E_{AC} used in the model is 0.2 eV for the initial state where n_D is 1×10^{16} cm^{-3}, and linearly decreases to 0.1 eV with increasing n_D up to 1×10^{20} cm^{-3}, 0 eV with n_D up to 5×10^{21} cm^{-3}, and -0.006 eV with n_D up to 1×10^{22} cm^{-3}. The pre-factor of thermal conductivity k_{th0} linearly increases with increasing n_D (Figure 2-5(b)).

To introduce the compliance current (I_{CC}) during Forming and Set, the conductance of the CML is defined by Eqs (2-14(a)-(c)). Here, V_1 is the voltage applied to the CML top surface and V_2 represents the actual voltage applied on the memristor TE, as shown in Figure 2-4. If the

electrical field applied to CML is smaller than E_{MAX}, the conductance of CML will be I_{cc_Sigma}, 10^5 S/m, while when the electric field becomes larger than E_{MAX} the conductance of the CML is determined by Eq. 2-14(a). During Reset, σ_{CML} is always I_{cc_Sigma}, 10^5 S/m.

$$\sigma_{CML} = \frac{I_{cc} \cdot h}{|V_1 - V_2| \cdot w \cdot d}, \qquad if\ \ E_{CML} \geq E_{max} \qquad (2-14a)$$

$$\sigma_{CML} = I_{cc_Sigma}, \qquad if\ \ E_{CML} < E_{max} \qquad (2-14b)$$

$$E_{max} = \frac{I_{cc}}{I_{cc_Sigma} \cdot w \cdot d} \qquad (2-14c)$$

where h = 10 nm, w = 40 nm, d = 20 nm, I_{CC} = 500 μA.

To model the transistor *I-V*, conductance of CML is defined in Eqs (2-15(a)-(d))[34].

$$\sigma_{CML} = \left(\frac{\frac{W}{L} C_{Ox} \mu \left(V_{gs} - V_t - \frac{m(V_1 - V_2)}{2} \right)}{1 + \frac{V_1 - V_2}{E_{sat} L}} \right) \frac{h}{d \cdot w \cdot (V_1 - V_2)}$$

$$if\ \ V_1 - V_2 < V_{dsat} \qquad (2-15a)$$

$$\sigma_{CML} = \left(\frac{\frac{W}{L} C_{Ox} \mu \left(V_{gs} - V_t - \frac{m V_{dsat}}{2} \right)}{1 + \frac{V_{dsat}}{E_{sat} L}} + a\left(V_{gs} - V_t\right)(V_1 - V_2 - V_{dsat}) \right) \frac{h}{d \cdot w \cdot (V_1 - V_2)}$$

$$if\ \ V_1 - V_2 \geq V_{dsat} \qquad (2-15b)$$

$$V_{dsat} = \frac{2(V_{gs} - V_t)/m}{1 + \sqrt{1 + 2(V_{gs} - V_t)/m E_{sat} L}} \qquad (2-15c)$$

$$E_{sat} = \frac{2 v_{sat}}{\mu} \qquad (2-15d)$$

where L = 130 nm, W = 450 nm, m = 1.3, T_{ox} = 4 nm, μ = 50 cm^2/(V·s), V_T = 0.3 V, v_{sat} = 8×10^6 cm/s, a = 8×10^{-6}, h = 10 nm, w = 40 nm, and d = 20 nm.

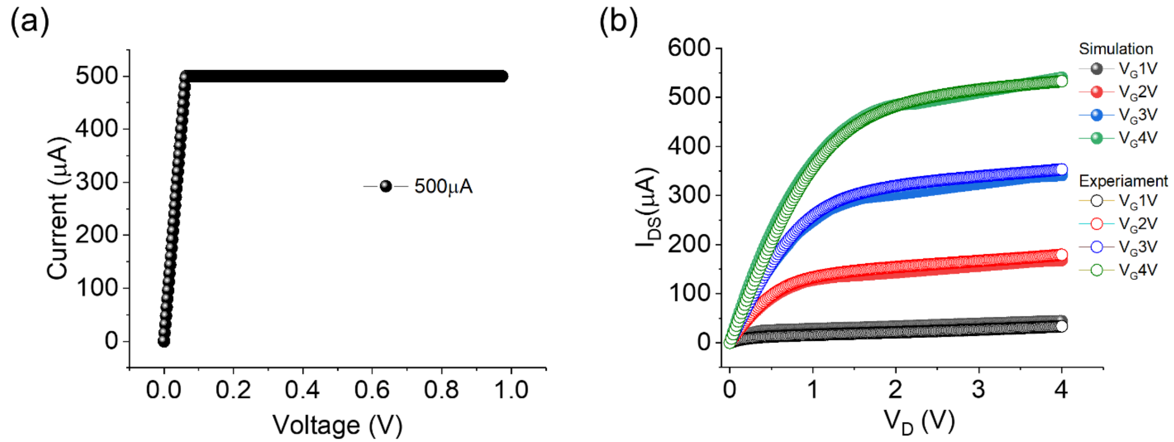


Figure 2-6: Current modulation layer *I-V* behavior, without memristor devices.
(a) Simulated DC *I-V* with I_{CC} of 500 µA. (b) Simulated and measured transistor DC *I-V* curves with varying V_G from 1 V to 4 V. The simulation results very well matched with experimental results.

Figure 2-6 shows the *I-V* behavior of the CML layer. The CML layer can effectively model the effects of I_{CC} and the transistor *I-V*, and match well with experimental results.

2.3 Forming process

Figure 2-3(c) shows the measured and the simulated DC *I-V* characteristics during the Forming, Set and Reset processes, with 500 µA of I_{CC} (Figure 2-6(a)). A very good match of the simulation results with the measured data can be found. The Forming and Set transitions occur at negative voltages, around -2.04 V and -0.9 V, respectively, while the Reset transition starts around 0.8 V for this model device. Figure 2-3(d) plots 2-D maps of n_D for 3 consecutive switching cycles, showing the creation and elimination of the V_O depletion gap during repeated cycling. Note the model can run in a self-contained manner (*e.g.*, no adjustment of parameters) from the initial state and through repeated cycling without degradation.

Before the Forming transition, the current level is very low due to the highly resistive Ta_2O_5 layer, where the current conduction mechanism is dominated by Poole-Frenkel (PF) emission. In local regions within the Ta_2O_5 layer with a few closely spaced Vo defects (Figure 2-3(d)), the PF current is larger due to the electric field focusing effect and leads to localized heat generation. The higher local electric field and elevated temperature (> 500 K) due to Joule heating eventually cause the original V_Os in the Ta_2O_5 layer to migrate towards the TE during the first stage of Forming (Figure 2-7). This process leads to a small filament to be formed from these original V_Os in the oxide film. However, during this stage V_O migration from the V_O reservoir is negligible, since the temperature at the interface between the Ta_2O_5 layer and the TaO_x layer is still too low (< 500 K) to trigger strong V_O drift/diffusion from the V_O reservoir (Figure 2-7(b)). This effect is more clearly observed in pulse simulations, by applying a short pulse with a fixed amplitude to the virgin device (Figure 2-8(a)). Negligible changes in conductance are observed in stages ①~② (Figures 2-8(b) and 2-8(c)) when the temperature at the Ta_2O_5/TaO_x interface is still low, even though the original V_Os in the Ta_2O_5 layer have already migrated to the TE (Figure 2-8(d) ①−②). Once the temperature at the Ta_2O_5/TaO_x interface reaches a critical temperature ~ 500 K (stage ③), the influx of V_Os from the V_O reservoir increases exponentially due to strong V_O drift/diffusion aided by the elevated temperature. Eventually, the extra V_Os supplied from the V_O reservoir form the conducting filament, connecting the TE to the conductive TaO_x layer and resulting in the abrupt conductance increase (stages ③−③'', Figures 2-8(c) and 2-8(d)). As a result, the switching speed of the Forming process strongly depends on the temperature of the Ta_2O_5/TaO_x interface, which in turn is a function of the applied voltage. When a more negative voltage is applied during the Forming process, the temperature at the Ta_2O_5/TaO_x interface increases more rapidly and results

in faster switching (Figures 2-8(e) and 2-8(f)). Such an exponential dependence of Forming time vs. voltage is commonly observed experimentally.

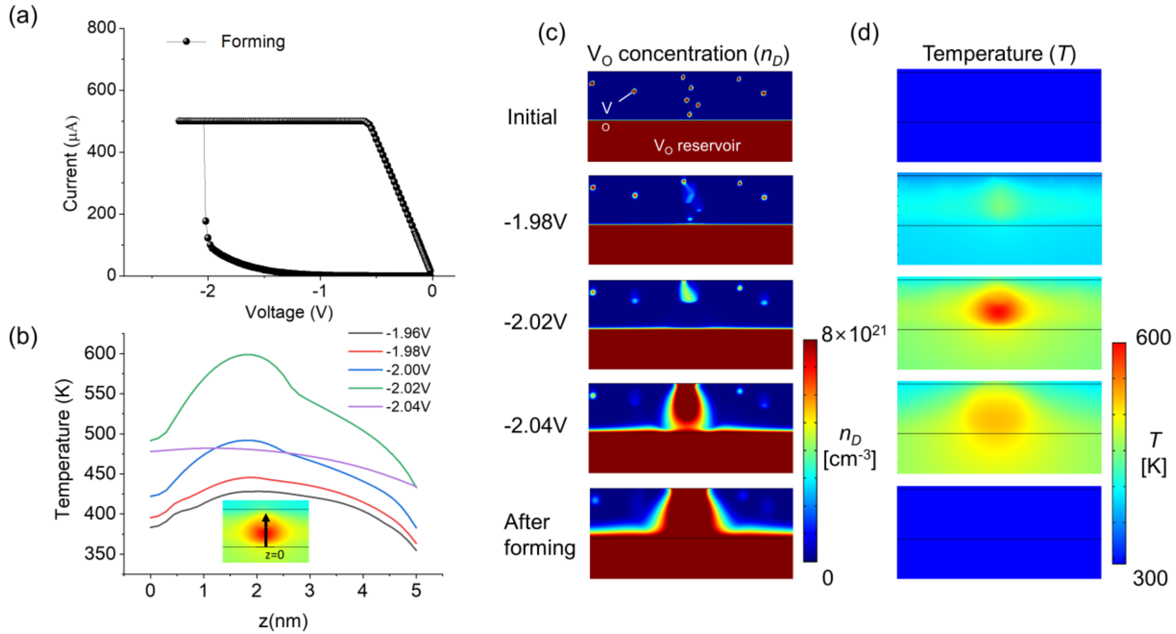


Figure 2-7: Evolution of V_O concentration (n_D) and temperature (T) during DC Forming. (a) DC *I-V* characteristics during Forming. (b) 1D profile of temperature along the z-direction during Forming with different voltages. 2-D maps of (c) n_D and (d) T during Forming with different voltages.

It should be noted that this transition is driven by a positive feedback process, where a high CF temperature leads to rapid CF growth, which in turn increases the CF temperature due to increased Joule heating effects. After the initial CF growth, prolonged positive feedback during the CF formation should be avoided to improve device reliability, normally by limiting the maximum programming current. Otherwise, permanent damage to the dielectric might occur, and the device cannot be Reset again. To model the current compliance effect, we introduced a current modulation layer in the model to control the programming current (Figure 2-6) during CF formation. Our model clearly shows that once the current level reaches I_{CC} during Forming, *i.e.* 500 μA in this example, the CF temperature starts to decrease and the positive feedback stops,

which suppresses further V_O drift/diffusion (Figures 2-8(b) and (c)). This leads to controlled and reliable filament formation in the oxide layer (Figures 2-3(d) and 2-8(d)).

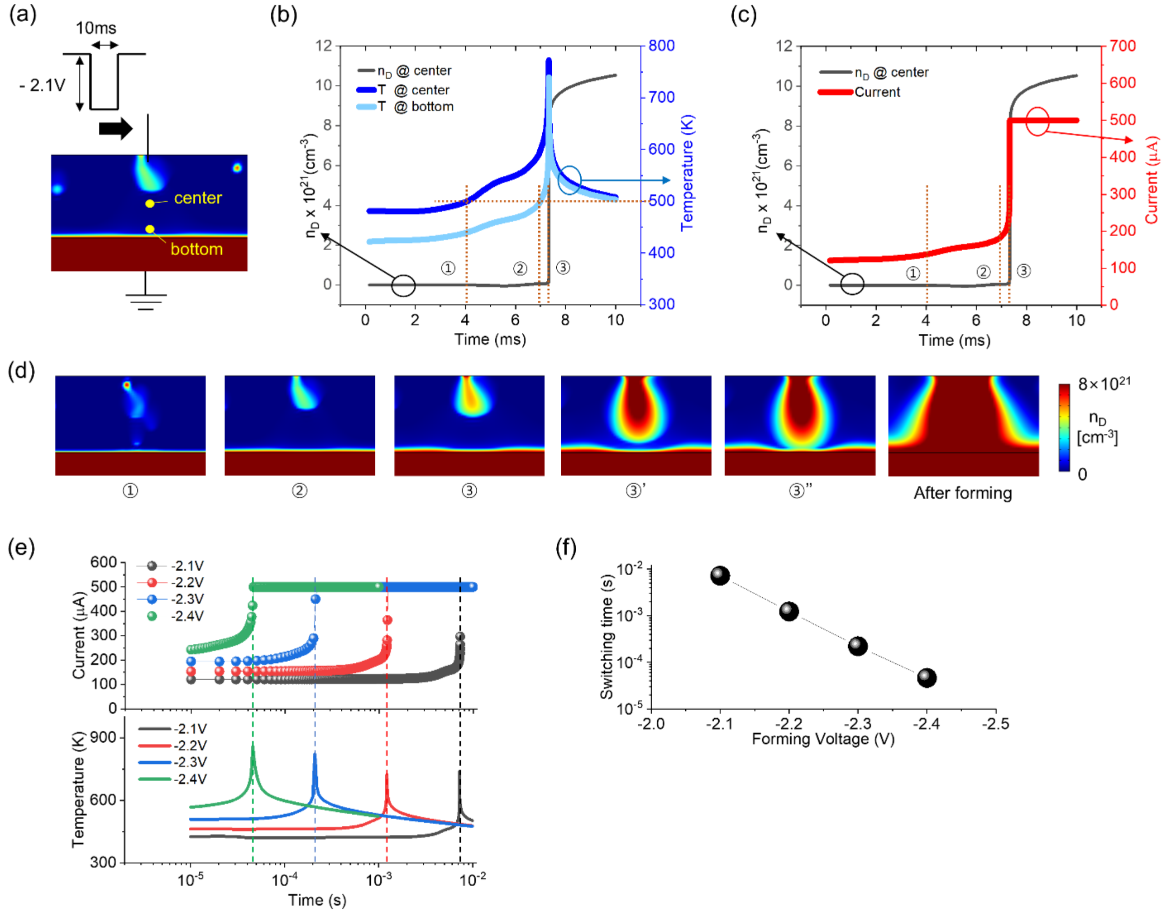


Figure 2-8: Forming process.
(a) Schematic of the device during pulse Forming. (b) n_D and T evolutions in the bottom and center regions marked in (a). (c) Current and n_D evolution during the Forming transition. (d) Evolution of the internal V_O configuration at different stages during Forming. (e) Current and temperature evolutions in the bottom region at different Forming voltages. (f) Forming speed vs. Forming voltage.

Importantly, the proposed model also clearly reveals that the device-to-device variation of the Forming voltage can be traced to the different initial V_O defect profiles in the switching layer (Figures 2-9(a)-(c)). By simply changing the initial V_O defect locations while keeping the number of V_Os constant in the switching layer, our simulation produces different Forming voltages for

different devices (Figure 2-9(d)). These results are consistent with experimental results, where devices even fabricated on the same chip show different Forming voltages from -1.7 to -2.2 V (average: -1.99 V, normal distribution (1σ): 0.11 V), as shown in Figure 2-9(e). These findings clearly highlight that even though the relatively uniform film is deposited, precise control of the Forming voltage can be challenging due to the stochastic V_O distributions. Precise control of the defect locations, e.g. through engineered electrode structures or ion blocking barriers[35,36], may be necessary if the tight distribution of the Forming voltage is required.

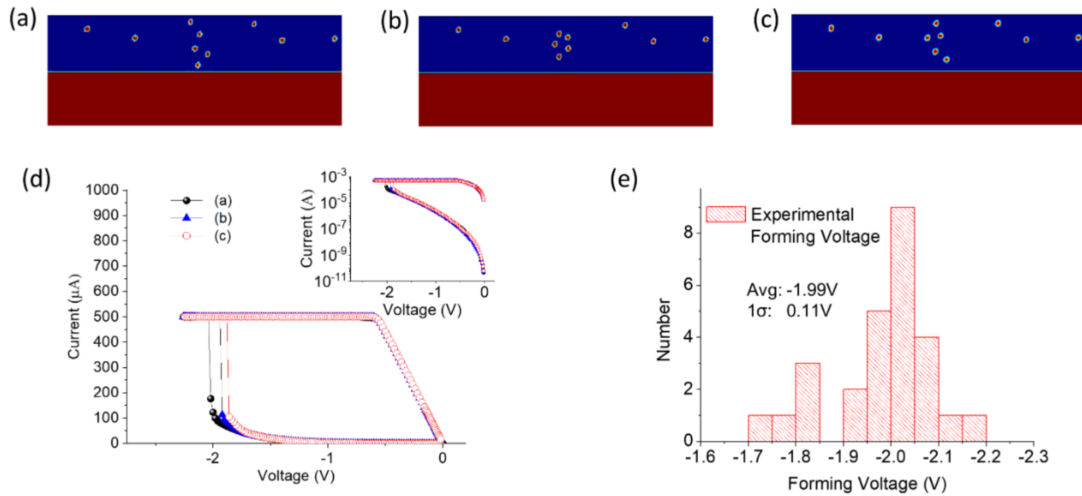


Figure 2-9: Variation of the Forming voltage originated from different initial states.
(a)~(c) Different initial states with the same number of V_Os in the switching layer. (d) Simulated Forming *I-V* characteristics with different initial states in (a)~(c). (e) Experimentally measured Forming voltage distribution from 27 samples fabricated on the same chip.

2.4 Reset and Set cycling (1R with I_{CC})

During the 1st Reset after the initial Forming process, our model shows the CF rupture occurs near the TE, caused by internal V_O redistribution driven by a combination of electric field and thermal effects (Figure 2-10(a)). Similar to the Forming process, the Reset process strongly

depends on the local temperature. As the CF is usually formed as a cone shape with narrow width near the TE (as verified by our simulation shown in Figure 2-3(d)), this region experiences more pronounced temperature rise due to Joule heating, as shown in Figure 2-10(a), and the elevated temperature activates the Vo drift/diffusion at this location. Once the V_Os in this region gain enough thermal energy at sufficient Reset voltage (e.g. 0.9 V), they will be repelled from the TE region by the applied electric field and create a V_O depletion gap between the TE and the CF, resulting in a decrease of the device conductance.

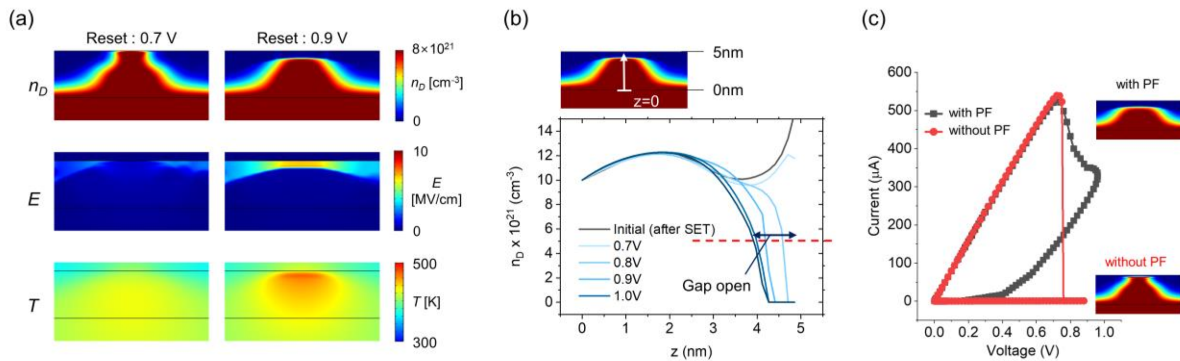


Figure 2-10: Simulated Reset process.
(a) 2-D maps of V_O concentration (n_D), Electric field (E), and temperature (T) in the device during the Reset process at different voltages. (b) 1D profile of n_D along the vertical direction during Reset with different Reset voltages. (c) Reset with and without the Poole-Frenkel (PF) effect in the depletion gap. A very narrow gap and failed cycling are obtained without PF.

Figure 2-10(b) shows the 1D profile of n_D along the vertical (z) direction during the Reset with different Reset voltages. These results verify that the gap region can be further increased by a larger Reset voltage, resulting in a further increase of the device resistance. Additionally, we found that it is important to maintain a sufficiently elevated temperature in the filament region during Reset to achieve reliable switching behavior. In our model, the PF conduction allows

sufficient current to continue to flow through the V_O depletion region during Reset and prevents the CF temperature from dropping abruptly soon after the gap is created. Figure 2-10(c) shows Reset simulation results with and without the PF conduction term. When the PF conduction term is removed from the model (i.e. removing the $\sigma_{PF}(E,T)$ term from Eq (2-8)), the temperature quickly drops as soon as the depletion gap appears, and subsequent V_O drift/diffusion processes essentially stop. As a result, this process fails to produce a sufficiently large V_O depletion gap and leads to cycling failure, as shown in Figure 2-10(c).

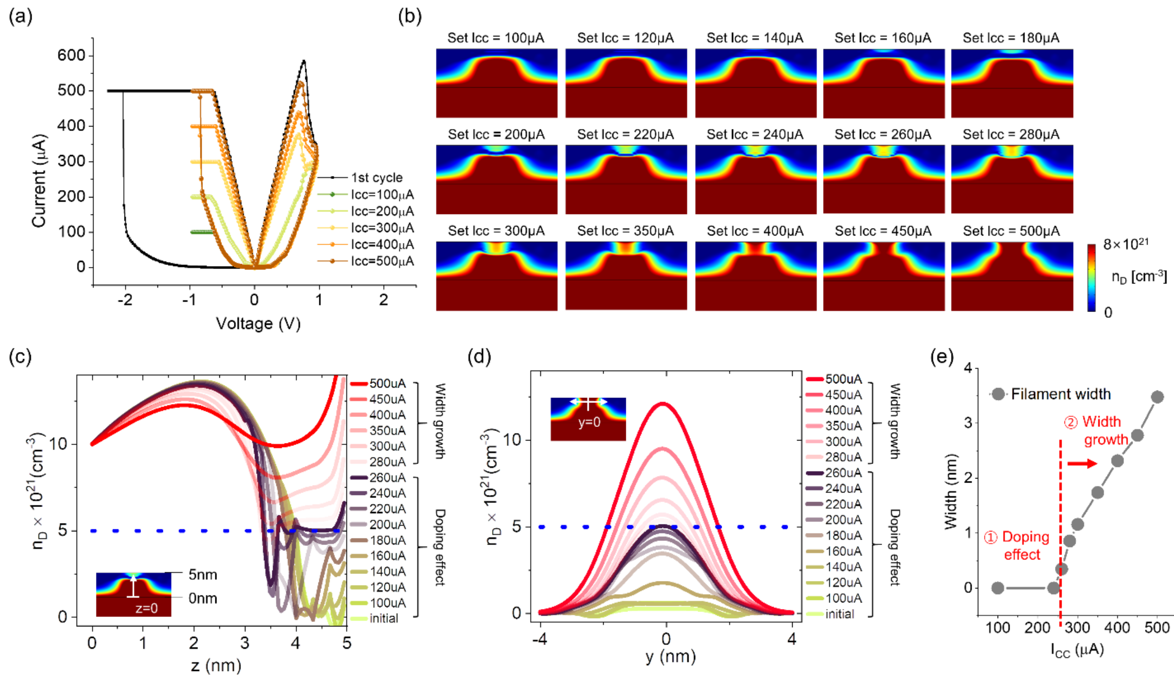


Figure 2-11: Two different filament growth mechanisms during Set with different I_{CC}s.
(a) Simulated Set and Reset DC *I-V* characteristics, and (b) 2-D maps of V_O concentration (n_D) at different I_{CC}s. The measurements start after Forming with 500 µA I_{CC} and the 1st Reset. Bulk-type doping effect dominates in the low I_{CC} (~260 µA) cases, while lateral filament growth can be observed in the high I_{CC} (280~500 µA) cases. (c) Vertical 1D n_D profiles after Set, for different I_{CC}s. The z = 0 position is the Ta_2O_5/TaO_x interface. (d) Horizontal 1D n_D profiles after Set, for different I_{CC}s. The y = 0 position is the center of the filament. (e) Filament width as a function of I_{CC}.

After the Forming process with 500 μA I_{CC} and the 1st Reset process, the Set process is simulated in a self-contained manner, with different levels of Set programming currents (I_{CC}s) controlled by the current modulation layer. The Set/Reset cycling can be reliably repeated through our simulations. Additionally, as shown in Figures 2-11(a) and (b), both the DC *I-V* characteristics and the CF shapes are governed by I_{CC} during the Set transition. Importantly, we observed two different filament growth mechanisms based on the I_{CC} level: a bulk-type doping mechanism and a filament width growth mechanism. At low Set I_{CC} (100 ~ 260 μA), n_D is gradually increased in the gap region, *i.e.* representing a bulk-type doping effect in the gap region, but the n_D level has not reached the metallic conduction level (5×10^{21} cm^{-3}) where E_{AC} becomes ~ 0 (Figure 2-5(a)) so this process corresponds to the conductance increase prior to CF completion. With high Set I_{CC} (280 ~ 500 μA) the filament is formed in regions with $n_D > 5\times10^{21}$ cm^{-3} and lateral filament growth can be observed with continued Set programming. This process corresponds to CF expansion.

These effects are also clearly observed in the 1D n_D profile plots along the vertical (z) and horizontal (y) directions for different I_{CC}s, as shown in Figures 2-11(c) and 2-11(d). The V_Os are first accumulated near TE and migrate to fill the gap region (Figure 2-11(c)), similar to the Forming process. After n_D reaches the metallic n_D level ($n_D = 5\times10^{21}$ cm^{-3}), further increase of I_{CC} leads to both increase in n_D and lateral expansion of the CF (Figure 2-11(d)). In this regime, the width of the metallic CF is linearly broadened as I_{CC} is increased (Figure 2-11(e)). The filament width modulation mode, however, yields a small dynamic range, since the device conductance is already high after the initial CF formation, while a much higher dynamic range is observed in the doping region, as we will discuss in more detail in the next Section.

Our analysis shows that different Forming I_{CC} leads to different filament shapes. For instance, decrease of I_{CC} (300 μA) during Forming results in a small filament size, producing a

more resistive HRS after Reset, compared with the case with the same Reset voltage but Formed with 500 μA I_{CC} (Figures 2-12(a)-(d)); while increasing I_{CC} (800 μA) during Forming leads to a large filament size with a more conductive HRS after Reset (Figures 2-12(e)-(h)). The filament shapes with different Set I_{CC}s (100 ~ 500 μA) after 300 μA I_{CC} Forming and 1st Reset are also analyzed and compared to the cases with 500 μA Forming I_{CC} (Figure 2-13). Interestingly, even with the same Set I_{CC}, the filament shapes and conductance levels are not the same for the two Forming conditions (I_{CC} = 300 μA and 500 μA), which is another evidence that the conductance of a memristor depends on the history of external stimulation[17,37]. Furthermore, the long-term potentiation (LTP) behavior with incremental Set I_{CC}s with different Forming conditions (300 μA vs. 500 μA) was investigated. It was found that Forming with a lower I_{CC} provides a larger dynamic range (on/off ratio), although the conductance update slope becomes sharper which makes it more challenging to fine-tune the conductance/weight (Figure 2-14).

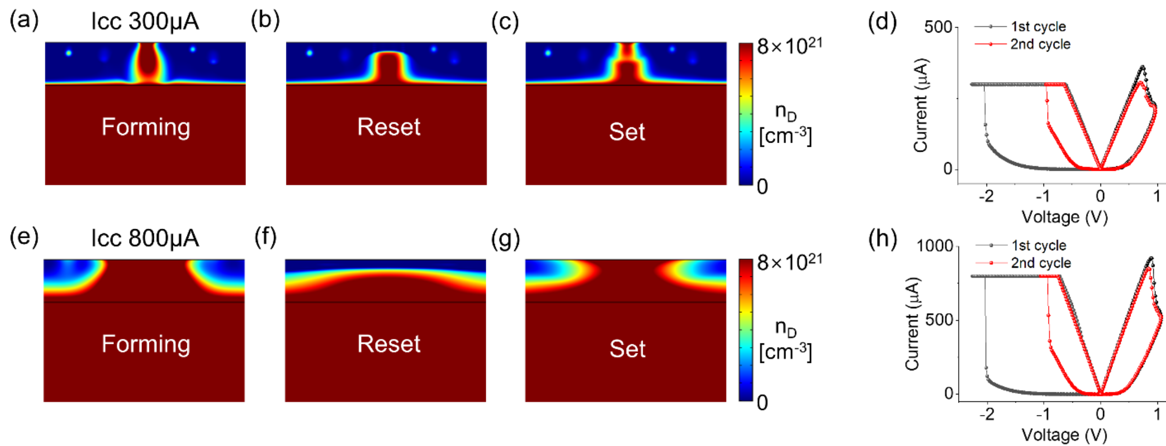


Figure 2-12: Simulated RS behavior with different I_{CC}s.
(a-c): 2-D maps of n_D for (a) Forming, (b) Reset, and (c) Set processes with 300 μA of I_{CC}, along with the simulated DC *I-V* characteristics (d). (e-g) 2-D maps of n_D for (e) Forming, (f) Reset, and (g) Set process with 800 μA of I_{CC}, along with the simulated DC *I-V* characteristics (h).

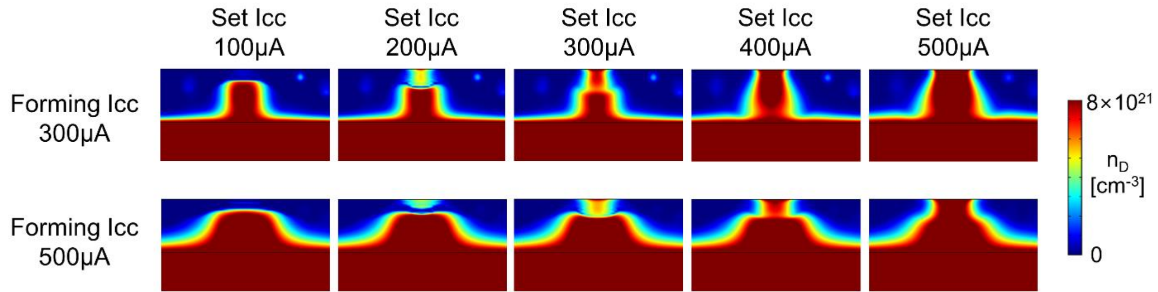


Figure 2-13: 2-D maps of n_D profile after Set, for different Set I_{CC}s.
The devices first go through (a) Forming with 300 μA I_{CC} and 1st Reset, or (b) Forming with 500 μA I_{CC} and 1st Reset, respectively. Different V_O configurations are observed during Set, for devices formed at these different Forming conditions, even if the same Set I_{CC}s are used.

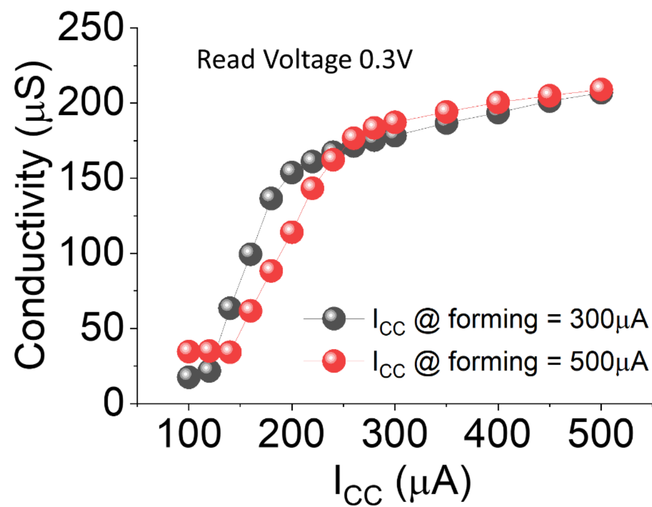


Figure 2-14: Simulated LTP behaviors with incremental I_{CC}, for different Forming conditions.
Forming with lower I_{CC} provides a larger dynamic range, but a steeper slope which may make it more difficult to fine-tune the conductance.

This observation explains the discrepancy of filament shapes between the Forming and Set states even with the same I_{CC} (Figure 2-3(d)). Specifically, the Forming transition starts from the initial state and requires a larger voltage and a higher temperature to create a filament, while the Set process starts on the conditions where the filament is already created and partially ruptured.

The different initial states lead to a different electric field and temperature profiles during Forming and Set, and produce different filament shapes in the end, as shown in Figures 2-3(d), 2-12 and 2-13. On the other hand, during Set/Reset cycling the same filament growth/rupture conditions can be maintained, leading to reliable Set/Reset cycles as shown in Figure 2-3(d).

2.5 1T1R simulation

In 1T1R devices, the transistor, controlled by the gate voltage (V_G), can effectively modulate the maximum current and suppress the off current level, thus allowing more precise control of the memristor conductance. In particular, current overshoot due to discharging current from parasitic capacitances can be effectively suppressed[38,39], allowing more reliable Forming and Set processes. The transistor also serves as an excellent selector that suppresses sneak currents in an array. As a result, the 1T1R structure is widely used for the practical implementation of large memristor arrays[11,12,40,41], especially for inference applications and edge computing[42,43] where the data processing occurs close to the point of data creation.

We used n-type enhancement-mode transistors to implement the 1T1R devices (Figure 2-15(a) and Figure 2-6(b)). The CMOS chip with standalone transistors are fabricated in a commercial fab, and TaO_x-based memristor devices are integrated on the Drain side of the transistor afterward (Figure 2-15(a) and Figure 2-16). The transistor *I-V* model is successfully implemented in the current modulation layer in our model (Figure 2-6(b)). In the integrated 1T1R structure, the bottom electrode of the bilayer memristor is connected to the Drain of the transistor to apply effective negative Forming and Set voltages to the memristor (Figure 2-16(b)).

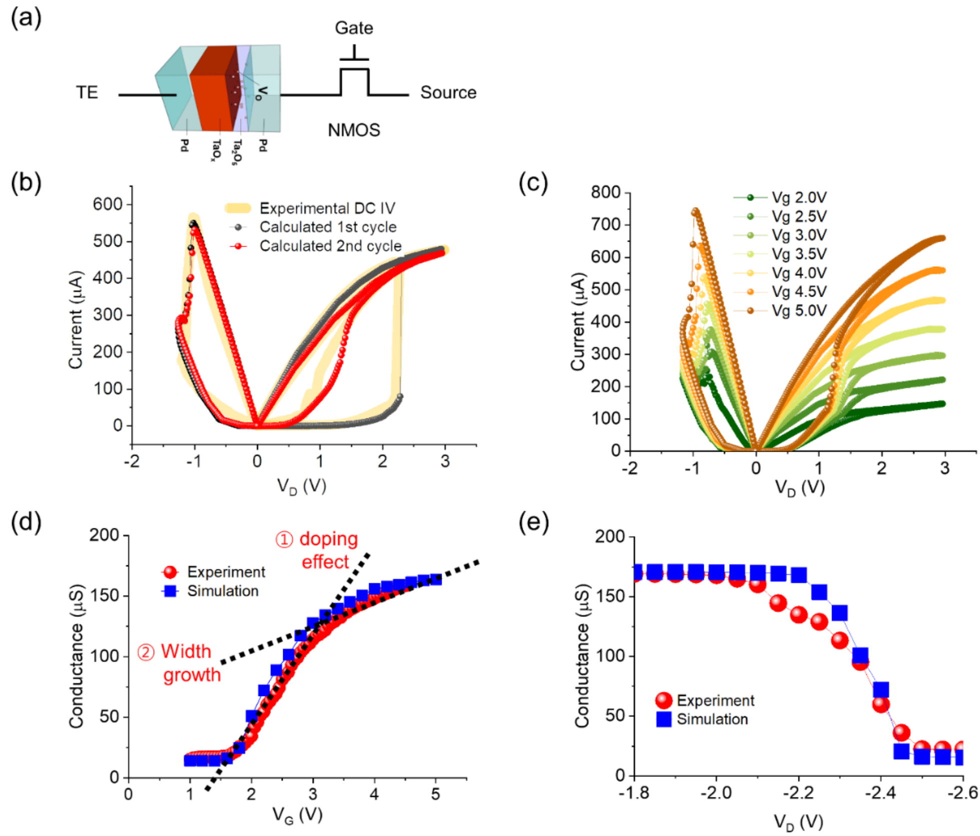


Figure 2-15: 1T1R characteristics with different gate voltage (V_G).
(a) 1T1R schematic. (b) Measured and simulated 1T1R DC *I-V* curves with V_G = 4 V during Forming and Set. (c) Simulated Set and Reset curves with different V_G. (d) Measured and simulated LTP behaviors with incremental V_G with 3 V and 10 µs V_D pulses. Different slopes and dynamic ranges can be clearly observed. (e) Measured and simulated LTD behavior with different V_D with a 1 µs pulse.

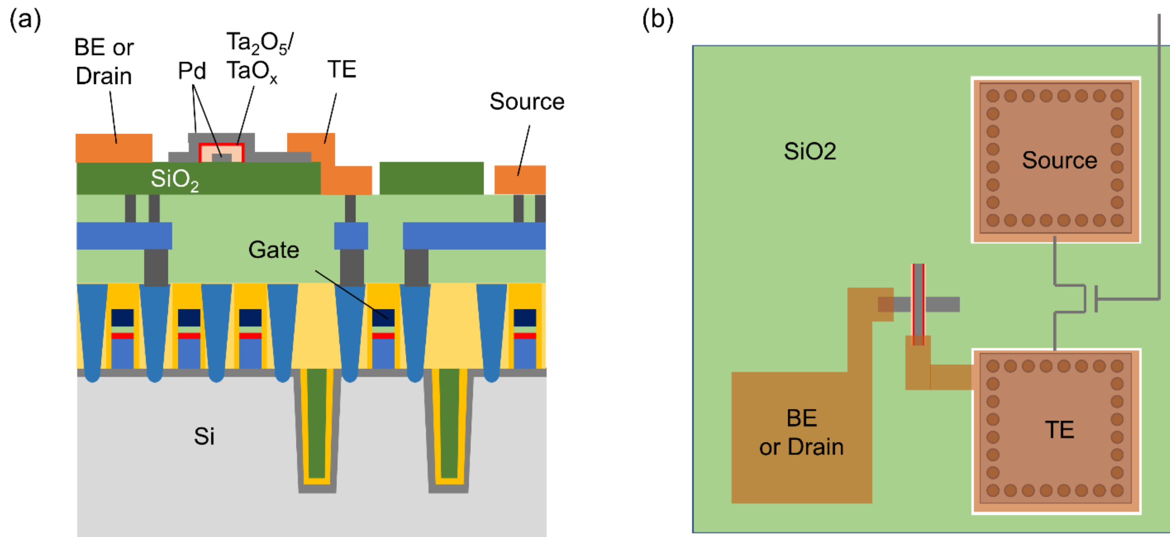


Figure 2-16: Schematic of the 1T1R structure.
(a) A cross-sectional view of the 1T1R structure with the Pd/Ta_2O_5/TaO_x/Pd bilayer memristor device. (b) Top view. The BE of the memristor is connected to the Drain of the transistor to provide negative Forming and Set voltages to the memristor device.

Similar to the 1R case with I_{CC}, the simulation starts from the initial state where V_Os are non-uniformly distributed in the switching layer. The model can accurately capture the Forming, Reset, and Set switching characteristics. Figure 2-15(b) shows that during Forming and Set switching, the maximum current is limited by the transistor saturation current, e.g. I_{CC} ~ 470 μA at $V_G = 4$ V and $V_D = 3$ V. During Forming and Set transitions, the strong positive feedback process between Joule heating and filament formation is stopped when the memristor current reaches the transistor saturation current, leading to reversible and stable CF formation in the switching layer. In the Reset process, the transistor is fully turned ON to minimize the series resistance effect. Because only positive bias is available for NMOS to switch the memristor in the 1T1R form, we applied the positive bias to the Source (0 ~ 1.2 V) of the original transistor structure, leading to a negative voltage at the Drain (Figure 2-15(b)). The simulated consecutive DC *I-V* switching cycles

match well with the measured results. Here, the Forming, Set and Reset voltages are slightly increased compared to those of 1R device due to the series transistor resistance.

The conductance level of the 1T1R device can be precisely controlled by changing the gate voltage V_G during Set. Results of different Set conditions after Forming with V_G = 4 V and the 1st Reset are shown in Figure 2-15(c). Modulation of V_G from 2 V to 5 V during Set clearly shows the ability to control the conductance levels in the 1T1R device. The model also clearly captures the pulsed long-term potentiation (LTP) and long-term depression (LTD) behaviors in the 1T1R device, consistent with experimental results. For LTP, 3 V and 10 µs V_D pulses were applied to the Drain with varying V_G. Even in the pulse condition, bulk-type doping and lateral filament growth regions (Figure 2-15(d)) are clearly identified, with different slopes and dynamic ranges with increasing V_G. Similar to the effects of I_{CC} in the 1R structure, a low V_G (1.5 V ~ 3 V) leads to bulk-type doping effect with a large dynamic range, while a large V_G (3 V ~ 5 V) leads to the filament width growth effect with a small dynamic range. Here, the read condition was V_D = 0.3 V and V_G = 5 V. Note that the conductance of the 1T1R device cannot be modulated by V_G lower than 1.5 V both in simulation and in the experiment, where the current level is not large enough to initiate V_O migration, as shown in Figure 2-15(d).

Analog LTD behaviors can also be achieved by modulating V_D (Figure 2-15(e)), with excellent agreements of the modeling and experimental results. Note that LTD may be harder to control compared to LTP in the 1T1R system due to the changing V_{GS} during Reset transition since the device resistance is connected to the effective source side of the transistor and varies during Reset. Fortunately, online learning based on LTP behaviors only can be effectively implemented by using a pair of memristor devices[44]. For inference applications that do not require online learning, it is also sufficient to achieve precise conductance programming through the program-

and-verify (PNV) scheme[11,40], where the initial HRS state can be achieved by applying a large positive voltage to the Source (e.g. $-V_D$ in Figure 2-15(e)). This approach allows the weights to be precisely reprogrammed during the infrequent model updates.

2.6 Conclusion

In this chapter, a device model that can self-consistently and quantitatively describe the dynamic RS processes including Forming and Set/Reset cycles is successfully developed. Excellent agreements with experimental DC and pulse measurements in 1R and 1T1R devices were obtained. Forming was observed to originate from the initial non-uniform defect distribution, and electric field focusing and localized thermal effects were found to strongly affect the filament formation process. By controlling I_{CC}, the V_O configuration can be systematically tuned during CF growth, resulting in different RS behaviors. Two different filament growth modes were observed, leading to different conductance modulation slopes and dynamic ranges. In particular, a low programming current induces a bulk-type doping effect, resulting in linear conductance/weight updates and a large dynamic range. The modes and the observations will help continued device optimizations and applications in memory and low-power neuromorphic computing applications.

References

1. Zidan, M. A., Strachan, J. P. & Lu, W. D. The future of electronics based on memristive systems. *Nat. Electron.* **1**, 22–29 (2018).
2. Ielmini, D. & Wong, H. S. P. In-memory computing with resistive switching devices. *Nat. Electron.* **1**, 333–343 (2018).
3. Xia, Q. & Yang, J. J. Memristive crossbar arrays for brain-inspired computing. *Nat. Mater.* **18**, 309–323 (2019).
4. Lee, M. J. *et al.* A fast, high-endurance and scalable non-volatile memory device made from asymmetric Ta2O5-x/TaO2-x bilayer structures. *Nat. Mater.* **10**, 625–630 (2011).
5. Torrezan, A. C., Strachan, J. P., Medeiros-Ribeiro, G. & Williams, R. S. Sub-nanosecond switching of a tantalum oxide memristor. *Nanotechnology* **22**, 485203 (2011).
6. Pi, S. *et al.* Memristor crossbar arrays with 6-nm half-pitch and 2-nm critical dimension. *Nat. Nanotechnol.* **14**, 35–39 (2019).
7. Prezioso, M. *et al.* Training and operation of an integrated neuromorphic network based on metal-oxide memristors. *Nature* **521**, 61–64 (2015).
8. Yao, P. *et al.* Face classification using electronic synapses. *Nat. Commun.* **8**, 15199 (2017).
9. Alibart, F., Zamanidoost, E. & Strukov, D. B. Pattern classification by memristive crossbar circuits using ex situ and in situ training. *Nat. Commun.* **4**, 3072 (2013).
10. Bayat, F. M. *et al.* Implementation of multilayer perceptron network with highly uniform passive memristive crossbar circuits. *Nat. Commun.* **9**, 2331 (2018).
11. Li, C. *et al.* Efficient and self-adaptive in-situ learning in multilayer memristor neural networks. *Nat. Commun.* **9**, 2385 (2018).

12. Li, C. *et al.* Analogue signal and image processing with large memristor crossbars. *Nat. Electron.* **1**, 52–59 (2018).

13. Sheridan, P. M. *et al.* Sparse coding with memristor networks. *Nat. Nanotechnol.* **12**, 784–789 (2017).

14. Du, C. *et al.* Reservoir computing using dynamic memristors for temporal information processing. *Nat. Commun.* **8**, 2204 (2017).

15. Choi, S., Shin, J. H., Lee, J., Sheridan, P. & Lu, W. D. Experimental Demonstration of Feature Extraction and Dimensionality Reduction Using Memristor Networks. *Nano Lett.* **17**, 3113–3118 (2017).

16. Cai, F. *et al.* A fully integrated reprogrammable memristor–CMOS system for efficient multiply–accumulate operations. *Nat. Electron.* **2**, 290–299 (2019).

17. Chua, L. O. Memristor-The Missing Circuit Element. *IEEE Trans. Circuit Theory* **18**, 507–519 (1971).

18. Waser, R., Dittmann, R., Staikov, C. & Szot, K. Redox-based resistive switching memories nanoionic mechanisms, prospects, and challenges. *Adv. Mater.* **21**, 2632–2663 (2009).

19. Lee, J. & Lu, W. D. On-Demand Reconfiguration of Nanomaterials: When Electronics Meets Ionics. *Adv. Mater.* **30**, 1702770 (2018).

20. Zhu, X., Lee, S. H. & Lu, W. D. Nanoionic Resistive-Switching Devices. *Adv. Electron. Mater.* **5**, 1900184 (2019).

21. Sun, W. *et al.* Understanding memristive switching via in situ characterization and device modeling. *Nat. Commun.* **10**, 3453 (2019).

22. Chua, L. Resistance switching memories are memristors. *Appl. Phys. A Mater. Sci. Process.* **102**, 765–783 (2011).

23. Strukov, D. B. & Williams, R. S. Exponential ionic drift: Fast switching and low volatility of thin-film memristors. *Appl. Phys. A Mater. Sci. Process.* **94**, 515–519 (2009).

24. Ielmini, D. Modeling the universal set/reset characteristics of bipolar RRAM by field- and temperature-driven filament growth. *IEEE Trans. Electron Devices* **58**, 4309–4317 (2011).

25. Kim, S., Choi, S. & Lu, W. Comprehensive physical model of dynamic resistive switching in an oxide memristor. *ACS Nano* **8**, 2369–2376 (2014).

26. Larentis, S., Nardi, F., Balatti, S., Gilmer, D. C. & Ielmini, D. Resistive switching by voltage-driven ion migration in bipolar RRAM - Part II: Modeling. *IEEE Trans. Electron Devices* **59**, 2468–2475 (2012).

27. Magyari-Köpe, B., Park, S. G., Lee, H. D. & Nishi, Y. First principles calculations of oxygen vacancy-ordering effects in resistance change memory materials incorporating binary transition metal oxides. *J. Mater. Sci.* **47**, 7498–7514 (2012).

28. Kim, S. *et al.* Physical electro-thermal model of resistive switching in bi-layered resistance-change memory. *Sci. Rep.* **3**, 1680 (2013).

29. Adapted from comsol.com. *(https://www.comsol.com/comsol-multiphysics)*

30. Yang, Y., Choi, S. & Lu, W. Oxide heterostructure resistive memory. *Nano Lett.* **13**, 2908–2915 (2013).

31. Larentis, S., Cagli, C., Nardi, F. & Ielmini, D. Filament diffusion model for simulating reset and retention processes in RRAM. *Microelectron. Eng.* **88**, 1119–1123 (2011).

32. Lee, J., Schell, W., Zhu, X., Kioupakis, E. & Lu, W. D. Charge Transition of Oxygen Vacancies during Resistive Switching in Oxide-Based RRAM. *ACS Appl. Mater. Interfaces* **11**, 11579–11586 (2019).

33. Ielmini, D., Nardi, F. & Cagli, C. Physical models of size-dependent nanofilament formation and rupture in NiO resistive switching memories. *Nanotechnology* **22**, (2011).

34. Hu, C. C. Modern Semiconductor Devices for Integrated Circuits. (Prentice Hall, 2010).

35. Liu, Q. *et al.* Controllable growth of nanoscale conductive filaments in solid-electrolyte-based ReRAM by using a metal nanocrystal covered bottom electrode. *ACS Nano* **4**, 6162–6168 (2010).

36. Lee, J., Du, C., Sun, K., Kioupakis, E. & Lu, W. D. Tuning Ionic Transport in Memristive Devices by Graphene with Engineered Nanopores. *ACS Nano* **10**, 3571–3579 (2016).

37. Strukov, D. B., Snider, G. S., Stewart, D. R. & Williams, R. S. The missing memristor found. *Nature* **453**, 80–83 (2008).

38. Walczyk, D. *et al.* Resistive switching characteristics of CMOS embedded HfO2-based 1T1R cells. *Microelectron. Eng.* **88**, 1133–1135 (2011).

39. Wan, H. J. *et al.* In situ observation of compliance-current overshoot and its effect on resistive switching. *IEEE Electron Device Lett.* **31**, 246–248 (2010).

40. Wang, Z. *et al.* Fully memristive neural networks for pattern classification with unsupervised learning. *Nat. Electron.* **1**, 137–145 (2018).

41. Wang, Z. *et al.* Reinforcement learning with analogue memristor arrays. *Nat. Electron.* **2**, 115–124 (2019).

42. Chen, W.-H. *et al.* CMOS-integrated memristive non-volatile computing-in-memory for AI edge processors. *Nat. Electron.* **2**, 420–428 (2019).

43. Xu, X. *et al.* Scaling for edge inference of deep neural networks. *Nat. Electron.* **1**, 216–222 (2018).

44. Ambrogio, S. *et al.* Equivalent-accuracy accelerated neural-network training using analogue memory. *Nature* **558**, 60–67 (2018).

Chapter 3

A Fully Integrated Memristor–CMOS System for Neuromorphic Computing

In this chapter, we discuss a fully functional, hybrid memristor-CMOS computing chip including a 54x108 passive memristor crossbar array directly integrated on CMOS circuitry which contains a full set of mixed-signal interface blocks and a digital processor. Previous implementations have largely relied on external printed-circuit boards (PCBs) or discrete components to provide control, signal conversion and data I/O, severely limiting the circuit functions[1-7]. Demonstrating the potential of memristor-based computing hardware requires the development of complete, functional systems, where the memristor crossbars are integrated with necessary analog interface circuitry (including analog-to-digital converters (ADCs) and digital-to-analog converters (DACs)), digital buses and ideally a programmable processor to control the digital and analog components. Integrating all the necessary functions on a chip will be key to enabling the practical implementation of memristor-based computing systems and allowing the prototypes to be scaled to larger systems.

In this work, the system supports charge-domain operation to overcome the non-linear *I-V* characteristics of memristor devices through pulse width modulation and custom analog-digital converters (ADC). The integrated chip allows the mapping of different neuromorphic and machine learning algorithms on-chip through simple software changes. To verify the hybrid neuromorphic chip operation, we demonstrate three widely used models – a perceptron network, a sparse coding algorithm, and a bilayer principal component analysis (PCA) system with an unsupervised feature extraction layer and a supervised classification layer – experimentally on the same chip[8].

3.1 System architecture and circuit design (designed by Prof. Flynn's group & Prof. Zhang's group)

The custom CMOS circuitry includes an OpenRISC processor with 64KB SRAM and a mixed-signal interface with 162 configurable channels as shown in Figure 3-1(a) and (b). The processor configures the mixed-signal interface via a set of global configuration registers and performs write and read operations through digital to analog converters (DACs) and analog to digital converters (ADCs). Each channel is set to either have an ADC or 1 of the 3 DACs connected to a row or column of the crossbar. For example, at forwarding pass mode, all rows are configured as DACs and all columns are configured as ADCs. At write mode, all rows and columns are configured as DACs. The ADC or DAC connection is set in the mode register and the type of DAC connections is set in the DAC register along with the 6b DAC pulse input. The timing generator handles both the ADC start signal and creates the duty-cycled pulse-train for the DACs.

Since the on-chip processor is mainly used for register manipulations, the reduced instruction set Alternate Lightweight OpenRISC processor (AltOR32) is used in the design to minimize area and power consumption. The SRAM is divided into 3 parts. The processor instruction and data memory are mapped to 32 KB SRAM data memory. The remaining 32KB are dual-port SRAMs that support simultaneous input/output and are assigned as two "ping-pong" memory banks for potential data buffering, although in this study the ping-pong memory banks were not used since all data can be fit in the data memory.

(a) (b)

Figure 3-1: Chip System Architecture and mixed signal interface design.
(a) The integrated memristor/CMOS chip consists of the digital controller and bus shown in green, the mixed-signal interface shown in red, and the memristor crossbar shown in blue. (b) The mixed-signal interface is comprised of global configuration registers and timing generation shown in brown, and 162 configurable channels (shown in green) to provide input and measure output to and from each row and column of the memristor crossbar.

During operation, the processor is first used to configure the mixed-signal interface by setting the configuration registers. The processor is then paused and the control is handed off to the timing generator of the mixed-signal interface. The timing generator operates for 64 cycles during which VMM operations and memristor weight updates are performed. The control then goes back to the processor.

The mixed-signal interface was designed by our collaborators, Prof. Mike Flynn's group and Prof. Zhengya Zhang's group. Specifically, Prof. Flynn's group developed a hybrid ADC architecture comprised of a 5b first stage first-order incremental ΔΣ ADC and a 9b second stage SAR ADC, as shown in Figure 3-2, to avoid the use of a large charge integrating capacitor while achieving high resolution. With 1b of stage redundancy, the ADC resolves 13b after digital correction.

A regulation loop sets the input voltage (V_{IN}) of the ADC to maintain a constant virtual ground voltage (V_{REG}, nominally 1.2V). The ADC uses a current divider to take a small portion

(configurable from 1/64 to 8/64) of the input current to handle a large input current without large integrating capacitors and power-hungry active integrator. The capacitor banks (Ch. 1, 2 and D) that work as passive integrators sequentially integrate the divided input charge and transfer the charge to the active integrator when the integrated voltage on a channel hit the hysteresis threshold. The dummy channel (Ch. D) continuously integrates the divided input charge during the inner-product cycle to avoid the loss of the input charge during the non-overlapping control time of the Ch. 1 and Ch. 2 and transfers the charge at the end of the inner product cycle.

The hysteresis generates the control signals of the active and passive integrators asynchronously depending on the input current. This reduces the output noise of the ADC when the input current is low since the number of active integration is reduced. The active integrator uses a three-stage fully-differential ring amplifier with an auxiliary first-stage based auto-zero to improve the energy efficiency of the ADC.

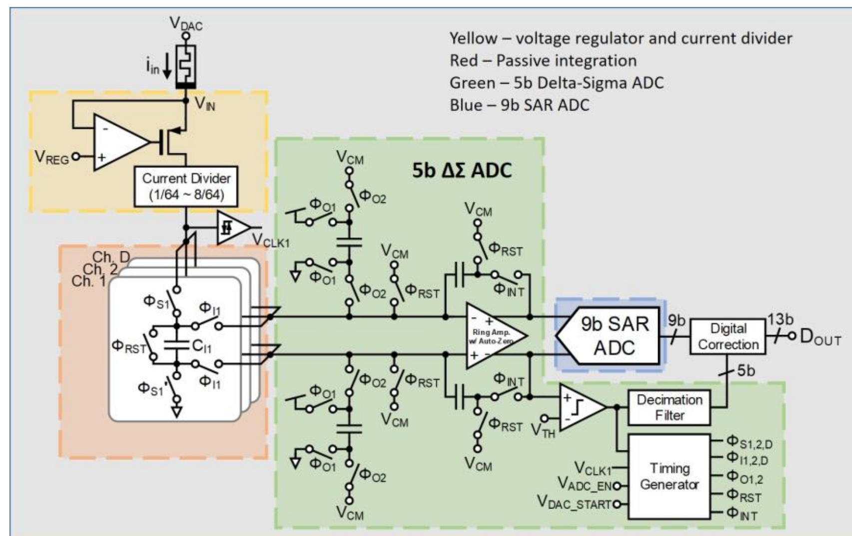


Figure 3-2: Schematic of 13b current-integrating ADC.

For DAC operation, we introduced the number of pulses to represent the input amplitude (effectively modulating the pulse width in the discrete-time domain) to eliminate errors due to pulse rise and fall time, instead of directly modulating the pulse width, as shown in Figure 3-3.

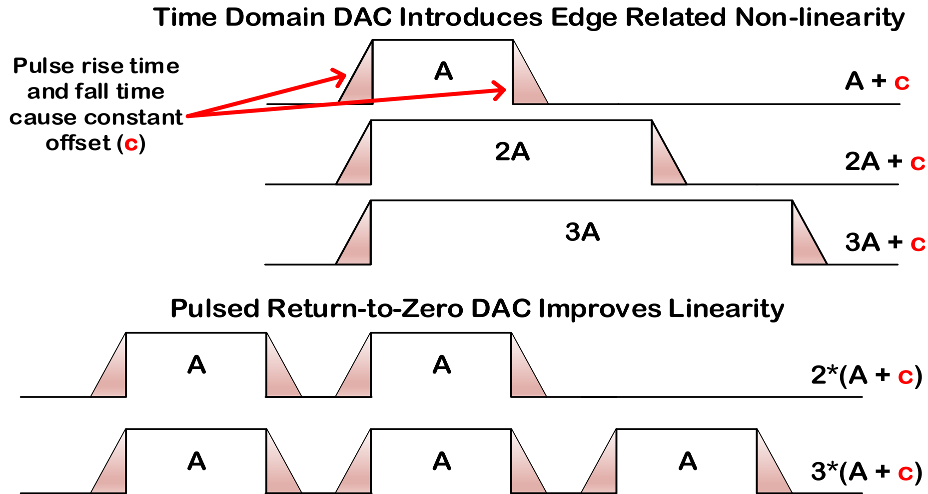


Figure 3-3: Pulsed-mode DAC scheme.

When performing VMM, the chip is configured at read mode. During the VMM operation, we apply a discrete-time pulse-train input and measure the accumulated charge from each column (row). The column (row) ADCs present a 1.2 V virtual ground while the row (column) DACs apply a 6-bit programmable train of fixed-amplitude 0.6 V “read” pulses (1.8 V-1.2 V). The integrating ADCs measure the collected charges over the input period.

When performing weight update, the chip is configured at write/erase mode (depends on the need to increase/decrease the weight). During the write operation, we apply discrete-time pulse-trains at both rows and columns. When writing (erasing), the row (column) DACs apply 6-bit programmable trains of fixed-amplitude “High voltage” pulses (1.9 V-1 V) and the column (row) DACs apply 6-bit programmable trains of fixed-amplitude “Low voltage” pulses (0.1 V-1V) with the same duration, which effectively generates pulses with 1.8 V (-1.8 V) voltage drop across

the device. The idle level of both “High voltage” and “Low voltage” are chosen at 1 V, which is the halfway of the voltage drop to provide write protection for unselected devices in the array.

3.2 Data path during training and inference on chip

As shown in Figure 3-4, at the first step, we program all instructions for necessary mathematical calculations and data storage in C code and compile the C code into binary machine code. This process is performed on a personal computer. Note the programming and compilation only need to be performed once for every task. The binary code will be used to run the training and inference algorithms, and allocate the memory space for the necessary input & output data and all the internal intermediate variables.

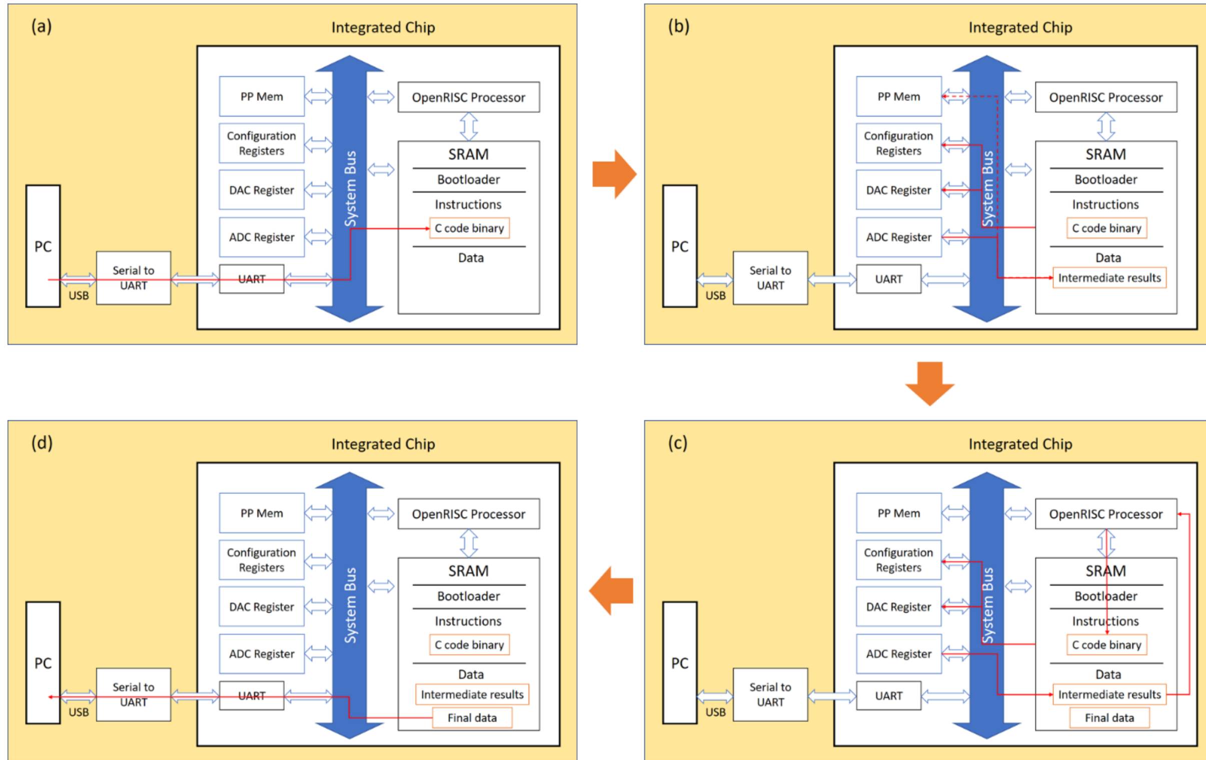


Figure 3-4: Data path during training and inference.
(a) All instructions for necessary mathematical calculations and data storage are first programed in C code and compiled. The entire compiled code is then loaded into the on-chip 32kB SRAM data memory by a bootloader. (b) The binary program instructions are executed through the OpenRISC processor to run the training and inference algorithms. The vector-matrix multiplication (VMM) results are read as charge values from the ADCs. (c) During training, the required weight updates are calculated through the OpenRISC processor. Afterwards, the DACs are configured to supply the desired pulse widths to the update the memristors. (d) The final output can be transferred to a personal computer and accessed by the user.

Afterward, we load the final binary code to the chip. Specifically, the binary code is sent through the USB port of the computer, and a Serial-to-UART (Universal Asynchronous Receiver-Transmitter) board converts the serial data in the USB to the UART protocol and sends the compiled code to the memristor chip through the UART interface. The entire compiled binary code is loaded into the on-chip SRAM by a bootloader.

After the binary code is loaded into the on-chip SRAM, the binary program instructions are executed, and the entire application will be run on-chip. The instructions set the circuit

configurations and DAC registers (i.e. row/column configurations and the pulse widths for weight updates and VMM operations). The VMM results are read as charge values from the ADCs and stored in the data memory.

Third, the VMM outputs are processed by the on-chip OpenRISC processor to calculate the required weight updates or for running other operations of the algorithm. The processor executes the algorithm as programmed by the C code and sends the results (i.e. batch update values) as instructions to the on-chip registers, and the DACs are configured to the new configurations for the next operations.

Finally, after all the instructions have been executed, the output can be supplied to the user through the Serial-to-UART interface.

We note in the networks we run most of the operations are multiplication and summations that can be readily implemented in the memristor array and the OpenRISC processor. There are two functions that require more complex calculations, namely the Sigmoid (for neurons in the PCA classification layer) and Softmax (for neurons in the SLP classification layer) activation functions. The Sigmoid is a one-to-one mapping function that can be approximated by a piece-wise linear function. As a result, we have successfully implemented the Sigmoid function on-chip using the OpenRISC processor, and all experiments for the LCA network and the entire PCA + Classification network are implemented on-chip, without communication with the computer during the learning and testing process. However, the Softmax function is an n-to-n mapping that computes the ratio of an exponential output of the current class over the sum of the exponential outputs of all classes, which is challenging to implement without floating-point operations.

In this prototype integrated chip, the OpenRISC processor we use is a stripped-down version called "AltOR32" (Alternative Lightweight OpenRISC 32-bit version) without the

floating-point unit, to reduce power, area and design costs since most of the calculations performed in our algorithms are multiplication and summations. The decimal values of the parameters were approximated with fixed point representation in the integer domain.

Therefore, in the Greek letter classification experiment that uses the Softmax function during the training process, we made the compromise of calculating the Softmax activation off-chip in software (Python), the activation value was then passed back to the chip for batch-gradient descent calculations. We note the current version of the system is developed as a prototype that demonstrates the integration of memristor arrays with functional periphery circuitry. Not all functions, such as floating-point support, were targeted. Designs that include floating-point unit supports, as well as improved training protocols and more efficient processor designs[9-11] will be incorporated in future studies.

3.3 Integration of memristor array on CMOS chip

A key challenge of the chip is to integrate the 54×108 WO_x-based memristor crossbar array directly on top of the CMOS circuits, as shown in Figure 3-5(a).

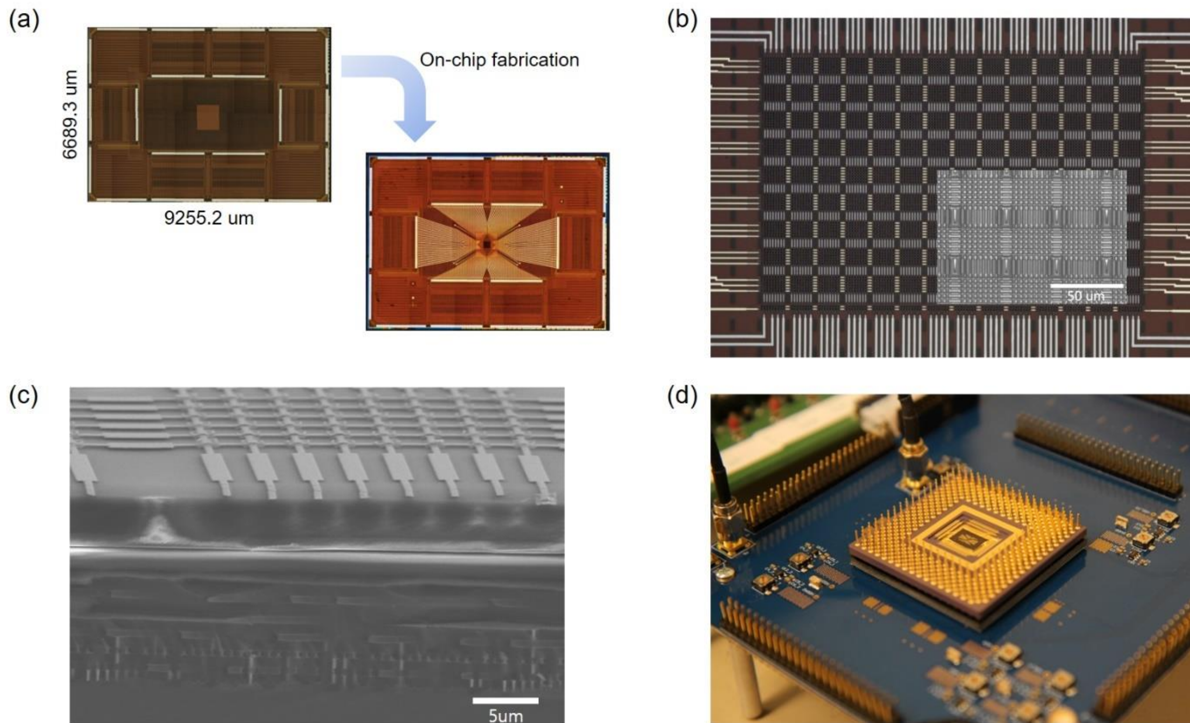


Figure 3-5: Fully integrated memristor/CMOS chip.
(a) Optical microscopic images of a memristor crossbar array integrated on the chip, with portions of the CMOS circuitry visible underneath the chip surface. (b) Magnified image of the integrated 54×108 memristor array region. Inset: scanning electron microscope (SEM) image of the WO_x memristor crossbar. (c) A cross-sectional SEM image of the integrated chip, showing the memristor crossbar array fabricated on top of the CMOS circuits and the different CMOS wiring layers underneath the memristor array. (d) The integrated chip wire-bonded on a pin grid array (PGA) package and testing set up used to power and test the integrated memristor/CMOS chip.

The process flow is the following. First, the bottom electrode (BE) patterns with 500 nm width are defined by e-beam lithography on CMOS chip, the 80 nm thick Au BEs are then deposited (with Ni/Cr adhesion layer underneath) by e-beam evaporation and lift-off processes. Next, 300 nm of SiO_2 is deposited by plasma-enhanced chemical vapor deposition (PECVD), followed by reactive-ion etch back to form a spacer structure along the sidewalls of the BEs. To minimize crosstalk, the switching layer is only deposited at the cross-point regions through patterns defined by e-beam lithography. The switching layer is formed by first depositing 20 nm thick W through DC sputtering and lift-off processes in the e-beam patterned regions, then through

rapid thermal annealing of the patterned W islands with oxygen gas at 425°C for 60 s to form the WO_x switching material. Afterward, the TEs (Pd (40 nm)/Au (90 nm)) with 500 nm width are patterned and deposited by e-beam lithography, e-beam evaporation, and lift-off processes. Finally, metallization processes are performed by photolithography to connect the crossbar electrodes with the CMOS landing pads that are left open during the CMOS circuit fabrication process. An in-situ etch process is performed to remove the native aluminum oxide on the CMOS landing pads, followed by deposition of 800 nm thick Al with DC sputtering and lift-off processes to ensure step coverage of the deeply recessed landing pads.

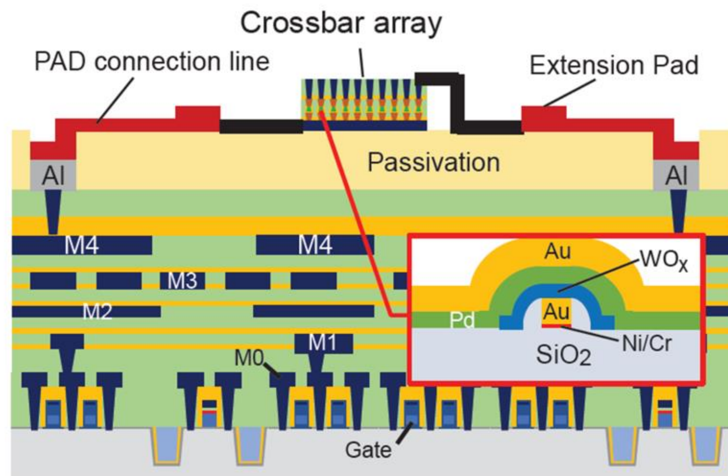


Figure 3-6: Cross-section schematic of the integrated chip.
This shows connections of the memristor array with the CMOS circuitry through extension lines and internal CMOS wiring. Inset, cross-section of the WO_x device.

Figure 3-5(b) and (c) are top-view and cross-sectional view images of the memristor/CMOS chip, respectively, showing the memristor array integrated on top of the chip surface. Figure 3-5(d) shows a photo of the integrated chip after packaging, along with the testing setup. Figure 3-6 shows a cross-section schematic of the integrated chip, showing connections of

the WO_x memristor array with the CMOS circuitry through extension lines and internal CMOS wiring.

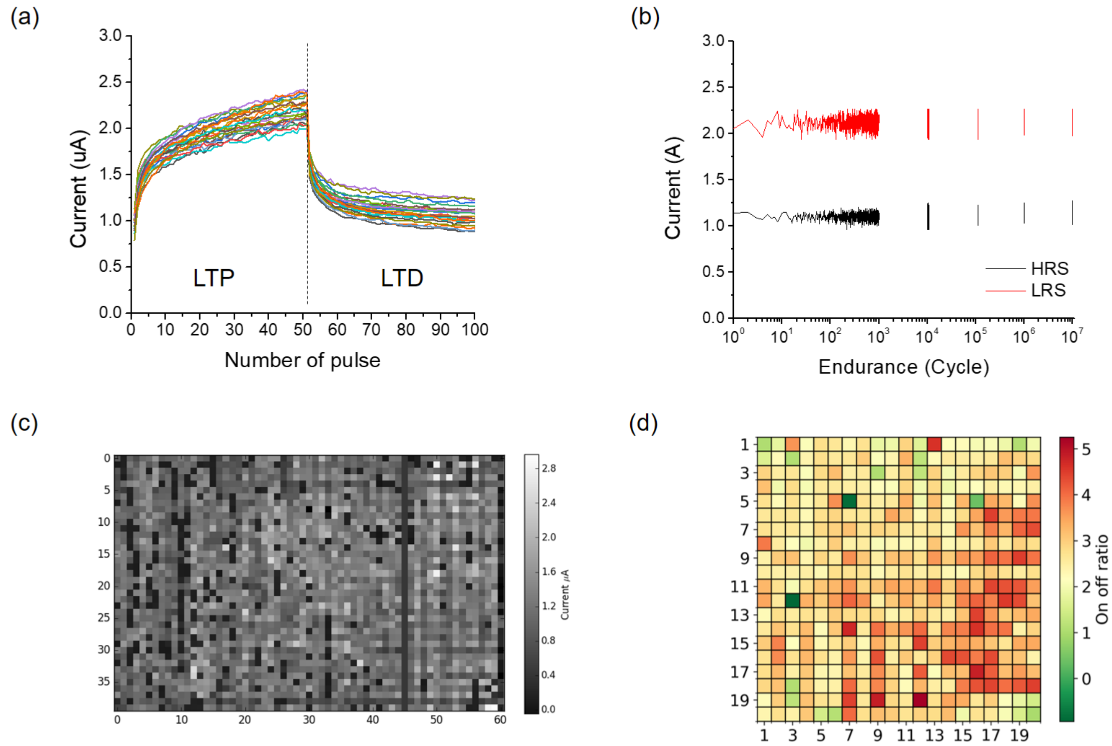


Figure 3-7: Measurement results of the memristor devices in the array.
(a) Programming memristor array on chip. Weight update curves from 22 devices measured from the crossbar array in the integrated chip are plotted. During the measurement, 50 consecutive programming pulses are applied (the LTP test), followed by 50 erase pulses (the LTD test). The current is measured at 0.6V after each pulse, using the on-chip processor and the integrated DAC/ADC circuitry. (b) Endurance test, showing HRS and LRS currents during the write/erase processes. After each order of the endurance cycle, the device was programmed and erased another 1000 times, with write/erase pulses of +1.8V and -1.8V, 200µs, respectively. (c) Distribution map of initial current level for a 40×60 subarray of the memristor crossbar. (d) Distribution map for a 20×20 subarray of the memristor crossbar. The color represents the on-off ratio measured after 50 consecutive programming pulses.

The memristor devices can be successfully programmed by controlling the number of applied write/erase pulses and read out by the integrated interface circuitry and controller with 4.5% of the device to device variability, even without any current-limiting transistors or external current compliance, as shown in Figure 3-7(a). The devices were programmed with 50 write pulse

at 1.8 V and 50 erase pulses at -1.8 V with 82 μs pulse width, using the on-chip processor and the integrated DAC/ADC circuitry.

The WO_x memristor device can be reliably programmed over 10^7 times (Figure 3-7(b)). Although this level of endurance can support certain online training algorithms, longer endurance may be desirable. Additionally, the cycle to cycle variation during the 10^7 programming cycles is ~ 3.4-4.2%. Figure 3-7(c) and (d) show the initial current level and the on/off ratio distributions of devices in the integrated array, respectively. The relative uniform distribution of devices within the integrated memristor array allows the system to implement a number of computing tasks on-chip. Note that although the small models we implemented in the current study can tolerate these cycle-to-cycle and device-to-device variations, device nonlinearity and variability need to be reduced to implement larger networks. Further device optimizations and using a monolithic integration technique that reduces the line resistance can lead to a better array and network performance.

3.4 Task 1: Training and classification using a single-layer perceptron

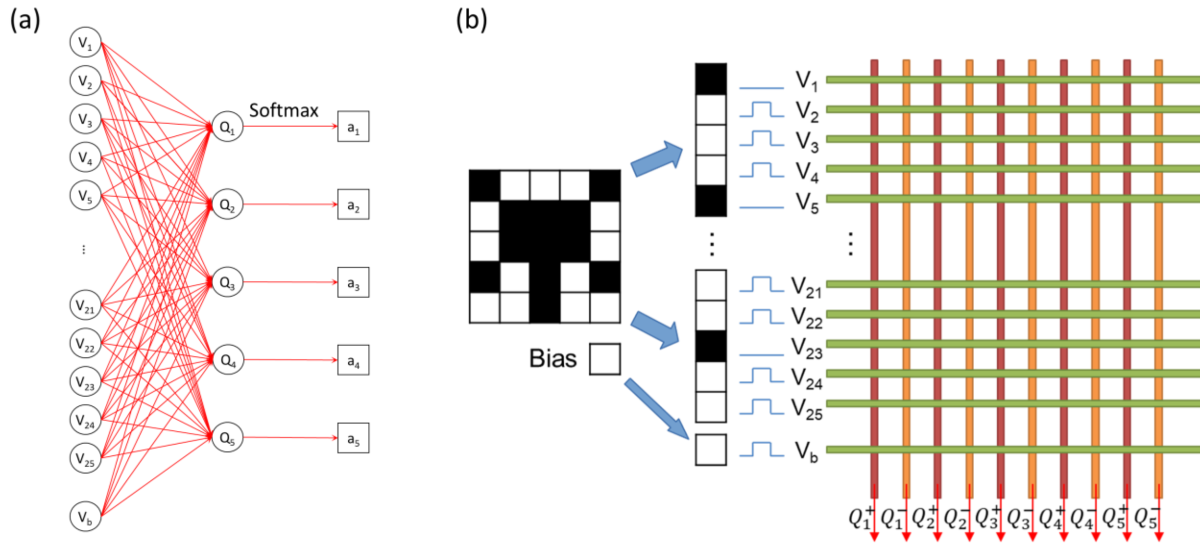


Figure 3-8: Schematic of the single-layer perceptron.
(a) Schematic of the single-layer perceptron for classification of 5 × 5 images. b, Implementation of the SLP using a 26 × 10 memristor subarray through the integrated chip. Input data (for example, the Greek letter 'Ω') are converted to voltage pulses of *V*read or 0 through the on-chip circuitry, depending on the pixel value.

A single-layer perceptron (SLP) network was first implemented to verify the operation of the integrated chip. 5×5 binary Greek letter patterns were used in the SLP training and testing. The SLP has 26 inputs (corresponding to the 25 pixels in the image and a bias term) and 5 outputs, with the input and output neurons fully connected with 26×5 = 130 synaptic weights, where the neuron with the highest output is identified as the winner by Softmax calculation and used to classify the corresponding class, as schematically shown in Figure 3-8(a).

In our implementation, the original binary input patterns are converted into input voltage pulses (V_{read} or 0) through the integrated processor and DAC circuitry and are fed to the rows of the memristor array. Specifically, when a white pixel is present, a pulse is applied to the corresponding row; while black pixels correspond to no pulse. The bias term is fixed at a constant value of 1 (treated as a white pixel) and is applied as an extra input. All the input pulses have the

same duration and amplitude in this test, as illustrated in Figure 3-8(b). Each synaptic weight w_{ij} is implemented with two memristor devices representing a positive and a negative weight, G_{ij}^{+} and G_{ij}^{-}, respectively, using the positive memristor conductance values:

$$Q_j = \sum_i w_{ij} x_i = V \sum_i G_{ij} t_i = V \sum_i (G_{ij}^{+} - G_{ij}^{-}) t_i = Q_j^{+} - Q_j^{-} \qquad (3-1)$$

where x_i is the input at row i and represented by a voltage pulse with amplitude V and width t_i. The charges are measured at the output columns and digitized by the ADCs, then converted to the neuron output y_j through the Softmax function:

$$y_j(Q_j) = \frac{\exp(\beta Q_j)}{\sum_k \exp(\beta Q_k)} \qquad (3-2)$$

where β is a scaling factor of the ADC output and k represents the output neuron index.

The integrated chip allows us to perform online learning. Specifically, the synaptic weights are updated during the training state using the batch gradient descent rule:

$$\Delta w_{ij} = \eta \sum_{n=1}^{N} \left(t_j^{(n)} - y_j^{(n)}\right) x^{(n)} \qquad (3-3)$$

where $x^{(n)}$ is the nth training sample of the input dataset, $y^{(n)}$ is the network output and $t^{(n)}$ is the corresponding label. η is the learning rate. The update value Δw_{ij} for the ith element in the jth class is then implemented in the memristors by applying programming pulses through the write DACs with a pulse width proportional to the desired weight change (quantized within the range of 0~63 timesteps, *i.e.* corresponding to 6-bit precision).

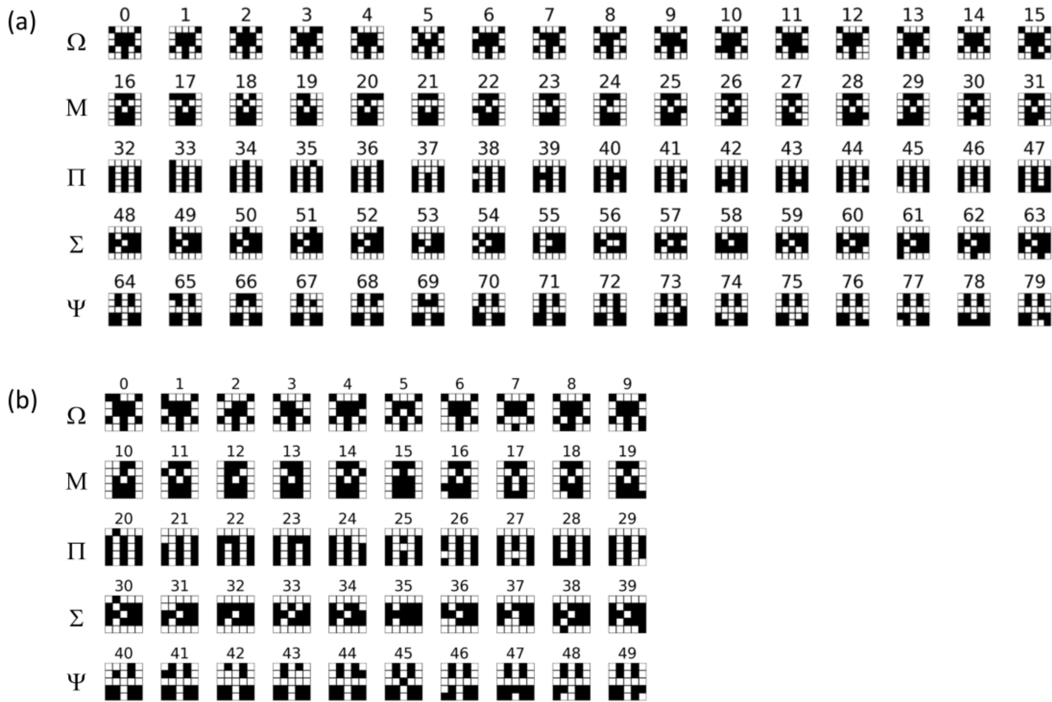


Figure 3-9: Training and test data set.
(a) Noisy training data set for the SLP. The training data set for each class includes the original image and 15 out of the 25 noisy images created by flipping 1 pixel in the original image. (b) Noisy testing data set for the SLP. The testing data set includes the 10 noisy images not in the training set, created by flipping 1 pixel in the original image for each class.

The SLP is mapped on the integrated chip using a 26×10 subarray. We trained and tested the SLP with noisy 5×5 Greek Letter patterns, for 5 distinct classes: 'Ω', 'M', 'Π', 'Σ', 'Φ'. To create data set, we flip one of the 25 pixels of the original image and generate 25 noisy images for each Greek letter. Together with the original image, they form a set of 26 images for each letter. We randomly select 16 images from the set for each class for training (Figure 3-9(a)) and use the other 10 images for testing (Figure 3-9(b)). This approach guarantees that the training set and the testing set do not overlap, and therefore improves the robustness of our testing results since the noisy test images are not used to train the network.

Training and testing results from the experimentally implemented SLP are shown in Figure 3-10(a-c). After 5 online training epochs, the SLP can already achieve 100% classification accuracy for both the training and testing sets. The average activation of the correct neuron during training is also clearly separated from the others, and the difference in neuron outputs between the winning neuron and the other neurons improves during training, as shown in Figure 3-10(a), verifying online learning has been reliably implemented in the experimental setup.

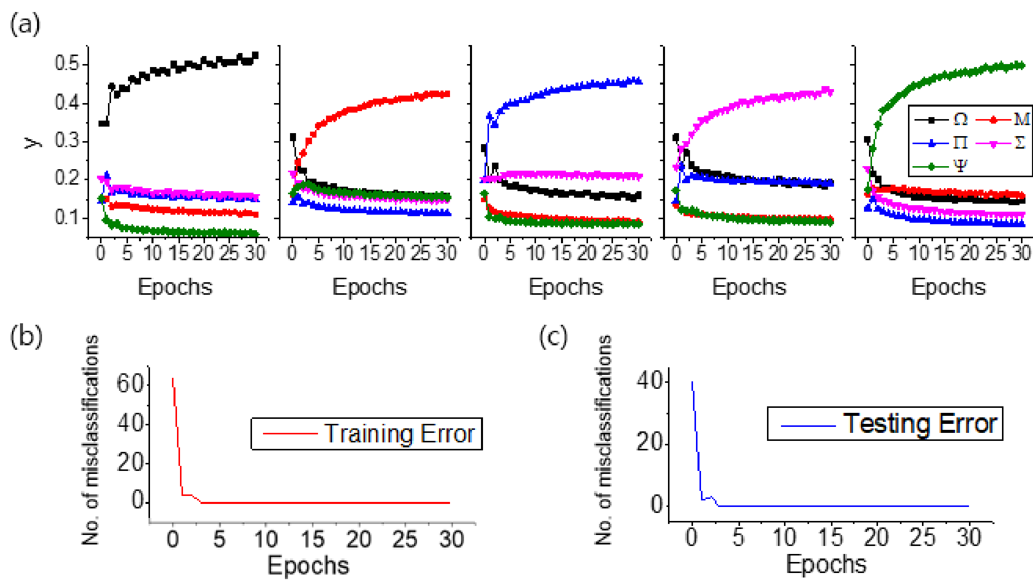


Figure 3-10: Classification results with SLP implementation.
(a) Evolution of the output neuron signals during training. Each data point represents the average output value for a specific neuron during a training epoch for a given input class. (b) Classification error of the training and (c) testing data set as a function of training epochs. The network can achieve 100% classification for both the training set and the testing set after five training epochs.

3.5 Task 2: Sparse coding algorithm implementation

The same hardware system was then used to implement a sparse coding algorithm. Sparse coding aims at representing the original data with the activity of a small set of neurons and can be traced to models of data representation in the visual cortex[12,13]. Sparse coding is an efficient

method for feature extraction and information compression and allows pattern recognition and classification to be performed in the compressed domain[14].

Following our previous work implemented at the board level[15], we mapped the *Locally Competitive Algorithm (LCA)* [16] on our integrated memristor/CMOS chip. In this approach, the membrane potential of an output neuron is determined by the input, a leakage term, and an inhibition term whose strength is proportional to the similarity of the neurons' features, *i.e*. an active neuron will try to inhibit neurons with similar features with itself. It can be shown mathematically that the lateral neuron inhibition can be achieved in the memristor crossbar by removing the reconstructed signal from the input to the network.

3.5.1 Locally competitive algorithm (LCA)

The locally competitive algorithm (LCA) is a sparse coding algorithm that uses a dictionary of feature vectors to encode an input signal with a small number of output coefficients while minimizing the reconstruction error.

The concept of sparse coding is as follows: Given an input signal, *x*, and a dictionary of features, *D*, sparse coding aims to represent *x* as a linear combination of features from *D* using a set of sparse coefficients *a*, with a minimum number of features. Mathematically, the objective of sparse coding can be summarized as minimizing an energy function containing both the reconstruction error term as well as a sparsity penalty term, defined as:

$$\min_{a}\left(|x - Da^{T}|_{2} + \lambda |a|_{0} \right) \qquad (3-4)$$

where $|\cdot|_2$ and $|\cdot|_0$ are the L^2- and the L^0-norm, respectively, and λ is a sparsity parameter that determines the relative weights of the reconstruction error (1st term) and the sparsity penalty (the number of neurons used, 2nd term).

The mathematical form of the LCA can be expressed as follows: x is an m-element ($m \times 1$) input vector, D is an $m \times n$ matrix, where each column of D represents an m-element feature vector (*i.e.* a dictionary element), a is an n-element ($1 \times n$) row vector representing the neuron activity coefficients, where the i_{th} element of a corresponds to the activity of the i_{th} neuron. After feeding input x to the network and allowing the network to stabilize through lateral inhibition, a reconstruction of x can be obtained as Da^T, *i.e.* linear combination of the neuron activities and the corresponding neurons' feature vectors. In a sparse representation, only a few elements in a are nonzero while the other neurons' activities are suppressed to be precisely zero.

The neuron dynamics during LCA analysis can be summarized by the following equation[16]:

$$\frac{du}{dt} = \frac{1}{\tau}\left(-u + x^T D - a(D^T D - I_n)\right) \quad (3-5a)$$

$$a_i = \begin{cases} u_i \text{ if } u_i > \lambda \\ 0 \text{ otherwise} \end{cases} \quad (3-5b)$$

where u_i is the membrane potential of neuron i, τ is a time constant, and I_n is the $n \times n$ identity matrix.

Implementing the inhibition effect $D^T D$ can be very computation intensive. To implement the algorithm in memristor hardware, the original equation (3-5a) can be re-written as

$$\frac{du}{dt} = \frac{1}{\tau}(-u + (x - \hat{x})^T D + a) \quad (3-6)$$

where $\hat{x} = Da^T$ is the reconstructed signal. Equation (3-6) shows that the inhibition term between neurons can be interpreted as a neuron removing its feature from the input when it becomes active, thus suppressing the activity of other neurons with similar features. The matrix-matrix operation $D^T D$ in equation (3-5a) is thus reduced to two sequential matrix-vector operations, one used to calculate $\hat{x} = Da^T$ and the other used to calculate the neuron activity from the updated input $r^T D$,

where $r = x - \hat{x}$ is the residue term. This approach allows us to implement LCA in memristor crossbars without physical inhibitory synaptic connections between the neurons.

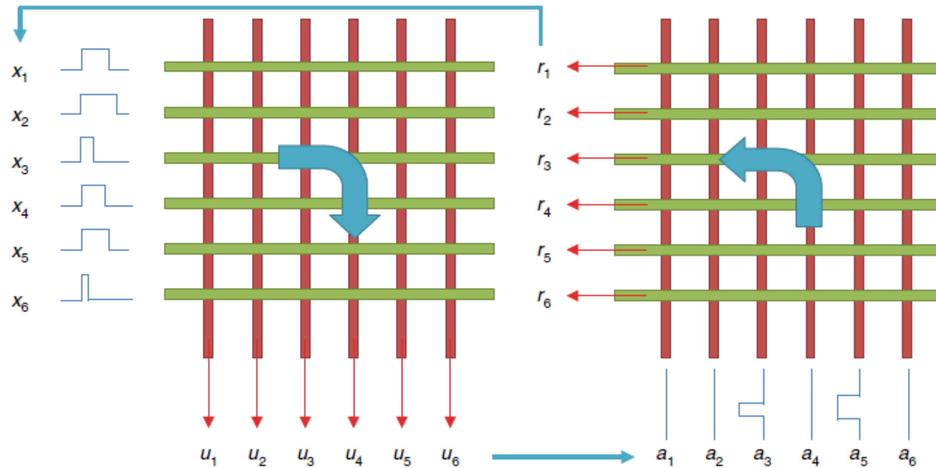


Figure 3-11: Schematic of the LCA algorithm.
Experimental implementation of the LCA algorithm using on-chip memristor arrays. In each iteration a forward pass is performed to update the membrane potential u of the output neurons based on the inputs x, followed by a backward pass to update the residual r based on the neuron activities a. The residual r becomes the input for the next iteration.

The LCA algorithm can be implemented in an iterative process through two VMM operations; in a forward direction to obtain the neuron activations, and in backward direction to obtain the reconstructed input. The residue term is then obtained by removing the reconstructed input from the original input and is then fed to the network, and the process is repeated until the network stabilizes. Figure 3-11 illustrates the iterative forward and backward processes employed in the LCA implementation.

3.5.2 Implementation of Sparse coding on chip

The bi-directional operation of the memristor array in the integrated memristor/CMOS chip allowed us to experimentally implement the sparse coding algorithm on-chip. Similar to the SLP

case, we use the crossbar array to perform VMM operations, here in both forward and backward directions. Since the chip offers full flexibility to implement different algorithms by re-programming the integrated processor, the LCA algorithm was implemented in the same chip used in the SLP study, through simple software changes.

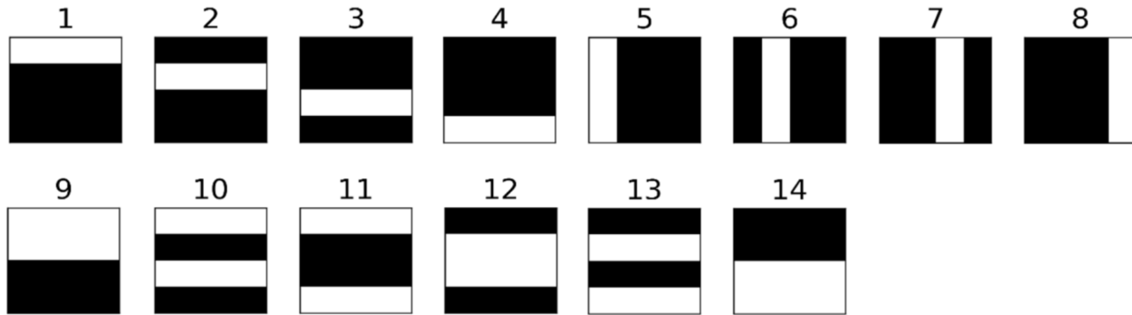


Figure 3-12: Dictionary elements based on horizontal and vertical bars.

4×4 inputs were used to test the experimental implementation of the LCA algorithm. By using linear combinations of horizontal and vertical bar patterns, the input dimension is reduced to 7. To satisfy the over-completeness requirement of the LCA algorithm, a dictionary containing 14 features of horizontal and vertical bar patterns are used, as shown in Figure 3-12. This setup produces a 2× over-complete dictionary[16] that enables the network to find a sparse, optimal solution out of several possible solutions.

The LCA algorithm was mapped to a 16×14 subarray in the memristor/CMOS chip, using the corresponding interface circuitry and the processor that provides the neuron functions. An example of the LCA network operation is shown in Figure 3-13. The experimentally implemented network correctly reconstructs the input image while minimizing the number of activated neurons. For example, it identifies the optimized solution with two neurons 6 and 13, instead of using three neurons 2,4 and 6, as shown in Figure 3-13(b) and (c). The dynamics of the LCA network operation

can also be correctly captured, as shown in Figure 3-13(a), where the effects of lateral neuron inhibition that lead to a more sparse solution than the initial solution can be clearly observed.

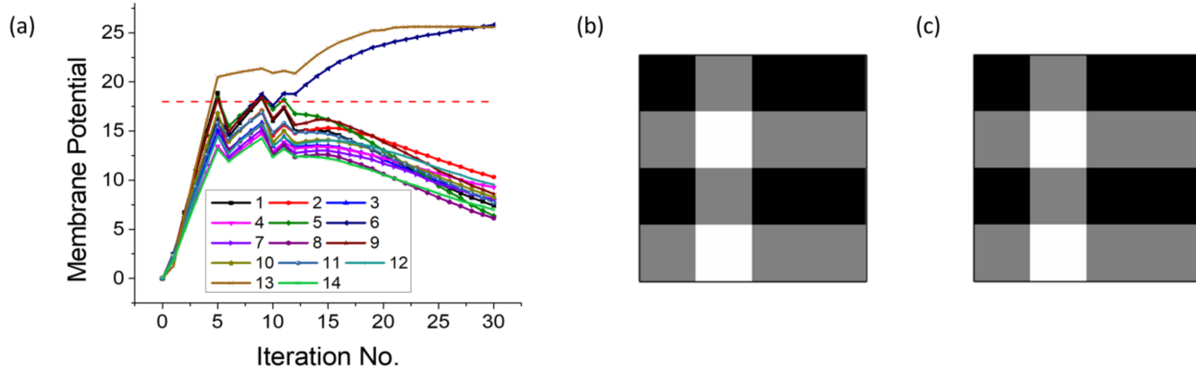


Figure 3-13: Sparse coding results.
(a) Experimentally obtained neuron membrane potentials as a function of iteration number during LCA analysis. The horizontal red dashed line marks the threshold parameter $\lambda = 18$. (b) Original test image. (c) Experimentally reconstructed image based on the neuron activities

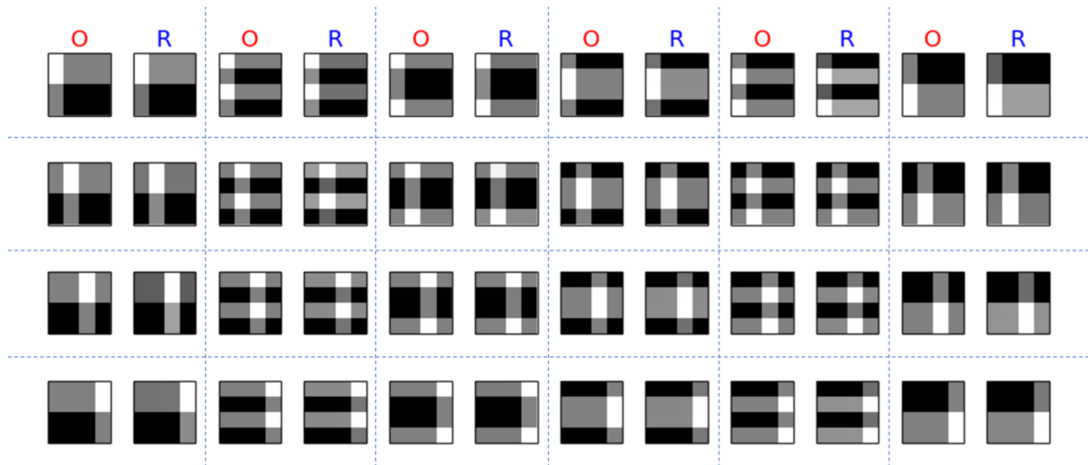


Figure 3-14: Additional examples.
Original input images (O) and reconstructed images (R). The same threshold $\lambda = 18$ is used in all experiments.

To verify the system's performance for other input patterns, an exhaustive test of all 24 possible patterns consisting of two horizontal bars and 1 vertical bar was performed using the on-chip memristor network, resulting in 100% success rate (Figure 3-14) measured by the network's ability to correctly identify the sparse solutions.

3.6 Task 3: A multi-layer neural network: PCA + Classification

We demonstrate a bilayer neural network using two subarrays in the same memristor crossbar, implementing unsupervised and supervised online learning to achieve feature extraction and classification functions, respectively. The bilayer network is used to analyze and classify data obtained from breast cancer screening based on principal component analysis (PCA). In this work, the first layer performs PCA of the original data, which reduces the 9-dimensional raw input data to a 2-dimensional space based on the learned principal components (PCs). The second layer is a 3×1 SLP layer (with differential weights and a bias term) which performs classification using the reduced data in the 2-dimensional space for the two classes, as schematically shown in Figure 3-15.

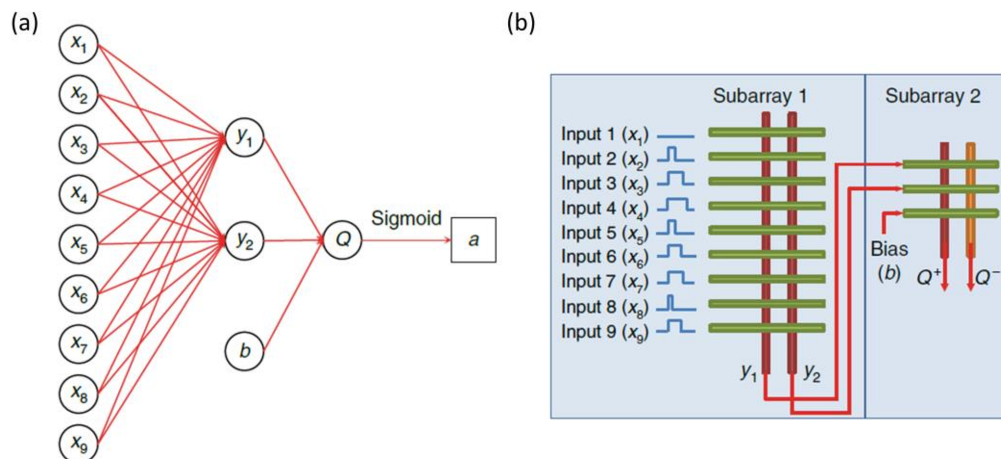


Figure 3-15: Schematic of the bilayer neural network.
(a) Schematic of the bilayer neural network for PCA analysis and classification. (b) The bilayer network is mapped onto the integrated memristor chip, using a 9 × 2 subarray for the PCA layer and a 3 × 2 subarray for the classification layer.

3.6.1 Principal component analysis (PCA)

PCA reduces data dimensionality by projecting data onto lower dimensions along with principal components (PCs), with the goal of finding the best summary of the data using a limited number of PCs[17]. The conventional approach to PCA is to solve the eigenvectors of the covariance matrix of the input data, which can be computationally expensive in hardware. A more hardware-friendly approach is to find the PCs through unsupervised, online learning.

Specifically, following our previous study[18], Sanger's rule, also known as the generalized Hebbian algorithm, is implemented in the integrated chip to obtain the PCs. The desired weight change for the *j^th^* principal component is determined by:

$$\delta g_{ij} = \eta y_j \left(x_i - \sum_{k=1}^{j} g_{ij} y_j \right) \qquad (3-7)$$

3.6.1.1 Mapping memristor conductance to synaptic weight in PCA

In the experiment, the weights of the 1st and 2nd PCs, g_{ij}, are mapped onto the memristor conductances through a linear transformation[18]. The network is trained online, using a subset of the original database consisting of 100 data points. During the training process, the 9-dimensional breast cancer data[19] is converted into input voltage pulses with pulse widths proportional to the data values, within the range of 0~63 time units. The output charge collected at column *j* then corresponds to the dot-product of the input vector and the conductance vector stored in column *j*, projecting data from the original 9-dimensional space to a 2-dimensional output space (when only two principal components are used). During training, the weights are then updated following equation (3-7), using programming voltage pulses generated through the write DACs with pulse widths proportional to δg_{ij}.

A linear conversion is used to map the memristor conductance in the array to the synaptic weights g_{ij} with range ([-1 1]) used in PCA, by using the relationship:

$$g = \frac{ADC - a}{b} \quad (3-8)$$

where ADC is the unconverted ADC output from the circuit, which is converted to the current/conductance value through factors a (ADC shift factor, which is about 1900) and b (ADC scaling factor, which is about 1500) in equation (3-8). The conversion based on equation (3-8) maps the maximum average current to weight 1 and minimum average current to weight -1.

For each input data, the dot-product of the input data and the jth feature, y_j is directly obtained from the ADC output of the j_{th} column in the 9×2 weight matrix. The column's weights are then updated based on Sanger's rule (equation (3-7)). During training, the desired weight updates are linearly converted into write pulse widths and applied to the memristor devices, without using nonlinear compensation schemes[20].

3.6.1.2 Dataset for PCA: Breast cancer dataset

A standard breast cancer dataset from the University of Wisconsin Hospital[19] was used as the input for the PCA network, which is available through the University of Californian, Irvine Machine Learning Repository[21].

The dataset consists of breast cell mass properties measured in 9 categories and each property is scored from 1 to 10. Each input to the memristor network is thus a nine-dimensional vector consisting of scores from the nine measurements. The bilayer network was trained using 100 training data (containing 50 benign and 50 malignant cases) from the dataset and tested with 500 data (containing 312 benign and 188 malignant cases), not in the training set.

3.6.1.3 Implementation of 1st layer for PCA on chip

Initially, the weights of the 1st and 2nd components are randomized in the memristor array (Figure 3-16(a)). Projection of the input along these vectors leads to severe overlapping of the benign and malignant cases in the 2-dimensional space, as shown in Figures 3-16(b) and (c). After 30 training epochs (an epoch is defined as a training cycle through the 100 training data), the PCs are correctly learned (Figure 3-16(d)), and the 2-dimensional projected data can be clearly separated into two clusters, as shown in Figure 3-16(e) and (f). Note the ground truth (benign or malignant) is not used in the PCA training or clustering process. They are included in the plots (represented as blue and red colors in Figure 3-16(e) and (f)) only to highlight the effectiveness of the clustering before and after learning the PCs.

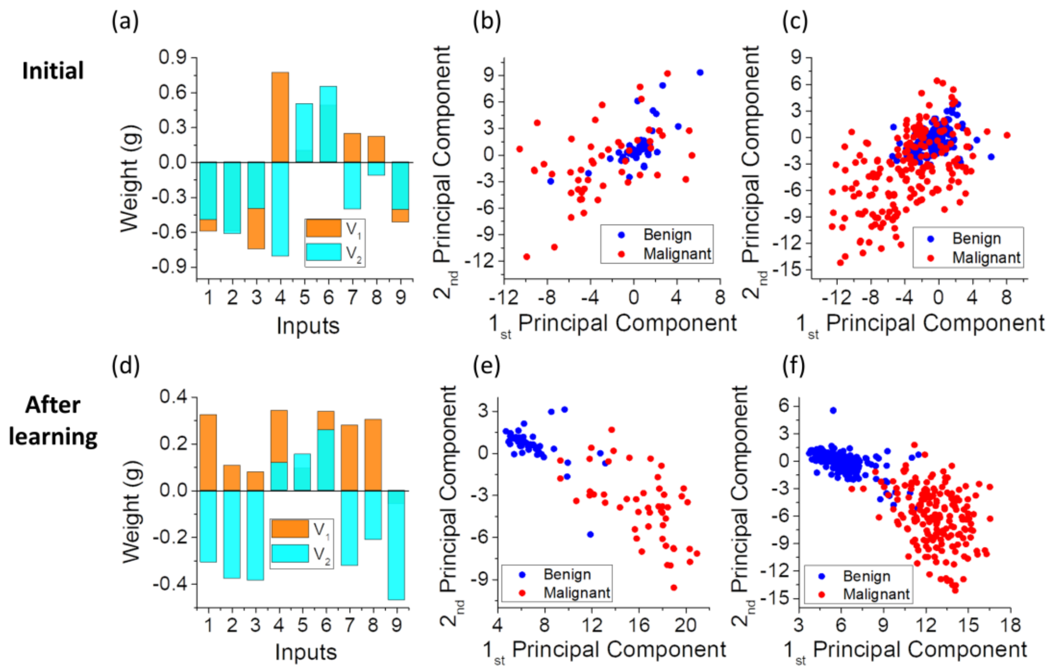


Figure 3-16: The results of 1st layer (PCA) implementation.
(a) Initial weights for the two PCs in the network. (b), (c) Before training, linear separation is not possible in the projected 2D space, for both training (b) and testing (c) data. (d) Weights for the two PCs after unsupervised online training obtained from the memristor network, using Sanger's learning rule and 30 training cycles. (e), (f) Clear separation can be observed in the 2D space for both training (e) and testing (f) data after projection along the trained PCs.

3.6.2 2nd layer for classification (SLP)

The PCA layer separates the original data into clusters but does not classify them. To achieve classification, we implemented a second layer, an SLP, in the same hardware system. The SLP processes output from the PCA layer and generates a label (benign or malignant). Since there are only two classes to distinguish, the SLP is trained online using logistic regression. A 3×2 subarray is used in the second layer, to account for the 2 inputs, the bias term, and the differential weights, as schematically shown in Figure 3-15.

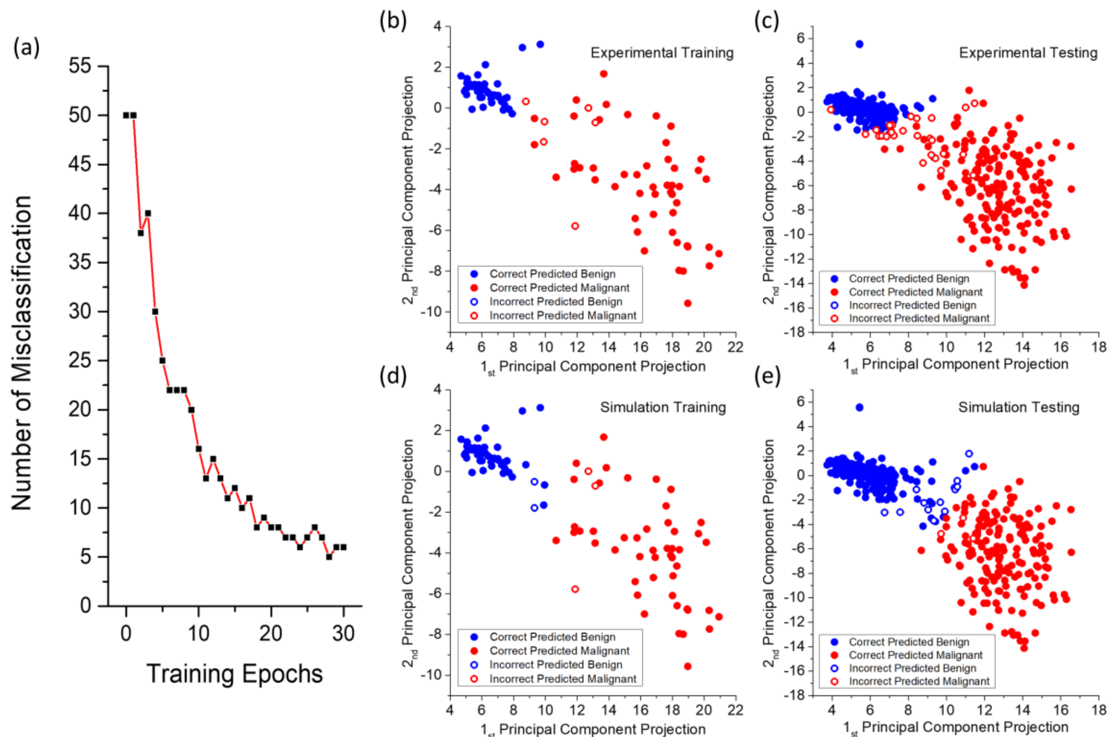


Figure 3-17: Classification results in the bilayer network.
(a) Evolution of the classification error during the online training process, from the experimentally implemented bilayer network on chip. (b), (c) Classification results experimentally obtained from the memristor chip, for the training (b) and testing (c) data. Blue and red colors represent the predicted benign and malignant data, respectively. The incorrectly classified results are marked as open circles. Classification rates of 94% and 94.6% are obtained for the training and testing data, respectively. (d), (e) Classification results of the bilayer network implemented in software. Blue and red colors represent the predicted benign and malignant data, respectively. Incorrectly classified results are marked as open circles. Classification rates of 95% and 96.8% are obtained for the training (d) and testing (e) data in software, respectively.

A supervised learning algorithm, logistic regression, is used to train the SLP layer of the bilayer network to classify benign and malignant cells. Logistic regression is commonly used for the classification of two classes. Suppose an N training sample dataset $\mathbf{x} = \left(x^{(1)}, \ldots, x^{(N)}\right)^T$ with label $\mathrm{t} = (t^{(1)}, \ldots, t^{(N)})^T$, where the nth training sample can be written as $x^{(n)}$ and the nth label $t^{(n)} \in \{0,1\}$. A cross-entropy energy function can be defined as:

$$E(\mathrm{w}) = -\sum_{n=1}^{N}\left\{t^{(n)} \ln y^{(n)} + \left(1 - t^{(n)}\right) \ln\left(1 - y^{(n)}\right)\right\} \qquad (3-9)$$

where $y^{(n)} = \sigma(z^{(n)})$ and $z^{(n)} = w^T x^{(n)}$

$\sigma(z)$ is the logistic sigmoid function defined by:

$$\sigma(z) = \frac{1}{1 + \exp(-z)} \qquad (3-10)$$

The likelihood of a training data $x^{(n)}$ belonging to class $t^{(n)} = 1$ is determined by the sigmoid function output $y^{(n)}$, with larger $y^{(n)}$ meaning $x^{(n)}$ more likely belong to the class. Taking the gradient of the error function leads to:

$$\nabla E(w) = \sum_{n=1}^{N}\left(y^{(n)} - t^{(n)}\right)x^{(n)} \qquad (3-11)$$

To minimize the energy function, the network is trained using the batch gradient descent defined as[22]:

$$w = w - \eta \sum_{n=1}^{N}\left(y^{(n)} - t^{(n)}\right)x^{(n)} \qquad (3-12)$$

After learning the PCs in the PCA layer, the original 9-dimensional data are fed through the PCA layer, and the clustered 2-dimensional data are used as inputs for the SLP layer. The same

100 training data used for the PCA layer training are used for the SLP layer training (Figure 3-17(a)), in a supervised fashion using the ground. Training is completed after 30 epochs.

Afterward, the 500 test data not included in the training set are applied to the network, passing first through the PCA layer then as 2-dimensional data into the 2nd SLP layer. After online training of the PCA and the SLP layers, the experimentally implemented 2-layer network can achieve 94% and 94.6% classification accuracy during training and testing (Figure 3-17(b) and (c)). The values are slightly lower than the ones obtained from software implementation (95% during training and 96.8% during testing, Figure 3-17(d) and (e)), due to the nonideality in the memristor weight update that results in a decision boundary that differs from that obtained from software (which assumes ideal linear weight updates) after the online training process.

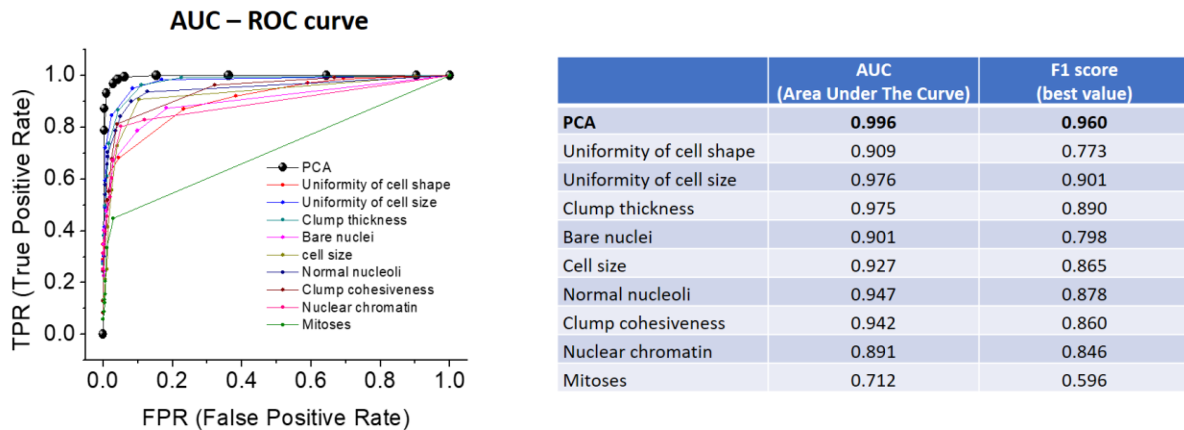
3.6.3 Evaluation of the PCA + classifier network

Beyond classification accuracy, the following statistical parameters can be used to further evaluate the performance of the network:

- Condition positive (P): the number of real positive cases in the data
- Condition negative (N): the number of real negative cases in the data
- True positive (TP): Sick people correctly identified as sick
- False positive (FP): Healthy people incorrectly identified as sick
- True negative (TN): Healthy people correctly identified as healthy
- False negative (FN): Sick people incorrectly identified as healthy
- Sensitivity (the true positive rate (TPR) or recall): The proportion of actual positives that are correctly identified as such (e.g., the percentage of sick people who are correctly identified as having the condition). i.e. TP/(TP+FN)
- Specificity (true negative rate (TNR)): The proportion of actual negatives that are correctly identified as such (e.g., the percentage of healthy people who are correctly identified as not having the condition). i.e. TN/(TN+FP)

- Precision (Positive Predictive Value (PPV)): measure of the classifier exactness. i.e. TP/(TP+FP)

The receiver operating characteristic (ROC) curve is a graphical plot that illustrates the diagnostic ability of a binary classifier system as its discrimination threshold is varied. The ROC curve is created by plotting the true positive rate (TPR) against the false positive rate (FPR, which equals (1 – specificity)) at various threshold settings. The area under the curve (AUC) represents the degree or measure of separability. It shows how much the model is capable of distinguishing between classes. The higher the AUC, the better the model is at predicting 0s as 0s and 1s as 1s. By analogy, the higher the AUC, the better the model is at distinguishing between patients with the disease and no disease. Therefore, the AUC - ROC curve is an important performance measure for a classification problem at various threshold settings.



	AUC (Area Under The Curve)	F1 score (best value)
PCA	**0.996**	**0.960**
Uniformity of cell shape	0.909	0.773
Uniformity of cell size	0.976	0.901
Clump thickness	0.975	0.890
Bare nuclei	0.901	0.798
Cell size	0.927	0.865
Normal nucleoli	0.947	0.878
Clump cohesiveness	0.942	0.860
Nuclear chromatin	0.891	0.846
Mitoses	0.712	0.596

Figure 3-18: AUC-ROC curve and F_1 score of the breast cancer task.
The experimentally PCA + classifier network show excellent AUC value of 0.996 and high F_1 score of 0.960, corresponding to sensitivity, specificity and accuracy values of 93.1%, 99.0% and 94.6%, respectively.

The F_1 score is a measure of a test's accuracy. It considers both the precision (PPV) and the recall (TPR) of the test to compute the score: PPV is the number of correct positive results

divided by the number of all positive results returned by the classifier, and TPR is the number of correct positive results divided by the number of all relevant samples (all samples that should have been identified as positive). The F_1 score is the harmonic average of the precision and recall, where an F_1 score reaches its best value at 1 (perfect precision and recall) and worst at 0.

The sensitivity, specificity, and accuracy for our experimentally implemented PCA + classifier network were calculated to be 93.1%, 99.0%, and 94.6%, respectively. From the AUC-ROC curve shown in Figure 3-18, the AUC of classification using the learned principal components is 0.996, suggesting the network is almost perfect at distinguishing between the positive class and the negative class. The F_1 score of the network is 0.960, proving the network has excellent precision and sensitivity for breast cancer evaluation.

3.7 Performance & area analysis of the integrated memristor/CMOS chip

3.7.1 Power estimate

The integrated chip suggests different computing tasks can be efficiently mapped on the memristor-based computing platform, by taking advantage of the bidirectional VMM operations in the memristor crossbars and the flexibility in the CMOS interface and control circuitry. In our prototype, the supporting analog interfaces, as well as digital control and the OpenRISC processor are implemented in 180 nm CMOS technology.

Both the digital processor power and the mixed-signal interface power are directly measured experimentally, by measuring the root mean square (RMS) current with a Fluke meter while running the chip. At the maximum frequency of 148 MHz, the digital power reads 235.3 mW and the total analog power reads around 64.4 mW. This corresponds to the energy consumption of 6.53 nJ/inner product or 1.12 pJ/op for the mixed-signal interface, where an

operation is defined as the multiplication and accumulate (MAC) process of a 4-bit input with a stored analog weight in the memristor array. The crossbar array power is obtained from the average device current at the read voltage, which yields ~7 mW for the 54×108 array. The total power at 148 MHz clock is thus 306.7 mW for the current chip based on 180 nm CMOS technology.

The energy efficiency is estimated using the 148 MHz clock speed and an average 4-bit input during inference, which gives around 9.87M VMM operations per second. Multiplying this number by 54×108 leads to 5.75×10^{10} operations per second. Therefore, the energy efficiency can be derived by dividing the number of ops/second with the total power, which results in 187.62 GOPS/W for the current memristor/CMOS chip.

The custom circuitry was designed in 180nm CMOS and features a generic digital processor along with a full set of mixed-signal analog to digital converters (ADC) and digital to analog converters (DAC). Two different approaches were used to estimate the power dissipation at the 40 nm technology node. The digital power was estimated using generalized scaling[23] and the mixed-signal power was estimated using a figure of merit (FOM) approach.

In digital circuits, the length scaling factor S, and the supply voltage scaling factor U are different during scaling. Specifically, from 180 nm to 40 nm, S = 180 nm/40 nm = 4.5, U = 1.8 V/1.0 V = 1.8. As a result, the digital power is reduced by a factor of $1/U^2 = 0.32$, while the circuit speed is improved by a factor of S = 4.5. Note the faster processor allows the same process to control more channels, so normalizing to the same number of 162 channels, the digital power at 40 nm is estimated to be

$$P_{40nm} = \frac{P_{180nm}}{U^2 S} \quad (3-13)$$

Using the measured digital power at 180nm, the estimated digital power at 40 nm is then 235.3 mW/($(1.8)^2\times4.5$), which is about 16.1 mW.

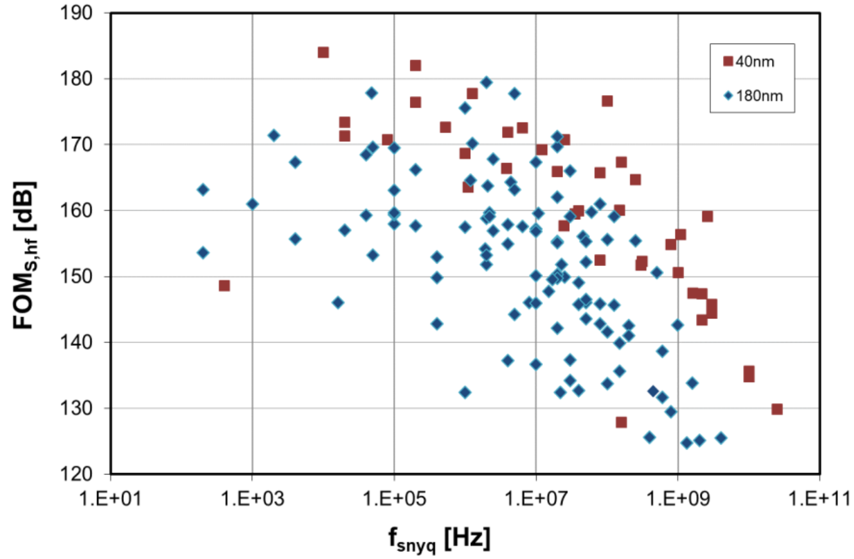


Figure 3-19: Schreier FOM for 180nm and 40nm ADCs.
The Schreier FOM is used for comparing high-resolution ADC performance across architectures. The data are collected from all ADCs published in ISSCC and VLSI conferences from 1997-2018.

The analog to digital converter performance is evaluated using a figure of merit (FOM) approach. An ADC FOM combines several converter performance metrics into one number for comparison across ADC architectures. The Schreier FOM is typically used for high-resolution converters and is given by the following equation:

$$FOM_s = SNDR + 10\log\left(\frac{BW}{P}\right) \qquad (3-14)$$

where SNDR is the signal to noise ratio and distortion, BW is one-half of the sampling frequency, and P is the power dissipation. To estimate the power scaling from 180nm to 40nm, the ADC Performance Survey[24] was used which aggregates all ADCs published in the ISSCC and VLSI circuits conferences from 1997 - 2018. A subset of data for 180 nm and 40 nm ADCs is shown in Figure 3-19.

The mean FOM between sampling frequencies from 100 kHz to 10 MHz for each technology was determined and compared. The mean FOM for 40 nm was determined at 172 dB

and a conservative estimate of 165 dB was used for power estimation. Using an estimated 165 dB, which corresponds to 176 μW per ADC, we can estimate the new total analog power at 40 nm to be 19 mW, leading to a total system power of 42.1 mW, assuming the 54×108 crossbar power remains at 7 mW. Therefore, by simply scaling the system to 40nm technology node, the estimated OPS/W at 148 MHz is 1.37 TOPS/W.

3.7.2 Area efficiency

As the first prototype, we intentionally made the circuit design flexible to serve as an evaluation platform, and we were limited to a relatively old CMOS technology (due to cost and tool compatibility limitations in a university cleanroom). As a result, the area efficiency is not ideal. The total chip area is 6.7 mm × 9.2 mm = 61.64 mm^2. The area of the memristor array is 475 μm × 292 μm ~ 0.14 mm^2. Therefore, the memristor array's area efficiency (the percentage of the array over the entire chip) is about 0.23%.

A large reason for the low area efficiency can be traced to the conservative design choice. To make the chip flexible, we designed 2 write DACs (for positive and negative writes), 1 read DAC and a 13bit ADC at each row and each column, so input data can be applied at every row and the output can be collected at every column and processed in parallel. The reverse operation is also supported for algorithms such as the LCA. The size of the three DACs and the ADC is around 74 μm × 1800 μm ~ 0.1332 mm^2 for each such reconfigurable channel. The total area of these components for the 162 channels (54 rows and 108 columns) occupies 21.57 mm^2, which is around 35% of the entire chip area. The rest of the chip areas includes the 64 kB SRAM (32 kB data memory and 32 kB ping pong memory) and the OpenRISC core, which are 3.79 mm^2 and 10.04 mm^2, respectively.

3.7.3 Optimization of Power and Area

Further optimizations of the system design, e.g. by replacing the generic processor with a custom-designed controller or Field-Programmable Gate Array (FPGA), and by replacing the fast and high-precision 13-bit ADC with simpler interface circuits and more optimized ADC designs, along with memristor device optimizations that reduce power consumption in the memristor crossbar, can further improve the system's performance and power efficiency.

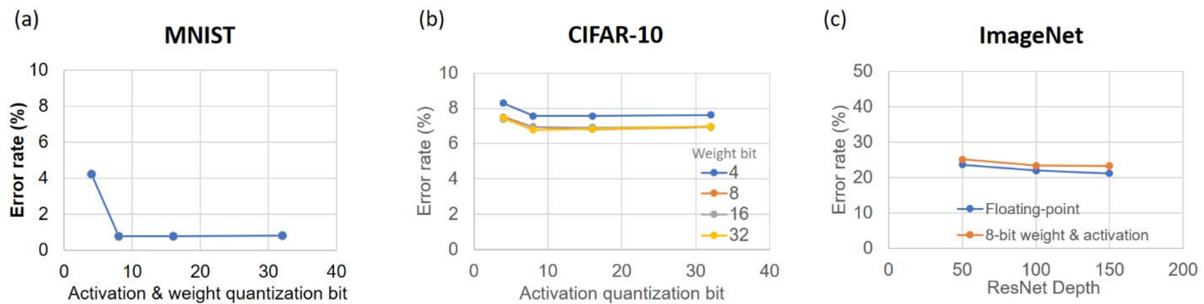


Figure 3-20: Error rates from different models as a function of weight and activation quantization effects.
(a) Error rates obtained using quantized weight and activation, for 32-bit, 16-bit, 8-bit, 4-bit fixed-point for the MNIST data set [25]. (b) Error rates as a function of activation quantization effect, for weights quantized at 32-bit, 16-bit, 8-bit, 4-bit, for CIFAR-10 [26]. (c) Floating-point vs 8-bit quantized network error rates for various network depths of the ResNet model, tested on ImageNet [27].

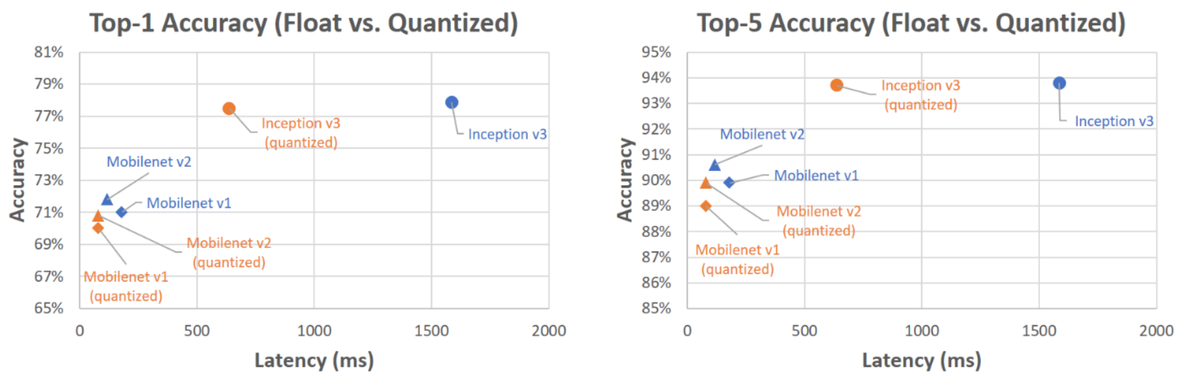


Figure 3-21: Quantization effect for common CNN models.
The models are implemented using 8-bit weight and 8-bit activation. All latency numbers are measured on Pixel 2 cell phones using a single core, adapted from tensorflow.org (https://www.tensorflow.org/lite/performance/model_optimization)

For many applications, we can replace the 13-bit ADC used in this evaluation chip with 8-bit ADCs without loss of accuracy. For example, Figures 3-20 and 3-21 show simulation results comparing floating-point activations (corresponding to the ADC outputs) vs. 8-bit activations for commonly used algorithms, as well as weight quantization effects for some models[25-27]. Reducing the required ADC precision to 8-bit will significantly decrease the ADC area, as shown in Figure 3-22(b). The chip area will also be naturally reduced with more advanced technology nodes[28,29]. For instance, state-of-the-art 40 nm 8-bit ADCs occupy ~1650 μm^2, which will reduce the ADC area by a factor of 20 compared with the current chip[29]. The DAC area is relatively much smaller. For example, the total DAC area for a 128 ×128 memristor crossbar array would be 173 μm^2 with the 40 nm technology node.

The ADC area can be further reduced through optimization of the array interface structure. Instead of using a dedicated ADC at each crossbar column and row, multiple columns can share a single ADC by sequentially sampling one column at a time, e.g. via time-multiplexing, as shown in Figure 3-23. This approach is feasible since ADCs can operate at speeds of several GHz while the read operation through the memristor takes a longer time (e.g. 10 ns). Using this shared ADC approach, the physical ADC area can be significantly reduced. Note here we are trading the system speed for the peripheral circuitry area. However, due to the in-memory operations and high level of parallelism, the memristor-based systems do not need to operate at very high frequency to achieve the desired outputs. For example, our previous analysis has shown that even at a system speed of 10 Mhz, the very high energy efficiency of 60.1 TOPS/W can still be achieved[30].

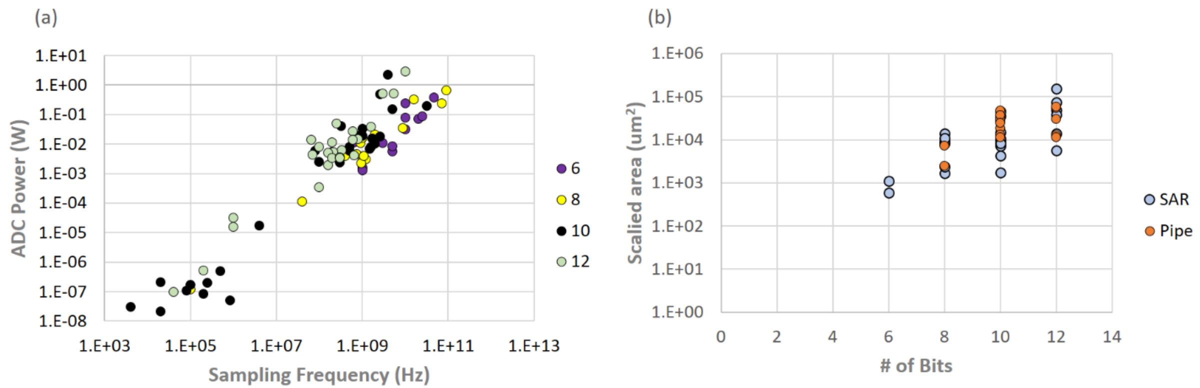


Figure 3-22: ADC design choices.
(a) Speed and power tradeoffs, and (b) resolution and area tradeoffs for reported ADC designs. The data are collected from publications at key technology conferences, and scaled to the 40nm technology node.

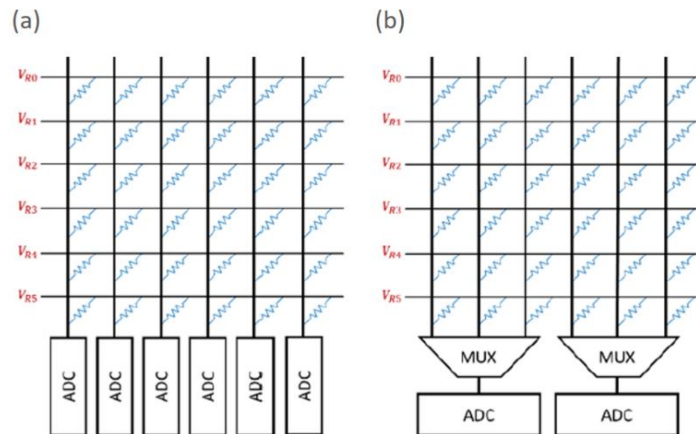


Figure 3-23: Shared ADC options.
Schematics showing dedicated ADC (a) and shared ADC (b) designs.

Another potentially promising approach for area efficiency is to exploit a concept we termed "array ADCs", where the core component of the ADC is shared among different columns but each column maintains a dedicated latch to accumulate data locally so the column outputs can still be processed in parallel. This approach can potentially reduce the overall ADC area without having to reduce the system clock. Similar approaches have been used in column-parallel CMOS sensor arrays[31-34] or multi-channel biomedical and physical detectors[35-41]. These ADCs are

designed for parallel multi-node sampling applications, which is an excellent match for the tasks discussed here, although circuit optimizations have not been extensively studied since these prior applications do not require high-speed operations. Optimizing the ADC design using the array ADC concept will be a key direction in our future studies.

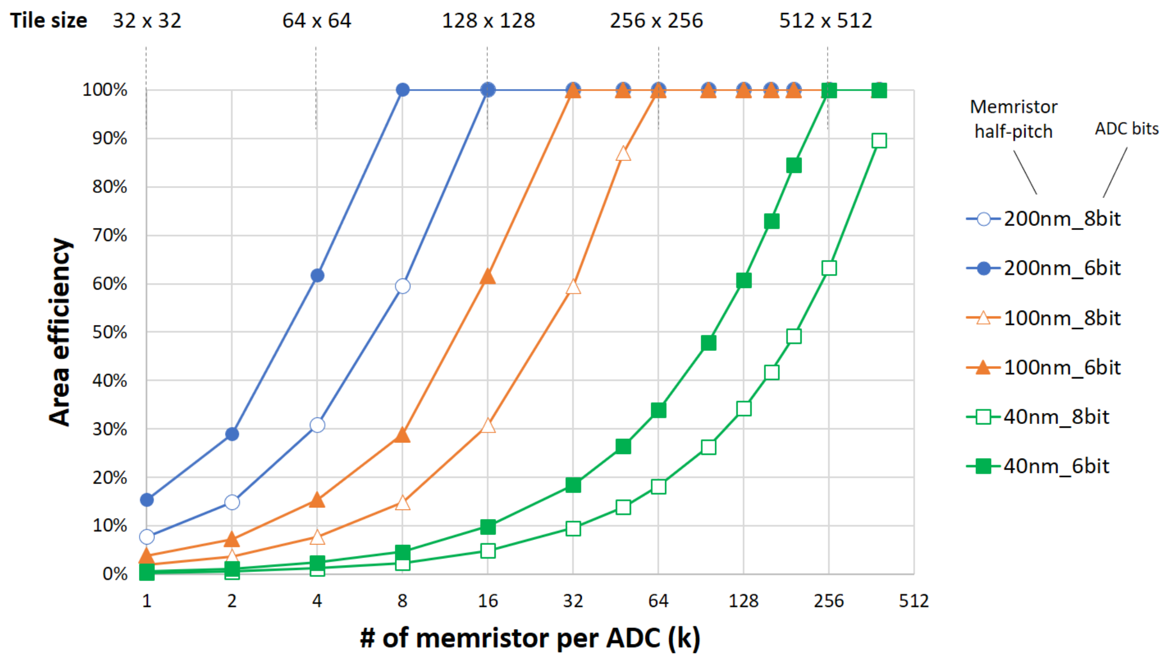


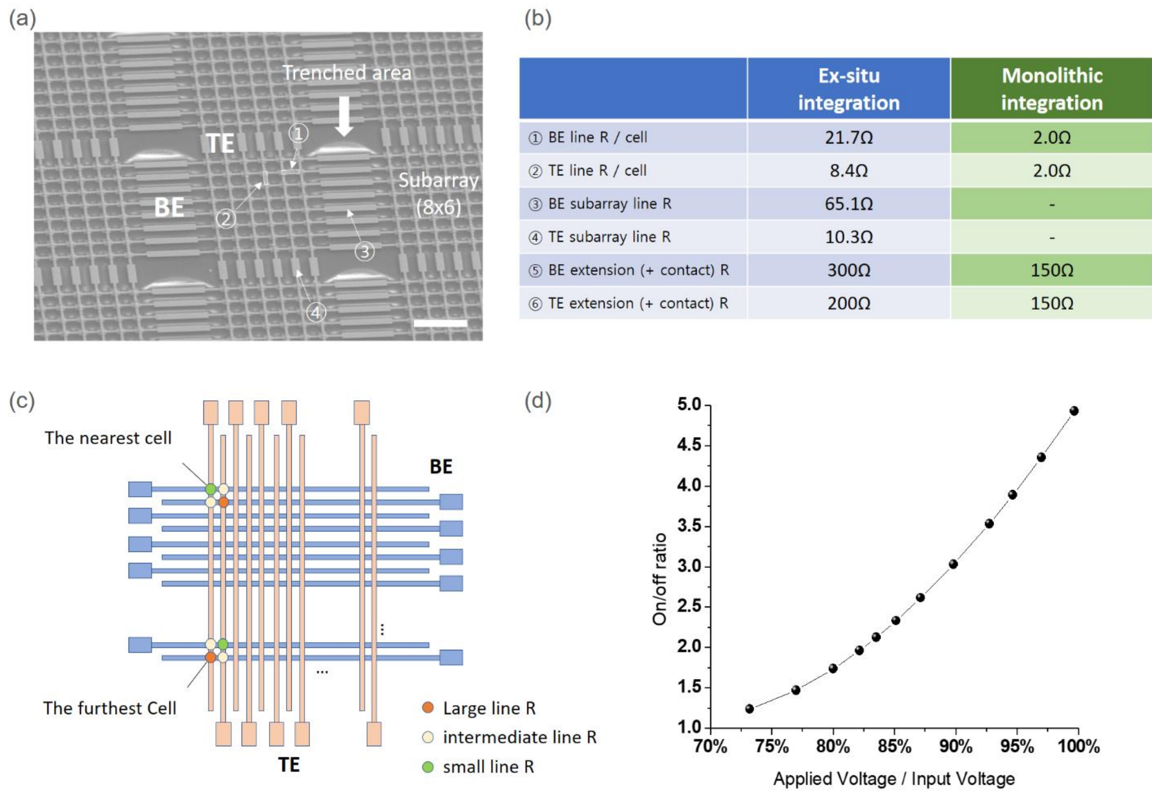
Figure 3-24: ADC area efficiency.
Expected area efficiency for different memristor half-pitch sizes. The ADC is based on conventional 40nm SAR ADC design, for 6 bit [28] and 8 bit ADCs [29].

With technology scaling, proper ADC resolution selection, and optimized design through shared ADC or array ADC approaches, we expect the area efficiency of the integrated chip to be significantly improved, and can potentially reach 100%, as shown in Figure 3-24.

3.8 Non-ideal effect of memristor/CMOS chip and future implementations

3.8.1 Device variability on integrated chip

Although this prototype achieved most of the goals we originally set, the limitation of having to perform the integration after all CMOS processes are completed, including the final passivation steps, leads to significant challenges and non-ideal device behaviors. For example, Figure 3-7(d) shows the on/off ratio distribution of devices in the integrated array after a weight update operation. The relative uniform distribution of devices within the integrated memristor array allows the system to implement a number of computing tasks on-chip. However, device variations can still be clearly observed. This effect is exacerbated by the line resistance effects, as shown in Figure 3-25.



	Ex-situ integration	Monolithic integration
① BE line R / cell	21.7Ω	2.0Ω
② TE line R / cell	8.4Ω	2.0Ω
③ BE subarray line R	65.1Ω	-
④ TE subarray line R	10.3Ω	-
⑤ BE extension (+ contact) R	300Ω	150Ω
⑥ TE extension (+ contact) R	200Ω	150Ω

Figure 3-25: Line resistance effects.
(a) Zoomed-in SEM image of the memristor array integrated on top of the CMOS chip. The 54×108 array used in this study is composed of 126 8×6 subarrays to work around the periodic deep trenched areas (~2000Å deep) created from the design rule. (b) Measured BE and TE line resistance values from the integrated array (labeled as ex-situ integration). The line resistances can be significantly reduced if the memristor array can be integrated at the local or intermediate interconnect levels (labeled as monolithic integration). (c) Schematic of the wiring patterns. The total line resistance seen by the device can change significantly even among neighboring devices due to the even/odd arrangement of the pad patterns. (d) Simulated results using the memristor model, showing the measured on/off ratio can be strongly affected by the voltage loss due to the line resistance. The on/off is calculated after 50 consecutive programming pulses.

As discussed earlier, a major challenge we encountered for the memristor/CMOS integration is that we do not have whole wafers to work with. The CMOS chip we obtained is from a Multi-project Wafer (MPW) process due to budget constraints. In these cases, all back-end-of-line (BEOL) processes are completed at the foundry, including the final passivation (with a very thick polymer passivation layer) and pad-open processes. The chips are then diced and sent to us.

This process leads to very rough surfaces for us to integrate memristor arrays with. An ideal process will be to integrate the memristor arrays at the local or intermediate interconnect level (e.g. metal 4 level) before the rest of the BEOL processes. However, this will require stopping the whole wafer at this intermediate step which is not possible with MPW processes.

To enable successful integration on these very rough and small-size chips, we have performed extensive optimizations in our fabrication process. During the CMOS chip design, we intentionally designed a flat area that is around 1mm × 1mm in the center of the chip. However, even this area contains periodic trenches (~2000 Å deep) to comply with the design rules. Therefore, to avoid having active devices in the trench region we divide the 54x108 memristor array into 8x6 small subarrays. The small subarrays are then connected with thicker and wider wires to form the larger array, as shown in Figure 3-25(a). This approach allowed us to successfully fabricate the memristor arrays and achieve a high fabrication yield. However, the additional wiring increases the metal line resistance. In addition, to maintain device yield and device uniformity, the bottom electrode cannot be made very thick which also leads to larger line resistance.

The line resistance affects the larger array operation. Specifically, although this effect is relatively small during the VMM operation due to the high memristor resistance (~ 120-580 kΩ) at 0.6 V used for the inference stage, the device exhibits non-linear I-V characteristics such that the device resistance at the write voltage (1.8 V) is much lower (~ 1.7-8.2 kΩ). As a result, programming devices in the larger array becomes challenging due to the line resistance. More importantly, non-uniform programming occurs as a result of the differences in the BE and TE wire length seen by different devices.

To analyze the effect of the series line resistance on the array operation, we performed detailed SPICE (Simulation Program with Integrated Circuit Emphasis) simulations using the

measured line resistance values and a dynamic memristor model (Equation (3-15) and (3-16)) to estimate the voltage loss effect.

$$I = (1 - w)\alpha[1 - \exp(-\beta V)] + w\gamma \sinh(\delta V) \qquad (3-15)$$

$$\frac{dw}{dt} = \lambda \sinh(\eta V) - \frac{w}{\tau} \qquad (3-16)$$

where Equation (3-15) is the I-V equation which includes a Schottky term (the 1st term) corresponding to conduction in the V_O-poor region and a tunneling-like term (the 2nd term) corresponding to conduction in the V_O-rich region[42].

The two conduction channels are in parallel and the internal state variable *w* represents the relative contribution from the two channels. Equation (3-16) is the dynamics equation which describes the change rate of the state variable *w* with respect to the applied voltage, including the drift effect under an applied electric field (the 1st term) and the spontaneous diffusion (the 2nd term). α, β, γ, δ, λ, η are all positive-valued parameters determined by material properties. τ is the diffusion time constant. In our array-level simulation, a series resistance *R* is also added to the device model. The parameters used in the SPICE simulation are listed below.

Parameter	Value	Parameter	Value
α	9e-7	β	4
γ	2.8e-7	δ	6
λ	0.045	η	6
τ	10	w_{max}	1
w_{min}	0	R	400Ω

Table 3-1: The parameters used in the SPICE simulation

Our SPICE simulations show that in the worst case (i.e. the target cell is furthest from the voltage supply), the voltage loss due to the line resistance is around 0.33 V, corresponding to an

18.5% drop of the write voltage. The reduction of the applied voltage leads to significantly reduced device response, as shown in Figure 3-25(d).

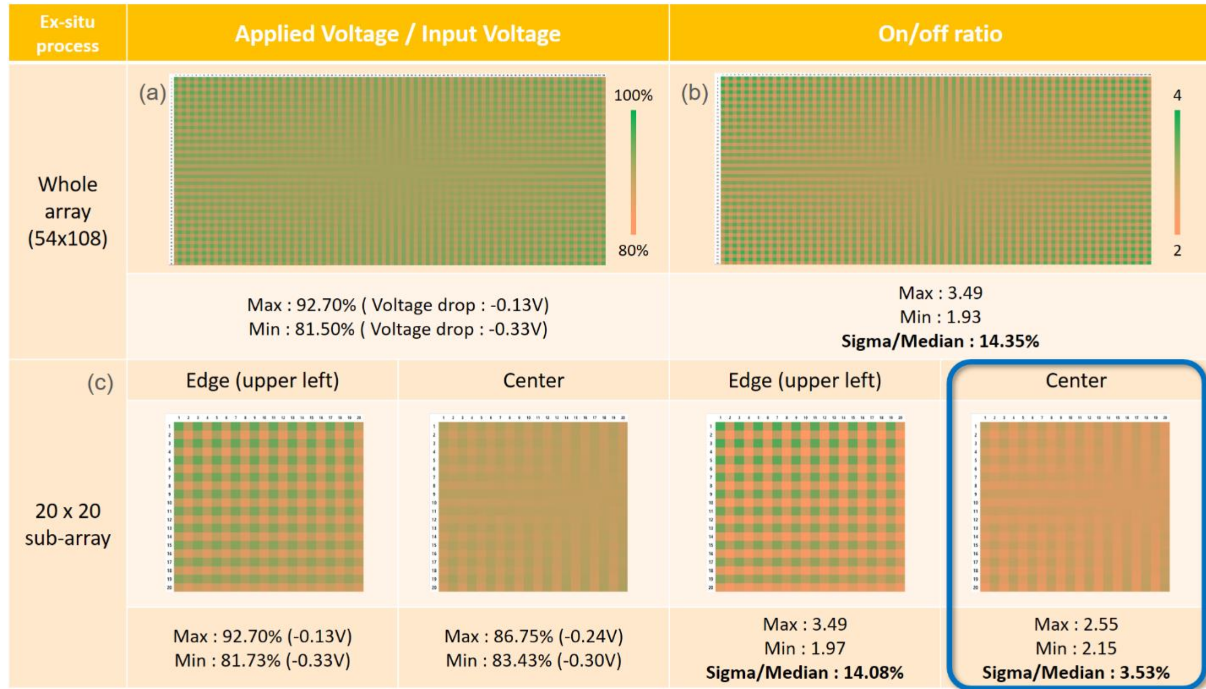


Figure 3-26: Voltage loss effects in the memristor array.
(a) The effect of voltage loss in the integrated 54×108 array, and (b) the resulting on/off ratio affected by the voltage loss effect, obtained from SPICE simulations. The voltage loss and on/off ratio values are represented by the color labels shown on the right. (c) The voltage loss effect at different regions in the array. The large line resistance effect, combined with the even/odd electrode design, leads to large variations among neighboring cells for cells near the edge. Better uniformity can be obtained for cells in the center of the array, with a reduced on/off ratio.

Analysis of the integrated array shows large device variations can be observed when trying to program the device, due to the different line resistances the devices see, as shown in Figure 3-26. Near the edges, the effect is significant due to the even/odd wiring patterns, as seen in Figure 3-25(b) and 3-26(c). To achieve a more uniform device response, we chose the center area of the array in our studies, where the voltage drop due to line resistance is roughly similar (Figure 3-26(c)) that allows us to reliably program the devices using the open-loop method.

Future studies that can allow more direct integration at the local or intermediate interconnect levels (which we termed monolithic integration) will effectively address this issue. For example, our detailed SPICE simulations show that monolithic integration will greatly reduce the line resistance and minimize measured device variability, and allow larger arrays to be successfully operated, as shown in Figure 3-27.

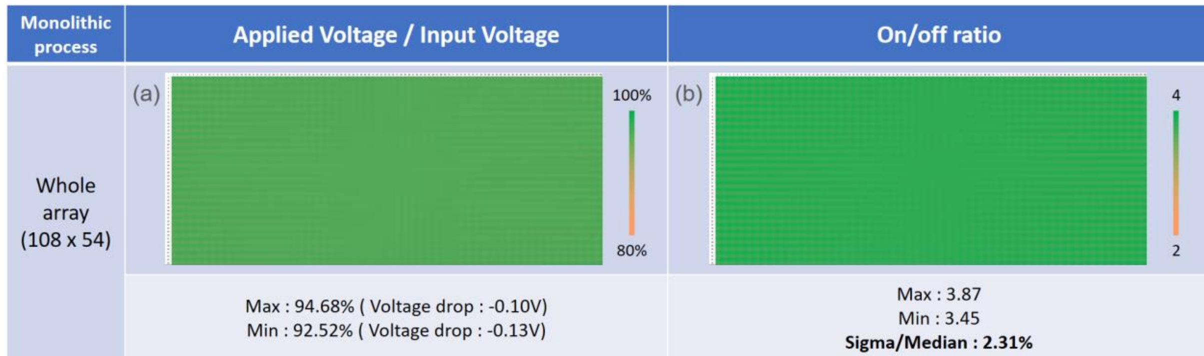


Figure 3-27: Improvements with monolithic integration.
SPICE simulations showing (a) the effect of voltage loss and (b) the expected on/off ratio from a 54×108 array integrated at the local interconnect level, showing much reduced line resistance effect and improved device uniformity.

3.8.2 Expected other issues for complex model and large datasets

During batch training, the memory needed to store the batch information may become a bottleneck as the task becomes complex and the pattern size increases. Batch gradient descent has also been associated with overfitting due to the low stochasticity of the process. For complicated tasks and large datasets, mini-batch training[43,44] may be an attractive option by splitting the training dataset into small batches, with typical batch sizes between 2 to 32. Additionally, recently proposed hybrid techniques that utilize CMOS capacitors to perform the lower-significance weight updates could also provide a realistic solution to the batch information storage challenge[11].

To achieve high accuracy with more complex and larger datasets, device properties should also be improved. The WO_x memristor device can be reliably programmed over 10^7 times. Although this level of endurance can support certain online training algorithms, longer endurance may be desirable. The small models we implemented in the current study can tolerate these cycle-to-cycle and device-to-device variations, however, device nonlinearity and variability need to be reduced to implement larger networks[45]. To this end, future device optimizations that can improve device uniformity and weight update linearity[4,46], along with architecture innovations, such as hybrid non-volatile memory (NVM)-CMOS neural-network implementations[11], mixed-precision[9], multi-memristive architectures[10], and other precision extension techniques[30] can be employed to address these requirements and allow larger models to be implemented in memristor-based systems.

3.8.3 Tiled architecture

To scale up the system for larger networks, rather than simply increasing the crossbar size, a promising approach may be to tile small crossbars together in a modular fashion[30,47,48], as schematically shown in Figures 3-28 and 3-29. In this approach, each tile is a self-contained, integrated memristor - CMOS unit (macro), which is then tiled together using digital interfaces to construct larger systems[49]. In Chapter 5, we will discuss it in more detail.

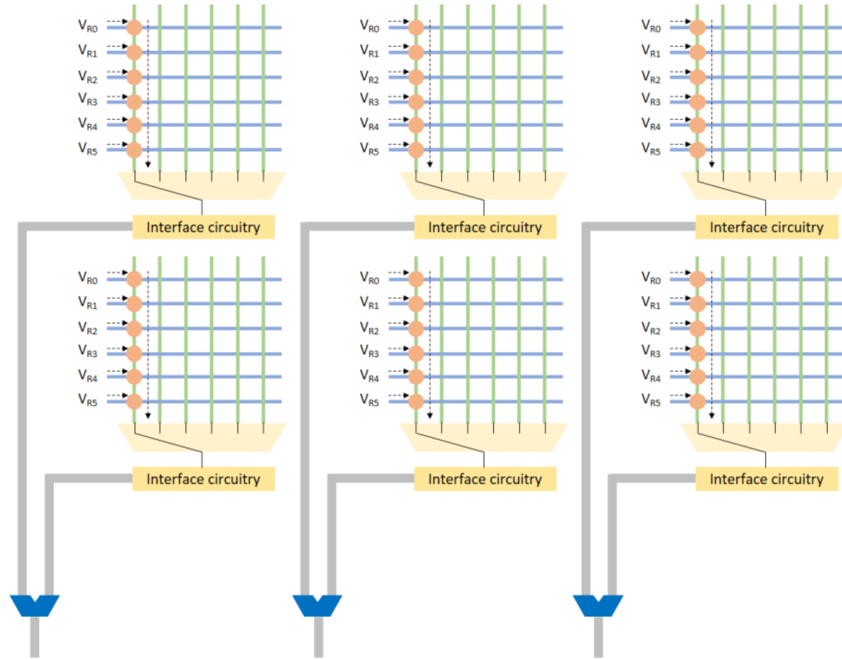


Figure 3-28: Tiled architecture based on small crossbar arrays.
Every tile consists of a memristor array (e.g. 128 × 128) integrated on top of CMOS circuitry forming a self-contained unit (macro). The tile-to-tile communication is performed in digital domain.

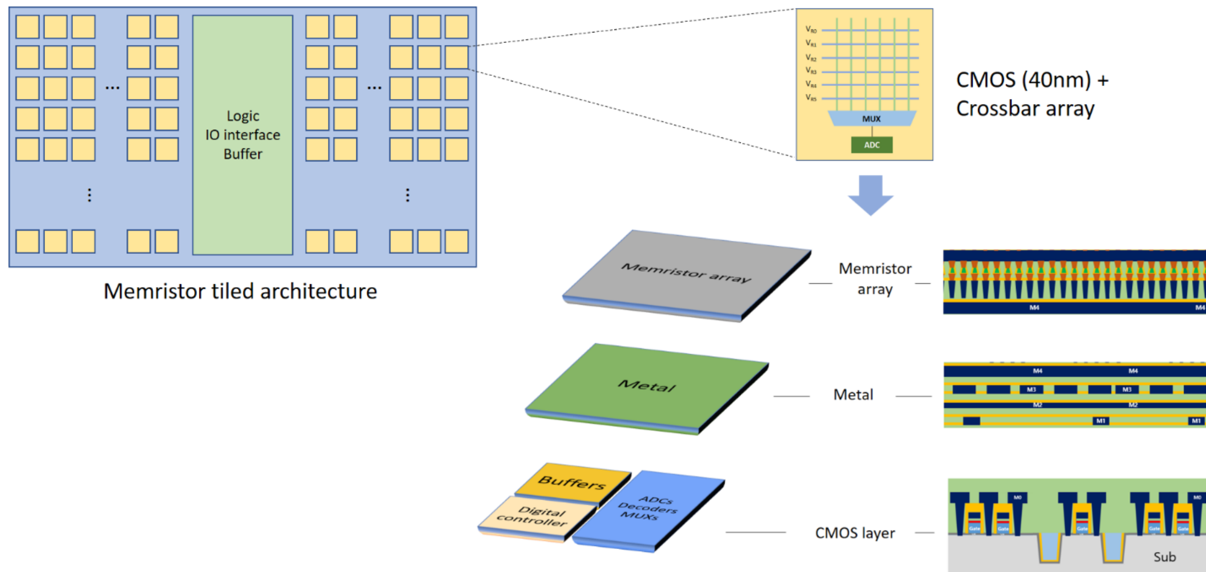


Figure 3-29: Tiled architecture with high area efficiency.
Each macro consists of a small memristor array (128 × 128) integrated with underlying CMOS circuitry (DACs, ADCs, MUXs, register buffer, etc.). Larger systems are built by tiling the macros through digital interfaces.

3.9 Conclusion

In this study, we successfully designed and fabricated a fully-functional, programmable neuromorphic computing chip with a passive memristor crossbar array integrated with a complete set of analog and digital components and an on-chip processor. The integrated chip allows the mapping of different neuromorphic and machine learning algorithms on-chip through simple software changes. Three different and commonly-used models, perceptron, sparse coding and principal component analysis with an integrated classification layer, were demonstrated. 100% classification accuracy was achieved for 5×5 noisy Greek Letters in the SLP implementation, the reliable sparse coding analysis was obtained from an exhaustive test set using 4×4 bar patterns, and 94.6% classification rate was experimentally obtained from the breast cancer screening dataset using the same integrated chip.

The integrated memristor – CMOS systems potentially offer efficient hardware solutions for different network sizes and applications[4,5,7,9,11,50-57]. An initial application of such systems may be edge computing such as those used in the Internet of Things (IoT) to process data near its source, allowing real-time data processing with high speed and low energy consumption[58,59]. We expect continued device, circuit and architecture innovations as discussed above, along with algorithm advances such as quantized neural networks[60,61] can allow the system to be scaled up for more complex and demanding tasks.

References

1. Prezioso, M. *et al.* Training and operation of an integrated neuromorphic network based on metal-oxide memristors. *Nature* **521,** 61–64 (2015).
2. Yao, P. *et al.* Face classification using electronic synapses. *Nat. Commun.* **8,** 15199 (2017).
3. Bayat, F. M. *et al.* Implementation of multilayer perceptron network with highly uniform passive memristive crossbar circuits. *Nat. Commun.* **9,** 2331 (2018).
4. Li, C. *et al.* Efficient and self-adaptive in-situ learning in multilayer memristor neural networks. *Nat. Commun.* **9,** 2385 (2018).
5. Li, C. *et al.* Analogue signal and image processing with large memristor crossbars. *Nat. Electron.* **1,** 52-59 (2017).
6. Gao, L., Chen, P.-Y. & Yu, S. Demonstration of Convolution Kernel Operation on Resistive Cross-Point Array. *IEEE Electron Device Lett.* **37,** 870–873 (2016).
7. Sheridan, P. M. *et al.* Sparse coding with memristor networks. *Nat. Nanotech.* **12,** 784–789 (2017).
8. Cai, F. Correll, J. M., Lee, S. H., Lim, Y., Bothra, V., Zhang, Z., Flynn. M. P. & Lu. W. D. A fully integrated reprogrammable memristor–CMOS system for efficient multiply–accumulate operations. *Nat. Electron.* **2**, 290–299 (2019).
9. Le Gallo, M. *et al.* Mixed-precision in-memory computing. *Nat. Electron.* **1**, 246–253 (2018).
10. Boybat, I. *et al.* Neuromorphic computing with multi-memristive synapses. *Nat. Commun.* **9**, 2514 (2018).
11. Ambrogio, S. *et al.* Equivalent-accuracy accelerated neural-network training using analogue memory. *Nature* **558**, 60–67 (2018).
12. Olshausen, B. A. & Field, D. J. Emergence of simple-cell receptive field properties by learning

a sparse code for natural images. *Nature* **381**, 607–609 (1996).

13. Olshausen, B. A. & Field, D. J. Sparse coding with an overcomplete basis set: A strategy employed by V1? *Vision Res.* **37**, 3311–3325 (1997).
14. Sheridan, P. M., Du, C. & Lu, W. D. Feature Extraction Using Memristor Networks. *IEEE Trans. Neural Networks Learn. Syst.* **27**, 2327–2336 (2016).
15. Sheridan, P. M. *et al.* Sparse coding with memristor networks. *Nat. Nanotech.* **12,** 784–789 (2017).
16. Rozell, C. J., Johnson, D. H., Baraniuk, R. G. & Olshausen, B. A. Sparse coding via thresholding and local competition in neural circuits. *Neural Comput.* **20**, 2526–63 (2008).
17. Lever, J., Krzywinski, M. & Altman, N. Points of Significance: Principal component analysis. *Nat. Methods* **14,** 641–642 (2017).
18. Choi, S., Shin, J. H., Lee, J., Sheridan, P. & Lu, W. D. Experimental Demonstration of Feature Extraction and Dimensionality Reduction Using Memristor Networks. *Nano Lett.* **17,** 3113–3118 (2017).
19. Mangasarian, O. L., Street, W. N. & Wolberg, W. H. Breast Cancer Diagnosis and Prognosis Via Linear Programming. *Oper. Res.* **43,** 570–577 (1995).
20. Choi, S., Sheridan, P. & Lu, W. D. Data Clustering using Memristor Networks. *Sci. Rep.* **5**, 10492 (2015).
21. Dheeru, D. & Karra Taniskidou, E. {UCI} Machine Learning Repository. (2017).
22. Bishop, C. M. *Pattern recognition and machine learning*. *Library* **4,** (Springer, 2006).
23. Taur, Y. & Ning, T. H. Fundamentals of Modern VLSI Devices. (Cambridge University Press, 2009).
24. Murmann, B. ADC performance survey 1997-2018.

http://web.stanford.edu/~murmann/adcsurvey.html (2018).

25. Hashemi, S., Anthony, N., Tann, H., Bahar, R. I. & Reda, S. Understanding the impact of precision quantization on the accuracy and energy of neural networks. *2017 Design, Automation & Test in Europe* 1474–1479 (2017).
26. Lin, D. D., Talathi, S. S. & Annapureddy, V. S., Fixed point quantization of deep convolutional networks. *arXiv preprint arXiv:1511.06393.* (2015).
27. Jacob, B., *et al.* Quantization and training of neural networks for efficient integer-arithmeticonly inference. *arXiv preprint arXiv:1712.05877.* (2017)
28. Choo, K. D., Bell, J. & Flynn, M. P., Area-efficient 1GS/s 6b SAR ADC with charge-injection-cell-based DAC, *Proc. IEEE Int. Solid-State Circuits Conf.*, 460-461, (2016).
29. Chen, H., Zhot, X., Yu, Q., Zhang F. & Li, Q., A >3GHz ERBW 1.1GS/s 8b Two-Step SAR ADC with Recursive-Weight DAC, *IEEE Symposium on VLSI Circuits (VLSI-Circuits),* 97-98 (2018).
30. Zidan, M. A. *et al.* A general memristor-based partial differential equation solver, *Nat. Electron.* **1**, 411-420, (2018).
31. Le-Thai, H., Chapinal, G., Geurts, T. & Gielen, G. G. E., A 0.18- μm CMOS Image Sensor With Phase-Delay-Counting and Oversampling Dual-Slope Integrating Column ADCs Achieving 1e−rms Noise at 3.8-μs Conversion Time. *IEEE J. Solid-State Circuits* **53,** 515-526 (2017).
32. Xu, R., Liu, B. & Yuan, J., Digitally Calibrated 768-kS/s 10-b Minimum-Size SAR ADC Array With Dithering. *IEEE J. Solid-State Circuits* **47,** 2129-2140 (2012).
33. Li, T.-L. *et al.* A Column-Parallel Hybrid Analog-to-Digital Converter Using Successive-Approximation-Register and Single-Slope Architectures with Error Correction for

Complementary Metal Oxide Silicon Image Sensors. *Jpn. J. Appl. Phys.* **52,** 04CE04-1 - 04CE04-7 (2013).

34. Jeon, B.-K., Hong, S.-K. & Kwon, O.-K., A Low-Power 12-bit Extended Counting ADC Without Calibration for CMOS Image Sensors. *IEEE Trans. Circuits Syst. II, Exp. Briefs http://dx.doi.org/10.1109/TCSII.2017.2717044* (2017).
35. Kim, C. *et al.* A 92dB dynamic range sub-μVrms-noise 0.8μW/ch neural-recording ADC array with predictive digital autoranging. *IEEE International Solid-State Circuits Conference (ISSCC),* 470-472 (2018).
36. Kassiri, H. *et al.* All-Wireless 64-Channel 0.013mm2 /ch Closed-Loop Neurostimulator with Rail-to-Rail DC Offset Removal. *IEEE International Solid-State Circuits Conference (ISSCC),* 452-453 (2017).
37. Gagnon-Turcotte, G., Ethier, C., Köninck, Y. D. & Gosselin, B., A 0.13μm CMOS SoC for Simultaneous Multichannel Optogenetics and Electrophysiological Brain Recording. *IEEE International Solid-State Circuits Conference (ISSCC),* 466-468 (2018).
38. Bugiel, S. *et al.* Ultra-Low Power Fast Multi-Channel 10-Bit ADC ASIC for Readout of Particle Physics Detectors. *IEEE Trans. Nucl. Sci.* **63,** 2622-2631 (2016).
39. Gao, W., Gao, D., Wei, T., Hu-Guo, C. & Hu, Y., A 12-bit Low-Power Multi-Channel Ramp ADC Using Digital DLL Techniques for High-Energy Physics and Biomedical Imaging. *IEEE International Conference Solid-State and Integrated Circuit Technology (ICSICT),* 227-229 (2010).
40. O'Driscoll, S., Shenoy, K. V. & Meng, T. H., Adaptive Resolution ADC Array for an Implantable Neural Sensor. *IEEE Trans. Biomed. Eng* **5**, 120-130 (2011).
41. Gao, W., Gao, D., Hu-Guo, C. & Hu, Y., Design of a 12-Bit 2.5 MS/s Integrated Multi-

Channel Single- Ramp Analog-to-Digital Converter for Imaging Detector Systems. *IEEE Trans. Instrum. Meas.* **60,** 1942-1951 (2011).

42. Chang, T. *et al.* Synaptic behaviors and modeling of a metal oxide memristive device. *Appl. Phys. A* **102,** 857–863 (2011).
43. Masters, D. & Luschi, C. Revisiting Small Batch Training for Deep Neural Networks. *arXiv preprint arXiv:1804.07612*. (2018).
44. Keskar, N. S., Mudigere, D., Nocedal, J., Smelyanskiy, M. & Tang, P. T. P. On Large-Batch Training for Deep Learning: Generalization Gap and Sharp Minima. *arXiv preprint arXiv:1609.04836.* (2016).
45. Chen, P.-Y., Peng, X. & Yu, S. NeuroSim: A Circuit-Level Macro Model for Benchmarking Neuro-Inspired Architectures in Online Learning. *IEEE Trans. Comput. Des. Integr. Circuits Syst.* **37,** 3067–3080 (2018).
46. Choi, S. *et al.* SiGe epitaxial memory for neuromorphic computing with reproducible high performance based on engineered dislocations. *Nat. Mater.* **17,** 335–340 (2018).
47. Zidan, M. A. *et al*. Field-programmable crossbar array (FPCA) for reconfigurable computing. *IEEE Trans. Multi-Scale Comput. Syst*. https://doi.org/10.1109/TMSCS.2017.2721160 (2017).
48. Mikhailenko, D., Liyanagedera, C., James, A. P. & Roy, K. M2CA: Modular memristive crossbar arrays. *Circuits and Systems (ISCAS) 2018 IEEE International Symposium on. IEEE*, pp. 1-5, (2018).
49. Zidan, M. A., Strachan, J. P. & Lu, W. D. The future of electronics based on memristive systems. *Nat. Electron.* **1**, 22–29 (2018).
50. Krestinskaya, O., James, A. P., Chua, L. O., Neuro-memristive circuits for edge computing:

A review, *arXiv preprint arXiv:1807.00962.* (2018)

51. Xia, Q. and Yang, J.J. Memristive crossbar arrays for brain-inspired computing. *Nat. Mater.* **18**, 309–323 (2019)
52. Ielmini, D. & Wong, H.-S. P. In-memory computing with resistive switching devices. *Nat. Electron.* **1**, 333–343 (2018).
53. Prezioso, M. *et al.* Training and operation of an integrated neuromorphic network based on metal-oxide memristors. *Nature* **521,** 61–64 (2015).
54. Yao, P. *et al.* Face classification using electronic synapses. *Nat. Commun.* **8,** 15199 (2017).
55. Burr, G. W. *et al.* Experimental Demonstration and Tolerancing of a Large-Scale Neural Network (165 000 Synapses) Using Phase-Change Memory as the Synaptic Weight Element. *IEEE Trans. Electron Devices* **62,** 3498–3507 (2015).
56. Hu, M. *et al.* Memristor-Based Analog Computation and Neural Network Classification with a Dot Product Engine. *Adv. Mater.* **1705914,** 1–10 (2018).
57. Xu, X. *et al.* Scaling for edge inference of deep neural networks. *Nat. Electron.* **1**, 216–222 (2018).
58. Shafiee, A. *et al.* ISAAC: a convolutional neural network accelerator with in-situ analog arithmetic in crossbars. In *Proc. 43rd International Symposium on Computer Architecture* 14–26 (IEEE, 2016).
59. Gokmen, T. & Vlasov, Y. Acceleration of deep neural network training with resistive cross-point devices: design considerations. *Front. Neurosci.* **10**, 33 (2016).
60. Hubara, I., Courbariaux, M., Soudry, D., El-Yaniv, R. & Bengio, Y. Quantized neural networks: Training neural networks with low precision weights and activations. *arXiv preprint arXiv:1609.07061.* (2016).

61. Jacob, B., *et al.* Quantization and training of neural networks for efficient integer-arithmeticonly inference. *arXiv preprint arXiv:1712.05877.* (2017)

Chapter 4

Short Term Memory and Reservoir Computing System

4.1 Short term memory effect on WO_x memristor

For non-volatile memory (NVM) applications[1–5] and in-memory computing[6–10] with memristor devices, the conductance of the memristors should remain unchanged during the network operations. Otherwise, the stored information in memristors is no longer valid, resulting in critical errors on the computation of VMM and very low accuracy for deep learning models. However, the short-term memory (*i.e.* volatile) effect[11–13], which refers to the fact that the device can only hold its conductance value for a short period of time, can potentially be used to natively perform other computing tasks, particularly in the temporal domain. For instance, these dynamic effects allow the devices to map temporal input patterns into different device states, which can be used to build a "reservoir" in reservoir computing schemes. In this Chapter, I will discuss a study in which WOx memristors with short-term memory effects are used in reservoir computing systems for image classification and non-linear system mapping.

4.1.1 Short term dynamics on WO_x memristor

In a typical WOx memristor, the energy barrier for oxygen vacancy migration is low and oxygen vacancy drift by an electric field and spontaneous diffusion can co-exist in the device[12]. Depending on the purpose of the applications, the oxidation condition, and device structure can be tailored to achieve either short-term memory or long retention properties, which in turn allow the devices to be used in different applications.

Specifically, the memristor dynamics, considering spontaneous oxygen vacancy diffusion, can be described by the following equations:

$$I = (1 - w)\alpha[1 - \exp(-\beta V)] + w\gamma \sinh(\delta V) \qquad (4-1)$$

$$\frac{dw}{dt} = \lambda \sinh(\eta V) - \frac{w}{\tau} \qquad (4-2)$$

where equation (4-1) is the *I*-*V* equation which includes a Schottky term (the 1st term) corresponding to conduction in the V_O-poor region and a tunneling-like term (the 2nd term) corresponding to conduction in the V_O-rich region[14]. Equation (4-2) is the dynamics equation which describes the change rate of the state variable *w* with respect to the applied voltage, including the drift effect under an applied electric field (the 1st term) and the spontaneous diffusion (the 2nd term). α, β, γ, δ, λ, η are all positive-valued parameters determined by material properties. τ is the diffusion time constant, which corresponds to the retention or the decay speed of the memristor device.

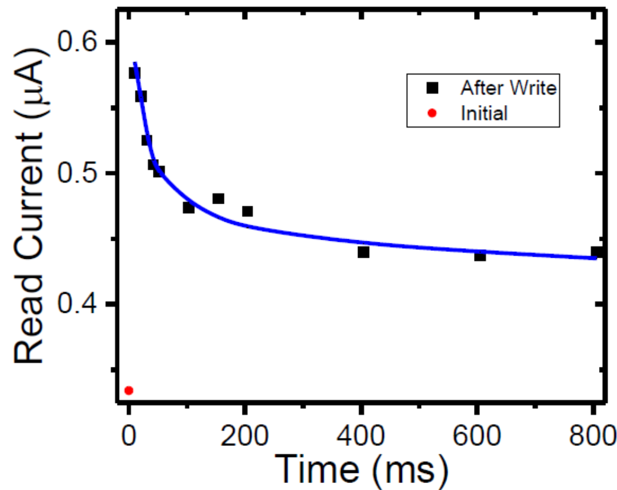


Figure 4-1: Conductance decay in a WOx memristor.
The device was first programmed by 5 write pulses (1.4 V, 1 ms) then its conductance was monitored by periodic read pulses (0.4 V, 500 μs). τ = 50ms.

With oxidation at a low temperature such as 350~375 °C for 45s and tungsten bottom electrode, we can achieve a pronounced short-term memory effect in WOx memristor devices. The time constant τ in the short-term memory devices is typically around 50ms, as shown in Figure 4-1. If we use stronger oxidation condition, e.g. 400~425°C for 60s, with inert, e.g. gold (Au), a bottom electrode, the device can obtain much longer retention. In this case, we can use the memristor devices to store synaptic weights and to perform matrix operation directly in the memristor arrays.

To demonstrate the temporal dynamics of the WO_x device[15], a pulse stream composed of write pulses having the same amplitude (1.4 V, 500 μs) but at different timeframes are applied to the device and the response of the memristor, which is represented by the read current through a small read pulse (0.6 V, 500 μs) following each write pulse, is recorded. The results are shown in Figure 4-2. It can be clearly observed that with consecutive short pulses, the conductance of a memristor with short-term memory effect is gradually increased, while the conductance state will decay without any stimulation. As a result, the final device state depends on the temporal pattern of the input, with different patterns leading to different device states. This native short-term memory effect allows a reservoir to be built with a small number of memristor devices (including using just one device), significantly reducing the hardware implementation complexity of reservoir computing systems.

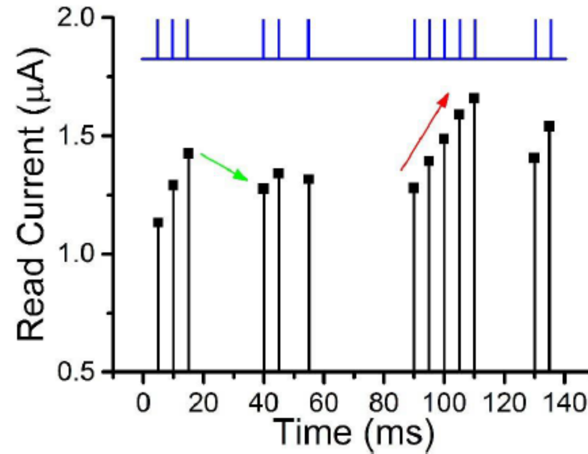


Figure 4-2: Memristor's temporal response to a pulse train.
Write pulses (1.4 V, 500 μs) with different timing (blue lines) were applied and the response, represented by current measured by a small read pulse (0.6 V, 500 μs) after each write pulse is recorded. The memristor shows distinctive response to specific temporal patterns applied to it.

4.1.2 Integration of WOx device for short term memory

A 32×32 WO_x array with short-term memory effects is used in reservoir computing studies. The array fabrication is as follows. 60 nm of W was first sputter deposited on a Si carrier wafer with a 100 nm thermally grown oxide. The bottom electrodes (BEs) with 500 nm width were patterned by e-beam lithography and reactive ion etching (RIE) using Ni as a hard mask. Afterward, the Ni hard mask was removed by wet etching. 300 nm of SiO_2 was then deposited by plasma-enhanced chemical vapor deposition, followed by RIE etch back to form a spacer structure along the sidewalls of the BEs. The spacer structure allows better step coverage of the top electrodes (TEs) at the cross-points and also restricts the resistive switching regions to a flat surface. The resistive switching WO_x layer was formed through rapid thermal annealing of the exposed W electrode surface with oxygen gas at 375 °C for 45 seconds. Afterward, the TEs (Pd (70 nm)/Au (90 nm)) were patterned by e-beam lithography, e-beam evaporation, and liftoff processes. Another RIE process was used to remove the WO_x between the TEs to isolate the devices and to

expose the BEs for electrical contacts. Finally, photolithography, e-beam evaporation, and liftoff process were performed to form wire bonding pads of 150 nm thick Au. Figure 4-3 shows SEM image of the 32 × 32 WO_x memristor array for temporal data processing.

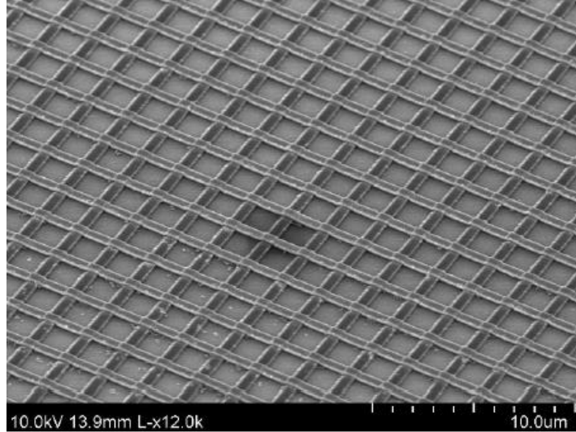


Figure 4-3: SEM image of the 32×32 WO_x memristor array.
Image Credit: Dr. Chao Du

4.2 Reservoir computing (RC) with memristor devices

Temporal data, including videos, speech, and other signals that evolve with a time that can across many different time scales, are of high technological and societal importance but are difficult to process with conventional DNNs. To process temporal data, networks with internal dynamics such as recurrent networks (RNNs) have been developed. However, generic RNNs are expensive to train. To address these challenges, reservoir computing (RC) systems[16,17] were proposed, where the original inputs can be non-linearly projected into a high-dimensional feature space through a dynamic reservoir. With this approach, the original features that may not be linearly separable can become linearly separable in the new feature space, and can then be further processed with a simple linear network. RC systems have been shown to outperform classical fully trained RNNs in many tasks[18–20]. To perform the nonlinear transformation of temporal data, a key

requirement of the reservoir is to have a “fading memory”, or “short-term memory” effect, so that the system can respond to inputs at the near past but not far past. With this approach, the reservoir only needs to be excited (mapping input features to different excited reservoir states) whereas the connectivity structure inside the reservoir remains fixed at all times, and thus does not require training. To further process the transformed data, a second network, called a readout function, is trained and generates the final output. Since the most difficult task of separating the features is implemented in the reservoir, a simple and small readout network is typically sufficient, thus dramatically reducing the training cost of the overall system.

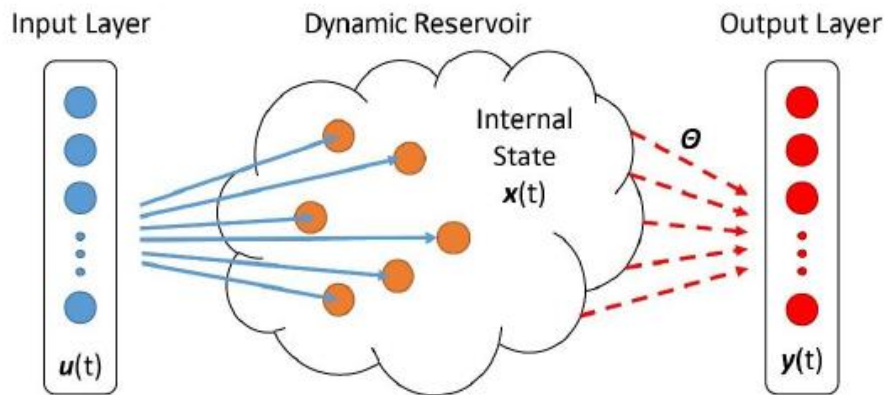


Figure 4-4: Schematic of an RC system.
This shows the reservoir with internal dynamics and a readout function.

Implementing the reservoirs using conventional systems can however be expensive. In this study, we attempt to utilize the internal short-term ionic dynamics of the memristor devices to build RC systems[15]. The internal ionic dynamic processes allow the memristors to map temporal input patterns into different reservoir states (represented as memristor resistance, as shown in Figure 4-2), which can be further processed via a simple readout function, as shown in Figure 4-4.

4.2.1 Classification task with the RC system

In our study, WO_x memristors with short-term memory properties are used to effectively implement RC systems. The first task we demonstrated is image classification. Take digit "2", a 4×5 input image, as an example. The image has and 20 pixels, with values of either black ("0") or white ("1"), as shown in Figure 4-5. It is then divided into 5 rows, each row containing 4 consecutive pixels and is fed into a memristor as a node the reservoir, using a 4-timeframe input stream. A timeframe (3 ms in width) will contain a write pulse (1.5 V, 1 ms) if the corresponding pixel is a white pixel, or no pulse (equivalently a pulse with an amplitude of 0 V) if the corresponding pixel is a black pixel[21]. Therefore, information of the image for the digit "2", which is represented by the spatial locations of the white pixels in each row, is represented by temporal features streamed into the reservoir, *i.e.*, a pulse stream with pulses applied at different timeframes.

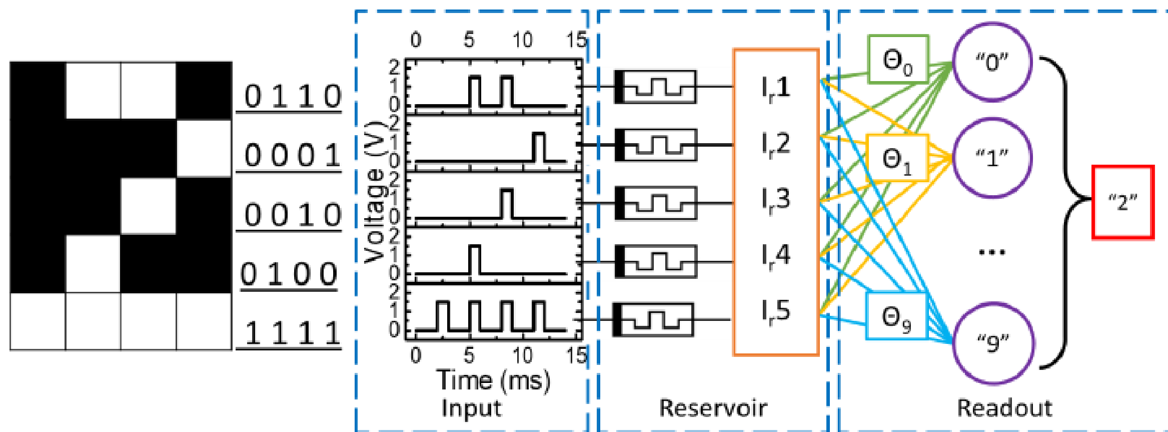


Figure 4-5: Reservoir for simple digit recognition.
Left: 4×5 pattern image, digit "2" as an example. Right: the reservoir containing the inputs (pulse transformed from the image), the liquid (consisting of 5 memristors) and the readout function (a network with 10 output neurons).

The goal is to extract information of the image, *i.e.* the digit number "2" here, by collectively processing the temporal features in the 5 input pulse streams. Here only 5 memristors

were used to process the image, with each memristor processing the input pulse stream from a specific row in the image. The reservoir state is represented by the collective resistance states of the 5 memristors. After the application of the input streams, the reservoir state is thus dependent on the input temporal patterns and can be used to analyze the input, as shown in Figure 4-5

Specifically, when a pulse is applied, the state of the memristor will be changed (reflected as a conductance increase) and if multiple pulses are applied with short interval a larger increase in conductance will be achieved, while long intervals without stimulation will result in the memristor state (conductance) decaying towards its resting state, *i.e.*, the initial state before any pulse is applied. Therefore, different temporal inputs will lead to different states of the device and consequently the overall reservoir state. In this specific setup, each memristor's state after stimulation will thus represent a specific feature for the given row in the original image, and the collective device states, representing the reservoir state, can be used to perform pattern recognition through the (trained) readout function, *i.e.* identifying the digit as "2" of the original input (Figure 4-5).

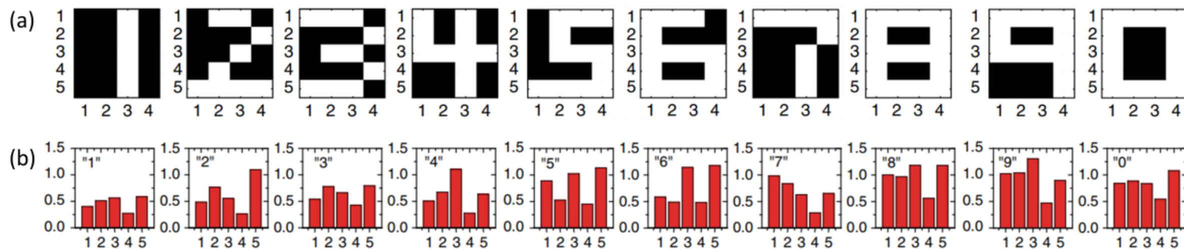


Figure 4-6: Test set and measured reservoir states.
(a) Images of the 10 digits used in this test. (b) Experimentally measured reservoir states after the memristors are subjected to the 10 inputs. The reservoir state is reflected as the read currents of the 5 memristors forming the reservoir.

The readout function here is a 5×10 network, with the reservoir state, measured by the read currents from the 5 memristors in the reservoir, as the input, and 10 output neurons (labeled 0-9)

representing the predicted digit value of the input image, schematically illustrated in Figure 4-5. During classification, the output from the 10 output neurons is calculated from the dot product of the 5 inputs and the weights associated with each output neuron, and the output with the maximum dot product is selected and its label number is used as the predicted digit value. The readout function is trained in a supervised fashion based on Softmax regression (explained with details in Chapter 3) where the weights are adjusted to minimize output error during training.

With the 10 images (from 0 to 9) and 200 training iterations, the RC system can correctly recognize all inputs from the 10 original images, as shown in Figure 4-6. To test the effects of cycle-to-cycle variations of the device, the 10 images were repeatedly tested ten times without retraining the readout function, and 100% accuracy was verified experimentally in the memristor-based RC system for this simple task.

4.2.2 Other complex temporal data processing with the RC system

In addition, the memristor-based RC system also can be used to process real-world problems such as handwritten digit recognition with performance comparable to those achieved in much larger networks. With the MNIST data set, an 88.1% recognition accuracy was obtained from the RC system experimentally with only 88 memristor devices (22 rows, 4 sections, 2 rates) where the unused boarder area was removed to reduce the original 28×28 image into a 22×20 image with 22 rows and 20 pixels per row. Increasing the reservoir to 112 memristors (28 rows, 4 sections, 3 rates) improves the performance to 91.5% accuracy[15].

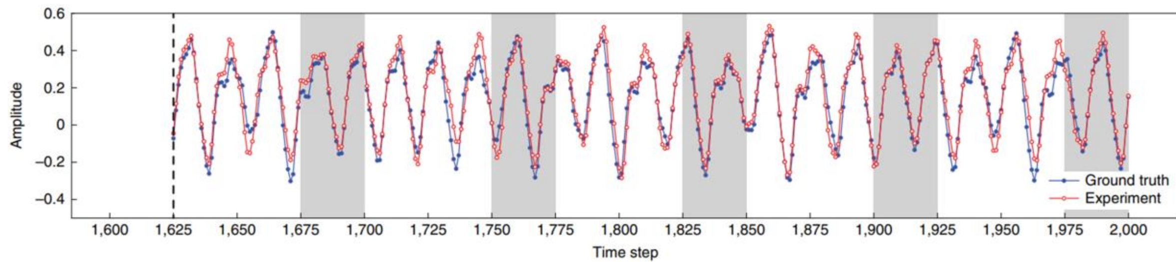


Figure 4-7: Long-term forecasting of Mackey–Glass time series using a reservoir system
The predicted output from the network is fed back to the network as input for the next time step.

It should be noted that in the RC system, the performance strongly depends on the dimension of the reservoir space. However, instead of increasing the number of physical nodes in the reservoir which can increase hardware implementation cost as well as the training cost of the readout layer, the concept of virtual nodes developed in delay systems can be an attractive alternative[22]. The state of virtual nodes depends on the node's own previous state, the current state of adjacent nodes, and the masked input signal, allowing them to be nonlinearly coupled. By using randomly generated masks for input signals, diverse responses can be obtained from the virtual nodes, and these systems have been shown to be able to achieve performance comparable to that of conventional and well-designed reservoirs. In another implementation of the memristor-based RC system by our group, the reservoir size was increased by using the virtual node concept, and spoken-digit was successfully recognized with the accuracy of 99.2%. More interestingly, since the network can capture the temporal features of the input, it was successfully used to perform prediction/forecasting functions. For example, in speech recognition, the speaker's intended word was correctly predicted before the speaker finished it. In another example, the network was able to capture the complex features and make predictions of a chaotic system such as the Mackey–Glass series, a deterministic system but very difficult to predict. Periodic updates can be used to

bring the reservoir state back to the original dynamics and make it possible to maintain even long-term prediction of the chaotic system[23], as shown in Figure 4-7. This project has been successfully transferred to other group members.

4.3 Conclusion

In this chapter, we show the internal short-term dynamics of memristor devices that can be used to implement RC systems to analyze and predict temporal data such as handwritten digit recognition, spoken digit recognition, and long-term forecasting of chaotic systems with very high accuracy. Unlike conventional machine-learning algorithms such as multilayer perceptrons (MLPs) and convolutional neural networks (CNNs) which expect data with fixed dimensions and typically work best with static inputs such as images, memristor-based RC systems are well suited for processing temporal data with very low power. By transforming the original time-dependent input into the excited memristor state space, temporal patterns in the inputs can be efficiently analyzed through a simple read-out function after the memristor reservoir. Continued material and device optimizations will help further broaden the appeal of the memristor-based RC systems for practical applications.

References

1. Hilson, G. IMEC, Panasonic push progress on ReRAM. https://www.eetimes.com/document.asp?doc_id= 13273 (2015).
2. Clarke, P. Crossbar ReRAM in Production at SMIC. https://www.eetimes.com/document.asp?doc_id= 13311 (2017).
3. Shen, W. C. *et al.* High-K metal gate contact RRAM (CRRAM) in pure 28nm CMOS logic process. *IEEE Int. Electron Devices Meet.* 31.6.1-31.6.4 (2012). doi:10.1109/IEDM.2012.6479146
4. Fackenthal, R. *et al.* A 16Gb ReRAM with 200MB/s write and 1GB/s read in 27nm technology. *IEEE Int. Solid-State Circuits Conf. Dig. Tech. Pap.* **57**, 338–339 (2014).
5. Yu, S. & Chen, P. Y. Emerging Memory Technologies: Recent Trends and Prospects. *IEEE Solid-State Circuits Mag.* **8**, 43–56 (2016).
6. Zidan, M. A. *et al.* Field-Programmable Crossbar Array (FPCA) for Reconfigurable Computing. *IEEE Trans. Multi-Scale Comput. Syst.* **4**, 698–710 (2018).
7. Borghetti, J. *et al.* Memristive switches enable stateful logic operations via material implication. *Nature* **464**, 873–876 (2010).
8. Prezioso, M. *et al.* Training and operation of an integrated neuromorphic network based on metal-oxide memristors. *Nature* **521**, 61–64 (2015).
9. Yao, P. *et al.* Face classification using electronic synapses. *Nat. Commun.* **8**, 15199 (2017).
10. Cai, F. *et al.* A fully integrated reprogrammable memristor–CMOS system for efficient multiply–accumulate operations. *Nat. Electron.* **2**, 290–299 (2019).

11. Chang, T., Jo, S. H. & Lu, W. Short-term memory to long-term memory transition in a nanoscale memristor. *ACS Nano* **5**, 7669–7676 (2011).

12. Du, C., Ma, W., Chang, T., Sheridan, P. & Lu, W. D. Biorealistic Implementation of Synaptic Functions with Oxide Memristors through Internal Ionic Dynamics. *Adv. Funct. Mater.* **25**, 4290–4299 (2015).

13. Ohno, T. *et al.* Short-term plasticity and long-term potentiation mimicked in single inorganic synapses. *Nat. Mater.* **10**, 591–595 (2011).

14. Chang, T. *et al.* Synaptic behaviors and modeling of a metal oxide memristive device. *Appl. Phys. A Mater. Sci. Process.* **102**, 857–863 (2011).

15. Du, C. *et al.* Reservoir computing using dynamic memristors for temporal information processing. *Nat. Commun.* **8**, 2204 (2017).

16. Lukoševičius, M. & Jaeger, H. Reservoir computing approaches to recurrent neural network training. *Comput. Sci. Rev.* **3**, 127–149 (2009).

17. Torrejon, J. *et al.* Neuromorphic computing with nanoscale spintronic oscillators. *Nature* **547**, 428–431 (2017).

18. Jaeger, H. & Haas, H. Harnessing Nonlinearity: Predicting Chaotic Systems and Saving Energy in Wireless Communication. *Science* **304**, 78–80 (2004).

19. Jaeger, H., Lukoševičius, M., Popovici, D. & Siewert, U. Optimization and applications of echo state networks with leaky- integrator neurons. *Neural Networks* **20**, 335–352 (2007).

20. Verstraeten, D., Schrauwen, B. & Stroobandt, D. Reservoir-based techniques for speech recognition. *IEEE Int. Conf. Neural Networks - Conf. Proc.* 1050–1053 (2006). doi:10.1109/IJCNN.2006.246804

21. Burger, J. & Teuscher, C. Variation-tolerant computing with memristive reservoirs. *Proc. 2013 IEEE/ACM Int. Symp. Nanoscale Archit. NANOARCH 2013* 1–6 (2013). doi:10.1109/NanoArch.2013.6623028

22. Appeltant, L. *et al.* Information processing using a single dynamical node as complex system. *Nat. Commun.* **2**, 1–6 (2011).

23. Moon, J. *et al.* Temporal data classification and forecasting using a memristor-based reservoir computing system. *Nat. Electron.* **2**, 480–487 (2019).

Chapter 5

Optimization of Memristor Devices and Systems, and Future Works

5.1 Device non-ideality issues for neuromorphic computing

Memristors have made remarkable progress over the last ~ 15 years. Beyond already being offered as commercial products for memory applications, memristors have been extensively studied for neuromorphic computing applications, providing significant benefits for real-time data processing with high throughput and low energy consumption. However, current memristor devices are still far from being ideal. First, due to its stochastic switching behavior based on individual ion/cation migration, significant device-to-device and cycle-to-cycle variations exist[1,2]. Unlike binary memory applications which just need to achieve enough read window margin to distinguish between 'ON' (LRS) from 'OFF' (HRS), these variations can be a major factor affecting the accuracy of memristor-based computing systems, leading to undesirable computation error.

Additionally, the impact of conductance update linearity and symmetry can be critical on the neural network training accuracy[3]. Ideally, during training, the calculated weight update (Δw) should be directly transformed to the number of programming pulses applied to the memristor, without having to read out the current weight and adjusting programming conditions accordingly. In other words, a single programming pulse (LTP or LTD pulse) should change the weight by a constant value, independent of the conductance state. However, due to the non-linear dependence of conductance on filament shape in filament-based memristors, typical memristors suffer from

nonlinear and asymmetric weight update characteristics. This leads to conductance changes that depend on the present conductance state, making it much more challenging to implement online training. Closed-loop weight update (program-and-verify) might not be a good alternative either due to very slow and complex operations that will significantly slow down online training, and the need of additional memory space to store current weight values. The non-linear weight changes can be attributed to the non-linear dependencies of ion migration on the local electric field, which is, in turn, a function of the filament shape, and the non-linear dependency of the device conductance on the effective filament length change. Particularly, in common oxide-based memristors, the V_O migration energy barrier is high. This leads to stable V_O filaments but also introduces a large degree of stochasticity during filament formation. Additionally, once a filament starts to grow it will quickly become the dominant one due to the field-enhancing effect at the growth front, leading to non-linear weight updates as the filament length is changed. More uniform V_O migration can be achieved with lower activation barriers but at a cost of shorter retention. In addition, a large dynamic range and good resistance stability are also crucial to reduce the error in VMM operations and to maintain high classification accuracy[4]. There is still no perfect device that can meet all these requirements. Continued improvements on the ionic process control and device optimization are essential.

In the next three sections, I will discuss three ongoing projects that aim to improve the memristor device through new, engineered materials (Section 5.2), and to improve the system performance through a modular, tiled architecture (Sections 5.3, 5.4), respectively.

5.2 Device optimization for online training

5.2.1 High entropy oxide (HEO) memristor

One research direction to improve memristor devices is to design new materials that can provide uniform V_O migration but without suffering from retention loss. One such material we started working on is based on the concept of high-entropy oxides (HEOs)[5]. It has been shown recently that entropy[5] can be used to stabilize new oxide phases, similar to the case in high entropy alloys, and thus it allows the creation of new oxides that cannot otherwise natively exist. For instance, unusual dielectric properties with a colossal dielectric constant and unusually high Li-ion mobility has been achieved in entropy-stabilized materials[6,7]. We hypothesize that HEOs allow the tuning of the material-dependent parameters such as the V_O activation energy and the hopping distance by varying their composition over a large range beyond the binary constituents. Additionally, the high entropy of mixing increases the thermodynamic stability of the homogeneous amorphous structures, by reliably accommodating a large amount of mobile species and allowing for tuning the composition over a broad range without phase segregation. The formation of the crystalline phase by Joule heating during device operation, another device failure mechanism[8,9], can also be suppressed in HEOs. HEOs also provide an intrinsic charge compensation mechanism between the substituents[10] that can lead to desired V_O generation. For example, when oxides with cations having lower oxidation state are added (e.g., the substitution of ZrO_2/HfO_2 into MoO_3/WO_3), oxygen vacancies can be spontaneously created to maintain charge neutrality. This capability can ensure the more uniform distribution of V_Os in the switching film and overcome the challenges faced by filament type devices, as discussed earlier.

5.2.2 High entropy oxide (HEO) materials

In this work, HEO materials consisting of six binary transition-metal oxides (ZrO_2, HfO_2, Nb_2O_5, Ta_2O_5, MoO_3, and WO_3) with similar ionic radii are studied, as shown in Figure 5-1. HfO_2, Ta_2O_5, and WO_3 have already been extensively studied and exhibit good RS behaviors[11–14]. Even though the three transition metals are adjacent to each other in the periodic table, each binary oxide exhibits different RS characteristics.

Figure 5-1: The periodic table and adjacent 6 transition metals for high entropy oxide (HEO) memristor.

For example, WO_3 has demonstrated good incremental (analog-type) conductance modulation with an interface-type switching mechanism[11], although retention of the resistance states is not ideal due to the high mobility (low activation energy) of the V_Os[11,15], while HfO_2-based devices show excellent retention but mostly abrupt (digital-type) conductance switching between two states. Both digital- and analog-type switching can be achieved in the same device with Ta_2O_5-based devices[16–18] but with limited on/off ratio. Directly mixing the three oxides to develop a material that may offer the desired switching characteristics from each, however, is not possible due to the mixing enthalpy costs. On the other hand, with the addition of similar oxides, e.g. ZrO_2, Nb_2O_5, MoO_3, from transition metals one row above to increase the entropy, stable

HEOs can theoretically be obtained that can potentially provide the parameter tenability not offered by individual oxides.

5.2.3 Characterization of HEO device and Investigation of systematic trend

Thin films were deposited by our collaborators of Prof. Jamie Phillips' group through pulsed laser deposition (PLD) based on targets synthesized from powders of the six binary oxides, as shown in Figure 5-2 (a). Powders from the six binary oxides were mixed with equimolar amounts of cations and milled. Then, the target was annealed at 1200 °C in a furnace for more than 48 hours to complete the mixing of the cations in the powders and pressed into 1-inch pellets. Thin films were deposited by PLD using the custom-made target, utilizing an excimer laser at a wavelength of 248 nm. Figure 5-2 (c) shows results from Energy-dispersive X-ray spectroscopy (EDS) of the deposited film, confirming the existence of six metal elements, Zr, Nb, Mo, Hf, Ta, and W in the film. Thin films were deposited with laser energy of 80 mJ, a substrate temperature of 200°C, a target-substrate distance of 6 cm and an oxygen partial pressure of 0-100mTorr, resulting in a controlled film growth. Still, the conditions of target synthesis and film deposition need further optimization to improve the film quality, e.g. reducing particle deposition and minimize the spatial variation of deposition rate.

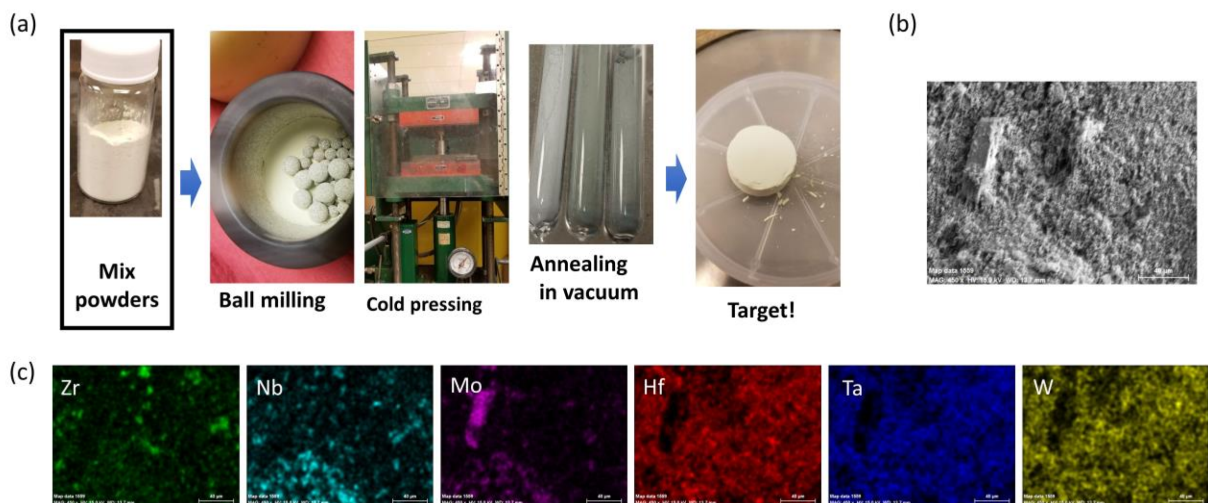


Figure 5-2: HEO target synthesis and deposition results.
(a) HEO target synthesis process. (b) Deposited film with PLD. (c) EDS mapping of the 6 elements in the film.

HEO-based devices with an Au/Pd/Ta/HEO/Pd stack were subsequently fabricated in a crossbar structure, shown in Figures 5-3 (a) and (b), on a SiO_2/Si substrate. Figure 5-4 (a) shows the resistive switching (RS) of the HEO memristor device. In this structure, the reactive Ta top electrode acts as an V_O supply layer, resulting in controlled bi-polar switching characteristics. The device also exhibits reliable incremental conductance updates with pulse trains. Although further optimization is needed, for instance, to systematically tune the film composition and stoichiometry, the first batches of HEO memristors already exhibit good RS characteristics comparable to the best Ta_2O_5 devices in terms of analog switching characteristics (LTP & LTD).

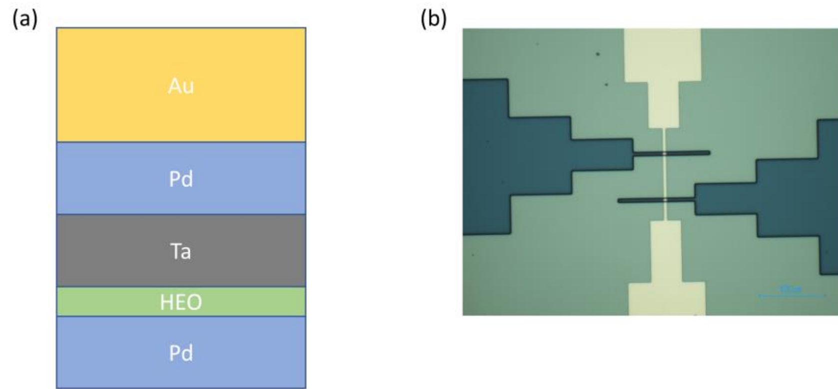


Figure 5-3: Structure of HEO devices.
(a) Schematic of the Au/Pd/Ta/HEO/Pd device stack. (b) Optical microscopy image of devices fabricated in a crossbar structure.

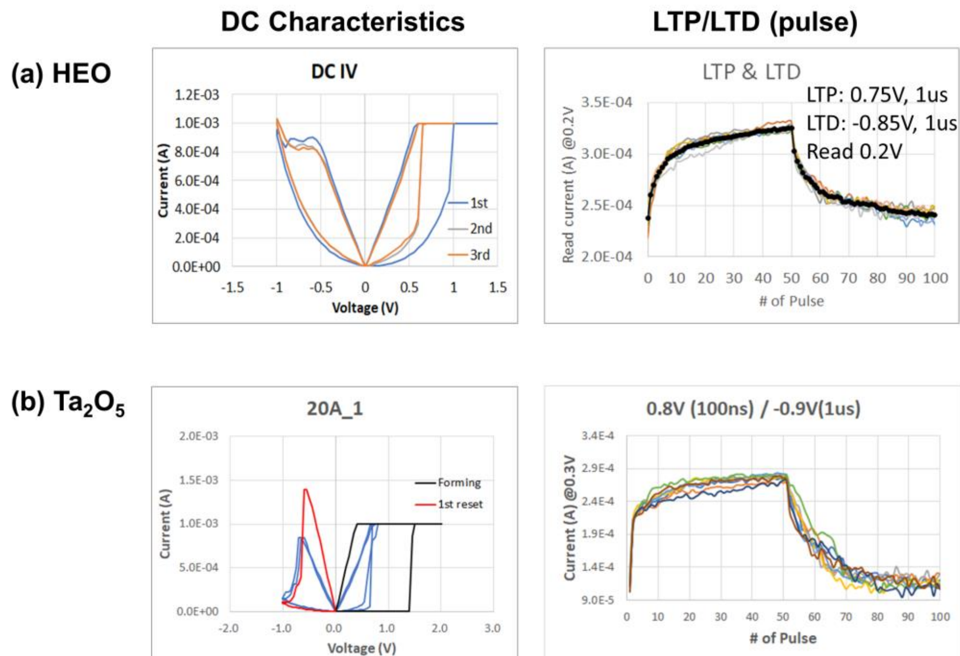


Figure 5-4: DC and pulse characteristics of HEO and Ta_2O_5 devices.

5.2.4 Future Plans:

Our collaborators, Prof. Emmanouil Kioupakis' group, will perform state-of-the-art first-principles calculations based on hybrid density functional theory to obtain the fundamental

material properties of the targeted amorphous HEO phases, such as (1) the structural parameters (atomic positions and bonding), (2) thermodynamics and phase stability (enthalpy of formation for crystalline and amorphous HEO phases), and (3) electronic structure (bandgap and orbital character of band extrema), as a function of composition and stoichiometry. First-principles calculations will then be performed to study the atomic-level microstructure of the amorphous structures with different composition ratios, and conduct nudged elastic band calculations with first-principles to determine the important kinetic parameters such as the energy barrier for ionic migration. The correlation between the barrier height and the local environment will be employed to help design the composition ratio for experimental HEO film growth and device fabrication and characterization. The initial focus will be on the ternary alloy (Zr-Hf-Nb-Ta-Mo-W oxide), and will subsequently expand to encompass other transition-metal oxides, such as TiO2, CuO, etc. that also exhibit interesting RS characteristics[19] and can lead to new oxygen bonding configurations and parameter tuning.

On the experimental side, after reliably obtaining initial RS characteristics based on the HEO film, programming conditions and device structures (TE/BE material, film thickness, etc.) will be performed to improve analog behavior (nonlinearity weight update). We will verify how the device properties change as the composition of HEO is changed. Device simulation will be performed with the calculated activation energy for V_O migration and hopping distance to explain device characteristics as a function of the HEO film properties.

Systematic measurements will be performed to extensively characterize the thin films. Rutherford Backscattering Spectrometry (RBS) can effectively quantify the stoichiometry of cations in deposited materials. In addition, to verify the homogenous distribution of multiple cations at the atomic scale, high-resolution scanning tunneling microscopy (STEM) with built-in

spectroscopy including EELS will be conducted in thin film samples. Individual elemental mapping and the comparison of composition ratio from multiple points obtained by EDS in STEM mode will provide the spatial distribution for each element at the atomic scale.

5.3 1T1R Array for DNN Inference Implementation

In Deep Neural Network (DNNs), the inference is the stage in which a pre-trained model is used to infer/predict the unseen test data and comprises a similar forward pass as training to predict the values[20]. Unlike training, it does not include a backward pass to compute the error and update weights, allowing a neural network to stay unchanged during inference. For inference applications, weight update will be performed very infrequently in the neural network, when one needs to change the model and weights. As a result, the speed of weight update is no longer critical so approaches such as write-verify can be used to fine-tune weight storage, and nonlinearity of weight update is no longer be a big issue as long as one can program weight value precisely to the neural network. This makes DNN inference a very attractive application for memristor-based computing systems.

5.3.1 Issue of online training with 1T1R system

Another argument for targeting the inference application, instead of training application, is the likely use of 1T1R arrays in the initial implementations due to the challenges of integrating reliable selectors in passive 1R arrays. As discussed in chapter 2.4, bulk-type doping effect was revealed by our simulation model during Set through I_{CC} modulation with Tr, enabling larger dynamic range and linear conductance modulations, which is valuable for low power neuromorphic applications. However, the 1T1R structure might not be suitable for neural network

online training, because error backpropagation to update the weight (Δw_{ij}) using gradient descent is calculated by the partial derivative of the errors from the output layer and does not require previous weight values in the weight matrix. To map the deep neural network (DNN) with a memristor/CMOS system, Δw_{ij} is converted into the number of pulses or corresponding pulse width to modulate the conductance of each memristor in the network, where nonlinearity of weight update should be very small. In the 1T1R neural network, however, the memristor conductance after programming does not depend on the pulse width since the positive feedback during the CF formation (Set) stops at the transistor maximum current level. As a result, the programmed memristor conductance solely depends on the gate voltage (V_G), *i.e.* saturation current, which makes it difficult to use conventional update rule during network online training. To update the network, the $w_{ij} + \Delta w_{ij}$ value has to be converted into V_G in the 1T1R system, resulting in additional memory blocks to store previous weight value (w_{ij}) of each memristor in the array.

5.3.2 1T1R array implementation for inference applications

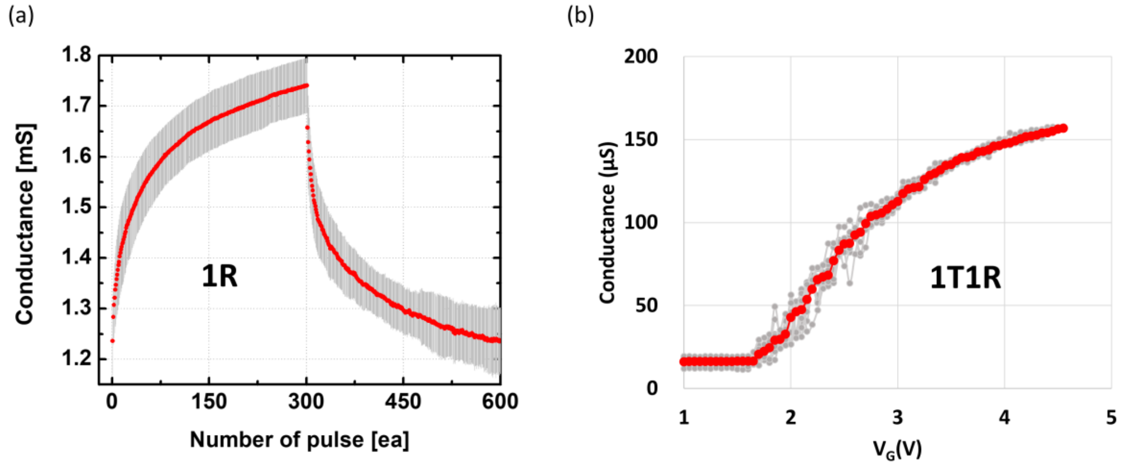


Figure 5-5: Analog behavior of typical 1R and 1T1R.
(a) Analog conductance update characteristics obtained from the1R structure. 30 devices were programmed by 300 consecutive write pulses (1.15 ± 0.1 V, 1 µs), followed by 300 erase pulses (−1.4 ± 0.1 V, 1 µs). *Reference* [41] (b) Analog conductance update characteristics obtained from the 1T1R structure. 9 devices were programed by write pulse (V_D:3V, PW:100ns, V_G:1V to 4.5V, 0.05V step).

The 1T1R system might be very useful for inference applications instead, especially at the edge, since it allows precise tuning of the memristor conductance (weight) with a large dynamic range that leads to low power operations compared to 1R system[21], as shown in Figure 5-5. Edge computing needs to processes large amounts of data near its source and only the data that needs to be stored is sent to the cloud, leading to the dramatic decrease of bandwidth and energy consumption compared to sending all data to the cloud for processing. This makes edge computing highly attractive, particularly since data are being exponentially generated at devices near the edge and certain applications demand latency, reliability (e.g. network congestion/outage) and privacy that will be challenging to meet with cloud-based computing.

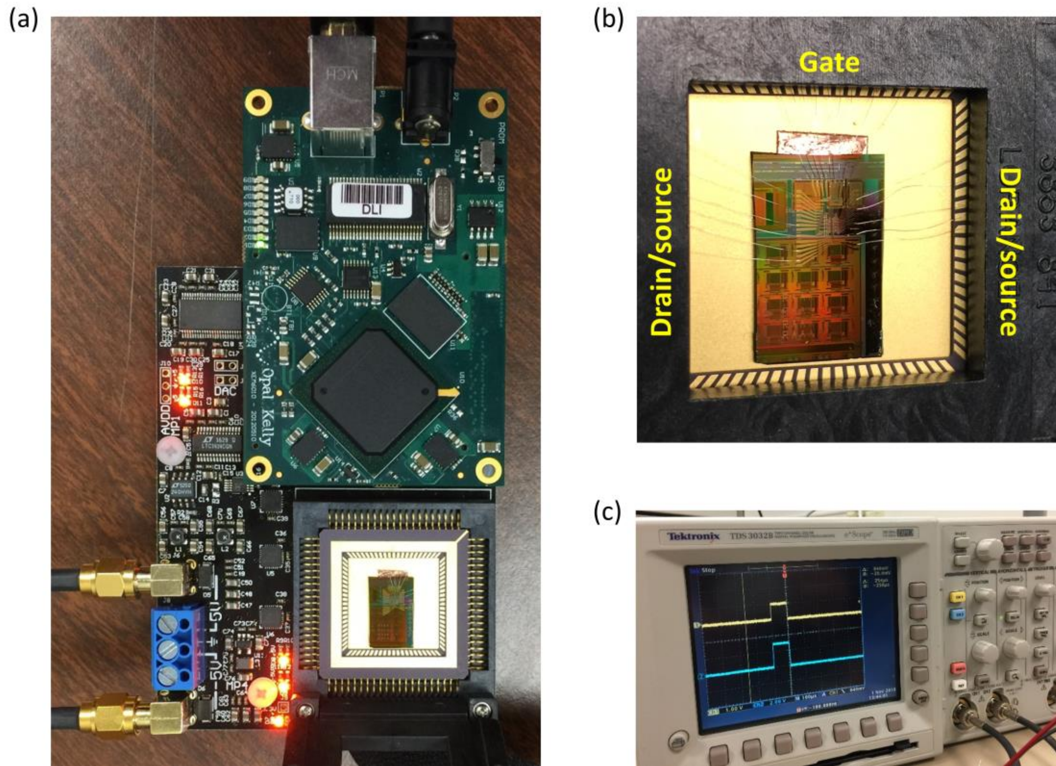


Figure 5-6: Test set-up for 1T1R array.
(a) Photo of the test board with 1T1R array. (b) The 1T1R array is wire-bonded and mounted on the board. (c) Verification of pulse inputs (V_D and V_G) for read and write of 1T1R array with oscilloscope.

We build prototype inference systems using 1T1R arrays and custom-built test board. The array is wire-bonded and mounted on the test board, as shown in Figure 5-6(a) and (b). The test board allows arbitrary pulse signals to be sent to and electronic current collected from either individual devices or multiple devices in multiple rows and columns, simultaneously and in parallel (Figure 5-6(c)).

5.3.3 Future Plans:

We plan to implement principal component analysis (PCA) and more complex tasks with this 1T1R system using pre-trained neural networks. In view of device-to-device and cycle-to-

cycle variability and V_{th} mismatch of the transistor, we will develop an optimized program-and-verify (PNV) scheme for 1T1R system. System performance will be precisely analyzed in terms of area and power if it is well scaled to state-of-the-art technology.

5.4 Deep neural network (DNN) accelerator based on tiled architecture

As discussed in the previous chapter, DNN accelerators, especially for inference operations, based on analog in-memory computing with memristor arrays can perform vector-matrix multiplication (VMM) in analog domain efficiently by accumulating total current or charge at each column[22,23]. At the same time, the high density and non-volatile properties of the memristor array make it possible to store entire DNN models on-chip, thus eliminates the inefficient off-chip memory access and promises much higher energy efficacy.

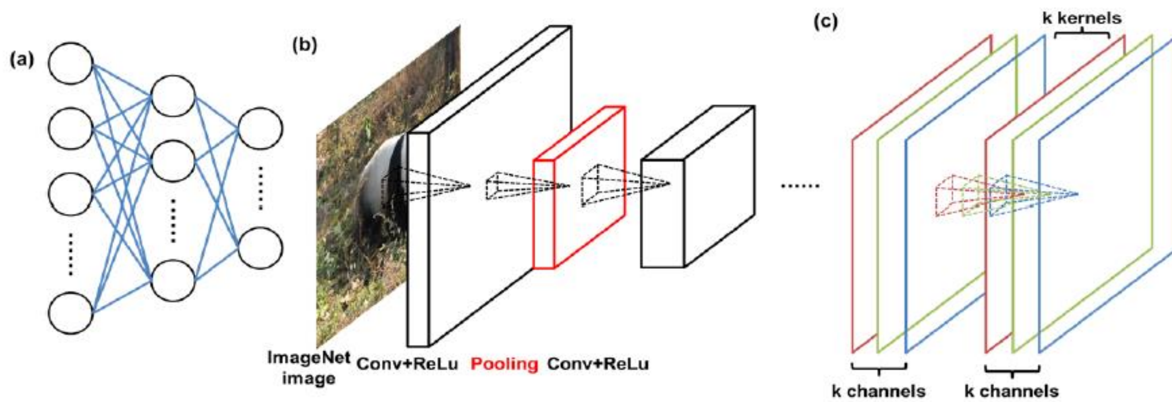


Figure 5-7: Different types of layers in DNN.
(a) Fully connected layers, (b) convolution layers, and (c) depth-wise convolution layers.

Although studies on memristor-based accelerators have extensively conducted, most of the studies have focused on small networks and simple databases such as MNIST[24], which may not

capture the challenges faced when implementing advanced models and complex tasks. We expect larger, practical models that can be implemented in memristor-based hardware in a reconfigurable tiled architecture, where weights in a single layer are mapped onto multiple crossbar arrays tiled through digital interfaces[25]. Efforts have been started to build this tiled hardware system using memristors fabricated in a commercial fab.

5.4.1 Tiled architecture implementation

We plan to use the 65nm technology node for CMOS integration and the 1T1R structure for memristor array implementation to decrease the off current and sneak current, and improve conductance level tuning, as discussed earlier. Weights to be stored in the memristor array will be calculated based on 8-bit quantization-aware-training and written by the program-and-verify (PNV) scheme through V_G modulation. Figures 5-8 and 5-9 show the full system overview and system architecture for the VMM operation. The RISC-V processor can execute program instructions to run the algorithm. Instruction and data memory (SRAM) will be used to store instructions and intermediate data and input data. The first chip will have 4 tiles of 1T1R array, with an array size of 256 × 64. Since the memristor has only positive conductance values, two memristors will be used to represent a single weight. For instance, odd columns will represent positive weight values and even columns will represent negative weight values.

The input can be applied in a bit-serial fashion, where 8 cycles are required for each 8-bit input (Clock Frequency: 100MHz). Total 160ns is needed to complete an 8-bit MAC operation.

Figure 5-8: Full system overview.
Image credit: prof. Flynn

Figure 5-9: Optimized tiled architecture.
(a) System level design for DNN accelerator based on 1T1R memristor arrays. (b) Schematic of a memristor/CMOS block.
Image credit: prof. Flynn

We plan to first build a simple CNN model for the MNIST classification task, using the 4 tiles of memristor arrays, as shown in Figure 5-8. For this model, there are 4 layers, including 3 convolutional layers and 1 fully connected layer. In the first layer, there are 22 filters and each

filter size is 3×3×1. After convolution, nonlinear activation function (i.e. ReLu) and 2×2 max pooling will be performed. The 2nd layer includes 27 3×3×22 filters, followed by ReLu and 2×2 max pooling. The 3rd layer includes 64 3×3×27 convolution filters, ReLu and 4x4 max pooling, resulting in 1×1×64 outputs. The results are flattened and turned into a single vector (64×1) that can be applied to the next stage. Finally, the last fully connected layer produces the final output probabilities for each label. Basically, the convolutional and pooling breaks up the image into features and analyzes them, while a fully connected layer takes the output of the convolutional layer and predicts the best label to describe the images. With an 8-bit quantization, the model achieves 97.21%. accuracy.

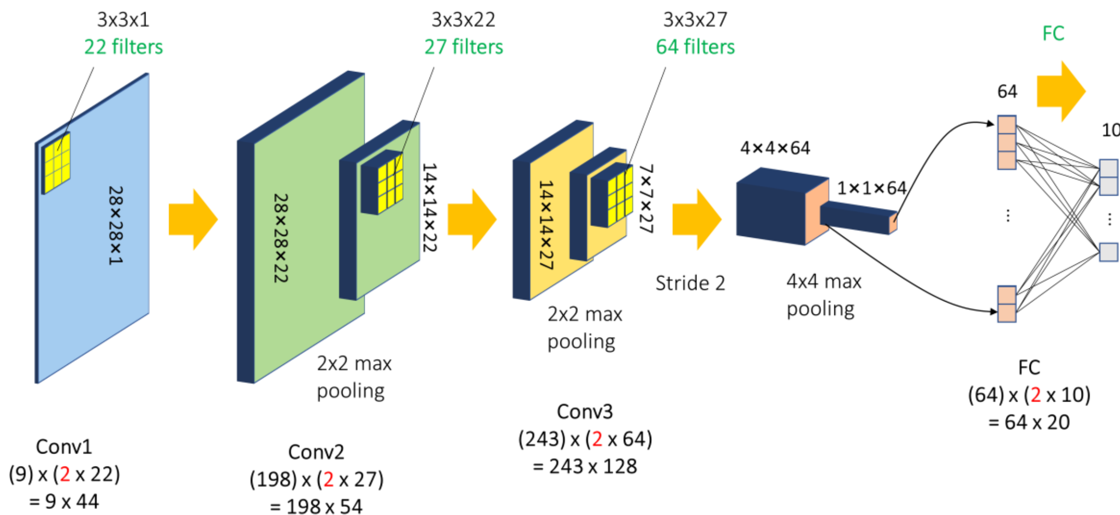


Figure 5-10: Schematic of the CNN model for MNIST.
The proposed CNN model consists of 3 convolutional layers and 1 fully connected layer.

The weight values of each filter will be stored in the memristor array, where the required array size is 9×44, 198×54, 243×128, and 64×20 for each layer, including even/odd weights, as shown in Figure 5-10. Almost all weight values in the model are close to 0 or very small, which

corresponds to 1MΩ in our memristor device, resulting in low output current, e.g. less than 45μA, that will be collected by the ADC. This corresponds to ~1/16 of the theoretical maximum current value, which helps to improve the ADC design with fixed bits.

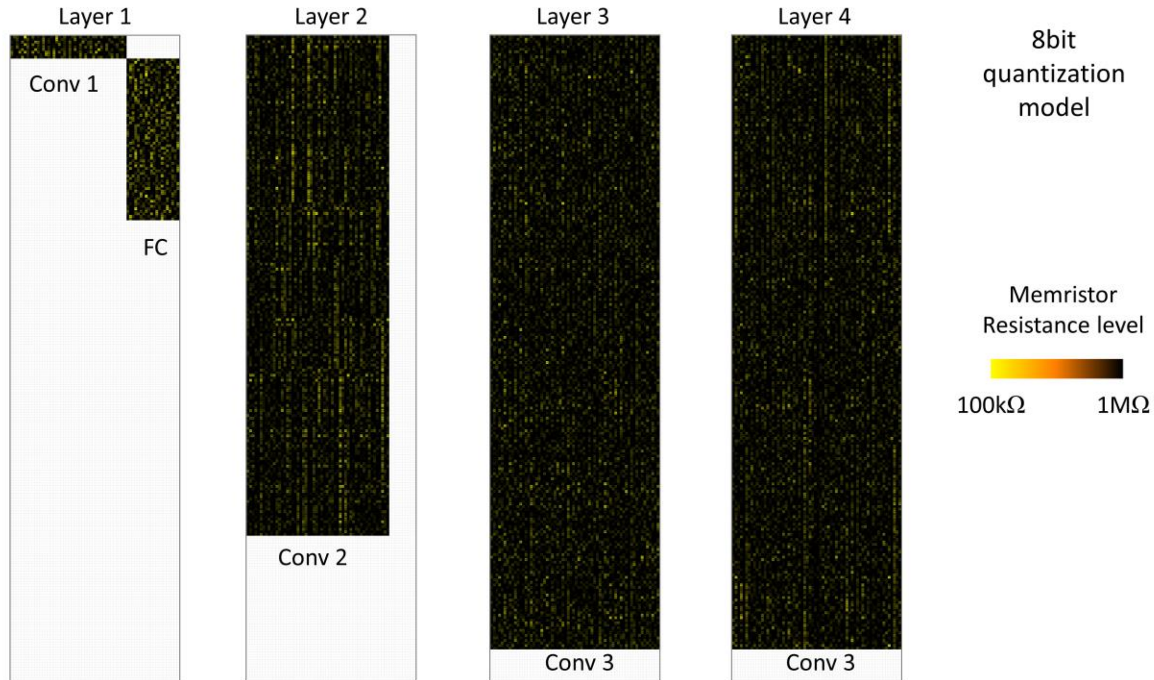
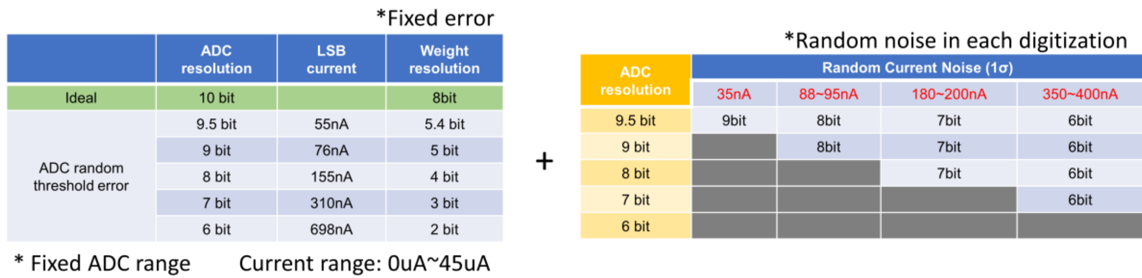


Figure 5-11: Weight mapping of the CNN to the 4 tiles.

Based on this analysis, ADC accuracy with fixed errors and random noises was evaluated. We assume 10-bit ADC will be designed for the output quantization, although we only take 8-bit value during operation. However, due to V_t mismatch and nonlinearity, the designed 10-bit ADC accuracy will be lower, for example, it can be even 9.5bit, 9bit, 8bit, and 7bit. This corresponds to different LSB (least significant bit) current and weight resolution if we use a single fixed ADC range. Even worse, if we include random noise, the ADC resolution will be further dropped depending on the noise level. For instance, if the random current noise is 200nA, then the maximum ADC resolution will be 7bit, as shown in Figure 5-12.


*Fixed error

	ADC resolution	LSB current	Weight resolution
Ideal	10 bit		8bit
ADC random threshold error	9.5 bit	55nA	5.4 bit
	9 bit	76nA	5 bit
	8 bit	155nA	4 bit
	7 bit	310nA	3 bit
	6 bit	698nA	2 bit

* Fixed ADC range Current range: 0uA~45uA

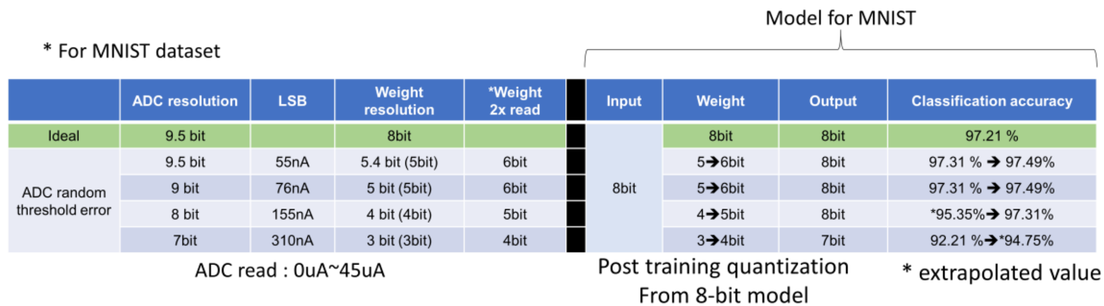
+

*Random noise in each digitization

ADC resolution	Random Current Noise (1σ)			
	35nA	88~95nA	180~200nA	350~400nA
9.5 bit	9bit	8bit	7bit	6bit
9 bit		8bit	7bit	6bit
8 bit			7bit	6bit
7 bit				6bit
6 bit				

Figure 5-12: Weight mapping.
ADC resolution drop can be critical due to fixed error and random noise term.

Based on the different LSB current levels, we can simulate the MNIST model operation with different weight- and ADC-precisions. We found that the MNIST task is quite tolerable, resulting in over 90% accuracy, even for low weight and ADC precisions. In addition, during single cell programming a higher (e.g. 2x read voltage) can be used, effectively expanding the current resolution by 1 bit, which can further improve the classification accuracy, as shown in Figure 5-13.


* For MNIST dataset

Model for MNIST

	ADC resolution	LSB	Weight resolution	*Weight 2x read	Input	Weight	Output	Classification accuracy
Ideal	9.5 bit		8bit			8bit	8bit	97.21 %
ADC random threshold error	9.5 bit	55nA	5.4 bit (5bit)	6bit	8bit	5→6bit	8bit	97.31 % → 97.49%
	9 bit	76nA	5 bit (5bit)	6bit		5→6bit	8bit	97.31 % → 97.49%
	8 bit	155nA	4 bit (4bit)	5bit		4→5bit	8bit	*95.35%→ 97.31%
	7bit	310nA	3 bit (3bit)	4bit		3→4bit	7bit	92.21 %→*94.75%

ADC read : 0uA~45uA Post training quantization From 8-bit model * extrapolated value

Figure 5-13: Model accuracy with lower precision ADC.

These results are summarized by plotting the model accuracy vs. ADC and weight precision, as shown in Figure 5-14. With 4-bit weight and 8-bit output, ~95% accuracy can be achieved. Note this simulation is a post-training quantization model from the original 8-bit

quantization model. With quantization aware training, the accuracy is expected to be further improved.

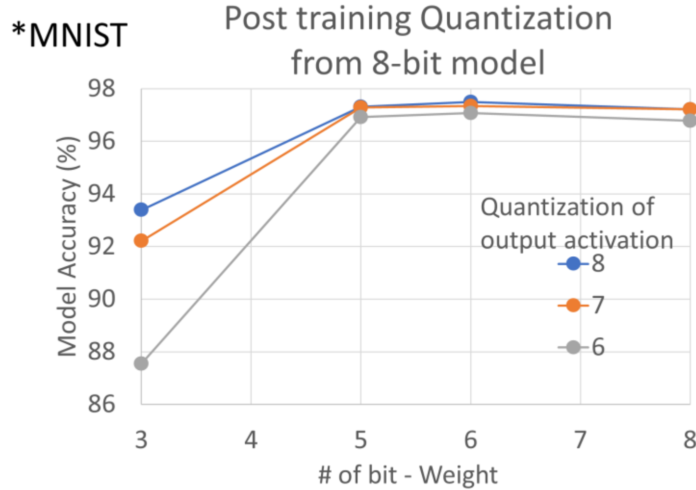


Figure 5-14: Model accuracy vs. ADC and weight precision.

5.4.2 Future Plans:

We plan to build the 4 tile system with 256×64 arrays. The 1T1R chip will be fabricated in a commercial fab, and TaO_x-based memristor devices will be integrated between metal 4 and metal 5. There will be 6 operation modes in the array: Idle, Forming, Set (potentiation), Reset (depression), Single read (after programming), and MAC read (even/odd).

First, in order to verify the 1T1R array operation and the Convolutional neural network (CNN) model we build, stand-alone 1T1R array (without CMOS circuit) will be tested. Program and read operation will be performed through external discrete components (DACs, ADCs, MUX, matrix switch, and FPGA) on a test board for this purpose. We first target 16 conductance states (4bit) for each memristor using the program-and-verify (PNV) scheme, where the 16 levels should tolerate device-to-device and cycle-to-cycle variations.

Afterwards, combined with integrated CMOS circuitry such as the RISC-V processor, mixed-signal interface, SRAM, and bus (designed by Prof. Flynn's group and Prof. Zhang's group), we will map the CNN model on the memristor arrays to perform power efficient MAC operations and run the CNN model for MNIST. The accuracy of the experimentally implemented system will be measured, along with power and throughput measurements, and compared with simulation results.

We will then expand the tiled system to practical applications such as ImageNet classification where the network size and the number of parameters will be significantly increased. Data flow and the system architecture will be further optimized to minimize stalls and bus bottlenecks. Improved model mapping that balances the workload of different layers to maximize parallelism and throughput will also be performed.

5.5 Summary

With the high density, speed, nonvolatility, and the ability to store and process information at the same physical locations, memristors can efficiently implement neural networks (NNs) and other in-memory computing architectures for data intensive applications such as machine learning. Further optimization of memristor devices and systems will be essential to improve the reliability, dynamic range, and update linearity to make such architectures practical. From the device point of view, the concept of high entropy oxide (HEO) may offer opportunities to custom-design the switching material due to the potential to combine different transition metal oxide to achieve good linearity and symmetry for weight update (WO_x) and good retention (TaO_x and HfO_x). In the short term, 1T1R arrays are likely employed first due to its maturity. These systems might be best used for inference applications due to the large dynamic range and precise conductance controllability.

For practical applications, scaling up the system with larger networks is highly desirable. Instead of increasing network size, tiling crossbars together in a modular fashion can be a promising approach that offers better scalability and minimizes parasitic effects. With further optimizations of the ADC design and system architecture, the system's power efficiency and throughput will be further improved. These systems thus offer great promise for a broad range of data-intensive applications ranging from IoT devices, autonomous systems, to large scale enterprise use scenarios.

References

1. Menzel, S., Kaupmann, P. & Waser, R. Understanding filamentary growth in electrochemical metallization memory cells using kinetic Monte Carlo simulations. *Nanoscale* **7**, 12673–12681 (2015).
2. Qin, S. *et al.* Atomistic study of dynamics for metallic filament growth in conductive-bridge random access memory. *Phys. Chem. Chem. Phys.* **17**, 8627–8632 (2015).
3. Chen, P. Y., Peng, X. & Yu, S. NeuroSim+: An integrated device-to-algorithm framework for benchmarking synaptic devices and array architectures. *Tech. Dig. - Int. Electron Devices Meet. IEDM* 6.1.1-6.1.4 (2018). doi:10.1109/IEDM.2017.8268337
4. Sun, X. & Yu, S. Impact of Non-Ideal Characteristics of Resistive Synaptic Devices on Implementing Convolutional Neural Networks. *IEEE J. Emerg. Sel. Top. Circuits Syst.* **9**, 570–579 (2019).
5. Rost, C. M. *et al.* Entropy-stabilized oxides. *Nat. Commun.* **6**, (2015).
6. Bérardan, D., Franger, S., Dragoe, D., Meena, A. K. & Dragoe, N. Colossal dielectric constant in high entropy oxides. *Phys. Status Solidi - Rapid Res. Lett.* **10**, 328–333 (2016).
7. Bérardan, D., Franger, S., Meena, A. K. & Dragoe, N. Room temperature lithium superionic conductivity in high entropy oxides. *J. Mater. Chem. A* **4**, 9536–9541 (2016).
8. Miao, F. *et al.* Anatomy of a nanoscale conduction channel reveals the mechanism of a high-performance memristor. *Adv. Mater.* **23**, 5633–5640 (2011).
9. Kumar, S. *et al.* Direct Observation of Localized Radial Oxygen Migration in Functioning Tantalum Oxide Memristors. *Adv. Mater.* **28**, 2772–2776 (2016).

10. Rak, Z. *et al.* Charge compensation and electrostatic transferability in three entropy-stabilized oxides: Results from density functional theory calculations. *J. Appl. Phys.* **120**, (2016).

11. Chang, T., Jo, S. H. & Lu, W. Short-term memory to long-term memory transition in a nanoscale memristor. *ACS Nano* **5**, 7669–7676 (2011).

12. Yang, J. J. *et al.* High switching endurance in TaOx memristive devices. *Appl. Phys. Lett.* **97**, 232102 (2010).

13. Lee, M. J. *et al.* A fast, high-endurance and scalable non-volatile memory device made from asymmetric Ta2O5-x/TaO2-x bilayer structures. *Nat. Mater.* **10**, 625–630 (2011).

14. Lee, H. Y. *et al.* Low power and high speed bipolar switching with a thin reactive ti buffer layer in robust HfO2 based RRAM. *Tech. Dig. - Int. Electron Devices Meet. IEDM* 3–6 (2008). doi:10.1109/IEDM.2008.4796677

15. Du, C., Ma, W., Chang, T., Sheridan, P. & Lu, W. D. Biorealistic Implementation of Synaptic Functions with Oxide Memristors through Internal Ionic Dynamics. *Adv. Funct. Mater.* **25**, 4290–4299 (2015).

16. Kim, S., Choi, S. & Lu, W. Comprehensive physical model of dynamic resistive switching in an oxide memristor. *ACS Nano* **8**, 2369–2376 (2014).

17. Kim, S. *et al.* Experimental demonstration of a second-order memristor and its ability to biorealistically implement synaptic plasticity. *Nano Lett.* **15**, 2203–2211 (2015).

18. Jeong, Y. J., Kim, S. & Lu, W. D. Utilizing multiple state variables to improve the dynamic range of analog switching in a memristor. *Appl. Phys. Lett.* **107**, (2015).

19. Yang, J. J. *et al.* Memristive switching mechanism for metal/oxide/metal nanodevices. *Nat. Nanotechnol.* **3**, 429–433 (2008).

20. Xia, Q. & Yang, J. J. Memristive crossbar arrays for brain-inspired computing. *Nat. Mater.* **18**, 309–323 (2019).

21. Jeong, Y. J., Lee, J., Moon, J., Shin, J. H. & Lu, W. D. K-means Data Clustering with Memristor Networks. *Nano Lett.* **18**, 4447–4453 (2018).

22. Cai, F. *et al.* A fully integrated reprogrammable memristor–CMOS system for efficient multiply–accumulate operations. *Nat. Electron.* **2**, 290–299 (2019).

23. Choi, S., Shin, J. H., Lee, J., Sheridan, P. & Lu, W. D. Experimental Demonstration of Feature Extraction and Dimensionality Reduction Using Memristor Networks. *Nano Lett.* **17**, 3113–3118 (2017).

24. Chen, P. Y., Peng, X. & Yu, S. NeuroSim: A circuit-level macro model for benchmarking neuro-inspired architectures in online learning. *IEEE Trans. Comput. Des. Integr. Circuits Syst.* **37**, 3067–3080 (2018).

25. Zidan, M. A. *et al.* Field-Programmable Crossbar Array (FPCA) for Reconfigurable Computing. *IEEE Trans. Multi-Scale Comput. Syst.* **4**, 698–710 (2018).