BRAIN-INSPIRED COMPUTING: NEUROMORPHIC SYSTEM DESIGNS AND APPLICATIONS

by

Chenchen Liu

B.S. in Material Physics,

Shaanxi University of Science and Technology, China, 2010

M.S. in Electronic and Communication Engineering,

Peking University, 2013

Submitted to the Graduate Faculty of

the Swanson School of Engineering in partial fulfillment

of the requirements for the degree of

Doctor of Philosophy

University of Pittsburgh

2017

UNIVERSITY OF PITTSBURGH

SWANSON SCHOOL OF ENGINEERING

This dissertation was presented

by

Chenchen Liu

It was defended on

May 8th, 2017

and approved by

Hai (Helen) Li, Ph.D., Associate Professor, Department of Electrical and Computer Engineering

Yiran Chen, Ph.D., Associate Professor, Department of Electrical and Computer Engineering

Ching-Chung Li, Ph.D., Professor, Department of Electrical and Computer Engineering

William E. Stanchina, Ph.D., Professor, Department of Electrical and Computer Engineering

Hao Jiang, Ph.D., Associate Professor, Department of Electrical and Computer Engineering

Dissertation Director: Hai (Helen) Li, Ph.D., Associate Professor, Department of Electrical and Computer Engineering

BRAIN-INSPIRED COMPUTING: NEUROMORPHIC SYSTEM DESIGNS AND APPLICATIONS

Chenchen Liu, PhD

University of Pittsburgh, 2017

In nowadays big data environment, the conventional computing platform based on von Neumann architecture encounters the bottleneck of the increasing requirement of computation capability and efficiency. The "brain-inspired computing" Neuromorphic Computing has demonstrated great potential to revolutionize the technology world. It is considered as one of the most promising solutions by achieving tremendous computing and power efficiency on a single chip. The neuromorphic computing systems represent great promise for many scientific and intelligent applications. Many designs have been proposed and realized with traditional CMOS technology, however, the progress is slow. Recently, the rebirth of neuromorphic computing is inspired by the development of novel nano-technology.

In this thesis, I propose neuromorphic computing systems with the ReRAM (Memristor) crossbar array. It includes the work in three major parts: 1) Memristor devices modeling and related circuits design in resistive memory (ReRAM) technology by investigating their physical mechanism, statistical analysis, and intrinsic challenges. A weighted sensing scheme which assigns different weights to the cells on different bit lines was proposed. The the area/power overhead of peripheral circuitry was effectively reduced while minimizing the amplitude of sneak paths. 2) Neuromorphic computing system designs by leveraging memristor devices and algorithm scaling in neural network and machine learning algorithms based on the similarity between memristive effect and biological synaptic behavior. First, a spiking neural network (SNN) with a rate coding model was developed in algorithm level and then mapped to hardware design for supervised learning. In addition, to further speed and accuracy improvement, another neuromorphic system adopt-

ing analog input signals with different voltage amplitude and a current sensing scheme was built. Moreover, the use of a single memristor crossbar for each neural network layer was explored. 3) The application-specific optimization for further reliability improvement of the developed neuromorphic systems. In this thesis, the impact of device failure on the memristor-based neuromorphic computing systems for cognitive applications was evaluated. Then, a retraining and a remapping design in algorithm level and hardware level were developed to rescue the large accuracy loss.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

ACKNOWLEDGEMENTS

I would like to acknowledge the support of my advisor, Hai (Helen) Li, whose support made this work possible, and to Air Force Research Lab (AFRL), Defense Advanced Research Projects Agency (DARPA), and National Science Foundation Project for directly providing much of the financial support. I'd like to thank Professor Hai (Helen) Li and Professor Yiran Chen for their excellent guidance during the research. Professor Hai (Helen) Li gives me guidance to become a qualified student and helped to me to develop my potential in my future career. Special thanks go to Professor Ching-Chung Li, Professor William E. Stanchina, and Professor Hao Jiang for being my committee members. Besides, I'd like to express my gratitude to the members from Evolutionary Intelligent (EI) lab at Swanson School of Engineering for their consistent supports during my research. Finally, I'd like to thank my family in China for their great encouragement during the whole Ph.D. research.

1.0 INTRODUCTION

In deep learning networks, the matrix-vector (and matrix-matrix) multiplications are basic operations that determine the overall computation speed, accuracy, and power consumption [4]. Accelerating the execution of matrix-vector multiplication emerges as an important task and extensive studies have been carried out. Revolutionary paradigms on general-purpose platforms, *e.g.*, GPU [5] and CPU [6], and domain-specific hardware like FPGA [7] have been developed. However, the computation efficiency improvement is hindered by the traditional von Neumann architecture, resulting in high hardware cost and energy consumption [8].

The recent rebirth of neuromorphic computing inspires a new solution of implementing neural networks in specialized VLSI designs to overcome the above difficulty. Firstly proposed in the 1980s, neuromorphic computing refers to implementing the computation in neural systems by utilizing a specific VLSI hardware system [9]. In traditional CMOS domain, many systems in digital, analog and mixed-signal formats have been presented and demonstrated, aiming at improving computing and data communication efficiency, *i.e.*, high computation speed with low cost [10]; [11]. The "brain chip" proposed by a group of MIT researchers [12] is a good example. The design with 400 transistors tends to mimic analog signal transmission in human brain. Neuromorphic computation platform can also be realized on *field-programmable gate array* (FPGA) or *field-programmable analog array* (FPAA) for low power signal processing and reconfigurability [7]; [13]. Alternatively, IBM reported *TrueNorth* – a spike-timing-based biosynaptical chip, in which synapses were built with SRAM cell in a crossbar structure. Extremely low power and energy consumption in data transferring was achieved [14]; [15].

With the appearance of new technologies and devices such as spintronic device [16], phase change device [17]; [18] and resistive device [19]; [20], new neuromorphic design and architecture adopting these novel devices are widely investigated. For example, a neuromorphic hardware using

spin devices in crossbar structure was proposed in 2012 [21]. It obtained more than $15\times$ lower energy consumption, comparing to the state of art CMOS designs. Among of all the emerging devices, resistive device (a.k.a.memristor) emerges as one of the most attractive candidates because it naturally performs alike synapse and owns the features of good scalability, low energy, multiple-state operation and CMOS compatibility [22].

The resistive switching effect as the basic principle in resistive memory (i.e. ReRAM) cell operations has been studied and used in memory applications since 1960 [23]; [24]; [25]. In the following years, various oxide materials with fast resistive switching characteristics including NiO, SiO_x, Al_2O_3, and Ta_2O_5 were investigated [19]. However, there was no practical prototype reported until 2002 when a 64-bit ReRAM array based on perovskite oxide devices was fabricated at $0.5\mu m$ CMOS process [26]. In 2004, a binary *transition metal oxide* (TMO) ReRAM was integrated at $0.18\mu m$ CMOS technology [27]. Notably, in 2008 HP Labs [28] described the ReRAM devices with analogue resistive states as *memristors*, approving the existence of the fourth basic circuit element predicted by Professor Leon Chua in 1971 [29].

Since then, extensive efforts have been given to the ReRAM development and applications. For instance, the ReRAM technology can not only be used as high density memory but also be leveraged in realizing reconfigurable systems [30] and matrix-based computation [31]. Usually, the resistive devices are organized in a cross-point array structure that offers ultra-dense data storage with a unit area of only $4F^2$, where F represents the technology feature size. Hence, the basic computation in deep neural network, that is matrix-vector multiplication (a.k.a. sum-of-products) can be implemented naturally and efficiently by the resistive crossbar, as is illustrated in Figure 1.

However, accessing such a passive resistive network inevitably induces current flows through unselected paths, or, *sneak paths* [1]; [32]. The extra current on sneak paths can be regarded as noise that degrades the effective programming voltage in write operation or contaminates the real information of the target cell during a read. Great efforts at device level [33]; [34]; [35] as well as circuits level [30]; [36]; [37] have been made to alleviate the impact of sneak paths. Importantly, Qureshi et al. [37] utilized an *operational amplifier* (op-amp) to accurately retain the voltage of unselected cells at a fixed level and hence suppress the sneak path leakage. Notably, the large area of the op-amp design severely constraints the allowable number of op-amps and therefore the data access bandwidth. In this work, we propose a *weighted sensing scheme* which assigns different

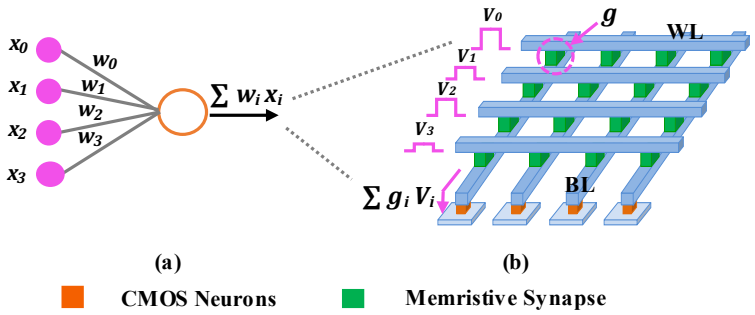

Figure 1: Implementing sums-of-product computation in neural network to a memristor crossbar.

weights to the cells on different bitlines. Thus, these bitlines can share only one op-amp to read out the associated data simultaneously, effectively reducing the area/power overhead of peripheral circuitry while minimizing the amplitude of sneak paths. Our simulation shows that the proposed scheme can successfully read out multi-bit data, even when RRAM device has a very low $\mathrm{R_{off}}/\mathrm{R_{on}}$ ratio of 5.

On the system-level front of the development of the neuromorphic systems based on resistive crossbar, Hu *et al.* proposed a neuromorphic engine with memristor crossbar in an analog computing approach – the input and output signals are represented by analog signals with high parallel computation in 2012 [3]. However, it requires *digital-to-analog converter* (DAC) and *analog-to-digital converter* (ADC) to transfer the input/output data, indicating large design cost.

In this work, we propose a novel and practical spiking neuromorphic design built on resistive crossbar arrays. A rate coding scheme where pre- and post-neuron signals are represented by digitalized pulses are adopted. The computing result is transferred into digitalized output spikes via an integrate-and-fire circuit (IFC) as the post-neuron. The computational robustness of the system was analyzed after including design considerations and verified by circuit simulations. Moreover, we applied the design for digital image recognition using two typical neural network configurations—feedforward and Hopfield networks. Compared with the previous memristor crossbar-based analog computing engine [3], our design can dramatically reduce more than 50% energy consumption. Further more, a chip design with a single layer and a 2-layer feedforward neural networks are implemented in IBM130 nm technology using the full custom design flow and then fabricated.

After that, a new memristor crossbar based computing engine which leverages a current-based sensing scheme for higher computation speed and accuracy is proposed. The design supplies analog voltage signals to wordlines in parallel. A current buffer amplifier directly senses out bitline current so the tail voltage of bitline maintains at a constant level. Two approaches are developed to transfer the current computing results into digital/analog voltage signals - one approach is connecting the IFC developed above after an current sensing scheme, and the other approach is developing a current-to-voltage converter basing on the current sensing scheme. Matrix-vector computing system with the first approach is implemented and applied in a 3-layer feedforward neural network. Compared to the spiking design, this design obtained 8.1% improvement in recognition accuracy. Similarly, the matrix-vector computing system is designed using the analog-mixed-signal design flow in IBM130 nm technology and then fabricated. Moreover, single layer and 2-layer feedforward neural network with approach with current-to-voltage converter are implemented and simulated at GlobalFoundry 130nm technology node. This proposed computing system demonstrates good classification accuracy: a maximum computation accuracy of 91.8% can be reached in a single layer design while a 96.12% computation accuracy value is obtained in the two-layer design.

In addition, as the development of memristor technology is still maturing, device defects and fabrication yield may be a significant concern. Specifically, the single-bit failure (SBF) denotes a device that freezes in a high conductance state ("stuck-on") or a low conductance state ("stuck-off"). Although neural networks usually can tolerate a certain number of imperfect synaptic weights, high SBF rate degrades the computation accuracy significantly. For example, we tested a feed-forward neural network for MNIST database: as the SBF rate increases to 20%, the average recognition accuracy rapidly dropped from 92.64% to 39.4%, which is far below an acceptable range. Redundancy schemes have been widely adopted in memory designss [38]. But it is not efficient for the memristor-based analog computations with high precision requirement. In this work, we propose a defect rescuing methodology that leverages the application-specific features to improve the hardware efficiency. Our results on the two-layer network show that by remapping only 5% defects of the most significant weights, the recovery rate further increases to 99.3%.

The rest of our paper is organized as follows:

- We itemize the detailed work progress and expected contributions of our ReRAM technology study and their application in storage memory and neuromorphic computing in Chapter 2.
- We then present the details and expects implementation of our weighted sensing approach for solving sneak path leakage problem in memory design in Chapter 3.
- We show our spiking neuromorphic computing system implementation and fabrication details in Chapter 4.
- We demonstrate our new neuromorphic computing system for higher speed and accuracy in Chapter 5.
- Finally we introduce our work in rescue the memristor-based neuromorphic design with high defects, as is shown in Chapter 6.

2.0 RESEARCH SCOPE AND EXPECTED CONTRIBUTIONS

The proposed work can be decoupled as following three main research scopes: 1) to investigate and provide a more efficiency approach to solve the sneak path leakage issue in ReRAM technology; 2) with a better understanding of the ReRAM cell (Memristor) and its application in neuromorphic computing, to design a spiking neuromorphic computing system with memristor crossbar and apply it in artificial neural network (ANN) for image classification. Then, finish the chip design with two feedforward neural networks - single layer and two-layer in IBM130 nm technology for fabrication; 3) to design a level-base neuromorphic computing systems for higher computing speed and accuracy compared to the spiking neuromorphic system mainly for matrix-vector computation, and evaluate its computation accuracy by applying it in neural network for image reconfiguration. Then, finish the chip design with a matrix-vector computation system in IBM130 nm technology for fabrication; 4) to improve the reliability of the memristor-based neuromorphic designs for cognitive applications considering the current development status of memristor device.

For research Scope 1, we proposed a weighted sensing scheme in ReRAM technology to solve the sneak path leakage issue in the ReRAM crossbar array. The major task of this approach is that assigns different weights to the cells on different bitlines. Thus, these bitlines can share only one op-amp to read out the associated data simultaneously, effectively reducing the area/power overhead of peripheral circuitry while minimizing the amplitude of sneak paths. Our major expect technical contributions of research Scope 1 are:

- We evaluate and analyze the significant of sneak path leakage in a passive resistive crossbar.
- We extend Qureshis work that connecting one operational amplifier (op-amp) after each bitline of the crossbar to eliminate the sneak path leakage, and propose a weighted sensing scheme to enable multiple bits sensing simultaneously.

- We implement the weight assignment in the crossbar array to guarantee storage data can be read out successfully at the same time.
- We evaluate the effectiveness of weighted sensing, the dependence of the op-amp gain parameter of our proposed design, the impact of device resistance ratio in the ReRAM design, and the area we can save by our proposed weighted sensing.

For research Scope 2, we proposed a spiking neuromorphic design with resistive crossbar. Different with the traditional neuromorphic computing system with resistive crossbar that uses voltage and current magnitudes to represent the data that participate in the computation, our design adopting spikes to represent signal, and thus our design provides the following features: 1) A digitalized interface in spikes with good noise immunity and energy efficiency for signal transferring; 2) Highly-efficient parallel analog operations of synaptic weighting function through resistive crossbar arrays; 3) A novel integrate-and-fire circuit converting the analog computation data to output spikes at a rate of up to 568.2M spikes/sec and energy of 0.48pJ-per-spike.

Our major technical contributions of research Scope 2 are:

1) In Hardware Level:

- We proposed a new integrate-and-fire circuit(IFC) design featuring high speed and low power consumption. The area of the IFC design at IBM 130nm technology is 175.3μm^2, which is compatible to that of traditional designs, while The energy consumption of our design is 0.48pJ-per-spike, which is about a quarter of the traditional designs.
- We designed the 1T1R (one-transistor-one-resistor) crossbar structure and a control scheme to eliminate the sneak path leakage and guarantee computing accuracy.

2) In System and Application Level:

- We explored the computational accuracy of our proposed spiking design based on a 32 $\times$ 32 crossbar array.
- We evaluated the performance and robustness of the proposed spiking neuromorphic design by using the application of digital image recognition. A feedforward network and a hopfield network are implemented and evaluated.

3) In Chip Level:

- We designed a chip with a single layer and a 2-layer feedforward neural networks for image classification in IBM130 nm technology using the full custom design flow and then fabricated.

For research Scope 3, we proposed a level-based neuromorphic design with resistive crossbar. Different with the spiking design, this design supplies analog voltage signals to wordlines in parallel and a current-based sensing scheme is developed for higher computation speed and accuracy. A current buffer amplifier directly senses out bitline current so the tail voltage of bitline maintains at a constant level. Two approaches are developed to transfer the current computing results into digital/analog voltage signals: 1) One approach is connecting the IFC developed in the spiking design after the current buffer amplifier. The current buffer amplifier is here to isolate the resistive crossbar and the IFC and transfer the computation results from the crossbar to the IFC. 2)The other approach is developing a current-to-voltage converter basing on the current sensing scheme, by which the current computation result from the crossbar can be transferred to analog voltage.

Our major technical contributions of research Scope 3 are:

1) In Hardware Level:

- We proposed a new current sensing scheme to copy the current computation result from the resistive crossbar or transfer the current result to analog voltage with high computation speed and accuracy.
- We designed the 1R - resistive device only crossbar structure which offers the minimal cell size of $4F^2$ while assuring computation accuracy.

2) In System and Application Level:

- Feed-forward neural networks with different array size and layer number for MNIST handwritten digit recognition are implemented basing on the neuromophic system in the two proposed approaches. The proposed designs are expected to have higher computing speed and accuracy.

3) In Chip Level:

- We designed a matrix-vector computation system using the analog-mixed-signal design flow in IBM130 nm technology and then fabricated.

For research Scope 4, we proposed a defect rescue neuromorphic design to restore the computing accuracy loss caused by the bit failure within the crossbar. Fist, the impact of device failure on the memristor-based neuromorphic computing systems for cognitive applications was analyzed through feedforward neural networks implementation for MNIST database recognition. Then, practical solutions to rescue the neuromorphic hardware with device defects was explored. The significance of synaptic weights was evaluated theoretically and statistically initially, and then a retraining and a remapping design in algorithm level and hardware level were developed.

Our major expect technical contributions of research Scope 4 are:

- *Learning weight significance*. We will classify the synaptic weights in a neural network into *significant* and *insignificant* categories based on their impact on the network's performance.
- *A retraining algorithm* is developed to compensate the SBF caused computation error by retuning the trainable weights. Two major constrains in *weight initialization* and *weight updating* are involved in accelerating the retraining process and mimicking the SBF defects in a memristor array.
- *A remapping algorithm* that utilizes a redundancy scheme can further improve the computation accuracy, especially when a large number of SBF defects fall in the significant weights category. Only the defects corresponding to the most significant weights will be remapped to the redundancy columns.

3.0 A WEIGHTED SENSING SCHEME FOR RERAM-BASED CROSS-POINT MEMORY ARRAY

In this chapter, we will present the details of the proposed "weighted sensing scheme" - an approach effectively reducing the area/power overhead of peripheral circuitry while minimizing the amplitude of sneak paths. The structure of this chapter is organized as the follows: Section 3.1 gives the background introduction on ReRAM technology and cross-point array; Section 3.2 describes the proposed weighted sensing scheme and the related implementation details; Section 3.3 presents and analyzes the simulation results; At last, we summary the work in this chapter in Section 3.4.

3.1 RERAM BASED CROSS-POINT ARRAY

3.1.1 ReRAM Cell Basics

Generally, ReRAM denotes to the random access memories that reply on resistance difference for data storage. For example, the *high resistance state* (HRS) and *low resistance state* (LRS) of a ReRAM device respectively represent logic '0' and '1', or versa vice. Among various conducting mechanisms to illustrate resistive switching phenomenons, the concept of conductive filaments has been widely recognized—the LRS/HRS of a ReRAM device is enabled through the formation/rupture of conducting channels [19][20][39].

Figure 2 illustrates a simple ReRAM switching model [1]. A SET process with an external voltage exceeding the set voltage ($\mathrm{V_{set}}$) transfers a memory cell to HRS. Oppositely, a negative reset voltage ($\mathrm{V_{reset}}$) is applied to switch it back to LRS in RESET procedure. A small voltage $\mathrm{V_{read}} < \mathrm{V_{set}}$ is used in a read operation so as not to disturb the stored data. In this work, we use $\mathrm{R_{off}}$ and $\mathrm{R_{on}}$ to represent the device resistance values at HRS and LRS, respectively.

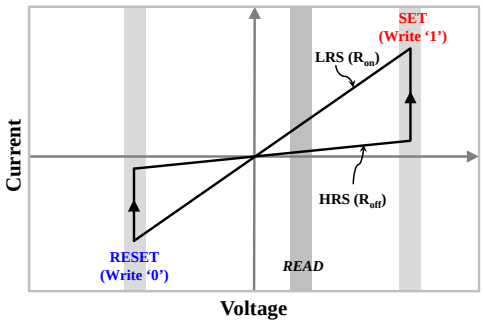

Figure 2: A ReRAM switching behavior model [1].

3.1.2 ReRAM Cross-point Structure Analysis

Cross-point array has been demonstrated as a telecommunication switching system since 1939. The nanometer ReRAM design exploits the similar structure in which two layers of metal wires are connected by memory devices at cross points as illustrated in Figure 3. Such a structure can achieve the minimal unit storage area of $4F^2$, where F represents technology feature size. For comparison, the area of an SRAM cell is $30\times$ larger ($> 120F^2$).

However, accessing such a passive resistive network inevitably induces current flows through unselected paths, or, *sneak paths* [1][32]. The extra current on sneak paths can be regarded as noise that degrades the effective programing voltage in write operation or contaminates the real information of the target cell during a read. The performance of a cross-point array is thereby dependent on the amount of sneak path leakage current, which is determined by the number of memory devices as well as the resistance value of these cells (that is, data pattern). The maximum array dimension is restricted by the worst-case condition to guarantee correct functionality.

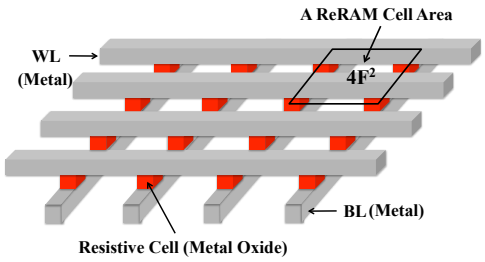

Figure 3: The ReRAM based cross-point array.

Great efforts at device level have been made to alleviate the impact of sneak paths. A popular solution is stacking a ReRAM device with a diode, which is regarded as a selected device [33]. Such a design can be applied to unipolar devices (e.g., PCM) using short/long pulses or high/low voltages at the same polarity to execute programming/erasing. A latest research effort tends to integrate a ReRAM device with a bipolar nonlinear selector [34]. Accordingly, high voltage is required in memory accesses, resulting in large power consumption. Complementary resistive switch is another interesting approach, which connects two bipolar ReRAM device anti-serially [35]. During operation, all the unselected cells stay at high resistance states so as to minimize the sneak path leakage. However, reading data from this structure is always associated with memory writes, increasing the read latency and degrading device reliability.

Improvements in crosspoint array and peripheral circuit designs have also been widely explored. Y.-C. Chen et al. proposed to insert a column of dummy cells as reference to partially compensate the impact of sneak path leakage [30]. The design leaves all the unselected cells floating, potentially inducing stability issue. *AC sensing* as a new design concept sets all the rows and columns to the same voltage potential and uses an AC signal to detect the stored information [36]. The complexity of such a small-signal design usually is high, especially after considering the process variations and signal fluctuations. Recently, Qureshi et al. [37] utilized an *operational amplifier* (op-amp) to accurately retain the voltage of unselected cells at a fixed level and hence suppress the sneak path leakage. Notably, the large area of the op-amp design severely constraints the allowable number of op-amps and therefore the data access bandwidth.

3.2 WEIGHTED SENSING SCHEME

3.2.1 Significance of Sneak Path Leakage

A cross-point array with bipolar ReRAM devices suffers from sneak path through neighbor cells. As shown in Figure 4, sneak paths leakage is referred to the current leakage flows through unselected cells when accessing the resistive network. In a read operation, the current leakage can be regarded as noise which contaminates the real information of the selected cells. We first evaluate the impact of sneak path leakage on the sensing margin of a cross-point array. All the circuit simulations were conducted under TSMC $180\mathrm{nm}$ technology under Cadence Spetre environment.

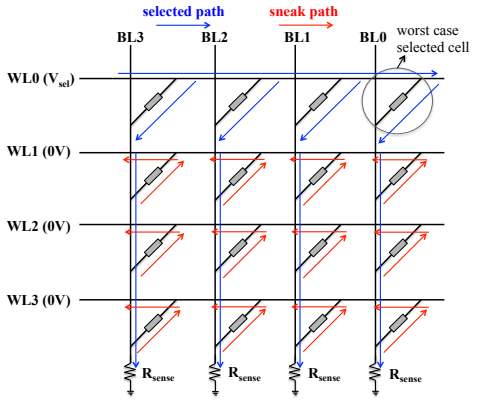

Figure 4: Sneak path leakage in a cross-point array.

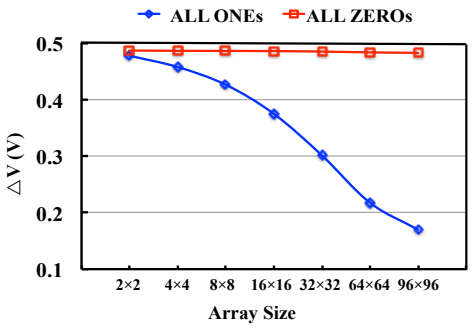

Figure 5: ΔV is determined by the cross-point array size and data pattern.

The sensing margin is represented by *the output voltage swing* (ΔV) which is the difference of sensing voltages when reading '1' and '0'. By referring to [32], we set $\mathrm{R_{on}}$ and $\mathrm{R_{off}}$ to $5\mathrm{K}\Omega$ and $1\mathrm{M}\Omega$ respectively. $\mathrm{R_{sense}}$ is the load resistor along bitline (see Figure 4) that behaves as the input resistance of sense amplifier. It is set to 100Ω. The evaluation also include the impact of wires assuming the resistance between two adjacent junctions is 2.5Ω [32]. We apply $0.5\mathrm{V}$ to the selected wordline and $0\mathrm{V}$ to the remaining wordlines.

Figure 5 shows the output voltage swing ΔV of the cell at the rightest top corner in Figure 4. Due to the interconnect resistance and sneak path leakage, it encounters the largest degradation and hence is considered as the worst-case situation [32]. Because the sensing margin is severely affected by the data patterns, we select two data patterns to represent the two extreme situations: all the unselected cells are at logic "1" (`ALL ONEs`) or at logic "0" (`ALL ZEROs`). Under `ALL`

ZEROs, ΔV drops slightly because the HRS of the unselected cells suppress the sneak path leakage. In contrast, when ALL ONEs is applied, all the unselected cells are at LRS and sink a significant amount of sneak path current. Moreover, as the cross-point array size increases, the impact of sneak path leakage grows and consequently the output voltage swing ΔV degrades dramatically. As such, cross-point memory design is constrained in a small size.

3.2.2 Design Principle of Weighted Sensing

Notably, sneak path leakage doesn't exist if the voltage drop across unselected cells perfectly remains at zero. Based on this concept, Qureshi *et al.* [37] proposed a new sensing scheme which connects each bitline with an *operational amplifier* (op-amp) for data sensing. Bringing in op-amps helps maintain close-to-zero voltage drop across unselected cells, which effectively suppresses the sneak path leakage current. However, the complex op-amp design induces significant overhead in design area, making it difficult to adopt such a solution in real implementation. In this work, we extended Qureshi's work and propose *a weighted sensing scheme to enable multiple bits sensing simultaneously*. The proposed design uses only one sensing circuit component to enhance the read efficiency and reduce the overhead of peripheral circuitry.

Figure 6 illustrates the principle of the proposed weighted sensing. The design assigns the data information to different cells with different weights. For example, the three cells in Figure 6 have weights of 1, 2, and 4, respectively. Consequently, their resistances fall into different ranges corresponding the ratio of the weight. Although an identical wordline voltage is supplied to the three cells on the same row in a read operation, currents through these cells are different due to their different weights. These currents are summed and collected at the end of bitlines and used to detect the data stored in these cells. For instance, the largest possible sensing current indicates all the three cells are at LRS, or logic '1'. The design requires to differentiate the sensing currents (voltages) of the eight possible combinations of three-bit data for the given example.

For ReRAM based cross-point memory array, the current summation of multiple bitlines can be integrated with the op-amp design. Thus, one op-amp is sufficient for multi-bit data detection while suppress the sneak path leakage. Note that our proposed read scheme is orthogonal to the conventional ReRAM design, in which a sensing component is shared by multiple columns belonging to different access blocks [40]. The weighted sensing utilizes one sensing circuit for multiple to-be-readout bits to further reduces the number of read circuit components.

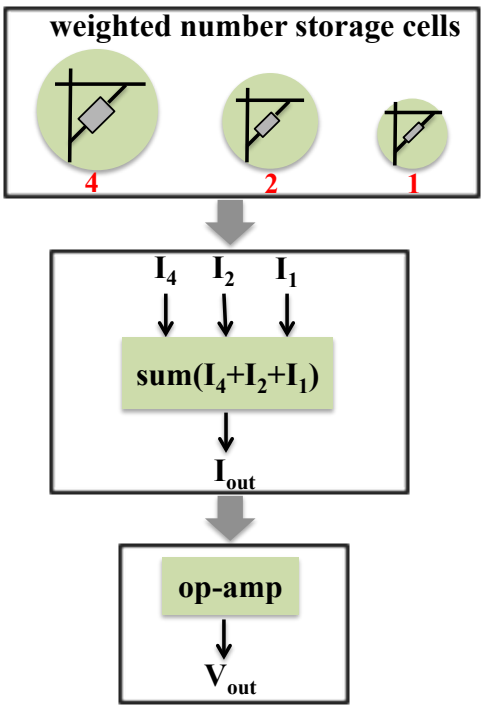

Figure 6: Weighted memory storage and the corresponding readout scheme.

3.2.3 Weighted Sensing Implementation Details

The weight assignment is the major difficulty in implementing the weighted sensing scheme. A straightforward solution in cross-point memory array is connecting ReRAM cells in parallel. As such, the whole memory array is divided into certain number of groups, each of which consists of different columns of memory cells. For instance, the example in Figure 7 has two groups: every two equivalent unit cells in the first group are connected in parallel while the second group has only one unit cell. Therefore, the two equivalent unit cells in the first group demonstrate an overall resistance of either $\mathrm{R_{on}}/2$ or $\mathrm{R_{off}}/2$ while the resistance of a cell in the second group shall be either $\mathrm{R_{on}}$ or $\mathrm{R_{off}}$. In this way, we assign the weights 1 and 2 to the two groups, respectively. The four logic combinations of the two group therefore corresponds to four different resistance levels: (‘00’ $\leftrightarrow 1.5\mathrm{R_{off}}$), (‘01’ $\leftrightarrow 0.5\mathrm{R_{off}} + \mathrm{R_{on}}$), (‘10’ $\leftrightarrow \mathrm{R_{off}} + 0.5\mathrm{R_{on}}$), and (‘11’ $\leftrightarrow 1.5\mathrm{R_{on}}$). The data pattern in different groups can be successfully read out by identifying the output signal corresponding to the different states. Notably, the write operation of the proposed weighted sensing scheme remains the same as that of the conventional ReRAM cross-point array.

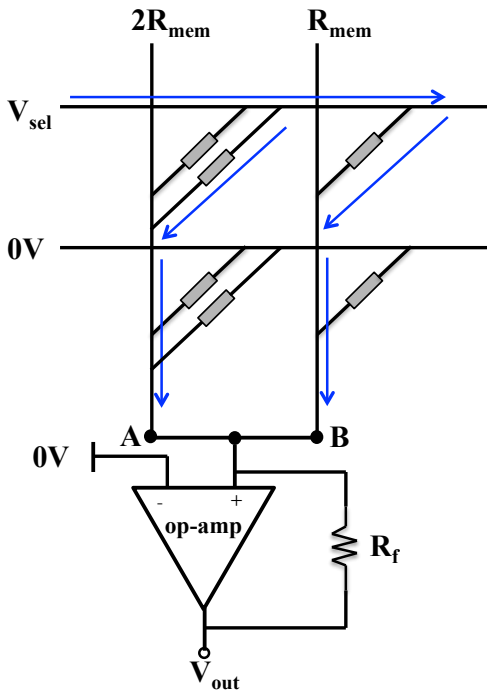

Figure 7: An example schematic of weighted sensing in cross-point array.

Compared to the conventional ReRAM cross-point array and peripheral circuitry design, more ReRAM columns are used in weighted sensing scheme and hence its array size increases along the wordline direction. Moreover, the expansion of memory array is also determined by the weighting granularity. Theoretically, n-bit data with weights of $1, 2, 4, 8, \cdots, 2^n$ can be accessed simultaneously. However, the design shall guarantee that each bit of data can be differentiated successfully. In other words, design trade-off exists among the array size, the sensing circuitry complexity and area, and the data access bandwidth. More detailed discussion shall be found in Section IV.

The adoption of op-amp holds the voltages of bitlines (*i.e.*, ports A and B in Figure 7) close to zero, that is, the voltage drop across unselected cells keeps approximate zero and the sneak path leakage is under well control. Summation of the two weighted cells can be naturally integrated with the op-amp design as illustrated in Figure 7. In the proposed design, we attach only one op-amp to a cross-point array to reduce the overhead of area and power consumption. The array is then divided into several sets of weighting groups and only one set can be read out during each read access.

3.3 RESULTS AND DISCUSSION

3.3.1 Effectiveness of Weighted Sensing

We first investigate the impact of data pattern and array size on the cross-point array design with the weighted sensing scheme. The weight assignment illustrated in Figure 7 is adopted in the simulation. Here, we utilize the same array configuration and ReRAM device parameters in Section III.*A*. Again, the farthest group from the wordline driver with the most significant voltage degradation is selected. During a read operation, the wordline voltage to the selected row is set to $0.5\mathrm{V}$ and the wordlines of the unselected rows are tied to ground. The selection of the op-amp parameters $\mathrm{R_f}$ and Gain (see Figure 7) determines the amplification factor and affects the output voltage. To match better with $\mathrm{R_{on}}$, $\mathrm{R_f}$ is set as $5\mathrm{K}\Omega$ and Gain is set to be 10^6.

Figure 8 shows the output signal of the cross-point array after applying the weighted sensing circuit. Four different resistance states correspond to four different output signal levels that can be easily differentiated. It means that all data information can be read out correctly in the weighted sensing scheme. Moreover, the output signals do not degrade with the increase of memory array size.

Figure 9 shows the output voltage swing ΔV after applying the weighted sensing circuit. In contrast to the large ΔV degradation occurred in traditional cross-point array shown in Figure 5, ΔV in the proposed design doesn't decrease much as array size increases. The difference between ΔV's obtained from the `ALL ONEs` and `ALL ZEROs` data patterns is negligible. The result approves that *the adoption of op-amp can efficiently reduce the sneak path leakage and minimize the impact of array size and data patterns*.

3.3.2 Dependence of Op-Amp Gain Parameter

Considering that the sensing accuracy is greatly affected by op-amp Gain parameter, we investigate the impact of Gain parameter under different array sizes. In the evaluation, we use the weight groups of 1 and 2 illustrated in Figure 6. And the farthest group from the wordline driver representing the worst-case scenario is selected.

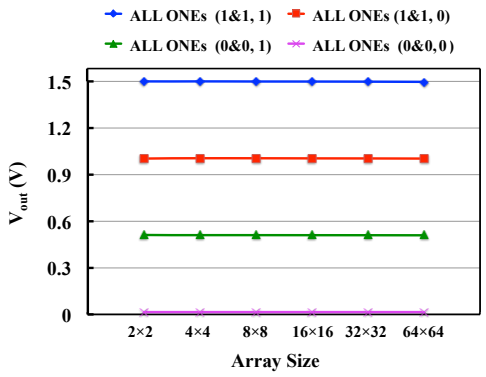

Figure 8: The output signal of the cross-point array with the weighted sensing ($\mathrm{Gain} = 10^6$).

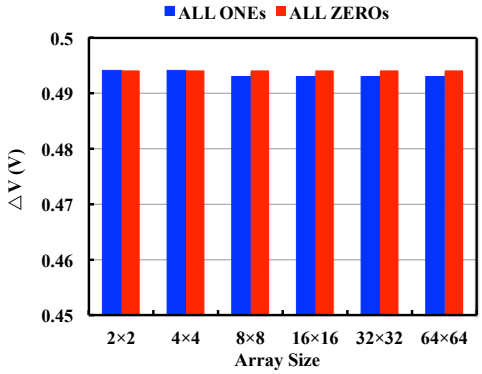

Figure 9: Output voltage swing ΔV of the cross-point array with the weighted sensing.

Figure 10 shows the output signal $\mathrm{V_{out}}$ when decreasing the Gain parameter from 10^6 to 10^3. Comparing to the results in Figure 8, an obvious drop at $\mathrm{V_{out}}$ with the increase of array size can be observed in Figure 10. Particularly, the output voltage degrades greatly when the data pair is '11'.

The major reason is that the virtual ground at bitline (i.e., ports A and B in Figure 7) cannot be well maintained when the Gain parameter is not large enough. The non-zero voltage drop across unselected cells and the sneak path leakage through these cells cannot be ignored.

Moreover, we summarize the relation of the sensing accuracy and the op-amp Gain parameter in Figure 11. Here, the sensing accuracy is defined as the ratio of a real $\mathrm{V_{out}}$ over its ideal value that is obtained by assuming an ideal op-amp and an ideal cross-point array without sneak path leakage. The simulation results demonstrate significant degradation of the sensing accuracy as

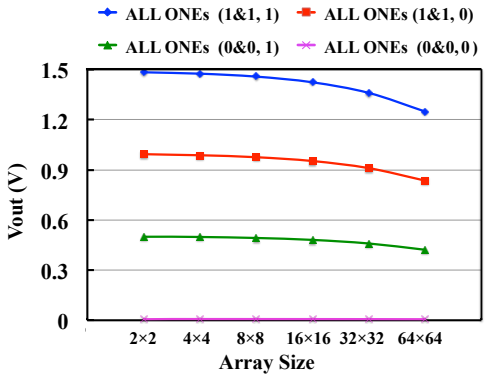

Figure 10: The output signal of the cross-point array at $\mathrm{Gain} = 10^3$.

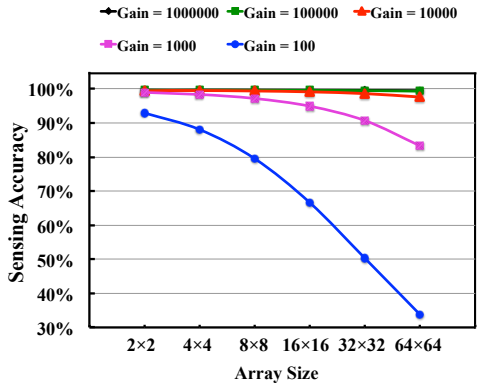

Figure 11: The relation of the sensing accuracy and the op-amp Gain.

Gain parameter reduces from 10^6 to 10^2. More specific, op-amp with $< 10^3$ Gain is not acceptable in cross-point array design for the poor performance. On the one hand, op-amps with higher Gain helps reduce sneak path leakage and therefore increase crossbar array size. On the other hand, both the induced area cost and the required sensing time increase dramatically.

3.3.3 Impact of Device Resistance Ratio

The resistance ratio of ReRAM device (i.e., $\mathrm{R_{off}}/\mathrm{R_{on}}$) also affects the output signal of cross-point array. In general, a smaller $\mathrm{R_{off}}/\mathrm{R_{on}}$ indicates smaller difference between HRS and LRS states, making data differentiation more difficult. We examine the trend of the output voltage swing ΔV of cross-point arrays as $\mathrm{R_{off}}/\mathrm{R_{on}}$ changes from 100 to 5. Here, $\mathrm{R_{off}}$ is fixed at $1\mathrm{M}\Omega$ while $\mathrm{R_{on}}$ varies from $10\mathrm{K}\Omega$ to $200\mathrm{K}\Omega$.

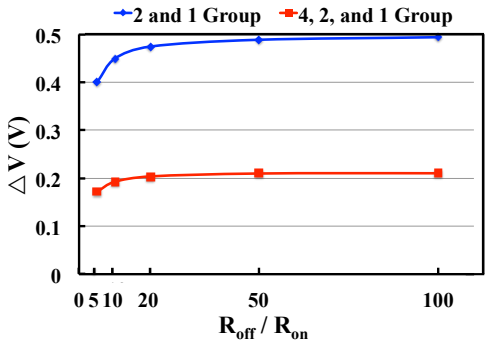

Figure 12: The dependence of ΔV on $\mathrm{R_{off}}/\mathrm{R_{on}}$ ratio. Here, $\mathrm{R_{off}}$ is fixed at $1\mathrm{M}\Omega$, and $\mathrm{R_{on}}$ varies from $10\mathrm{K}\Omega$ to $200\mathrm{K}\Omega$.

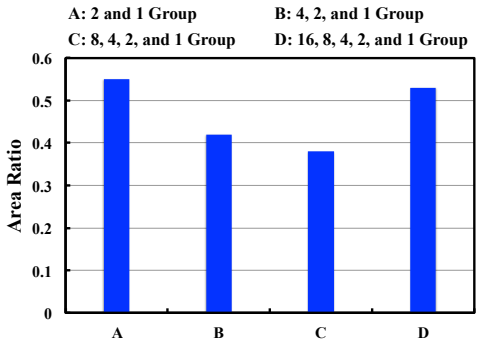

Figure 13: The estimated areas of 512×512 ReRAM cross-point arrays together with periphery circuit, under different weight assignments.

The simulation results are shown in Figure 12. The curve labeled with '2 and 1 group' corresponds to the results of the memory array design with the divided group data pattern of 2 and 1, the same as the example in Figure 7. Similarly, the curve labeled with '4, 2, and 1 group' represents the results after partitioning an array into three group patterns with weights of 4, 2, and 1, respectively. Simulation results show that ΔV remains sufficiently large for correct function as far as the ratio is larger than 5. When $\mathrm{R_{off}}/\mathrm{R_{on}} > 10$, ΔV changes slightly with $\mathrm{R_{off}}/\mathrm{R_{on}}$ ratio. Whereas, a large degradation on ΔV can be observed as $\mathrm{R_{off}}/\mathrm{R_{on}}$ becomes less than 10. The results also show that the cross-point array design with three divided group data patterns (4, 2, and 1) has smaller output voltage swings. This is because the design has eight different output signal levels corresponding to eight resistance states while its sensing amplification of the op-amp remains the same as the one utilized in the arrays with group data patterns. Selecting a large $\mathrm{R_f}$ to produce a larger output voltage swing can alleviate the situation and enlarge ΔV.

3.3.4 Area Reduction of Peripheral Circuitry

A motivation of this work is to decrease the number of op-amps and reduce its area overhead. The proposed weighted sensing scheme indeed tends to balance the areas of the cross-point array and its peripheral circuitry and obtain the best tradeoff. Figure 13 shows the estimated areas of 512×512 ReRAM cross-point arrays under different weight assignments, at $40\mathrm{nm}$ technology node. Here, we assume an op-amp area of $50\mu\mathrm{m}^2$ [41]. All the results are normalized to the area of the ReRAM cross-point design by Qureshi *et al*. [37].

Four types of group data patterns in the weighted memory were studied. The results shows nearly half of area can be saved by the weighted sensing scheme. As the group size in the weighted memory array increases, the area ratio decreases first and then increases again. This is because that the weighted memory grows in an exponential rate. When a large group size (*i.e.*, '16, 8, 4, 2, and 1 group') is applied, the overhead of the weighted memory will overcome the saving of less op-amps used in peripheral circuit. In summary, '8, 4, 2, and 1 group' demonstrates the lowest overall area including both array and peripheral circuity.

3.4 SUMMARY

The ReRAM based cross-point array has become one of the most promising candidate of non-volatile memory technology below $20nm$ technology node. However, the sneak path leakage issue remains unsolved and significantly limits its application. In this work, we propose a weighed sensing scheme. The design utilizes op-amp in read circuitry to well control the bitline voltage and suppress the impact of sneak paths. Furthermore, by assigning different weights to different data bits, these bits can share one op-amp for data detection. The area consumption is saved nearly a half in the weighted cross-point memory array. We thoroughly analyzed the effectiveness of the proposed weighted sensing scheme.

4.0 A SPIKING NEUROMORPHIC DESIGN WITH RESISTIVE CROSSBAR

In this chapter, we will present the details of the proposed spiking neuromorphic design with resistive crossbar. The structure of this chapter is organized as the follows: Section 4.1 gives the background of present neuromorphic system studies; Section 4.2 presents the computation method of the proposed spiking neuromorphic design; Section 4.3 presents the details of the hardware implementation and the computation accuracy analysis; Section 5.3.2 evaluates the performance and robustness of the proposed spiking neuromorphic design by using the application of digital image recognition, and compares the proposed design with previous approaches; Section 5.4 demonstrates the chip design of feedforward neural networks implementation basing on the spiking design; At last, we summary the work in this chapter in Section 4.6.

4.1 NEUROMORHIC COMPUTING BACKGROUND AND RELATED WORKS

4.1.1 Neuromorhic Computing Systems in Traditional CMOS Domain

In spite of the scaling of device size and the rising of clock frequency,"memory wall" phenomenon [8], i.e., the increasing gap between microprocessor performance and memory bandwidth, severely hinders the performance improvement of computing systems. The traditional von Neumann architecture that computes and stores data in separated locations becomes inefficient to many cognitive applications involving a large amount of data processing [42]. Neuromorphic computing systems inspired by the working mechanism of human brains are normally very efficient in their computation and data communication. For example, *TrueNorth*—the latest spike-timing-based neuromorphic hardware prototyped by IBM, achieved only $70mW$ whole-chip power consumption as well as $45pJ$-per-spike on-chip data transferring [14]; [15]. SRAMs, capacitors, and full custom analog and digital circuit modules have been developed to mimic biological synapses [9]; [10]; [11].

4.1.2 Neuromorhic Computing Systems with Resistive Crossbar

The high implementation cost of these designs is the major obstacle for their applications and continuous upscaling. More importantly, the two-terminal structure of a ReRAM device enables the construction of an ultra-dense crossbar array, in which a ReRAM cell is allocated at each crosspoint of horizontal and vertical metal wires. As illustrated in Figure 14, a crossbar array is particularly attractive for implementation of synaptic connections because of its similarity to the connection matrix in neural network models. By adjusting the amplitude and pulse width of programming voltage/current, a ReRAM device can be programmed to multiple discrete resistance levels or continuous resistance states [43]; [44] to represent a synaptic weight in neural network models.

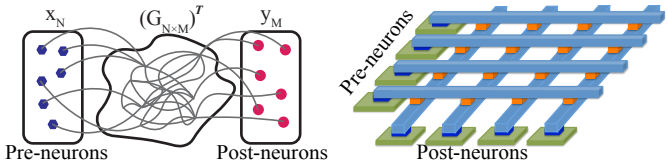

Figure 14: An abstract neural network model and a ReRAM crossbar array.

The applications of resistive crossbar arrays in acceleration of scientific and neuromorphic computing have been studied [45]; [46]. For example, memristor crossbar was used to implement matrix-vector computation [3]. The design uses voltage and current magnitudes to represent the data that participate in the computation, so it is generally susceptible to the signal noises and distortions. Moreover, the AD/DA conversions induce high design complexity as well as significant area and energy overheads. Recently, a spiking-based neuromorphic computing system was demonstrated for pattern recognition [47] and believed to have better tolerance to the signal noise. However, it requires complicated neuron circuits and can run only at an extremely low speed (e.g., ten operations per second).

4.2 DESIGN METHODOLOGY

Figure 15 depicts an overview of our proposed spiking computing architecture that leverages the compact resistive crossbar structure. The design adopts the rate coding model and represents data using the frequency of spikes (pulses) [48]. Through different bitlines (BLs) in a resistive crossbar

array, the synaptic weighting functions of different entries are executed in parallel. The *integrate and fire circuits* (IFC) as post-neurons generate output spikes based on the strength of the weighted pre-neuron signals from the crossbar.

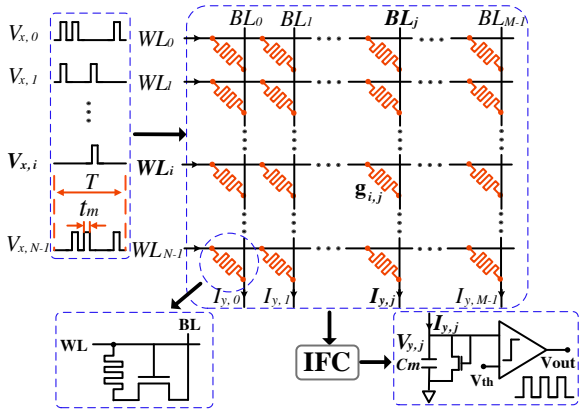

Figure 15: An overview of the spiking neuromorphic design with a resistive crossbar array.

A single-layer neural network with N pre-neurons and M post-neurons (Figure 14) can be implemented using a $N \times M$ resistive crossbar array in the following approach: First, the activity pattern of pre-neurons $\mathbf{x_{N\times 1}}$ is transferred into a set of pulses to wordlines (WLs). Here we assume the duration of an input pulse is t_m. The number of spikes on WL_i within a computation period T (that is, $n_{x,i}$) is determined by $x_i \in \mathbf{x}$. The synaptic weight between the j^{th} pre-neuron and the i^{th} post-neuron is mapped to conductance g_{ij} at the crosspoint of WL_i and BL_j. Thus, $\mathbf{G_{N\times M}}^T$ is constructed as the *connection matrix* of the network. The total weighted signal to post-neuron j is transferred to the current flowing through BL_j and accumulated on a capacitor C_m in IFC. Once the voltage on C_m reaches to a predefined threshold V_{th}, the IFC fires an output spike and resets C_m. The activity function of post-neurons $\mathbf{y}_{M\times 1}$ is represented by a set of spike numbers such as $[n_{y,0}, n_{y,1}, \cdots, n_{y,M-1}]^T$.

We use $V_{x,i}(t)$ and $V_{y,j}(t)$ to denote the voltages on WL_i and BL_j at time t, respectively. The current flows through all the connected resistive devices contributes to the total current on BL_j, such as

$$I_{y,j}(t) = \sum_{i=0}^{N-1} g_{ij} \left[V_{x,i}(t) - V_{y,j}(t)\right]. \tag{4.1}$$

On the other hand, the voltage across C_m and the current flowing through it also follows

$$I_{y,j}(t) = C_m \frac{dV_{y,j}(t)}{dt}. \tag{4.2}$$

By combining Eqs. (4.1) and (4.2), the increase of $V_{y,j}$ within a small epoch at time t can be derived as

$$\frac{dV_{y,j}(t)}{dt} = \frac{[1 - e^{-\frac{1}{C_m}\sum_{i=0}^{N-1} g_{ij}}]\sum_{i=0}^{N-1} g_{ij}V_{x,i}(t)}{\sum_{i=0}^{N-1} g_{ij}}. \tag{4.3}$$

When $\sum_{i=0}^{N-1} g_{ij} \to 0$, Eq. (4.3) can be approximated as

$$\frac{dV_{y,j}(t)}{dt} \approx \frac{1}{C_m}\sum_{i=0}^{N-1} g_{ij}V_{x,i}(t). \tag{4.4}$$

indicating that the change of $V_{y,j}(t)$ is approximately proportional to the weighted pre-neuron signals $\sum_{i=0}^{N-1} g_{ij} \cdot V_{x,i}(t)$. Moreover, the IFC fires a spike whenever $V_{y,j}$ reaches V_{th}. Thus, the spike number produced at post-neuron$_j$ is

$$n_{y,j}(t) \propto \int_{\tau=0}^{t}\sum_{i=0}^{N-1} g_{ij}V_{x,i}(\tau)d\tau. \tag{4.5}$$

Eq. (4.5) implies that the computation of connection matrix in neural network can be performed by resistive crossbar array using spike signals.

4.3 HARDWARE IMPLEMENTATION

Note that the analysis in Section 4.2 does not take into account the realistic factors in circuit implementations. First, the approximation of Eq. (4.4) is based on the assumption of $\sum_{i=0}^{N-1} g_{ij} \to 0$, which is satisfied only when all the resistive devices are at (or close to) the high resistance state. This cannot be generalized as a common condition in applications. Moreover, the delay overhead of IFC to generate pulses and reset C_m cannot be ignored. We implemented the spiking neuromorphic design with a 32×32 crossbar at IBM 130nm technology. The detailed design considerations of two key components—crossbar array and IFC—are presented. The computation accuracy of the design is also analyzed and verified through circuit simulations in the Cadence Virtuoso environment.

4.3.1 1T1R Crossbar Array

Sneak path is one of the major concerns in resistive crossbar array design [49, 50]. It refers to the intrinsic leakage flowing through unselected cells, which degrades the programming efficiency and the data detection accuracy. In fact, Eq. (4.3) demonstrates the impact of sneak paths: the accumulated weighted pre-neuron signals represented by term $\sum_{i=0}^{N-1} g_{ij} V_{x,i}(t)$ is shared by all the connected devices, i.e., $\sum_{i=0}^{N-1} g_{ij}$. In the case of only a few pre-neuron spikes to WLs, a large portion of the accumulated weighted pre-neuron signals could be consumed on unselected devices, deteriorating the computing accuracy.

Similar to many prior arts [51, 52], our analysis also indicates that a better access control mechanism, e.g., introducing a transistor or a selector to control the access of resistive devices, is necessary. For instance, when adopting the one-transistor-one-resistive device (1T1R) design as illustrated in Figure 15, cell_{ij} contributes a current to BL_j only when an input pulse is supplied to WL_i. The effective conductance of the cell_{ij} becomes $\tilde{g}_{ij} = g_{ij} // g_{on}$ where g_{on} is the conductance of the access transistor at ON state. When WL_i is grounded, the transistor is turned off and its extremely small conductance g_{off} causes $\tilde{g}_{ij} \to 0$. For ease of explanation, we introduce a new parameter δ_i to indicate if a spike occurs at WL_i ($\delta_i = 1$) or not ($\delta_i = 0$). Eq. (4.3) becomes

$$\begin{aligned} \Delta V_{y,j}(\Delta t)|_t &= \frac{V_I[1 - e^{-\frac{\Delta t}{C_m}\sum_{i=0}^{N-1}\tilde{g}_{ij}\delta_i}]\sum_{i=0}^{N-1}\tilde{g}_{ij}\delta_i}{\sum_{i=0}^{N-1}\tilde{g}_{ij}\delta_i} \\ &= V_I\left[1 - e^{-\frac{\Delta t}{C_m}\sum_{i=0}^{N-1}\tilde{g}_{ij}\delta_i}\right] \end{aligned} . \tag{4.6}$$

Here, V_I is the voltage amplitude of the pulses on WL_i. Eq. (4.6) shows that the 1T1R structure helps minimize the impact of sneak paths. $dV_{y,j}(t)/dt$, however, no longer follows a strict linear relation with the sum of the weighted pre-neuron signals, i.e., $\sum_{i=0}^{N-1}\tilde{g}_{ij}\delta_i$. This can be easily explained by Eqs. (4.1) and (4.2): as $V_{y,j}$ increases, $I_{y,j}$ gradually reduces, resulting in the degradation in $dV_{y,j}(t)/dt$.

In the work, we set the resistance range of ReRAM devices from $50K\Omega$ to $1M\Omega$ [43, 53]. The according conductance $g \in (1\mu S, 20\mu S)$. To suppress the impact of sneak paths, we chose NMOS transistor to control cell access and used the 1T1R cell structure to replace the resistive device, as shown in Figure 15.

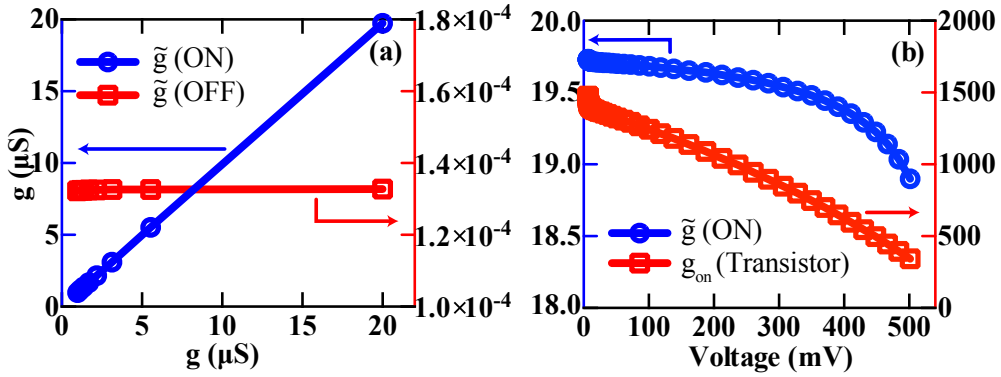

Figure 16: (a) $\tilde{g}_{ij}$ vs. g_{ij} at ON and OFF states. (b) The change of $\tilde{g}_{ij}$ and g_{on} as V_j varies.

Figure 16(a) compares the effective cell conductance $\tilde{g}$ and the ReRAM device conductance g. The result shows that they are very close when the select transistor is on. This is because the transistor's conductance g_{on} (at the order of mS) is much higher than g. When the transistor is turned off, the extremely small g_{off} ($\sim nS$) dominates the cell conductance. So $\tilde{g}$ of an OFF cell has negligible impact on the computation. Therefore, a BL current virtually comes only from those ON cells enabled by WL pulses and the computing on every BL is nearly independent from others.

We also investigated the relationship between $\tilde{g}$ and the BL voltage V_y when the cell is on. Figure 16(b) presents the result when $g = 20\mu S$. As V_y increases, both V_{GS} and V_{DS} of the select transistor decrease, leading to the reduction of g_{on}. Because the ReRAM device and the transistor in a cell are connected in series and g is much smaller than g_{on}, $\tilde{g}$ is primarily determined by g. The reduction of g_{on} causes only 4.0% in the change of $\tilde{g}$. As g decreases, the variance of $\tilde{g}$ induced by the change of V_y becomes even less significant.

4.3.2 Integrate-and-fire Circuit (IFC)

We observed that the delay of IFC is a critical parameter determining the performance of the spiking neuromorphic system. Let's set $k = V_{th}/V_I$. Then the time duration to switch $V_{y,j}$ from $0V$ to V_{th} can be derived by:

$$\Delta\tau = \frac{-C_m ln(1-k)}{\sum_{i=0}^{N-1} \tilde{g}_{ij}\delta_i}. \tag{4.7}$$

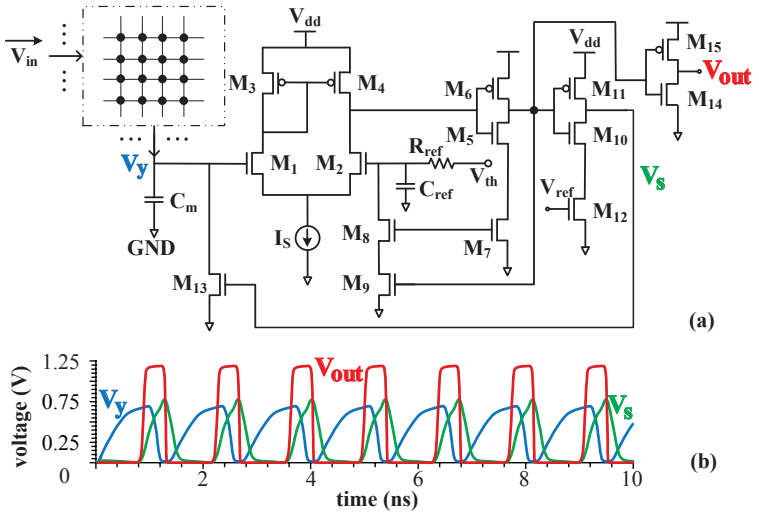

Figure 17: The integrate and fire circuit (a) the schematic and (b) the simulation waveforms.

The number of output pulses generated during an input pulse duration of t_m can be calculated by

$$n_{y,j} = \frac{t_m}{\Delta\tau + t_0} = \frac{t_m}{\frac{\alpha}{\sum_{i=0}^{N-1} \tilde{g}_{ij}\delta_i} + t_0}, \tag{4.8}$$

where t_0 is the delay overhead of the IFC. $\alpha = -C_m \cdot ln(1-k)$ represents the BL charging efficiency, which is determined by the integration capacitor C_m and the threshold voltage V_{th} of the IFC design.

We proposed a new IFC design featuring high speed and low power consumption. Figure 17(a) depicts its schematic. During the operation, the BL voltage V_y continues increasing until it reaches V_{th}. Then the differential pair (M_1–M_4) together with the following two cascaded inverters (M_5–M_7 & M_{10}–M_{12}) generates a high voltage at V_s, which in turn enables the discharging transistor M_{13}. Consequently, V_y decreases quickly and eventually turns off M_{13}. As such, the firing of one output spike at V_{out} is completed and a new iteration of integrate-and-fire starts.

To improve the IFC throughput, we tended to reduce its intrinsic operation delay and make it shorter than the integrating time $\Delta\tau$ in Eq. (4.7). A positive feedback loop (M_7–M_9) was deployed based on the traditional comparator for this purpose. Another approach was to minimize the discharge time of C_m once a spike is fired out, i.e., using a large M_{13} to provide sufficient discharging current.

We implemented and simulated the IFC design with IBM 130nm technology. V_I is set to $1.2V$, and V_{th} is set to $0.5V$ in which the system shows best computation accuracy. A MIM capacitor with a capacitance of $153fF$ (which is the minimum value offered by the PDK) is used as C_m. The deign parameters were carefully selected so that the intrinsic delay of the integrate-and-fire is shorter than even the minimum BL integrating time. Also, it will achieve fast output spikes if the frequency of which is still within the range that can be reliably captured by the sensing circuit. The waveforms of V_y, V_s, and V_{out} under the fastest firing frequency (568.2M spikes/sec) is shown in Figure 17(b).

The area of the IFC design at IBM 130nm technology is $175.3\mu m^2$, which is compatible to that of traditional designs, e.g., $120\mu m^2$ at 65nm technology in [54]. The energy consumption of our design is $0.48pJ$-per-spike, which is about a quarter of the one in [54] ($2pJ$-per-spike).

4.3.3 Computational Accuracy Analysis

The linearity between the obtained output spike number $n_{y,j}$ and actual computation on the crossbar $\sum_{i=0}^{N-1} g_{ij}\delta_i$ defines the computational accuracy of the neuromorphic system. We investigated the computational accuracy of our design based on a 32×32 crossbar array. Assume that an input spike has a $2ns$ period with 50% utilization rate (that is, $t_m = 1ns$). The resistance values and the input pulse numbers are randomly assigned to cover the entire range of $\sum_{i=0}^{N-1} g_{ij}\delta_i$. To examine the temporal scalability of the design, we conducted the simulations by varying T from $10ns$ to $80ns$ at a step of $10ns$. For better illustration, we present only four groups of results in Figure 18. All the eight sets of simulations well match the theoretical calculation from Eq. (4.8). We observed that the rising rate of $n_{y,j}$ becomes smaller as $\sum_{i=0}^{N-1} g_{ij}\delta_i$ increases. This is because a larger $\sum_{i=0}^{N-1} g_{ij}\delta_i$ and therefore a bigger $I_{y,j}$ results in a shorter $\Delta\tau$, making the impact of the IFC delay overhead t_0 more prominent (refer Eq. (4.8)). Nonetheless, a good computational accuracy (i.e., output linearity) is obtained when $\sum_{i=0}^{N-1} g_{ij}\delta_i$ is small (i.e., $< 0.15mS$). In fact, most of the operations of our neural network implementations in Section 5.3.2 fall into this small range. We also note that for different combinations of inputs and resistive array patterns with the same $\sum_{i=0}^{N-1} g_{ij}\delta_i$, the generated pulse number may be slightly different (no more than ± 1). Such a fluctuation comes from the difference in $I_{y,j}$'s waveform and amplitude generated by these combinations.

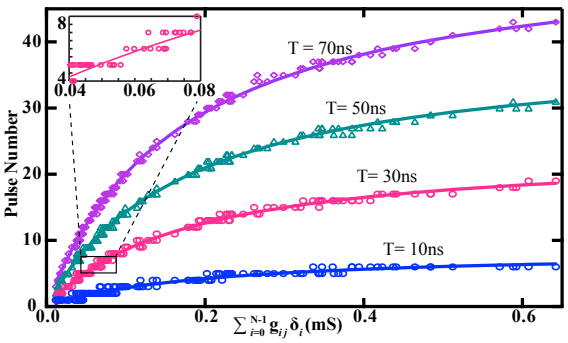

Figure 18: $n_{y,j}$ vs. $\sum_{i=0}^{N-1} g_{ij}\delta_i$ under various T.

Our proposed spiking neuromorphic system is designed mainly for learning and classification applications whose algorithms naturally tolerate the low resolution and variability in the computations. Moreover, the imperfect output linearity shown in Figure 18 may be also compensated during circuit implementation as long as the output spike and the weighted input spikes have a monolithic mapping relation.

Figure 19: (a) The standard training set of six patterns. (b) An example of noise pattern "5" under different single bit error rate.

4.4 EVALUATION IN APPLICATIONS

4.4.1 Experimental Setup

We evaluated the performance and robustness of the proposed spiking neuromorphic design by using the application of digital image recognition. Since the simulation time increases dramatically with the design scale, the crossbar with 32 rows was selected for training and recognition of images with 32 pixels. Six images corresponding to number $0 \sim 5$ in Figure 45(a) were used as the standard training set.

We implemented two typical neural network models: the *feedforward* network ("**F**") that has been widely utilized in approximate computing and the *Hopfiled* network ("**H**") to represent the popular recurrent systems. In both implementations, the back-propagation and delta rule [3] were adopted to perform training and programmed ReRAM devices to particular resistance states. Double crossbar array is adopted to obtain negative weights [3]. The 1T1R structure in the design can effectively suppress the sneak path leakage and therefore make programming very efficient.

The system performance was measured by the probability of failed recognition (P_F). The analysis was conducted based on Monte-Carlo simulations (i.e., 10,000 simulations per configuration). Noise patterns with a certain bit error rate (BER) were generated and used as the testing images. The example in Figure 45(b) shows that visually, it is already very difficult to identify an image with a BER $>9\%$.

In the work, we are particularly interested in the impacts of physical constraints, including the limited available resistance state levels and output spike number. We evaluated and compared the designs of which the ReRAM devices provide the continuous analog resistance states ("**A**") or only 8 discrete resistance levels ("**D**"). The efficiency of the digitized output spikes was studied by comparing the result obtained directly from the analog BL current ("**C**") and that achieved based on real output pulses ("**P**"). Thus, the configuration "**AC**" performs closely to the mathematical neural network model and was taken as the baseline in the following evaluations. The configuration "**DP**" corresponds to our proposed spiking neuromorphic design. Moreover, for comparison purpose, we also implemented the previous memristor crossbar-based computing engine ("**[3]**") which utilizes voltage amplitude to differentiate the data value [3]. Here, we assumed 8 discrete resistance levels of ReRAM devices and $20mV$ sensing margin of BL output op-amp design.

The reliability analysis was conducted by assuming that the ReRAM resistances ("**PV**") and the IFC spike generation speed ("**IFC**") follow normal distributions with a standard deviation of 10% and 5%, respectively. In Monte-Carlo simulations, the resistances of the crossbar array were fixed as they are determined by the offline training and process variations. The IFC performance was changed on-the-fly because it is affected by the signal fluctuation during the execution.

4.4.2 Feedforward Network Implementation

We realized a 1-layer feedforward network based on the scheme in Figure 15 for image recognition. It maps an input pattern to the output through a direct graph without iterations. A 32×6 crossbar array was trained so that output j has the strongest response to number j of the six training patterns. Figure 20 gives the simulation result when any standard pattern is used as the input. Output j generates the biggest spike number to pattern j but demonstrates much weaker responses to other patterns. Here, we say the recognition of a noisy testing image is *failed* when the corresponding output doesn't produce the most spikes or another output has the same number of spikes.

The computation period T selection: As shown in Figure 18, T determines the output spike granularity and hence greatly affects the system performance. We investigated the performance of FDP configuration by varying T from $10ns$ to $80ns$. Figure 21(a) presents the results expressed by the average failure rate of all the patterns ($P_{F,all}$) under different BERs. The results show that when T is less than $20ns$, the design cannot produce enough output spikes to differentiate the top two strongest outputs, resulting in a lot of failures. $P_{F,all}$ quickly drops to 5.57% when increasing T to

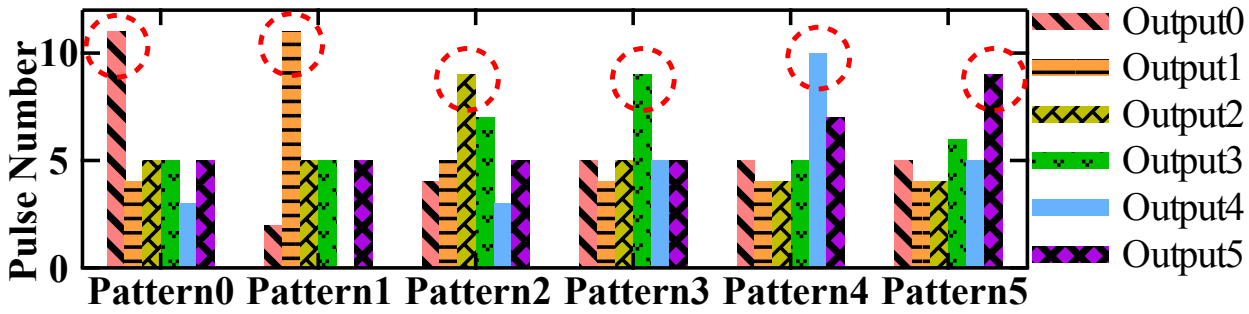

Figure 20: Pattern recognition result in forward neural network in ideal condition.

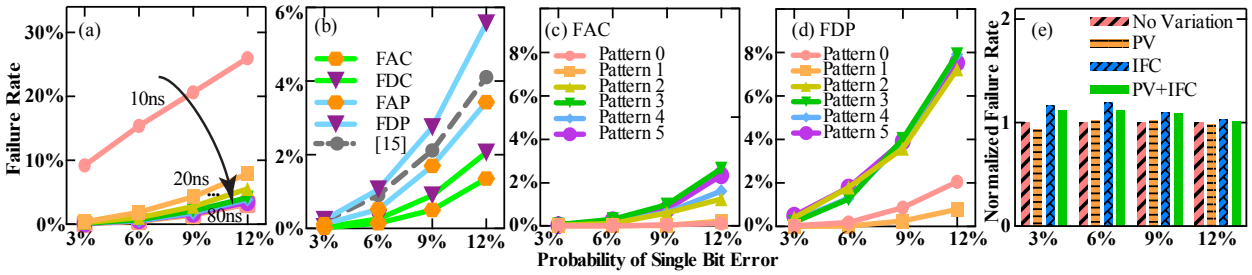

Figure 21: The simulations results of the feedforward implementation. (a) $P_{F,all}$ of FDP at various T's; (b) $P_{F,all}$ under different configurations; (c) $P_{F,ind}$ of FAC; (d) $P_{F,ind}$ of FDP; (e) The normalized $P_{F,all}$ of FDP after considering process variations and IFC fluctuation.

$30ns$. Further prolonging T demonstrates marginal improvement. To guarantee sufficient system performance, $T = 30ns$ was selected, corresponding up to 15 input pulses per computation period.

The impact of physical constraints was evaluated by comparing the recognition qualities of different configurations shown in Figure 21(b). FAC as the baseline has the least failures. Reducing the resistance states to 8 discrete levels inevitably results in quality loss when mapping the analog values of a connection matrix to the limited conductances in a crossbar. Compared to FAC, $P_{F,all}$ of FDC increases 0.71% at BER=12%. Changing from the analog BL current to the digitalized spikes (FAC vs. FAP) causes up to 2.08% more recognition failures, implying that this feedforward network implementation is more sensitive to the granularity of output signals. Overall, our proposed FDP obtains a $P_{F,all}$ of 5.57% at BER=12%, which is 4.21% higher than the baseline FAC and 1.46% worse than the computing engineer of [3]. Figure 21(c,d) show the statistical results of each individual pattern ($P_{F,ind}$) of FAC and FDP, respectively. Numbers 2$\sim$5 with high similarity in training patterns are more sensitive to the input defects during testing. The probability of failures is much smaller for numbers 0 and 1.

The system reliability was conducted by including the variations in ReRAM resistances and the IFC spiking generation. Figure 21(e) shows the relative $P_{F,all}$ of FDP under different conditions, all of which are normalized to the ideal one without any variations. The variations in ReRAM resistances can barely affect the system performance because it is buried under the resistance offset caused by the mapping from connection matrix to crossbar array. The impact of the fluctuation in IFC spiking generation is more obvious. Even though, $P_{F,all}$ under the worst scenario is still $<$5.65%.

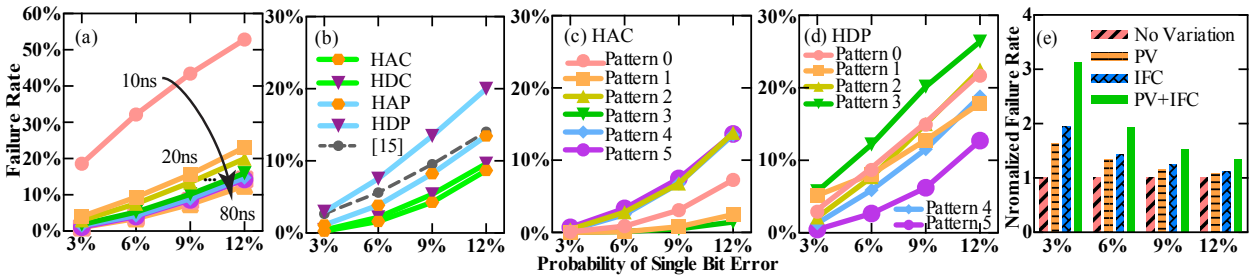

Figure 22: The simulations results of the Hopfield implementation. (a) $P_{F,all}$ at various T's; (b) $P_{F,all}$ under different configurations; (c) $P_{F,ind}$ of HAC configuration; (d) $P_{F,ind}$ of HDP configuration; (e) The relative $P_{F,all}$ after considering process variation and IFC fluctuation.

4.4.3 Hopfield Network Implementation

Hopfield network is a typical recurrent neural network, in which any two neurons are linked through a weighted connection. An input pattern distorted by noises goes through the network iteratively and gradually converge to a local minimum. In the implementation, we extended the crossbar size to 32×64 and used the combination of two columns to realize signed synaptic weights [3]. When a testing image cannot converge or the converged data does not match any training pattern, we say the recognition is *failed*. Note that this criterion is very restrict and therefore results in much higher failure rate compared to the feedforward design.

The computation period T selection: We investigated $P_{F,all}$ of HDP by changing T from $10ns$ to $80ns$ and showed the results in Figure 22(a). A significant reduction in $P_{F,all}$ can be clearly observed when raising T from $10ns$ to $30ns$. Further increasing T, however, cannot improve $P_{F,all}$ much. Thus, $T = 30ns$ is adopted in the following analysis.

The impact of physical constraints: Five configurations were utilized to analyze the impact of design constraints as shown in Figure 22(b). Compared to the feedforward design, the Hopfield implementation has much bigger $P_{F,all}$ due to the difference in failure evaluation criteria: the only requirement in the feedforward scheme is to distinguish the top two strongest outputs; while in Hopfield, a recognition is considered as failed even there is only one bit error between the output and its corresponding standard pattern.

Compared to the baseline HAC, constraining ReRAM resistance into 8 levels causes up to 0.92% increase in $P_{F,all}$ (HDC), while the digitalized pulse generation by IFC results in up to 4.77% degradation (HAP). The trend is very similar to that of the feedforward implementation while the impact on the Hopfield design is more severe due to the accumulation of errors in iterations. Overall, the maximum $P_{F,all}$ of HDP is 20.01%, which is 11.31% higher than the baseline HAC and 5.99% worse than the computing engineer of [3].

Figure 22(c,d) present the failure possibility of each individual pattern in HAC and HDP, respectively. The two configurations demonstrate different trends of $P_{F,ind}$'s. And all the $P_{F,ind}$'s in HDP grow rapidly as the BER of testing images increases. This is because a small network is utilized with a relative high capacity, which is defined as the ratio of the amount of training patterns over the pattern dimension, i.e., c=6/32=18.8%. It causes large overlaps of *the domain of attractions* of these training patterns [3]. Thus a testing image at the boundaries of two or multiple domains of attractions has a high possibility to converge into an unexpected local minimum, that is, a failure.

The system reliability: Here, we applied the similar reliability tests for the feedforward implementations. Figure 22(e) shows the statistical results from Monte Carlo simulations. All the them are normalized based on the $P_{F,all}$ of HDP without any variations. When BER is small and $P_{F,all}$ is low, these variations cause dramatic increase in $P_{F,all}$. As BER increases, the input defects dominate the failures and the impact of variations become less severe. After including all the variations, the maximal $P_{F,all}$ grows to 26.89%.

Table 1: System Evaluation and Comparison

NN	System	Area (mm^2)	Power (mW)	T (ns)	Energy (pJ)
Forward	This work	0.00135	0.577	30	17.32
	[3]	0.0042	2.1	16.6	34.86
	Diff.	-67.58%	-72.51%	+80.72%	-50.31%
Hopfield	This work	0.0144	5.054	30	343.60
	[3]	0.0387	22.4	16.6	842.59
	Diff.	-62.88%	-77.44%	+80.72%	-59.22%

4.4.4 Design Comparison

We compared our spiking neuromorphic design with the previous memristor crossbar-based computing engine [3]. Table 2 summarized the comparison for both feedforward and Hopfield implementations. The average power, latency, and energy were obtained based on the aforementioned application of image recognition. More specific, the computation period T represents the delay of the completed operation of the feedforward scheme or that of one iteration in the Hopfield network. The *energy consumption* denotes the total energy required to finish the entire recognition procedure so an average iteration number was taken to estimate the overall operation time in Hopfield implementation.

The feedforward and Hopfield schemes respectively adopt the 32×6 and 32×64 crossbar arrays. The key circuit components of our design include a crossbar array, WL drivers, and IFC. Besides a crossbar array, the computing engine in [3] also requires Op-amps, comparators, sample-and-hold (S&H) circuits, and analog-to-digital converters (ADCs) to detect, store, and transmit analog signals. We obtained the data from circuit simulations and estimated the design area based on the component layouts at IBM 130nm technology. We assumed that the extremely costly ADC is not necessary for the image recognition so it was not included into the design estimation for [3].

To provide sufficient output granularity, the operation of our spiking neuromorphic design is slower than the pure analogue execution in [3]. However, the digitalized interface in spikes naturally minimizes the use of analog components, making the proposed design a lot more attractive in area, power and energy consumptions, as shown in Table 2.

4.5 CHIP DESIGN

4.5.1 Two Designs on a Chip

Basing on the spiking neuromorphic computing engine proposed above, an on chip design was fulfilled. Two kinds of feedforward neural networks - single layer and two-layer for images classification are designed in hardware using full custom VLSI design flow. More specifically, there are four designs on a single chip.

- *Two single-layer designs* composing of 32×12 crossbar array for six patterns (black and white). The images are shown in Figure 23.

- *One two-layer design* composing of 64×40 and 20×10 crossbar arrays to test MNIST handwritten digit database that contain grey levels. The corresponding images are shown in Figure 24.

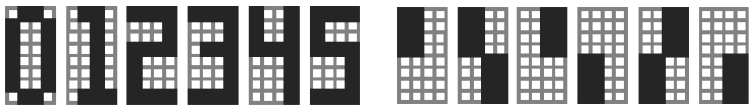

Figure 23: Images for classifying in the two single-layer design.

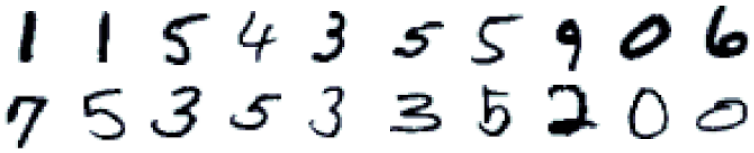

Figure 24: Images from MNIST database for classifying in the two-layer design.

The layout of the full chip and the pad ring arrangement is demonstrated in Figure 25 and Figure 26, respectively.

4.5.2 Hardcoded Cell Design

In this tape out, we adopted the hardcoded cell design - 1T1R (one transistor and one resistor) cell structure in the crossbar array design is adopted. As is demonstrated in Section 4.3, one transistor is connected series following with the resistor, and NMOS transistor is utilized to minimize the impact of sneak path leakage. Moreover, LVT transistor (low V_{th}) that offers higher drive ability is used in the on-chip design. The width of NMOS transistor is set to 2μm, considering the trade off between the voltage drop across the transistor and the cell size. In the implementation, eight hardcoded resistance levels are adopted, representing the 3-bits analog values of the resistive switching cell. The layout w/ diff. resistive values are depicted in Figure 27.

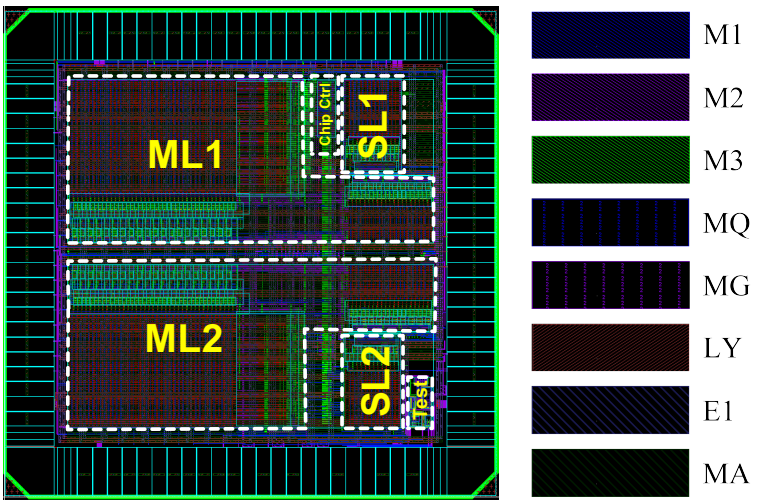

Figure 25: The full chip layout of the two designs on a chip.

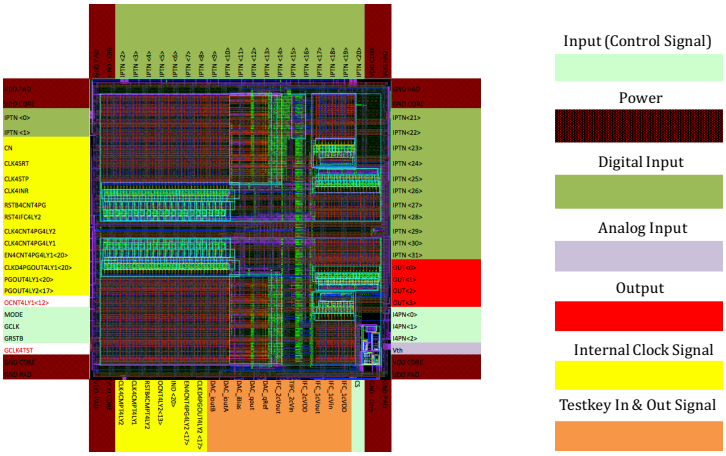

Figure 26: The pad ring arrangement on the chip.

4.5.3 Single Layer Feedforward System Design

4.5.3.1 Block Diagram Figure 28 shows the circuits block diagram of the single layer design. It can be divided to two major part: 1) Analog computation part: This part includes the resistive crossbar and the integrate and fire circuit (IFC) for analog computing. Digits inputs from the input driver are given to the crossbar array in parallel, and analog computation is fulfilled by the

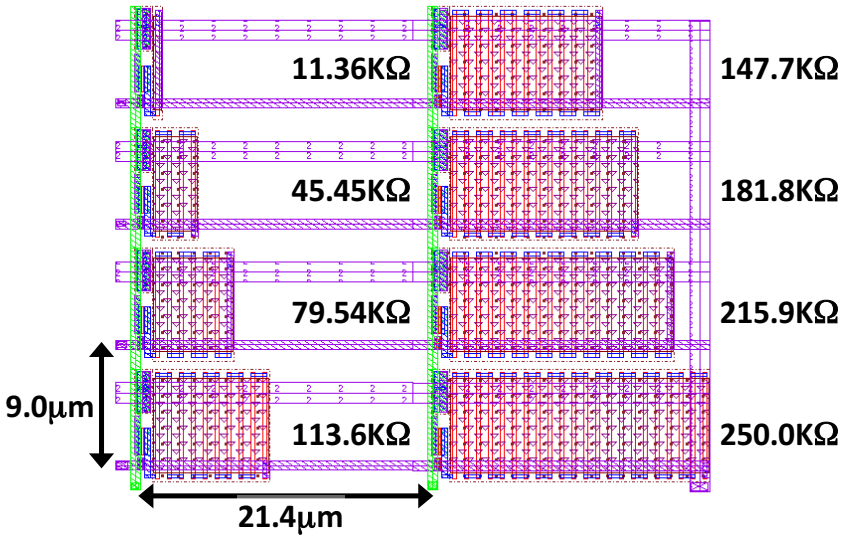

Figure 27: The layout w/ diff. resistive values in the hardcoded design.

crossbar and generate the sums-of-product results in current format. Then, the current output from each bitline of the crossbar are transferred to output spikes by the IFCs. 2) Digital computation part: There are three major parts for digital computation. One part is the circuits blocks generating input spikes from outside digital signals and providing enough drive abilities for the crossbar array. The other parts is the timing and control block, which generate internal timing signals to control the execution of the computing. The final part consist of counters, subtracters, comparators, and etc. to convert the output spikes from the IFC to digital signals and fulfill the computation for final output results.

4.5.3.2 Signal Waveform The signal waveform of external and internal timing signals are shown in Figure 29. The duration of an input pulse is set to be 5ns in the design. IPNT denotes the period to supply an input pattern. It varies from 60ns to 140ns, controlled by $I4PN < 2 : 0 >$. The signal waveforms in this example correspond to the computation period of 120ns when $I4PN < 2 : 0 >= 011$.

4.5.3.3 Post-layout Simulation Result Example Figure 30 is an image classification result example for classifying the six black and white digits images $0 \sim 5$ showing in Figure 23. It is the post-layout simulation result on the condition of TT corner with V_{dd} of 1.2V and temperature of $27°\mathrm{C}$. This result indicates that all images are classified successfully.

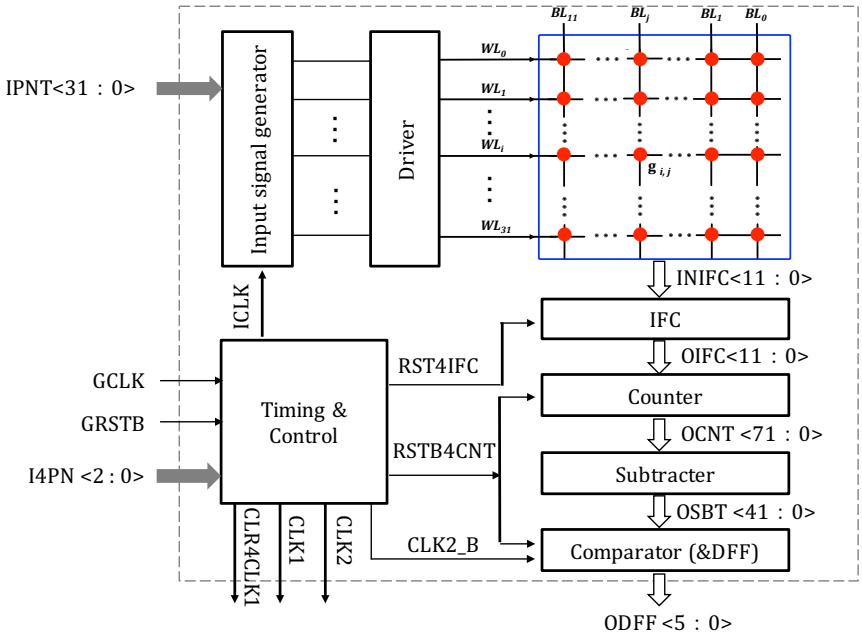

Figure 28: The circuits block diagram of the single-layer design.

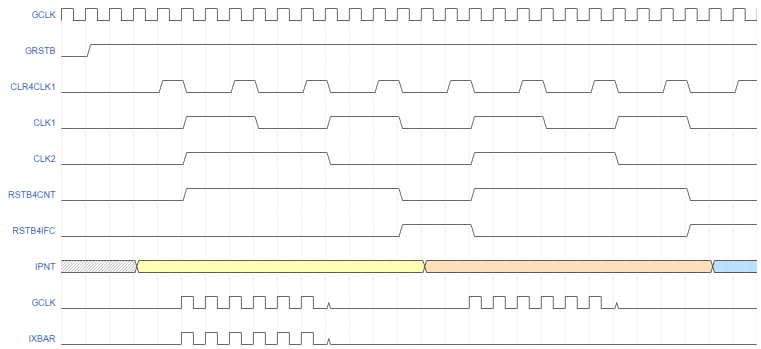

Figure 29: The ex- and internal timing signal waveform.

4.5.4 Two-Layer Feedforward System Design

4.5.4.1 Block Diagram Figure 31 shows the circuits block diagram of the two-layer design. Two layers consist of 64×40 and 20×10 crossbar arrays representing synaptic matrix is designed for images classification from MNIST digits database. In the design, three bits values in the form of spike numbers $0 \sim 7$ are given to the first layer crossbar array to represent the grey scale of the input images. Correspondingly, floating points images values are transformed to three bits values in the two-layer neural network training. The three bits are designed to guarantee a good classification accuracy with less design cost.

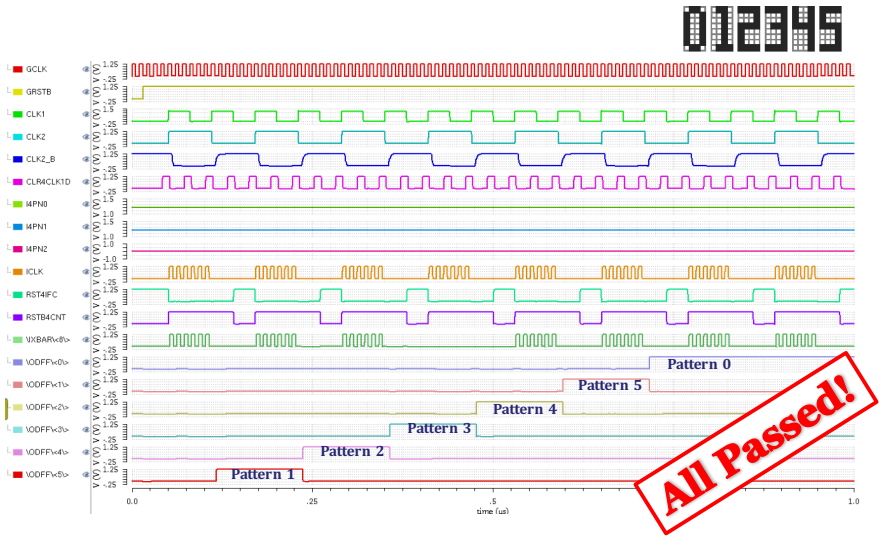

Figure 30: An image classification result example.

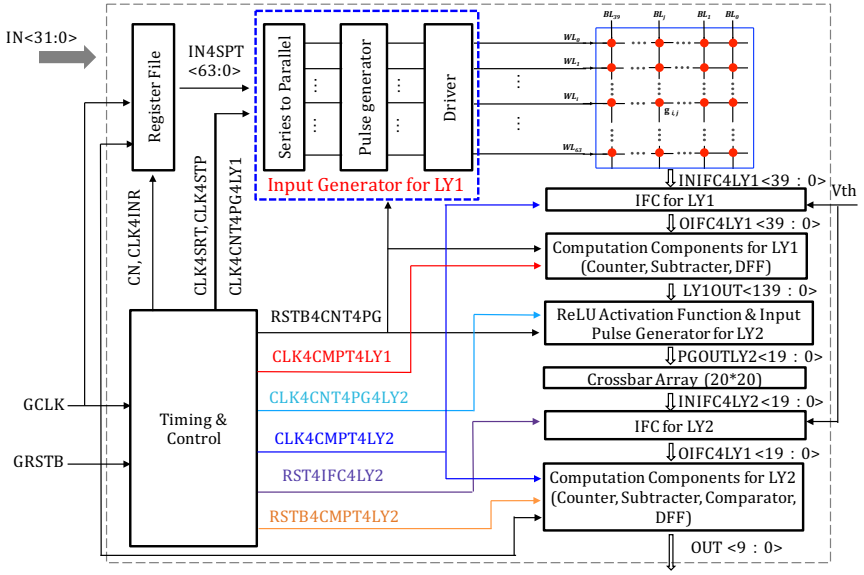

Figure 31: The circuits block diagram of the two-layer design.

Similar to the single-layer design, the two-layer design can also be divided to two major part: 1) Analog computation part: This part includes the resistive crossbars and the integrate and fire circuits (IFCs) for analog computing. Digits inputs from the input driver are given to the crossbar array in parallel, and analog computation is fulfilled by the crossbar and generate the sums-of-product results in current format. Then, the current output from each bitline of the crossbar are transferred to output spikes by the IFCs. Similar computation are executed in the second crossbar

array and the flowing IFCs. 2) Digital computation part: There are four major parts for digital computation. One part is the circuits blocks generating input pulses from outside digital signals and providing enough drive abilities for the crossbar array. Another part is the timing and control block, which generate internal timing signals to control the execution of the computing. It also includes a computation part for layer one which is similar to the computation part in single-layer design. The difference is that a rectifier (ReLU) activation function is implemented in the design, and an input pulse generator is also designed to generate input pulses from the output spike from the first layer IFCs. The final part is the computation part, including counters, subtracters, comparators and DFFs to transform the output spikes from the second layer IFCs to digital signals and compute for final output results.

4.5.4.2 Signal Waveform The signal waveform of external and internal timing signals are shown in Figure 32. The global clock (GCLK) period is designed to be 15ns. The duration of an input pulse is set to be 30ns in the design.

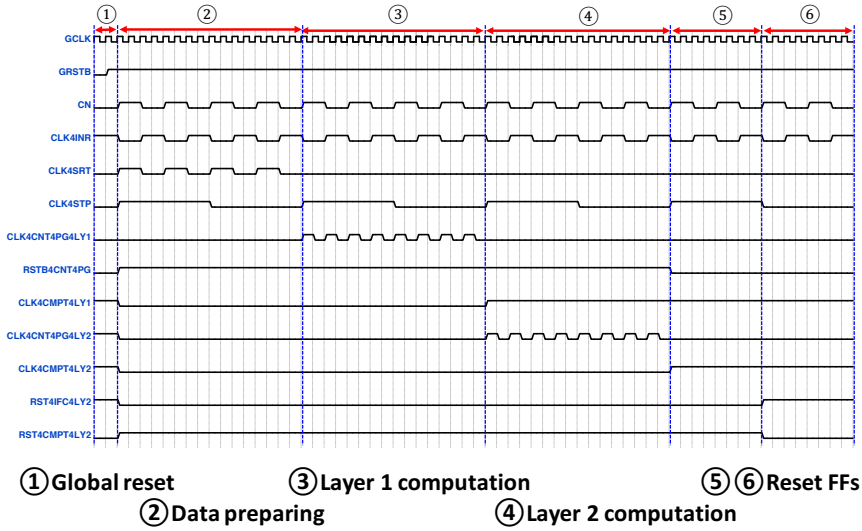

Figure 32: The ex- and internal timing signal waveform.

4.5.4.3 Post-layout Simulation Result Example Figure 34 is an image classification result example for classifying the ten digits images $0 \sim 9$ from MNIST database with eight grey scales showing in Figure 33. The 28×28 pixels images from MNIST database are compressed into an 8×8 pattern because of the constrain of the first layer crossbar size with only 64 wordlines in it. It is the post-layout simulation result on the condition of TT corner with V_{dd} of 1.2V and temperature of $27°\mathrm{C}$. This result indicates that all images are classified successfully.

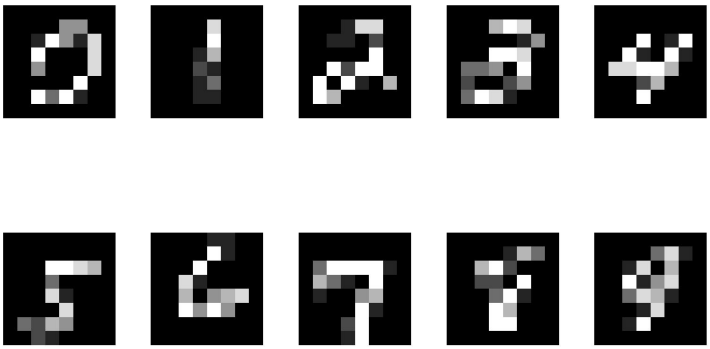

Figure 33: An image example from MNIST database (The 28×28 pixels images from MNIST database are compressed into an 8×8 pattern).

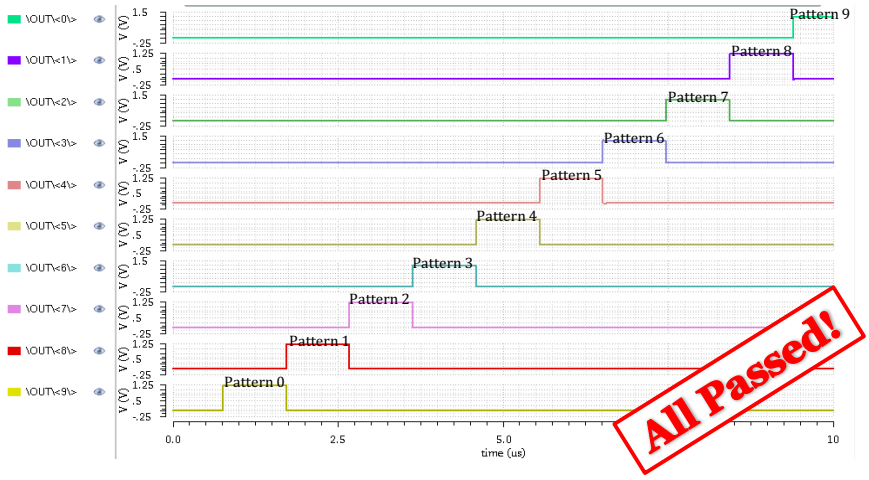

Figure 34: An image classification result example.

4.5.5 Fabricated Chip and Testing

The fabricated neuro-chip including 15 dies and 25 packaged chips has been received and are under testing, as is shown in Figure 35. We set up the testing bench shown in Figure 35. It provides the global clock signal from a high-frequency pattern generator and the input data from a FPGA board. The output signals are received and monitored through an oscilloscope. A PCB board was designed and made which is also shown in the figure.

Figure 36 demonstrate a test result example of the single-layer feedforward system for the six digits recognition - “1”, “2”, “3”, “4”, “5”, “0”, as is described in Figure 23. In this example, images with 3% defects were feed in using the digital pattern generator in a repeating sequence

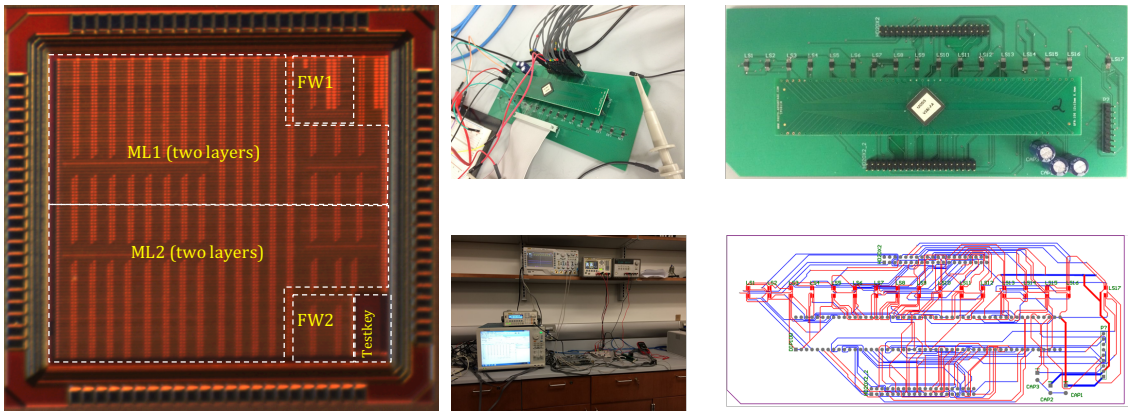

Figure 35: Die photo of the received neuro-chip and testing environment.

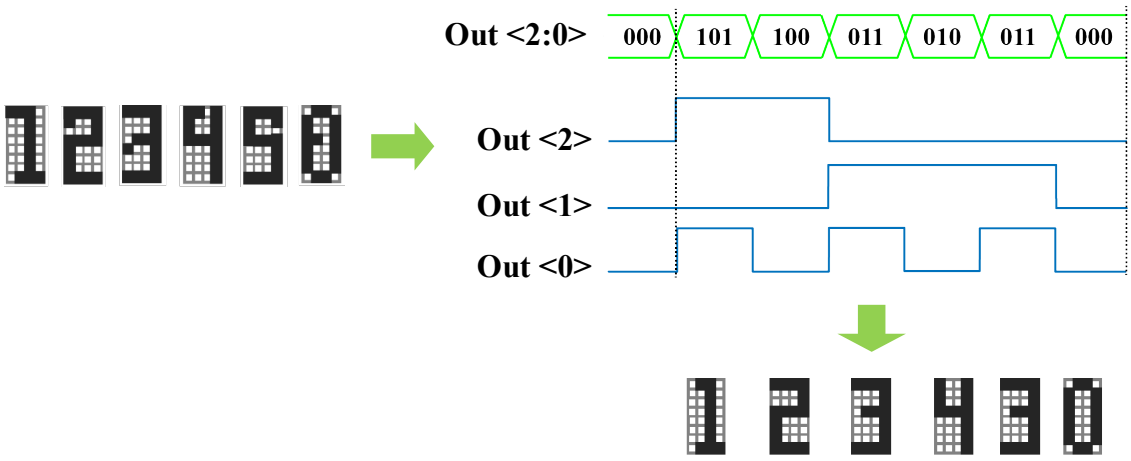

Figure 36: A test result example of the single-layer feedforward system for the six digits recognition.

of “0”, “1”, “2”, “3”, “4”, “5”. The 3% defects means 3% piexls of the images in Figure 23 were assigned to reverse digital values, for example, 0 is switched to be 1. The output results from the system were encoded to binary digits and captured by the logic analyzer. In the encoding circuit, the result “1” is represented by “101”, “2” is represented by “100”, “3” is represented by “011”, “4” is represented by “010”, “5” is represented by “001”, and “0” is represented by “000”. The results show that all the images except “5” are recognized successfully.

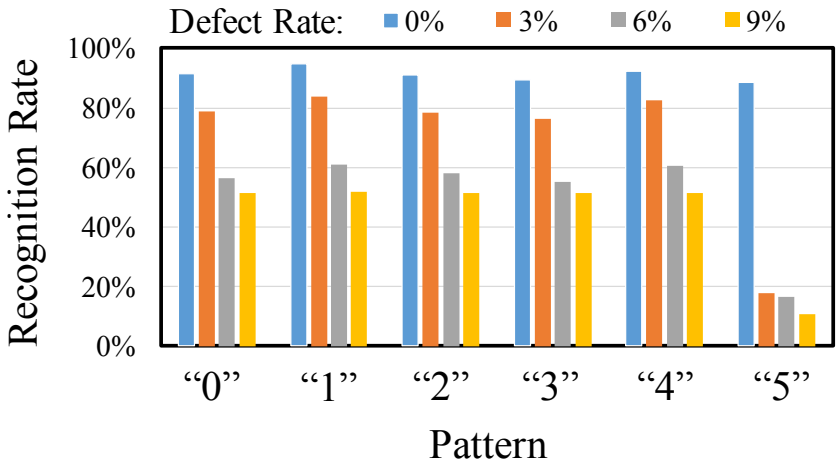

Figure 37: Testing results of the six digits with 0%, 3%, 6%, and 9% defects.

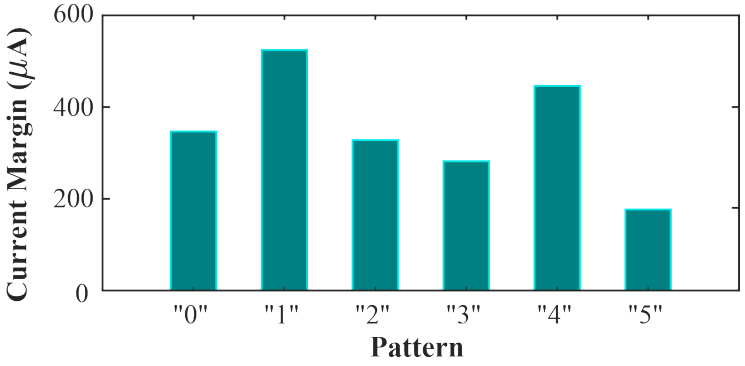

Figure 38: Statistical analysis of current margin of the six patterns.

Figure 37 depicts the statistical testing results of the six digits when 0%, 3%, 6%, and 9% defects are considered respectively. The results show that the number “5” has lowest recognition rate and more vulnerable to defects and process variation. The result is consistent with our analysis on current margin of different patterns, as is indicated in Figure 38. The current margin represent the current difference from the crossbar BLs with the largest and the second largest output.

4.6 SUMMARY

In this chapter, we proposed a novel spiking neuromorphic design built with a resistive crossbar array. It is a mixed-signal system that uses the digitalized spikes for data transferring and leverages the high density crossbar structure for parallel computation in analog format. Such a design naturally minimizes the use of analog components and therefore obtains significant savings in design area and energy consumption, compared with a previous crossbar-based computing engine with completely analog operation [3]. We carefully studied the feasibility of the proposed design in terms of the computation accuracy, efficiency, and reliability. The realization of neural network models demonstrated that our design has a good tolerance in resistive device imperfection but more vulnerable to the fluctuations in output spike generation. Moreover, a neuro-chip was design and fabricated consisting of a single-layer and a two-layer feedforward neural network for image classification. The post-layout simulation indicates that the both of the two systems have correct function and images can be classified successfully. The testing results show that about 90% recognition accuracy can be obtained in the single-layer feedforward network.

5.0 A MEMRISTOR CROSSBAR BASED COMPUTING ENGINE OPTIMIZED FOR HIGH SPEED AND ACCURACY

In the spiking neuromorphic system implemented in previous Chapter 4, the input information is supplied in the spiking format. The matrix computation result is detected by an *integrate-and-fire circuit* (IFC) and represented by output spikes. Such an approach demonstrates very high power efficiency. However, computing accuracy and speed are constrained by the limited spike number, limiting its application in matrix-vector computation. In this chapter, we propose a new memristor crossbar based computing engine for matrix-vector computation which leverages a current-based sensing scheme for higher computation speed and accuracy. The design supplies analog voltage signals to wordlines in parallel, and a current-based sensing scheme is developed.

The rest of this chapter is organized as follows: Section 5.1 gives preliminary of the proposed matrix-vector computation engine with memristor crossbar. Section 5.2 describes our proposed circuit design. Section 5.6.5 evaluates the computing accuracy and performance of our proposed design through a three-layer neural network for MNIST handwritten digit recognition. Section 5.4 presents the chip design of proposed matrix-vector computing engine. Section 5.5 summaries the work and discusses the challenges in the proposed design. Section 5.6 describes an neuromorphic engine with a current sensing scheme mainly for artificial network implementation. Section 6.4 summaries the chapter.

5.1 THE PROPOSED MATRIX-VECTOR COMPUTATION ENGINE PRELIMINARY

5.1.1 Matrix-vector Computation with Memristor Crossbar

As illustrated in Fig. 39, the memristor based cross-point structure can be used to realize a matrix-vector multiplication, *e.g.*, $\overrightarrow{Y_n}^T = \overrightarrow{X_m}^T \times M_{m\times n}$. More specific, we can use a crossbar array $G_{m\times n}$ to denote $M_{m\times n}$ by making $g_{i,j}$, the conductance of the cell at the cross-point of WL_i and BL_j, represent the corresponding data in $M_{M\times N}$. The matrix-vector multiplication is then transformed to

$$\overrightarrow{I_n}^T = \overrightarrow{V_m}^T \times G_{m\times n}. \tag{5.1}$$

Where, the input vector $\overrightarrow{V_m}^T = [v_1, v_2, \cdots, v_i, \cdots, v_m]$ is composed of analog voltages to WLs. The output current i_j at the end of BL_j produces a dot production, such as

$$i_j = \sum_{i=1}^{M} g_{i,j} \cdot v_i. \tag{5.2}$$

Thus, all the BLs currents form the output vector $\overrightarrow{I_n}^T = [i_1, i_2, \cdots, i_j, \cdots, i_n]$. For the high integration density and parallel operation, memristor crossbars greatly improve the efficiency in matrix computation and therefore inspired extensive studies on the hardware implementation and applications [3][43][55].

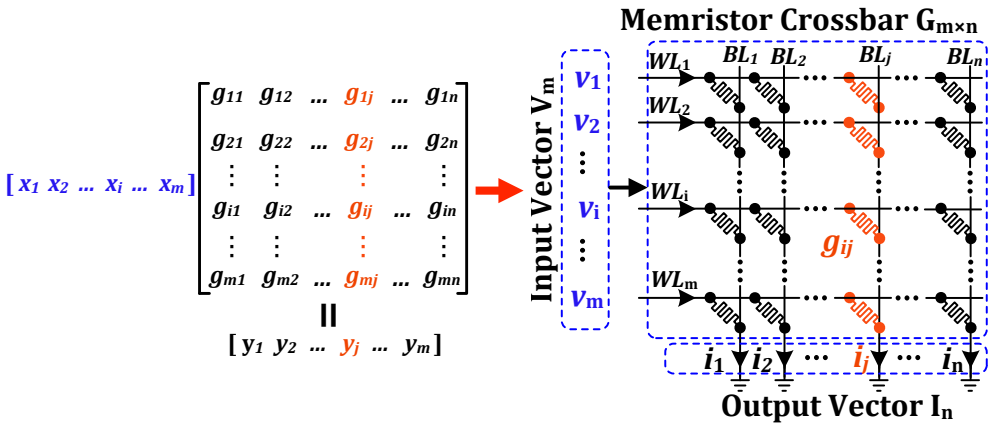

Figure 39: Mapping a matrix-vector multiplication to a memristor crossbar.

5.1.2 Motivation of the Proposed Work

In the spiking neuromorphic system implemented in previous Chapter 4, it uses an *integrate and fire circuit* (IFC) to detect the BL current and represent it by spike number. The charging and discharging overhead of IFC results in a non-linear relationship between the ideal BL current as a sum of weighted multiplication ($\sum_{i=1}^{M} g_{ij}V_i$) and the output spiking number: with BL current increase, the output spike number will increase first and then start to saturate, as shown in Fig. 40(a). It can also be observed from the figure that prolonging the computing period, *e.g.*, increasing it from 10ns to 50ns, will help produce more output spike number, resulting in better linearity and higher computation accuracy.

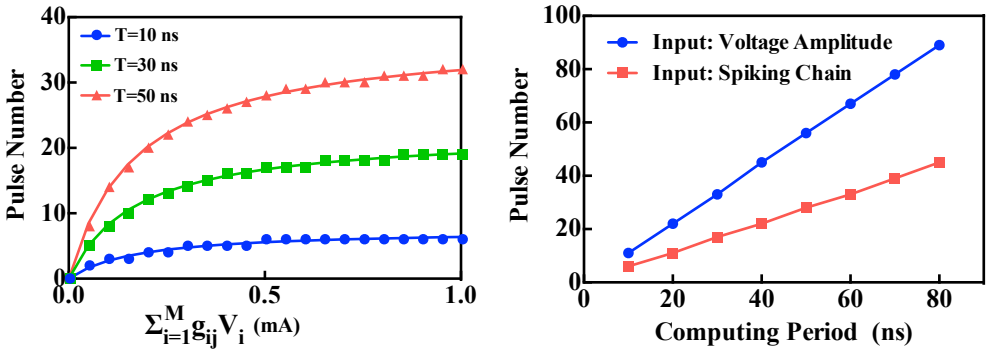

Figure 40: (a) The computation accuracy analysis of a spiking-based design [2]. (b) The relations of output spike number vs. computing period when representing input data by spike chain or by voltage amplitude.

Moreover, the input data in the spiking-based design [2] is converted to a chain of spikes: the spike frequency (*i.e.*, the number of spikes within a constant computing period) represents the scale of data. Such a digitalized interface guarantees the good noise immunity and high energy efficiency in signal transferring. However, it also induce a low utilization rate at time domain. Thus, the computing period need to be sufficiently long to satisfy the computation accuracy requirement. In contrast, when using voltage amplitude to represent the strength of input data as [3], the computing period will be fully utilized and therefore could be a lot shorter. For example, we compared the relations of the output spike number and the computing period of two designs and Fig. 40(b) shows the results when the BL current is set to 0.6mA. To generate the same amount of output spikes,

the computing period of the design representing input data by voltage amplitude is only about the half of the version using an input spike chain.

We also note that the spiking-based scheme adopted *one-transistor-one-memristor* (1T1M) cell structure. The use of selective transistor is highly recommended in data storage structure, to alleviate the sneak path leakage problem in crossbar [49]. However, it significantly enlarges the cell size ($\geq 6F^2$). Very importantly, more than 4% loss in computing accuracy has been induced by selective transistor after including its state resistance into consideration [2].

5.2 THE PROPOSED CIRCUIT DESIGN

We propose to develop memristor crossbar based computing engine by integrating a current sensing scheme. Instead of connecting each BL to an IFC, a current amplifier is used to detect BL current. Thus, the voltage of BL will be clamped to a fixed voltage level and the matrix-vector operation can closely follow the linear function of Eq. (5.2), without being affected much by the resistance distribution in memristor crossbar.

When implementing the proposed matrix computation engine, the non-ideal effects are mainly contributed by the memristor crossbar and the current amplifier. We investigated the computation engine design made of a 144×144 memristor crossbar and the current amplifier at 130nm technology node. In this section, we will describe the detailed design considerations and analyze the computation accuracy loss.

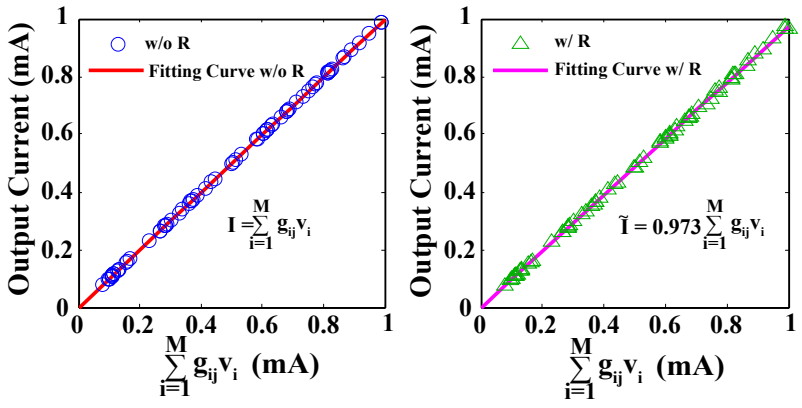

Figure 41: The impact of wire resistance on the relation of $I_{o,j}$ vs. $\sum_{i=1}^{M} g_{i,j} v_i$.

5.2.1 Memristor Crossbar Array

It has been well known that the sneak path leakage is a major concern in memristor crossbar design [49][50]. The sneak path leakage refers to the unexpected parasitic current leakage caused by those unselected cells. To solve the issue, *one-transistor-one-memristor* (1T1M) and *one-selector-one-memristor* (1S1M) cell structures have been investigated when using crossbar arrays for data storage [51][52] and neuromorphic systems [2].

Our approach conducts the computation in a parallel mode: all inputs are represented by analog voltage signals and sent to WLs of crossbar array simultaneously. The current amplifier help keep BL at a constant voltage level. In this way, all the cells are selected and accessed at the same time. The impact of sneak path leakage is negligible in such a *multiple inputs multiple output* (MIMO) operation [56] so the BL current can follow Eq. (5.2). Therefore, we are able to adopt the memristor-only cell structure which offers cell size of $4F^2$ while assuring computation accuracy.

The series wire resistance could also affect the BL current and therefor distort the realization of Eq. (5.2). To quantitatively evaluate the impact, we compared the relations of BL current ($I_{o,j}$) and $\sum_{i=1}^{M} g_{i,j}v_i$, with and without including the wire resistance into considerations. Each configuration conducted 2,000 simulations with randomly generated input data. Fig. 41 summarizes the simulation results, where only a small subset of data is included for better illustration.

In the simulations, we assumed 3-bit resistance states of memristor device, that is, eight resistance levels from $\mathrm{R_{on}} = 50\mathrm{K}\Omega$ to $\mathrm{R_{off}} = 1\mathrm{M}\Omega$. The resistance patterns of crossbar were randomly picked which cover the major portion of the range of $\sum_{i=1}^{M} g_{i,j}v_i$. For the $130\mathrm{nm}$ technology adopted in the work, the wire resistance per cell is about 0.52Ω. The simulation results show that when the wire resistance is not included, all the output currents strictly follow the theoretical analysis in Eq. (5.2) which is not affected by data patterns. A small shift occurs after including the wire resistance into the simulation. Even though, the linearity between the output current and $\sum_{i=1}^{M} g_{i,j}v_i$ can still maintain because of the large resistance value of memristor. The fitting curve obtained in our implementation is

$$\tilde{I}_{o,j} = \gamma I_{o,j}, \tag{5.3}$$

where $\gamma = 0.973$. Note that our result is consistent to [57], where the largest reading error is less than 5%. It was obtained at the farthest cell in the crossbar when all the remaining cells have the lowest resistance value of $\mathrm{R_{on}}$.

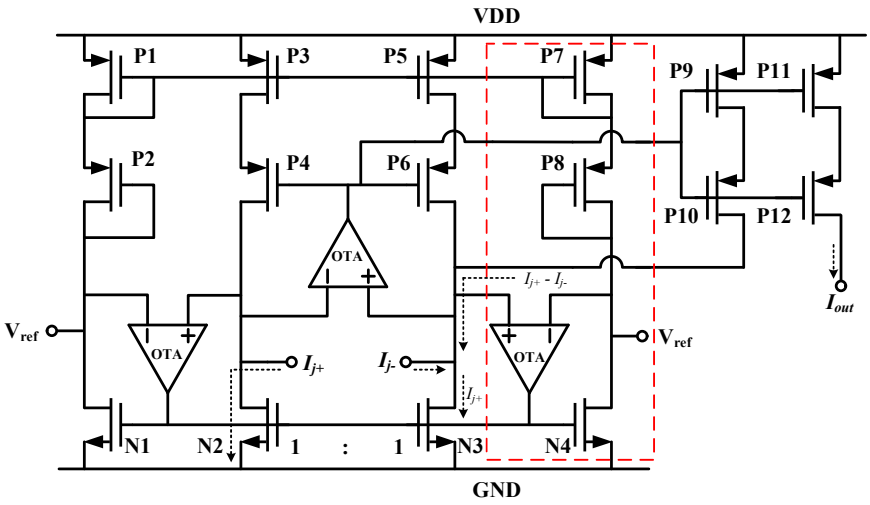

Figure 42: The current amplifier design.

5.2.2 Current Amplifier

Fig. 42 depicts the schematic of our current amplifier design. It is used to detect the output current from crossbar array. Since the conductance of memristor can represent a positive value, we can subtract the results from two crossbar arrays to obtain the computation with negative matrix elements [3]. To support the feature, we designed a bi-lateral current buffer amplifier with two input ports I_{j+} and I_{j-}, denoting the BL currents from the crossbars for the positive and negative elements, respectively. A system-level illustration is demonstrated in Fig. 44, which shall be explained in Section 5.6.5.

In this current amplifier design, V_{ref} is a reference voltage providing DC operating point. Three high-gain *operational impedance amplifiers* (OTAs) are used to clamp the voltage level of the two input ports I_{j+} and I_{j-} at V_{ref} during operation. OTAs also assist the function of the associated current mirrors, *e.g.*, keeping the same V_{ds} for $\mathrm{N1} \sim \mathrm{N4}$.

The input current I_{j+} injected into $\mathrm{N2}$ is duplicated to $\mathrm{N3}$. The subtraction of $I_{j+} - I_{j-}$ can be compensated by the branch of $\mathrm{P9} \sim \mathrm{P10}$. A cascode structure is adopted at the output stage to improve the accuracy of current mirror ($\mathrm{P9} \sim \mathrm{P12}$). For the design requiring only unilateral current input, the current amplifier corresponding to I_{j-} can be trimmed by removing the part within the red dashed box in Fig. 42.

Compared to other current amplifiers such as [58], our approach tends to minimize the output voltage variation under different amount of BL currents. Recall that in the spiking based design, a BL is directly connected to IFC, and the integration/firing operations are realized through charging/discharging a capacitor inside IFC [2]. A stable charging voltage at the capacitor is necessary in order to maintaining a fixed charging rate and therefore a constant spiking generation frequency. Our current amplifier was designed for the purpose. We evaluated the performance of I_{out} of current amplifier by fixing $I_{j-} = 0$ and sweeping I_{j+}. Here, V_{ref} is set to 200mV. The result in Fig. 52 shows that I_{out} follows well with I_{in}.

5.2.3 Overall Performance

After combining the non-ideal factors of the memristor crossbar and the current amplifier design, the output signals can be fitted by a first-order function such as

$$I^*_{out} = \gamma \sum_{i=1}^{M} g_{i,j} v_i. \tag{5.4}$$

The root mean square errors of I^*_{out} is 2.7% . The slight error of our design can be mitigated by many system-level designs. For example, in developing neural network models, this error can be included by substituting Eq. (5.4) into the training procedure so the computation accuracy at system level will not be affected much. In Section 5.6.5, we will use a simple neural network to evaluate the performanc

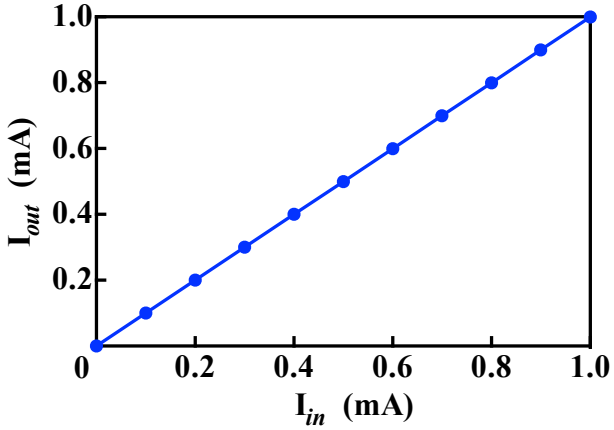

Figure 43: The characteristic of the current amplifier: I_{in} vs. I_{out}.

5.3 APPLICATION AND EVALUATION

We designed a neuromorphic system by using memristor crossbar with the proposed current sensing scheme. For each memristor crossbar, analog voltage signals are used as input data. The current generated at BL will go through a current amplifier and an IFC in sequence to produce output spikes. The performance and accuracy were evaluated through a three-layer neural network for MNIST handwritten digit recognition [59].

5.3.1 Neuromorphic System Implementation

As discussed in Section 5.2.2, the proposed current amplifier is used to sense out the BL current (I_{out}) with a close to ideal linearity, representing the sum of weighted multiplication. In real applications, a current output need to be transformed into a voltage signal. The spiking based architecture by Liu *et al.* [2] demonstrated a high power efficiency by eliminating the use of analog components like *analog-to-digital converters* (ADCs) and *digital-to-analog converters* (DACs). By leveraging the integrate-and-fire design concept, an approach of feeding I_{out} of the current amplifier to an IFC and therefore transferring the computation results into a digitalized format has been adopted in our design.

Fig. 44 demonstrates the system-level approach for neural network implementation. Instead of using a chain of spikes as the data input, analog voltage signals generated by DAC will be simultaneously supplied to the wordlines of crossbar to represent input vector. As discussed previously, this approach can produce more output spikes within the given computing period, offering better computation accuracy, compared to the original spike-based design. For the analog voltage inputs with different voltage amplitudes, the current amplifiers performs like buffers that isolate the crossbar array from IFCs. It helps eliminate sneak-path leakage and grantee good computing accuracy as discussed in 5.2.3.

As shown in Fig. 44, two memristor crossbars are used in each layer, in order to realize both the positive and negative matrices terms without modifying original neural network models. The two crossbars are respectively denoted as M^+ and M^- [3]. To conduct a subtraction operation and match the mathematical algorithm in training and recall processes, the current outputs from two corresponding BLs of the two crossbars will be connected to the two input ports of an current

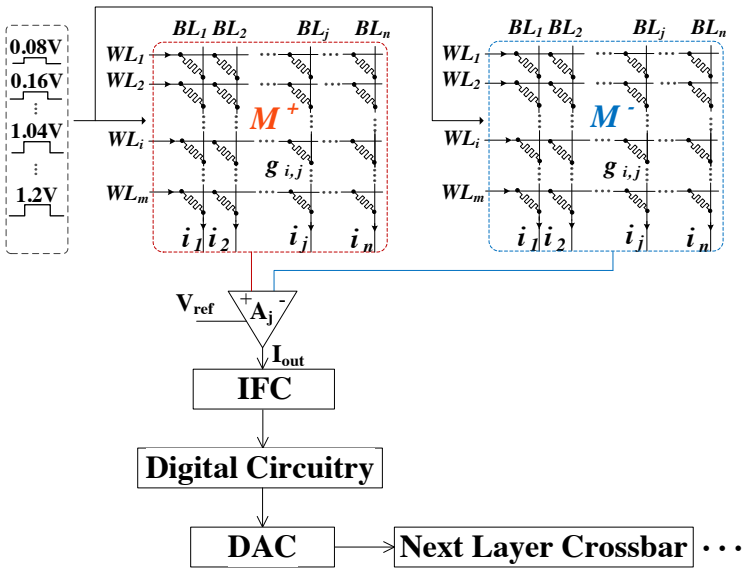

Figure 44: The system architecture used for neural network implementation.

amplifier. The computation output will be then transferred to the output of the current amplifier I_{out}, which is the input of IFC for output spike generation. The output spikes can be encoded to digital signals by digital circuitry and gives out digital voltage signal [2]. When implementing a multi-layer neural network system, the output digital signals of one layer are transferred to analog format and fed to the following layer.

In this work, we evaluated the proposed new design denoted as ***AnalogV*** in the following context and compare it with spiking based design in [2], which is denoted as ***Spiking***. Comparing the two type of system implementations, *AnalogV* offers an analog-digital flow, while *Spiking* provides a completed digitalized data transmission. The computation accuracy of *Spiking* is affected by the non-linearity effect shown in Fig. 40(a) and the selective transistor resistance in the 1T1M cell structure. The low output spiking rate in a certain computing period due to the non-linearity resulted by IFC charging and discharging described in [2] and the spiking input with 50% utilization rate are also major concerns. While, the system *AnalogV* proposed in this work can provide a higher output spiking rate because of the parallel analog voltage signal adopted in this design. Meanwhile, the crossbar of *AnalogV* is denser for it composed of only memristors. However, the design requires extra components, including current amplifier and DACs. The induced overheads have been carefully considered in the following evaluations.

5.3.2 Application of Digital Pattern Recognition

We tested and compared the two system designs on a three-layer feed-forward neural network. Here, 60,000 digital patterns from MNIST [59] were used for training, and a test set of 10,000 examples was selected randomly. During the training and testing, we resized digital patterns in 28×28 pixels from MNIST database to smaller patterns in 12×12 pixels to match the maximal allowable 144×144 memristor crossbar in this work. Each memristor represents a 3-bit data (8 resistance levels). Traditional back-propagation and delta rule were used for training and then each memristor cell in the crossbar was programmed to a target resistance [56]. Fig. 45 shows an example of patterns $0 \sim 9$ with 4-bit gray-scale used in our training and testing.

The crossbar sizes of layer $1 \sim 3$ are 144×128, 64×128 and 64×20, respectively. The numbers were determined for the best recognition accuracy obtained at software level. We evaluated and compared the performance of *AnalogV* and *Spiking* systems under different design considerations, including the input width, input bits, and resistance value variance. The impacts of matrix-vector computation accuracy on these systems were measured by the failure rates in pattern recognition testing.

5.3.3 Design Parameter Considerations

In both system implementations, integrate-and-fire circuits are adopted to transform the current computation results to readable digital voltage signals. As such, the computing accuracy of the integrate-and-fire circuits that is mainly affected by the computation period is a major concern in the failure rate of the two systems in pattern recognition. The computation period of the IFC will

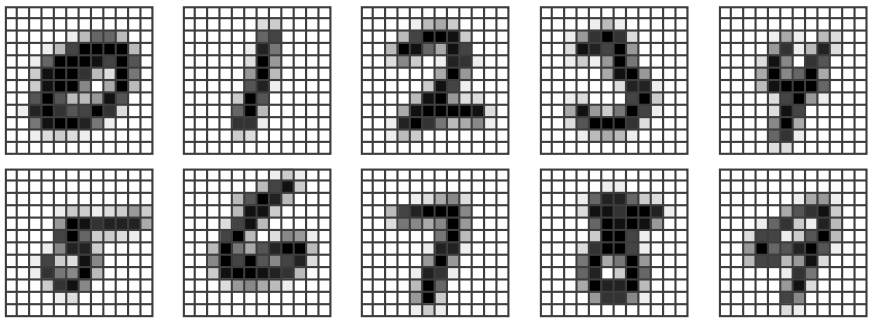

Figure 45: An example of patterns $0 \sim 9$ with 4-bit gray scale in testing.

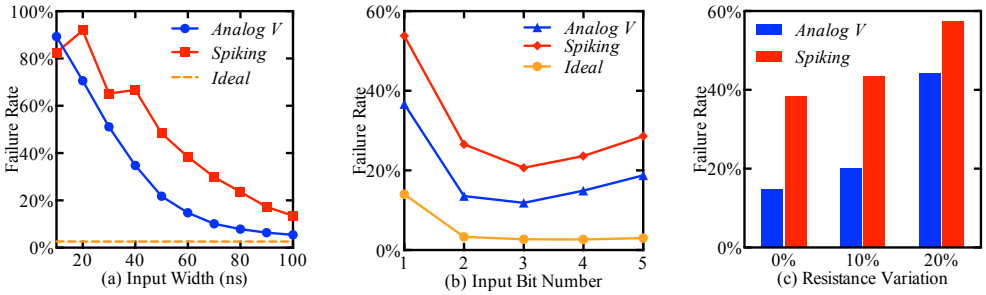

Figure 46: The MNIST recognition failure rates of ***AnalogV*** and ***Spiking*** system designs when applying different (a) input widths, (b) input bits, and (c) resistance value variations.

be determined by both input signal width and input bits. Therefore, the input signal width and input bits will be two important design parameters needed to be consideration. As discussed above, DACs are used in our proposed system *AnalogV* and its resolution will affect the computation accuracy of the system. In the hardware design, DAC with half-bit resolution error was implemented and the induced computation accuracy loss was considered in the following evaluation. Moreover, during programming memristor with multiple value, variations in resistance value cannot be ignored and will be considered.

5.3.3.1 Input Width Dependence We first tested the pattern recognizing failure rated under different input signal widths. Inputs were set to 4-bit value in this evaluation. Fig. 46(a) demonstrates the result when varying the input width from 10ns to 100ns. The failure rate under the ideal condition is 2.66%, which was obtained from software simulation without including any real implementation consideration. The simulation results showed that both *AnalogV* and *Spiking* systems are sensitive to the input width, and the new scheme *AnalogV* has the higher computation accuracy. More specific, the failure rate of *AnalogV* is 5.43% at the input width of 100ns, which is 8.11% less than that of *Spiking* design (13.54%).

The large input width dependency is mainly caused by IFC design so computation accuracy improves as the input width increases. Accordingly, the failure rate decrease as the input width grows up, especially when it is smaller than 60ns. *Spiking* demonstrates a much higher failure rate (38.43%) than *AnalogV* (14.75%) under the input width of 60ns. This is because sparse input pulses are given as input for *Spiking* and the real computation time is decrease by half. More output

spikes will be generated in system *AnalogV* in a certain input width as analog signals are applied to crossbar with 100% of utilization. The failure rate fluctuation of *Spiking* at small input width indicate the high instability of the system and high randomness of the testing results. The accuracy loss caused by the non-linearity of IFC in *AnalogV* is much smaller than the accuracy decrease in *Spiking* in the same computation speed. In the other words, our proposed system *AnalogV* could obtain much higher computing speed (about twice) than *Spiking* when they target at the same computation accuracy.

Another concern on *AnalogV* design is the frequency of DACs that generate discrete analog signals as crossbar inputs. In our implementation, DAC was designed with a maximum frequency of $100\mathrm{MHz}$, corresponding to $10\mathrm{ns}$ analog input width. The maximum frequency can be increased in real design but power and area will be scarified. The result in Fig. 46(a) indicates that the input width must be longer than $60\mathrm{ns}$ in order to achieve a reasonable failure rate. Therefore, the DAC with $100\mathrm{MHz}$ frequency is large enough and will not be a constraint in the system design.

5.3.3.2 Input Bits Dependence Fig. 46(b) shows the impact of pattern input bits, reflecting the pattern color depth on the system failure rate. Based on the above analysis, we evaluated the input bits dependence by using $60\mathrm{ns}$ input width. The two systems were first tuned till they can obtain similar accuracy. Comparing with prior design *Spiking*, *AnalogV* obtained lower failure rate when the input pixel has more than 2 bits It matches well with the simulation results discussed above. As can be seen from the ideal curve, there is a valley value. This is because more input bits per pixel of input images indicates higher complexity of neuromorphic system should afford, and the computation accuracy is limited by the available resistance states of memristors. Practically, we observed that the values of $\sum_{i=1}^{M} g_{i,j} \cdot v_i$ drops statistically, leading to higher quantification error in IFC. These two opposite factors together influence the trend shown in Fig. 46(b). We chose 4-bit color-depth to obtain a relatively high accuracy in difference systems.

5.3.3.3 Impact of Resistance Value Variation The impact of the resistance value variation in computation failure rate was evaluated too. The failure rates of both systems with 0%, 10%, and 20% resistance variation are summarized in Fig. 46(c). The systems failure rate increases with the increasing of resistance value variation. System *Spiking* is more vulnerable to the resistance value variation because of the lower output spike number in it. We observe a large failure rate increase when increasing the resistance value variation from 10% to 20%. The results show that the failure

rate of our proposed design *AnalogV* is under 20% when the variation is 10%, while the failure rate of *Spiking* increases to more than 45%.

Table 2: System Performance Comparison

	Area	**Power**	**Speed**	**Energy**
AnalogV (This work)	0.426mm^2	100.6mW	16.7MHz	6.04nJ
Spiking [2]	0.476mm^2	82.93mW	10MHz	8.29nJ
Difference	-10.5%	+17.6%	+40.1%	-27.1%

5.3.4 Design Comparison

The area, power, speed and energy consumption of the two systems were compared and summarized in Table 2. Area and power data are based on the systems implementation at 130nm technology node. The DACs and the current amplifiers used in *AnalogV* induce more area and power consumption. However, less IFCs and digital circuits are needed in *AnalogV* as the subtraction of computing result from the positive and negative crossbars can be executed by current amplifier. The 1T1M crossbar structure of *Spiking* induces more area cost.

Comparing to *Spiking*, *AnalogV* obtains 10.5% decrease in area while increases power consumption 17.6%. This is caused by the extra power consumed by the current amplifier and DACs. When testing the speed of *AnalogV* and *Spiking*, we selected 60ns and 100ns as input pulse widths respectively to maintain an approximate similar failure rate 15%. *AnalogV* executes much faster than *Spiking*, obtaining 40.1% speed improvement. Overall, *AnalogV* lowers the energy consumption 27.1% comparing with *Spiking*.

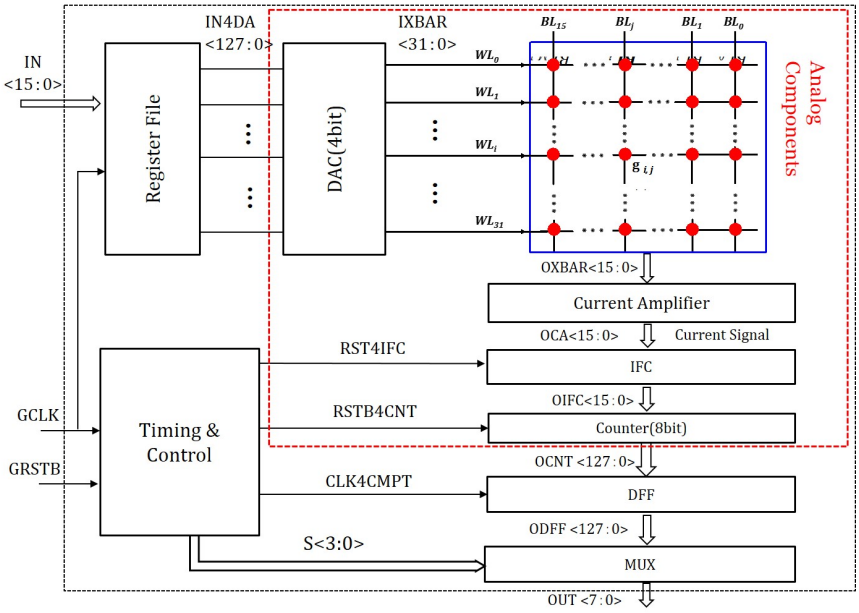

Figure 47: The circuits block diagram of the matrix-vector computation system.

5.4 CHIP DESIGN

Basing on the neuromorphic engine with current sensing proposed above, a matrix-vector computation system is designed for a on-chip design using the analog-mixed-signal design flow. The circuits block diagram and signal flow is shown in Figure 47. In this design, a 4-bit digital-to-analog converter (DAC) is needed before each wordline of the crossbar to transfer the external digital signals to analog voltage signals. Then, the analog voltage signals with voltage amplitude to represent the signal strength are given to the crossbar. The register files are designed before the DACs to store the external 16 bits signals ($< 15 : 0 >$)temperately and generate the 128 bits signals ($< 127 : 0 >$) for the inputs of the DACs after 8 clock cycles.

Similar to the on-chip design of the spiking engine in Chapter 4, we adopted the hardcoded cell design - 1T1R (one transistor and one resistor) cell structure in the crossbar array design. Output current from each bitline of the crossbar are copied without accuracy loss to the input of the IFCs by adopting the current amplifier. Counters, DFFs and MUXs are designed to convert the output spikes from the IFC to digital signals and fulfill the computation for final output results. Internal

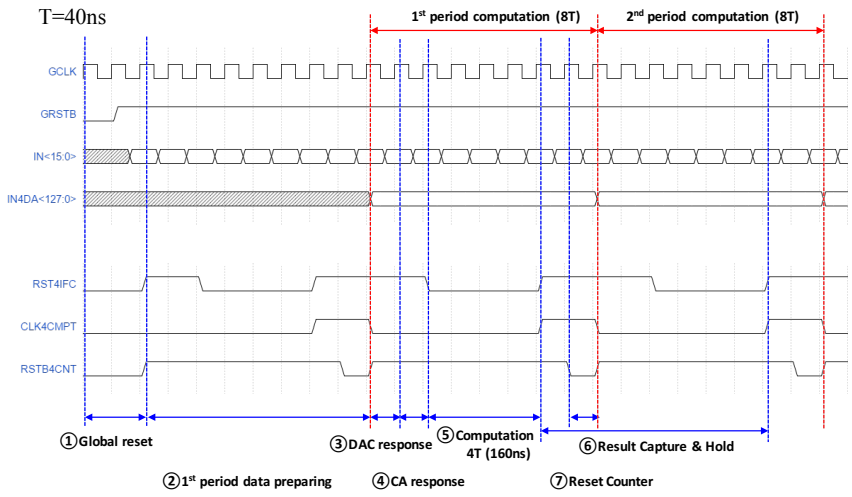

Figure 48: The ex- and internal timing signal waveform.

clock signals are generated by the timing and control circuit block to control the execution of the computing. The signal waveform of external and internal timing signals are shown in Figure 48. The layout of the matrix-vector computation system is shown in Figure 49.

5.5 CHALLENGE AND DISCUSSION

In this work, a memristor crossbar based computing engine using current sensing was proposed for matrix-vector computation. The parallel exaction was realized by applying analog voltages to every wordline of memristor crossbar. A current amplifier circuit was designed which mirrors the current from the crossbar with a slight accuracy degradation. To evaluate the computation accuracy of the scheme, we implemented a three-layer feed-forward neural network and plugged different memristor crossbar based computing engines for comparison. We thoroughly analyzed the newly proposed system and a prior reported design from perspectives of accuracy, speed, area, and energy. The results show that our system has lower area and energy consumption with a higher speed than prior spiking based design. The computation accuracy of the new system based on our computing engine can reach 94.6%. Overall, it demonstrates a good computation accuracy

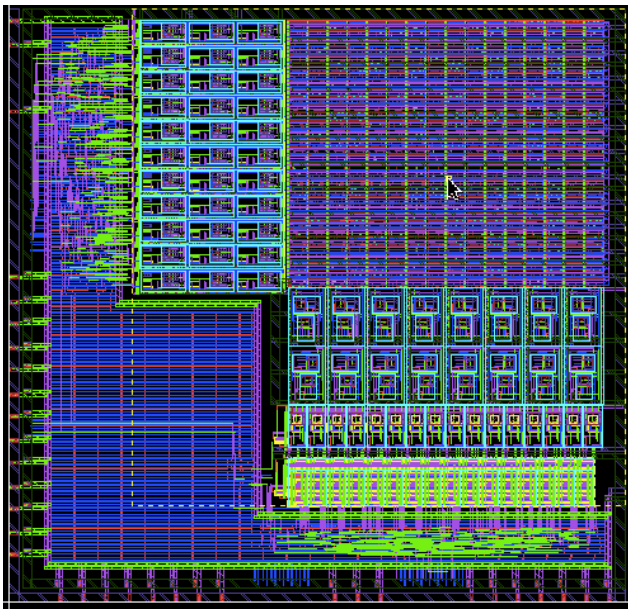

Figure 49: The layout of the matrix-vector computation design.

in matrix computation with a lower energy consumption. Moreover, a matrix-vector computation system is designed on a chip. Challenges in the proposed design is that IFCs are still used to generate readable digital signals which limit the computation accuracy. Therefore, a current-to-voltage converter based on a current amplifier is designed which converts the output current from the crossbar to analog voltage without computing accuracy loss. ADCs are needed when applying it in matrix-vector computation, which will bring extremely high design cost in the on-chip design. Therefore, this approach is mainly designed for artificial neural network implementation.

5.6 AN NEUROMORPHIC DESIGN FOR NEURAL NETWORK IMPLEMENTATION

In this work, we focus on improving the computation accuracy for memristor based neuromorphic engine. A current-to-voltage converter based on a current amplifier is designed which converts the output current from the crossbar to analog voltage without computing accuracy loss. Moreover, a rectified linear unit activation function is implemented by using the current-to-voltage converter.

A hardware implementation flow based on the proposed design approach is built for multi-layer as well as single-layer feed-forward neural network. Two neural networks including a single-layer system and a two-layer neural network will be implemented for MNIST handwritten digit recognition. Various design parameters are considered and their impact on classification accuracy are evaluated.

5.6.1 Design Motivation

In memristor crossbar based sums-of-product computation, it is essential to sense the BL currents out to obtain the computation result. A linear relationship between the output $O_{o,j}$ and $\sum_{i=0}^{n-1} g_{i,j}\, v_i$ shall be followed to guarantee a successful sensing without computation accuracy loss:

$$O_{o,j} = k \times \sum_{i=0}^{n-1} g_{i,j}\, v_i, \tag{5.5}$$

where, k is a constant value independent on any parameter related to the memristor-based crossbar.

Previously, Hu *et al.* adopted a voltage sensing scheme: a fixed resistor R_s is inserted at the end of each BL; the BL current is then transferred to the voltage across R_s and detected through an operational amplifier [3]. The design initiated an effective matrix computation. However, the use of R_s severely degrades the accuracy and limits its applications. More precisely, the output voltage at BL_j can be expressed as [3]:

$$v_{o,j} = \frac{1}{g_s + \sum_{i=0}^{n-1} g_{i,j}} \sum_{i=0}^{n-1} g_{i,j}\, v_i \tag{5.6}$$

where, g_s is the conductance of R_s. Eq. (5.6) implies that $v_{o,j}$ is not only determined by the sum-of-product $\sum_{i=0}^{n-1} g_{i,j}\, v_i$ but also affected by $\sum_{i=0}^{n-1} g_{i,j}$ denoted as the data pattern to BL_j. In other words, even for the same sum-of-product, BLs with different data patterns could generate different output voltages. The scenario is shown in Fig. 50(a). Here, we apply different data patterns – all R_{on}, R_{off}, and a random combination. They all have close-to-linear output/input relation but in different slope rates. The resistances of memristors range from $R_{on} = 10\text{K}\Omega$ to $R_{off} = 500\text{K}\Omega$, and R_s is $1\text{K}\Omega$. A possible solution could be developing a BL compensation circuit adaptive to various data patterns, which could be very costly.

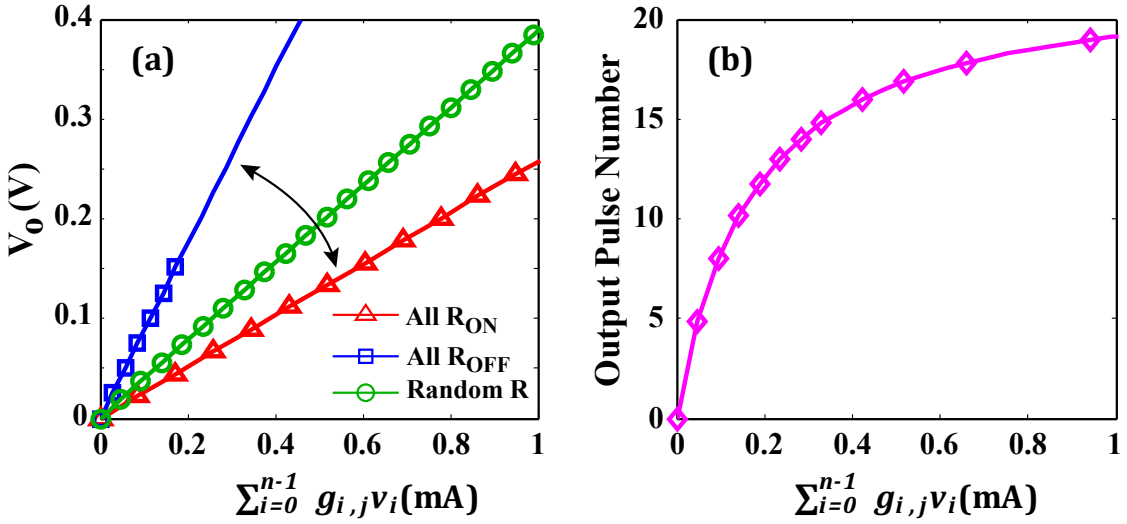

Figure 50: Computation accuracy analysis for (a) voltage-based sensing [3], (b) spiking design [2].

The spiking-based approach usually uses *integrate and fire circuits* (IFCs) to detect the output from BL and represent it by spike number [2]. However, the intrinsic delay of the IFC results in a non-linear relationship between the output spike number and the sum-of-product. A test case result is shown in Fig. 50(b). Recently, a traditional circuit – *trans-impedance amplifier* (TIA) was adopted to convert the output current $i_{o,j}$ to voltage signal directly [60]. This approach can retain a linear sensing in Eq. (5.5). However, its output range is narrow in real implementation, especially on-chip designs.

Another challenge faced in memristor-based neuromorphic systems is that two memristor crossbars are usually required to represent each neuron layer of an *artificial neural network* (ANN). That is because synaptic weight with negative values besides positive values are necessary. However, it doubles the use of memristor crossbars and induce in a lot more computation resources in data subtraction and transformation [3][2].

5.6.2 Design and Application of Our Proposed Design

In this work, we develop a memristor crossbar based neuromorphic engine with a current sensing scheme. In addition, the design space of the proposed sensing scheme in implementing the artificial neural network with a single crossbar for each neuron layer is explored.

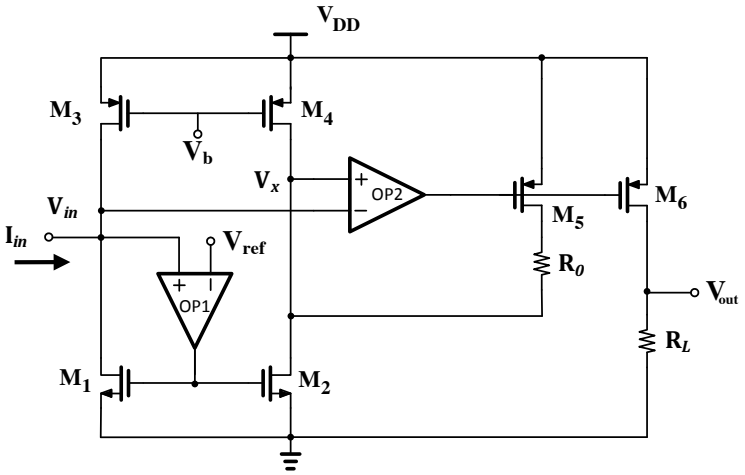

Figure 51: The design scheme of current-to-voltage converter.

We designed a current-to-voltage converter basing on a current amplifier which is connected to each bitline of crossbar directly. The voltage at the end of each BL is retained to a fixed value by the current amplifier, helping eliminate the sneak path leakage issue. Most importantly, the computation in crossbar is isolated with the current to voltage converting execution by the current amplifier. As such, the computation result of the crossbar will not be distorted and the linear function in Eq. (5.5) is implemented. The computation of the proposed neuromorphic engine is executed in a parallel format: input voltage v_i with different amplitude representing signal strength is supplied to each WL simultaneously; and the sum-of-product in each BL are sensed out through BLs all together.

In this section, we first present the design of current-to-voltage converter for the current sensing scheme and analyze its sensing accuracy. A computation engine made of a 32×16 memristor crossbar and the current sensing scheme at GlobalFoundry $130\mathrm{nm}$ technology node are implemented. Its performance in the sum-of-production is investigated. In this work, the resistance range of memristors is set from $R_{on} = 10\mathrm{K}\Omega$ to $R_{off} = 500\mathrm{K}\Omega$ with 16 resistance states.

5.6.3 Current-to-Voltage Converter

Fig. 51 demonstrates the schematic of our proposed current-to-voltage converter design. It is used to copy the output current of the crossbar and convert the current to output analog voltage by following a linear property. Different from the narrow output range of TIA, the proposed current-to-voltage converter has an output range larger than $0 \sim 1V$. In this section, the working function, converting accuracy and computing accuracy with memristor-based crossbar of the proposed design are analyzed.

Two *operational amplifiers* (op-amp), OP_1 and OP_2, are used in the design. OP_1 and M_1 together create a negative feedback, and a reference voltage V_{ref} is given to the negative port of OP_1 to clamp the positive port voltage to the same level. Therefore, the tail voltage of crossbar bitlines will be fixed to V_{ref} when connecting the proposed converter to the crossbar. This characteristic enables the aforementioned parallel computing execution, without inducing the sneak path leakage problem. The input current I_{in} combining the bias current through transistor M_3 are injected to M_1 which is mirrored to M_2 with a ratio of θ ($0 < \theta < 1$). $\theta = \beta/\alpha$, where α is the W/L ratio of M_1 and β is the W/L ratio of M_2. Here, the parameter θ shall be optimized to obtain a good linearity in current to voltage converting.

Another negative feedback loop is generated by OP_2 and M_5. The V_x depicted in Fig. 51 is set to be V_{ref} by the negative feedback loop, helping increase the accuracy of the current mirror pair of M_1 and M_2 apparently. R_0 in the current-to-voltage converter design also assists in maintaining the linearity during the current to voltage conversion. It makes the source voltage V_s of M_5 to follow V_s of M6 which corresponds to the output voltage. At last, the load resistor R_L senses the copied current out to an analog voltage signal.

The sensing accuracy of the proposed current-to-voltage converter is verified on a single device firstly and the result is shown Fig. 52. The reference voltage V_{ref} is set to *0.1V* for general sun-of-product exploration. A linear relationship between the output voltage I_{in} and input current I_{in} is obtained, that is

$$V_{out} = a\, I_{in} + b. \tag{5.7}$$

Where, $a = 2.47$ and $b = 0.004$, and the current is in an unit of mA. It indicates that current can be transformed to voltage signal by our proposed current-to-voltage converter with a linear

relationship. The offset of $b = 0.004$ is caused by the non-ideal property of the op-amps in real circuit implementation. Even though, a good sensing accuracy is still guaranteed by the linear relationship in sum-of-product. The output voltage has a range of $0 \sim 1V$ when input current varies from *0* to *0.4mA* in this implementation, while the input range can be extended with the same output range by adjusting the design parameter.

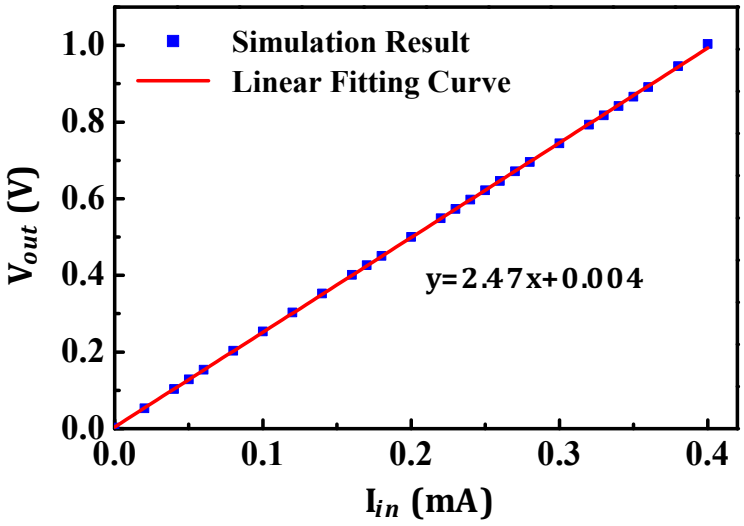

Figure 52: Sensing accuracy of the proposed current-to-voltage converter.

The sum-of-product computation accuracy of the proposed current sensing engine is evaluated with a 32×16 memristor crossbar. Results as shown in Fig. 53 indicates that the computation result $V_{o,j}$ follows a linear relationship with $\sum_{i=0}^{n-1} g_{i,j}\, v_i$:

$$V_{o,j} = k \times \sum_{i=0}^{n-1} g_{i,j}\, v_i + b^*. \tag{5.8}$$

where k is 2.46 and b^* is 0.012. The small difference between k and a is caused by the random noises caused by the crossbar. Ideally, Eq. (5.8) should follow the Eq. (5.5) in Section 5.6.1. Similar to Eq. (5.7), a non-zero parameter b^* exists for the same reason. And its value is slightly larger as memristor crossbar is involved. Based on these results, we can conclude that our proposed design has a good computation accuracy for the sum-of-products with memristor-based crossbar.

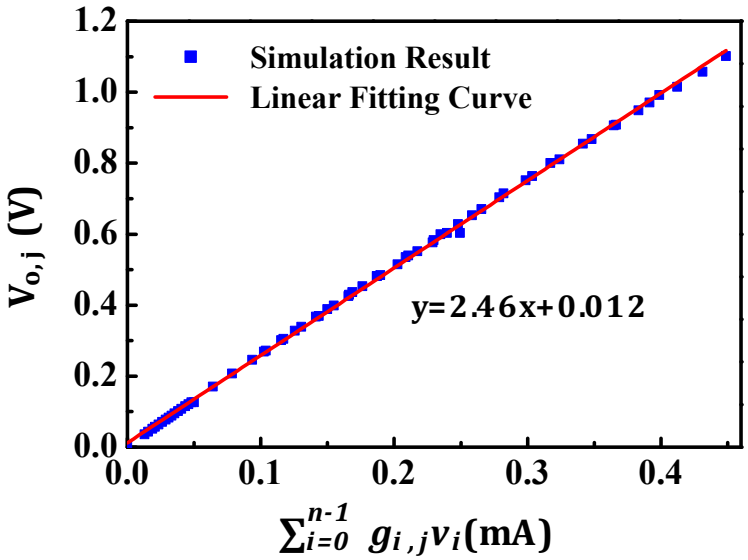

Figure 53: The computation accuracy of sum-of-product with proposed current sensing scheme.

5.6.4 Single Crossbar Utilization in Neuron Layer

We also explore the approach of utilizing a single crossbar to realize both positive and negative synaptic weights. The computation accuracy and the involved design considerations are analyzed and discussed. In artificial neural network model, a synaptic weight could be positive or negative. As discussed above, two memristor crossbars are commonly adopted and the weighted signals of the corresponding cells are subtracted with each other to represent negative weights. A lot more computation resources are introduced in such a scheme, including the extra crossbar, the doubled sensing circuits, and the additional subtraction circuitry. The design cost is especially higher in the multi-layer neural network design, as an additional computation flow with ADCs $\rightarrow$ DACs $\rightarrow$ Subtractors $\rightarrow$ ADCs are needed to generate analog inputs for the sequential hidden layers.

In this work, negative weighted signals are generated by the proposed current-to-voltage converters to save computation resource. Noticeably, the reference voltage V_{ref} in the proposed current-to-voltage converter is used to clamp the tail voltage of the crossbar BLs and suppress the sneak path leakage. Thus, the effective voltage across the memristor device would be $v_i^* = v_i - v_{ref}$. Different from the low v_{ref} setting above for general sum-of-products with all positive computation results, in this single crossbar for negative weighted generation, v_{ref} is set as the half

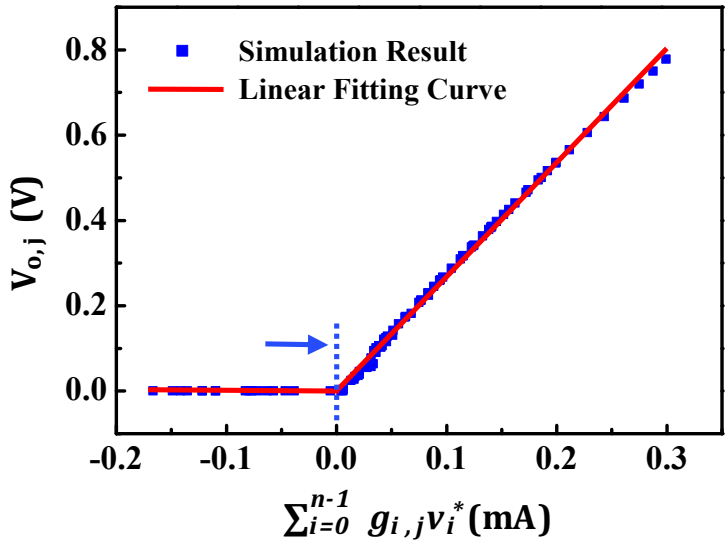

Figure 54: Computation accuracy of $\sum_{i=0}^{n-1} g_{i,j}\, v_i^*$ vs. $V_{o,j}$.

of V_{dd} $(0.6V)$ while input signals v_i are in the range of $0 \sim V_{dd}$ $(0 \sim 1.2V)$ with 16 analog voltage value (4 bits). Therefore, negative values of v_i^* will be generated when the inputs are lower than it, resulting in negative weighted signals of $g_{i,j}\, v_i^*$. Fig. 54 shows the relationship of the weighted signals $\sum_{i=0}^{n-1} g_{i,j}\, v_i^*$ and output of the current-to-voltage converter $V_{o,j}$, that follows

$$V_{o,j} = \begin{cases} k \times \sum_{i=0}^{n-1} g_{i,j}\, v_i^* + b, & \sum_{i=0}^{n-1} g_{i,j}\, v_i^* > 0 \\ 0, & \text{otherwise} \end{cases} \tag{5.9}$$

Here, $k = 2.69$ and $b = -0.0021$. A linear relationship is obtained when the output current of each BLs is positive, which guaranteed a good computation accuracy. Meanwhile, a grounded output voltage is generated when the sums of the weighted signals of a BL is negative. That is a *rectified linear unit* (ReLU) activation function without requiring any additional circuitry. In other words, extra circuits are not needed anymore to fulfill the activation functions in neural network implementation.

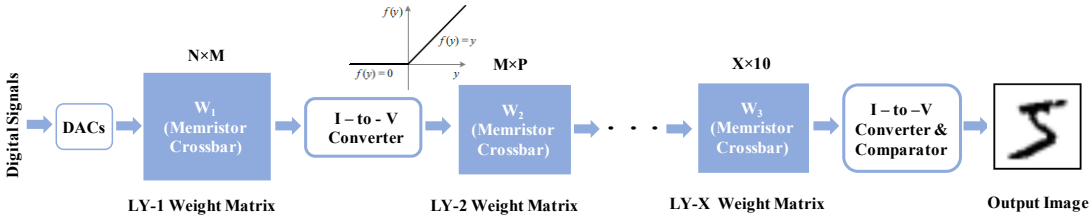

Figure 55: The system-level approach for a multi-layer feedforward neural network.

5.6.5 Neural Network Implementation and Evaluation

In this section, feedforward neural networks for MNIST handwritten digit recognition ($0 \sim 9$) are implemented by our proposed sums-of-product engine – adopting memristor-based crossbar for synaptic weight implementation and a current-to-voltage converter for neuron computing. The required computing resource and performance of the implemented neural networks for classification accuracy are evaluated. 60,000 digital patterns from MNIST were used for training, and a test set of 10,000 examples was selected randomly [59].

5.6.5.1 Feedforward Neural Network Implementation In the neuromorphic engine, digital images are encoded to inputs in floating-point formate and are then mapped to analog voltage signals with the closest possible amplitude. Thus *digital-to-analog converters* (DACs) are needed to generated the analog signals for input vector by digital signals from outside. Fig. 55 demonstrates the system-level approach for a multi-layer feedforward neural network implementation. A single-layer network is a simplified version with only one hidden layer weight matrix.

Each weight matrix layer is fulfilled by memristor-based crossbar and the weight is represented by the conductance of each memristor cell. During computation, analog voltage signals from DACs are given to the memristor crossbar simultaneously, forming the input vector. Weighed signals by dot productions from each memristor cell are summed and integrated at the end of BL of the crossbar. The computation current results from the crossbar will be injected to a *current-to-voltage converter* (I-to-V Converter) at the end of each BL.

Here, four important functions are implemented by the proposed current-to-voltage converter. Firstly, the current amplifier in the current-to-voltage performs like a buffer that isolates the crossbar array from the current-to-voltage procedure. It helps eliminate sneak-path leakage and grantee good computing accuracy as discussed in 5.6.2. Secondly, the negative besides positive weighted values can be generated by applying a higher reference V_{ref} to the current-to-voltage converter like half of V_{dd}, while providing inputs with a full $0 \sim V_{dd}$ input range. Therefore, only one memristor crossbar is needed in each layer without expecting much classification accuracy loss. Thirdly and very importantly, the injected current from the memristor crossbar is converted to analog voltage signals that can be sensed out in a single-layer design or transmitted to the next layer in a multi-layer design. Additionally, a *rectified linear unit* (ReLU) activation function can be implemented by the current-to-voltage converter as the simulation results in Section 5.6.2 indicate.

Therefore, a positive analog signal following a linear relationship with the output current from the crossbar are generated through an *I-to-V Converter*. In our design, this analog signal is transmitted forward to the next hidden layer and acts as its input signals. A sample-and-hold circuit is adopted here for maintaining signal stability and controlling timing. After the final neural layer with memristor crossbar, a positive analog voltage signal is given to an analog comparator. It helps capture the maximum output from the last layer by outputting it as logic 1 while others to be zero. As such, an entire computation is implemented and corresponding output image is obtained.

In this work, two feedforward neural networks are implemented by using GlobalFoundry 130nm technology: one is a single-layer neural network and the other is a two-layer neural network. 4-bit DACs are implement on this technology node and used in system implementation. The memristor has a resistance range of $10K\Omega \sim 500K\Omega$ with 16 analog levels (4 bits). Digit image classification accuracy of our proposed system are evaluated under different design considerations.

5.6.5.2 Classification Accuracy Evaluation As is discussed above, input vectors with floating-point values extracted from the image are mapped to discrete analog values that can be generated by DACs. Therefore, the resolution of DACs, meaning the analog voltage levels the DACs can provide, will affect the image recognition accuracy. In the neural network implementation, weights with continuous values obtained from software training are mapped to the memristor cell in the crossbar. However, available memristance states that can be programmed successfully with good

stability are limited. This discrete weight values resulting from the memristor cell will also influence classification accuracy. Moreover, the crossbar size and wire resistance in the crossbar in real implementation will also have an impact on the classification accuracy. All of these parameters have been considered and investigated in our neuromorphic system implementation.

5.6.5.3 Single-layer Neural Network The single-layer neuromorphic design is evaluated firstly with different input image size and input bits consideration. The input image size here refers to the real image size adopted in the implementation, corresponding to the crossbar size in WLs direction. For instance, a 784×10 crossbar is needed for the 28×28 original image size from MNIST data base without compression in the single-layer design. To reduce design cost in using such a large size crossbar, images are usually compressed to make smaller size crossbar available in the image recognition while not decrease computation speed. Here, the effect of this compression on image recognition error rate is investigated.

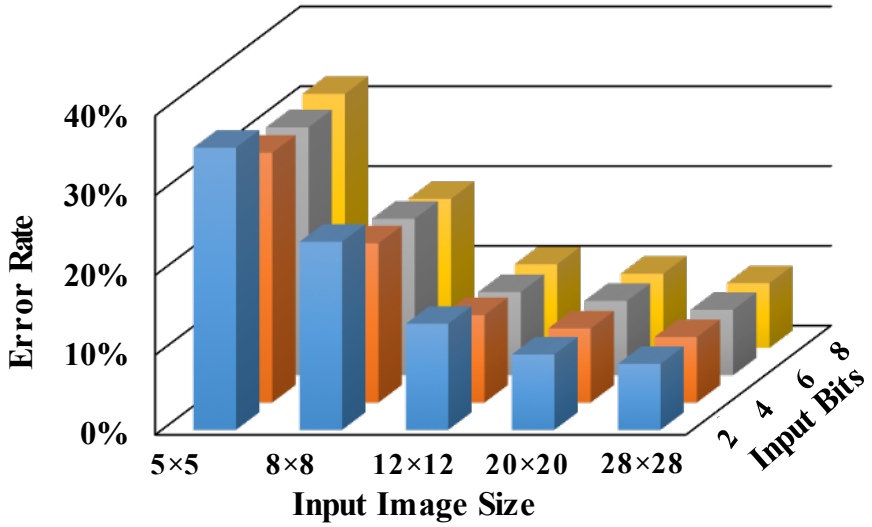

Figure 56: The MNIST recognition error rates of the single-layer system when applying different input image size and input bits.

Fig. 56 depicts the image recognition error rate when varying the image size from 5×5, 8×8, 12×12, 20×20, to 28×28, as well as increasing the input bits from 2, 4, 6, to 8. Each memristor cell can be programmed to a 4-bit value in this evaluation. The results indicate that the image recognition error rates increase with the decrease of input image size, especially when the image

size is smaller than 12×12. This is because smaller image size involved in heavier compression, resulting in worse image resolution. The error rate decrease is extremely small when the image size is larger than 12×12. The error rate dependency on input bits is also illustrated in Fig. 56. Identical to the discussion above, different input bits will bring in different error rates. However, there is nearly no error rate decrease when input bits are larger than 4-bit, especially when the input image size is large.

Overall, a 89.5% image recognition accuracy can be obtained in the 12×12 input image size with 4-bit input bits. With the original 28×28 input image size and an 8-bit input bits, there is a 91.9% accuracy rate in our implementation. 12×12 input image size and 4-bit input bits are adopted in our design and used in the following evaluation.

5.6.5.4 Two-layer Neural Network In our work, a two-layer neural network is also implemented and tested. Different with the crossbar size in WLs direction in Section 5.6.5.3, the impact of crossbar size in BLs direction in the two-layer design is evaluated with wire resistance is involved in. The error rates in different crossbar size without or with wire resistance consideration are demonstrated in Fig. 57. The wire resistance is considered here because it will decrease the access voltage and contaminate the computing accuracy in memristor crossbars. With the increase size in the BLs direction of the crossbar, its affect will be more severe. For the 130nm technology adopted in the work, the wire resistance per cell is about 0.52Ω.

Results in Fig. 57 presents a large error rate decrease from 14.5% to 3.4%, when increasing the crossbar size from 144×20 to 144×80, without considering wire resistance. However, error rate decrease is not obvious once the crossbar size reaches 144×80 and a high recognition accuracy can be obtained. Comparing the error rate with wire resistance considered to the result without wire resistance consideration, not much increase is observed. This is because the wire resistance is much smaller than the memristor resistance with $R_{on} = 10\text{K}\Omega$ and $R_{off} = 500\text{K}\Omega$. Moreover, resistance states mostly concentrate in the high resistance level in the training, decreasing the impact of the wire resistance furthermore.

The impact of memristance bits are also evaluated with involving in circuit performance. The results in Fig. 58 show that there is nearly no error rate increase when the resistance bits is larger than 5. However, a large error rate increase occurs when the resistance bits continues to be de-

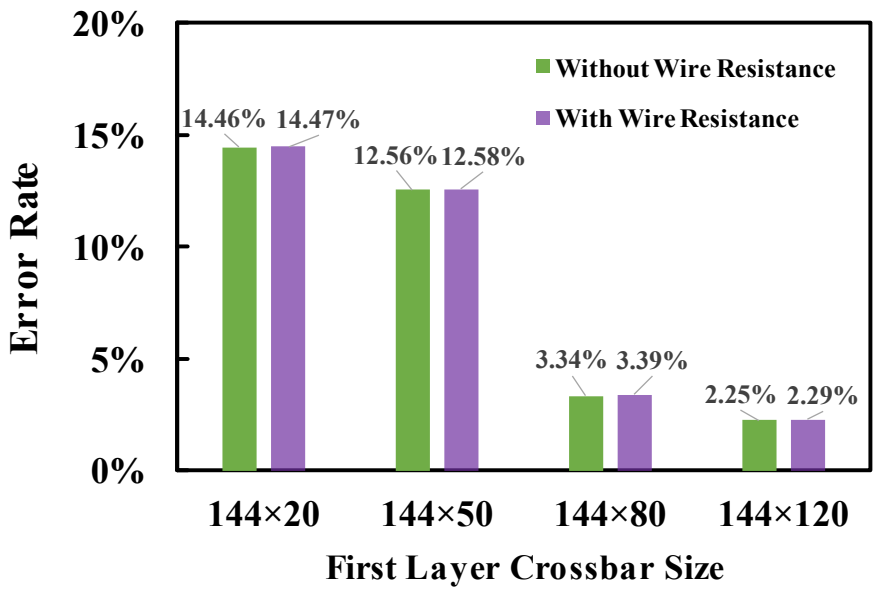

Figure 57: The MNIST recognition error rates in different crossbar size without or with wire resistance consideration.

creased lower than 5 bits. As discussed in Section 5.6.5.3, the current-to-voltage converter has a good linearity guaranteeing good computation accuracy. Therefore, the circuit variation is majorly introduced by the memristor and the DACs. In evaluation, a $\pm 5\%$ variation of both of the memristor and DACs are considered, which causes slight difference.

5.7 SUMMARY

In this Chapter, a memristor crossbar based computing engine using current sensing was proposed for high speed and accuracy. A parallel execution of sums-of-product computation is implemented by adopting a memristor-based crossbar and applying analog voltages to each wordline of the memristor crossbar simultaneously. Two current sensing scheme are proposed. First, a current amplifier circuit was designed which mirrors the current from the crossbar with a slight accuracy degradation, and then the current is computed by the integrate-and-fire circuit (IFC). The system demonstrates a good computation accuracy in matrix computation with a lower energy consumption. Moreover, a matrix-vector computation system is designed on a chip.

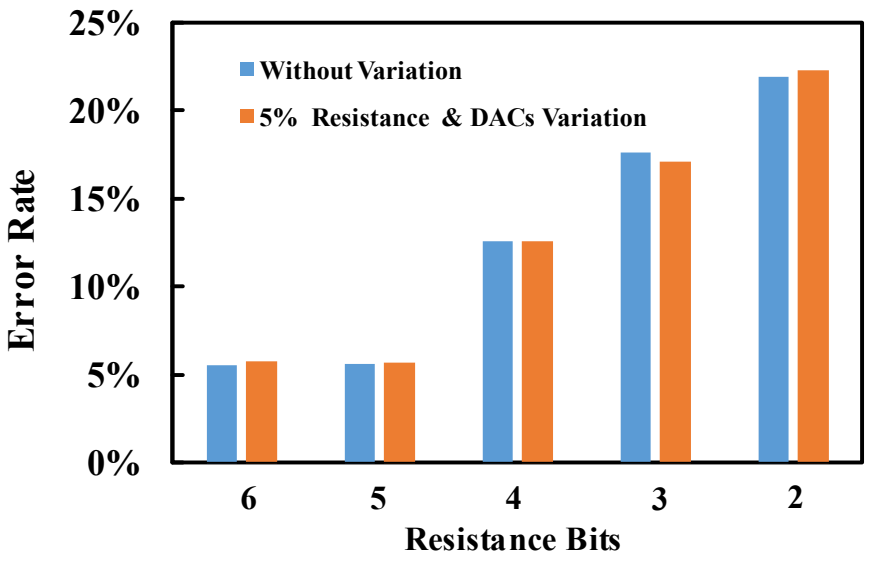

Figure 58: The MNIST recognition error rate with different resistance bits of memristor cell.

Second, A current-to-voltage converter based on a current amplifier is designed which converts the output current from the crossbar to analog voltage without computing accuracy loss. Moreover, a rectified linear unit activation function is implemented by using the current-to-voltage converter. A hardware implementation flow based on the proposed design approach is built for multi-layer as well as single-layer feed-forward neural network. Two neural networks including a single-layer system and a two-layer neural network are implemented for MNIST handwritten digit recognition. Various design parameters are considered and their impact on classification accuracy are evaluated. The proposed computing system demonstrates good classification accuracy: a maximum computation accuracy of 91.8% can be reached in a single layer design while a 96.12% computation accuracy value is obtained in the two-layer design. Overall, the proposed memristor-based neuromorphic engine demonstrates a good computation accuracy in neural network applications.

6.0 RESCUING MEMRISTOR-BASED NEUROMORPHIC DESIGN WITH HIGH DEFECTS

As is talked in the above chapters, memristor-based synaptic network has been widely investigated and applied to neuromorphic computing systems for the fast computation and low design cost. However, as memristors continue to mature and achieve higher density, bit failures within crossbar arrays can become a critical issue. These can degrade the computation accuracy significantly. In this Chapter, we propose a defect rescuing design to restore the computation accuracy. In our proposed design, significant weights in a specified network are first identified and retraining and remapping algorithms are described.

In this chapter, we will present the details of the proposed rescue neuromorphic design as the follows: Section 6.1 gives the development status of the memristor crossbar and analysis of impact of the single-bit defects; Section 6.2 presents the details of the defect rescuing design methodology. Section 6.3 evaluates the performance and robustness of the proposed rescue design in feedforward neural networks; At last, we summary the work in this chapter in Section 6.4.

6.1 OBSERVATION & MOTIVATION

6.1.1 Random SBF in a Memristor Array

In a neuromorphic application of memristor crossbar arrays, multiple stable conductance states are necessary for each memristor to represent synaptic weights in neural networks. Reported by HPE Labs recently, at least 64 conductance levels (6 bits) can be successfully programmed in Tantalum Oxide (T_aO_x) based crossbar arrays in a one-transistor-one-memristor (1T1M) design [61]. Figure 59(a) shows a 44×44 pattern in a 64×64 1T1M array It can be seen that the SBF defects

distribute randomly across the array and blur the programmed pattern. In fabrication, memristor arrays demonstrate very different defect patterns and yields. Among all the measured arrays, this example has the lowest yield of 84%. However, across the full lifetime of a memristor array used for neural network inference and training, the memristor cells can be heavily exhausted and damaged by aggressive programming and testing cycles.

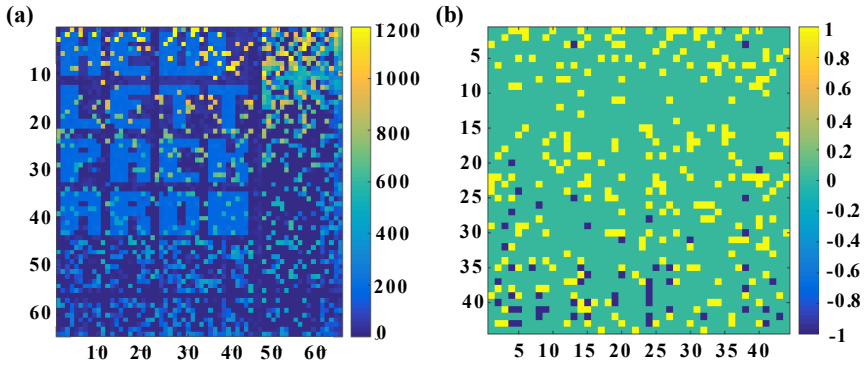

Figure 59: (a) The conductance distribution (in μS) and (b) the measured stuck-on ($+1$) and stuck-off (-1) defects of a 64×64 memristor crossbar array.

Figure 59(b) shows the stuck-on/off SBF distribution of the 44×44 sub-region. For ease of illustration, stuck-on defects are displayed as $+1$ and stuck-offs are presented as -1. It is worth pointing out that a defect is not fixed to the highest conductance g_H or the lowest value g_L. As our design utilizes the full analog conductance range of memristor, a device is taken as defect when its g_{error} is out of tolerance after programming, where $g_{error} = g_{final} - g_{target}$ denotes the conductance difference between the real programmed and the target values. In this work, a device with $g_{error} > 30\mu S$ is considered as a stuck-on. The measurement shows that the conductance of stuck-on devices ranges from $330\mu S$ to $1200\mu S$. The devices with $g_{error} < -30\mu S$ are defined as stuck-offs, most of which demonstrate less than $1\mu S$ conductance value. For the example in Figure 59(b), 18.4% of defects are stuck-off and the remaining are stuck-on.

The measurement data depicts a random distribution of the SBF defects. When utilizing a memristor-based functional unit for data storage, the data at defective cells can still be read and corrected if necessary. Traditional methods such as redundancy and error correcting codes (ECC) can effectively solve these issues for storage. For in-memory computations, however, the situation is more complex and errors in the conductances aggregate in a non-trivial manner, requiring new correction schemes.

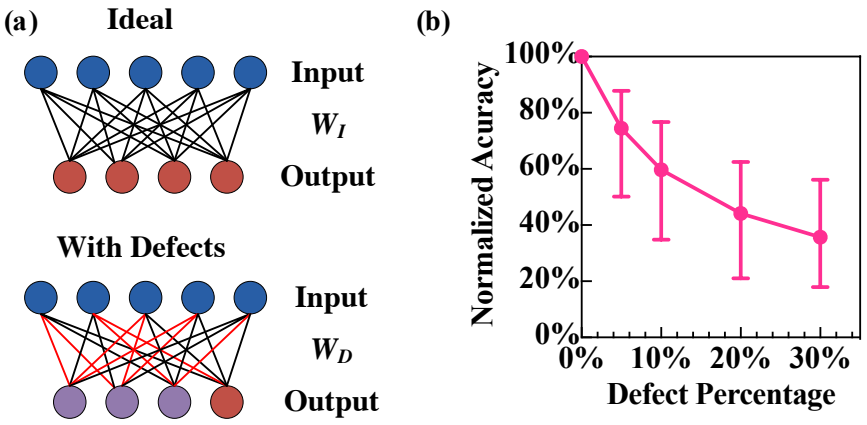

Figure 60: (a) An illustration of synaptic weight defects; (b) the impact of SBF on the classification accuracy of a two-layer neural network.

6.1.2 Impact of SBF on Neuromorphic Systems

We take a two-layer neural network for MNIST classification as the example to evaluate the impact of random SBF defects on the memristor-based neuromorphic design. Figure 60(a) depicts the network model which consists of an input layer, a fully-connected synaptic weight matrix W and an output layer. The well-tuned floating-point values of the synaptic weight matrix W_I will be mapped to the analog conductances that a memristor array can afford (e.g., 64 levels). Besides the slight precision loss during the mapping, unpredictable defects shall be considered. So we generate defect matrices W_D randomly and apply them to test the impact of SBF on the system performance.

Figure 60(b) shows the classification performance statistics when 5%$\sim$30% stuck on/off SBF are injected in a 784×10 memristor-based array. For each setup, the impact of defects is evaluated on 1,000 test cases that are generated by randomly spreading defects. Without defects, the network achieves a classification accuracy of 92.64%. The normalized accuracy rate defined as Acc_{real}/Acc_{ideal} in Figure 60(b) demonstrates the significant performance degradation due to defects: adding 10% SBF defects results in an average normalized classification accuracy rate of 59.7%, that is, 55.3% real classification accuracy for MNIST dataset.

Our observation of the large performance degradation here is different from previous research conclusion that comparable accuracy can be retained with weights penalties in deep neural networks (DNNs) [62, 63]. In the methods such as L1/L2 regularization and dropout, only those weights with near-to-zero parameters are penalized as they are less important. For example, the near-to-zero synaptic weights could be removed through network sparsification to realize recognition systems with high computation accuracy and efficiency [64, 65]. The SBF defects, however, can occur anywhere in a memristor array. Many of them could greatly affect the network performance. Moreover, the sparsification method completely removes some connections by forcing the corresponding matrix entry to 0. A defect could still contribute certain error to the computation result.

6.2 DESIGN METHODOLOGY

We propose a *defect rescuing design methodology* to improve the efficiency of memristor-based neuromorphic system. The impact of synaptic weights on system performance first is quantified and analyzed. We then develop a retraining algorithm in network learning and a remapping scheme in hardware implementation.

6.2.1 Weight Significance

We first quantify the impact of each weight on system performance, denoted as *weight significance*. Back-propagation is the key step in network training, which updates synaptic weights according to the errors received at neurons. A gradient decent algorithm is usually adopted in which the weight updating process can be formulated as [66]:

$$\Delta w_{j,i} = -\eta \frac{\partial E}{\partial w_{j,i}} = \eta f' \sum_{k \in downstream(j)} \left(w_{k,j} \frac{\partial E}{\partial w_{k,j}} \right) x_{j,i}. \tag{6.1}$$

Where E represents the global error, η is the learning rate, f' denotes the deviation of the activation function of the hidden layer, $w_{j,i}$ is the weight associated with the *i*th input to neuron *j*, and $\Delta w_{j,i}$ is the updated weight computed by propagating the error back from the downstream units *j*.

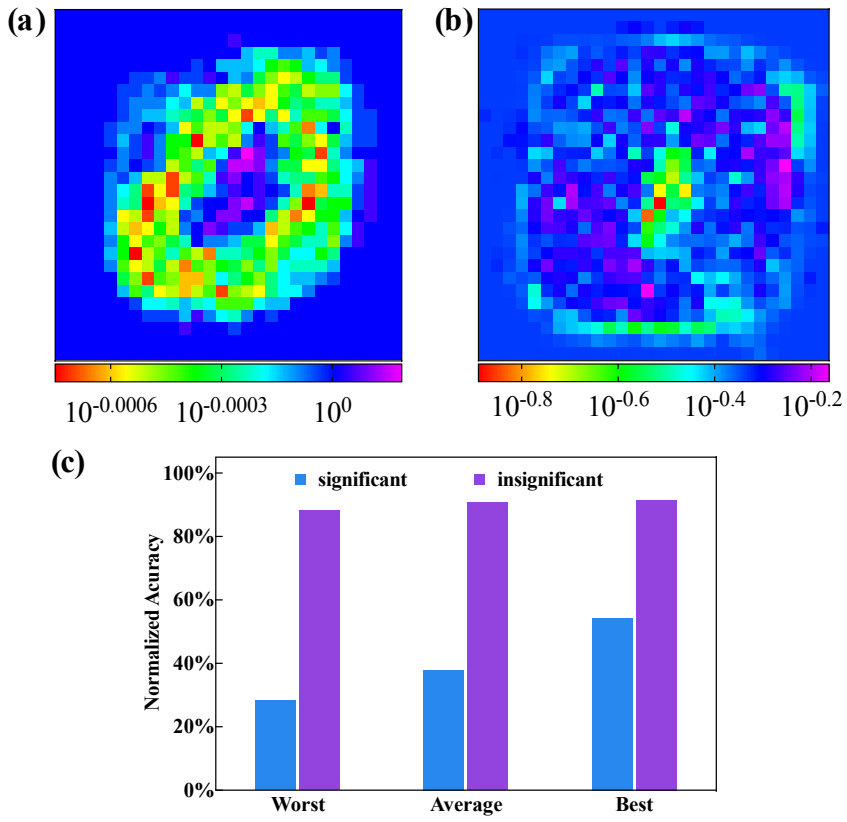

Figure 61: (a) Acc_{real}/Acc_{ideal} from experiments; (b) $\partial E/\partial w_{j,i}$ during training; (c) Accuracy degradation due to defects on significant and insignificant cells.

During the learning, each weight demonstrates different sensitivity to E, resulting in different Δw. Therefore, we can analyze the sensitivity of $w_{j,i}$ through network training, i.e., $\partial E/\partial w_{j,i}$, to classify the weight significance. The weight significance can also be ascertained directly from experiments by inserting defects in a well-tuned network, e.g., the two-layer neural network for MNIST classification in this work. When inputing an image to the network, we insert a single-bit defect and compare the real and the ideal accuracy rates Acc_{real}/Acc_{ideal}.

Figure 61(a) shows the statistical result of the network's first layer, the trend of which is consistent to the normalized sensitivity $\partial E/\partial w_{j,i}$ in Figure 61(b). We divide all the weights into *significant* and *insignificant* groups based on the value of $\partial E/\partial w_{j,i}$ and re-characterize their impacts at the system level. Figure 61(c) shows the accuracy degradation that induced respectively by the significant and insignificant weights when 10% SBF is considered. Here, 55% weights

are characterized to be insignificant with a threshold t of $\partial E/\partial w_{j,i}$. 1,000 random test cases are utilized and results in three scenarios are depicted in the Figure 61(c): *best*, *worst*, and *average* correspond to the highest accuracy, the lowest accuracy, and the mean of the highest and lowest values, respectively. As expected, defects on significant weights dramatically affect the system performance. Oppositely, insignificant weights are more tolerable to defects. It indicates that defects in some of the classified insignificant weights can be tolerated by network itself and induce negligible impact in accuracy degradation.

6.2.2 Network Retraining

Retraining is usually carried out to optimize the recognition accuracy after the a sparsification step of DNNs [64]. In this work, a hardware specific retraining methodology is developed to recover the accuracy loss induced by SBF defects.

Through a normal training process for a neural network, parameters are learned and the weight matrix is generated for a given application. Memristor cells in the crossbar array will be programmed to the corresponding conductance values, determined by the mapping algorithm in the memristor-based neuromorphic system. Some memristor cells will then be identified as stuck at certain conductance levels and not adjustable. Therefore, incorrect outputs are generated from the stuck weights. Our proposed retraining methodology attempts to recover the accuracy by re-tuning all remaining weights that are adjustable. Figure 62 illustrates a simple recovery model that is assisted with other trainable weights. The network retraining includes two major steps:

- *Weight initialization*: Instead of assigning random values to synaptic matrices, the pre-trained weight matrix W_{Ideal} will be used to initialize and accelerate the retraining process. The defect map obtained from chip testing will be applied, providing the initial values to the defected cells.
- *Weight updating*: Backpropagation is adopted to update the synaptic weights. In the retraining, $\Delta w_{j,i}$ at a defected location will be forced to be 0 so $w_{j,i}$ remains unchanged in iterations.

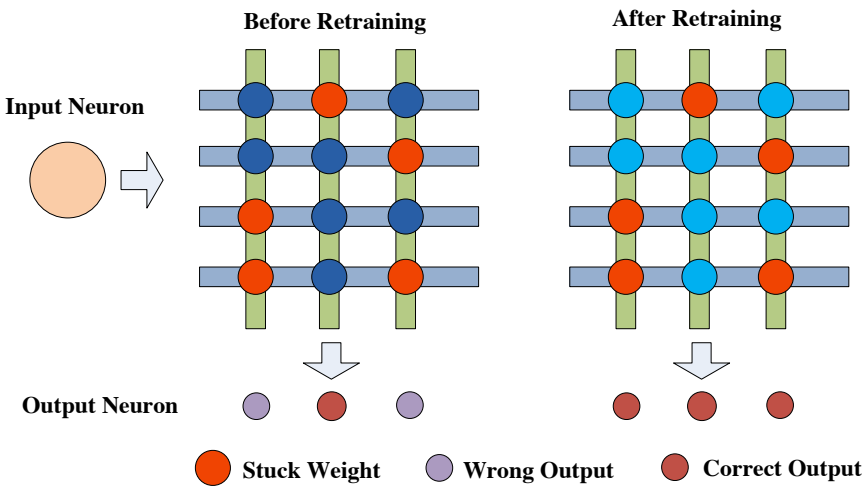

Figure 62: The illustration of the weight distributions before and after retraining.

In this way, the behavior of the memristor array with some non-adjustable weights is mimicked in the retraining. Weights with defects maintain constant values according to the measured mapping. Accuracy is then recovered by updating the trainable weights.

6.2.3 Defect Rescuing Design Flow

Based on the weight significance analysis and the network retraining, a defect rescuing flow presented in Figure 63 was developed for the memristor-based neuromorphic systems.

First, a pre-training procedure is executed. This targets an initial weight matrix W_{Ideal} through normal neural network training consisting of forward and back-propagation. W_{Ideal} will then be mapped to a conductance matrix implemented on a memristor array as described in [60]:

$$G \rightleftharpoons \alpha \cdot W + \beta, \tag{6.2}$$

where, α and β are two linear mapping coefficients. These follow the relationship $\alpha = (g_H - g_L)/(w_H - w_L)$ and $\beta = g_H - \alpha \cdot w_H$. $[g_L, g_H]$ is a selected conductance range for a linear computation in matrix-vector calculations. w_L and w_H are the minimum and maximum synaptic weight values in the well-tuned W_{Ideal}. In this way, a conductance within the range is normalized to a weight in $[-1, +1]$. Through array testing, defect information including stuck-on and -off conductances and their locations is obtained, forming a defect conductance matrix G_{defect}. Again, by following Eq. (6.2), G_{defect} will be translated to matrix W_{defect}. The recognition accuracy will be retested after considering W_{defect}.

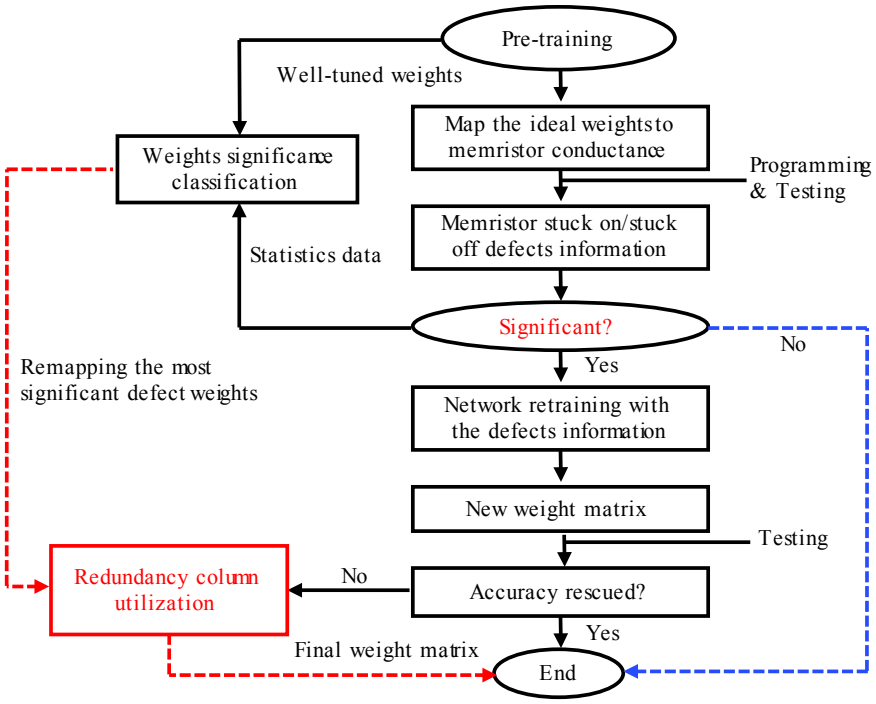

Figure 63: The proposed defect-rescuing neuromorphic design flow.

In accuracy checking, it is possible that only defects occur at insignificant weights. As shown in Figure 61(c), this results in negligible accuracy loss. In such a case, the retraining procedure may be omitted. Of most interest is when a certain accuracy degradation is caused by SBF defects. The proposed retraining is executed to rescue the accuracy loss by generating a new weight matrix, including the existing SBF defects. Then, the neural network is tested again with the new weight matrix. If the accuracy is recovered successfully after the retraining, the weights are finalized.

In the worst-case scenario when there are too many defects or the performance loss cannot be compensated by retraining, a remapping algorithm utilizing redundant memristor columns will be used. This additionally requires peripheral circuits such as the TIA and sample-and-hold blocks to support re-routing and operation from the redundant columns. In contrast to redundancy in the memory domain, only a small portion with the most significant defect weights will be mapped to redundancy columns in neuromorphic computing systems.

6.3 EVALUATIONS

In this work, the efficiency of the defect rescuing design is evaluated on feed-forward neural networks for MNIST handwritten digits classification [59]. Two networks with two-layer and three-layer structures are trained and tested. Here, 60,000 digital patterns from MNIST are used for training, and a test set of 10,000 examples are selected randomly.

One weight matrix *W* is included in the two-layer network with the array size of 784×10. The implementation of the three-layer neural network utilizes *W1* with the array size of 784×256 between the input layer and the hidden layer, and *W2* with the array size of 256×10 between the hidden layer and the output layer. The two networks obtain 92.64% and 97.82% classification accuracy at the software level, respectively.

Through neural network training, the synaptic weights of any array, i.e., W_{Ideal}, can be obtained and mapped to the conductances of memristor arrays, i.e., G_{Ideal}. The memristor normal operation range $[g_L, g_H]$ is set to be $[1\mu S, 300\mu S]$ to guarantee a linear matrix-vector computation based on the measurement data. The stuck-on and -off conductance ranges are respectively considered to be $[300\mu S, 1200\mu S]$ and $[0.01\mu S, 1\mu S]$ based on the measured worst-case conductances. As mentioned in Section 6.2.3, a defect weight matrix W_{defect} can be obtained from the measured G_{defect}. Based on this information, the impact of the SBF defects are evaluated in the two feedforward neural networks and our described defect rescuing flow is adopted to restore the accuracy.

6.3.1 Robustness of Retraining

Figure 64 summarizes the normalized accuracy defined as Acc_{real}/Acc_{ideal}, the ratio of the retrained accuracy with SBF defects compared with the ideal accuracy without defects. The results before and after applying the proposed retraining process on the two-layer and three-layer feedforward neural networks are presented. Under each condition, the result is obtained from 1,000 test cases of defects at random locations, stuck mode, and conductance values.

In the two-layer neural network, large accuracy degradation can be observed due to the random stuck-on/off SBF. The degradation increases with number of defects as shown in Figure 64(a). For example, only 42.5% accuracy can be achieved on average when considering 20% defects.

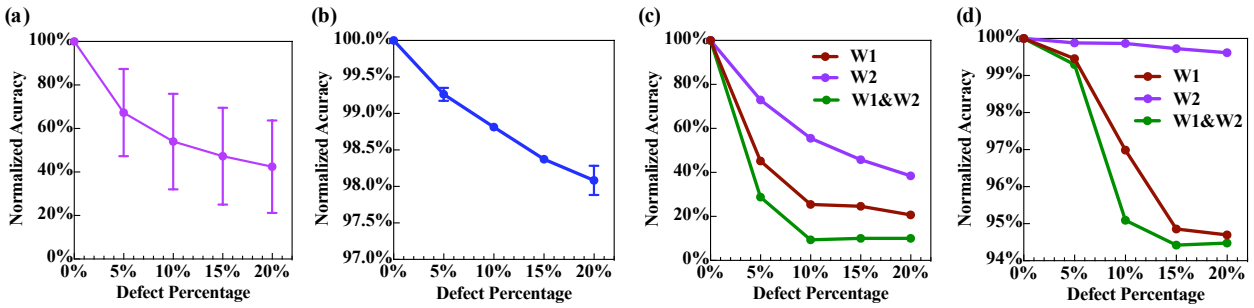

Figure 64: The impact of random stuck-on/off defects (a) and the recovered accuracy after retraining (b) in the two-layer neural network; The impact of random stuck-on/off defects to *W1* and *W2* (c) and the recovered accuracy after retraining (d) in the three-layer neural network.

The corresponding lowest and highest accuracy rates in the 1,000 tests are 21.2% and 63.7%, respectively. The reason for such a large variation between the best and worst cases is the highly varying significance of the defect weights, as previously discussed in Section 6.2.1. The retraining result for the same network is presented in Figure 64(b). This shows that the worst-case accuracy of 21.2% is recovered to 97.9%. On average, the accuracy can be rescued to 98.8% and 98.1%, considering 10% and 20% random SBF defects respectively. Moreover, the variations become much smaller after applying our retraining—less than 0.4%.

Figures 64(c) and (d) show the impact of the SBF on the three-layer network before and after retraining in three scenarios, assuming a certain percentage of defects only in *W1*, only in *W2*, or in both *W1* and *W2*. Similarly, 1,000 random cases are tested and the average accuracy is presented. Figure 64(c) indicates that *W1*, i.e., the weight matrix between the input layer and the hidden layer is more sensitive to defects than *W2*. This is because *W1* is designed to learn image features so as to have more severe impact [66]. As expected, the largest accuracy degradation happens when both *W1* and *W2* have defects. At a defect rate of 20%, the network shows only 10% classification accuracy. Utilizing the proposed retraining can recover the accuracy to 94.5% from the most destroyed network. We also observe that *W2* has higher resilience, with a retained accuracy of 99.6% compared to the ideal result without defects. The rescuing ability of retraining

is dominated by the more significant weight matrix *W1*. The results demonstrate that the retraining is robust and efficient in rescuing the accuracy loss caused by the random SBF.

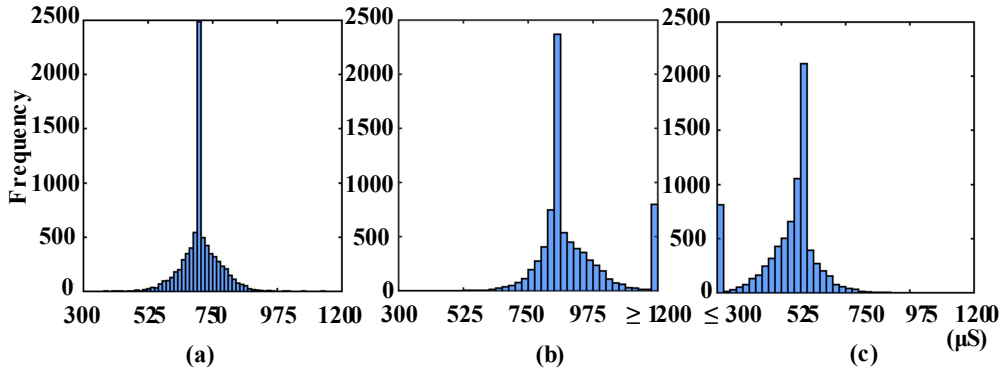

Figure 65: (a) The distribution of G_{Ideal}; (b) The conductance distribution with stuck-on defects after retraining; (c) The conductance distribution with stuck-off defects after retraining.

Figure 65 shows the conductance distributions of the two-layer network. Specifically, Figure 65(a) is the conductance distribution without defects, and Figure 65(b,c) show the weight distributions after retraining with 10% stuck-on or stuck-off defects. By comparing the distributions, we note that the trainable weights are re-tuned to compensate for the error caused by defects, consistent with our approach described in Section 6.2.2. The values of trainable weights shift toward the larger side when inserting stuck-on defects while it shifting to smaller when including stuck-off defects. In contrast to other work [67], our approach is able to accommodate both types of defects simultaneously with no observed challenges compared to a single type of defect.

6.3.2 Resilience of (In)significant Weights

Synaptic weights in a neural network can be characterized to be significant or insignificant, yielding different sensitivity to defects. Figure 66(a) shows that insignificant weights have lower accuracy degradation. For example, when 20% defects all fall at significant or insignificant weights, the test on significant weights shows 37.5% more degradation in accuracy. Here, 55% weights are classified to be insignificant when taking the two-layer network as the example.

In both cases, retraining leads to improved accuracy. But Figure 66(b) shows that the insignificant weights have better resilience to defects. By retraining as described here, the normalized accuracy can be recovered to 99.9% even with 30% defects in the insignificant weights. Correspondingly, the normalized accuracy can be recovered to 95.1% when 30% defects happen at the significant weights.

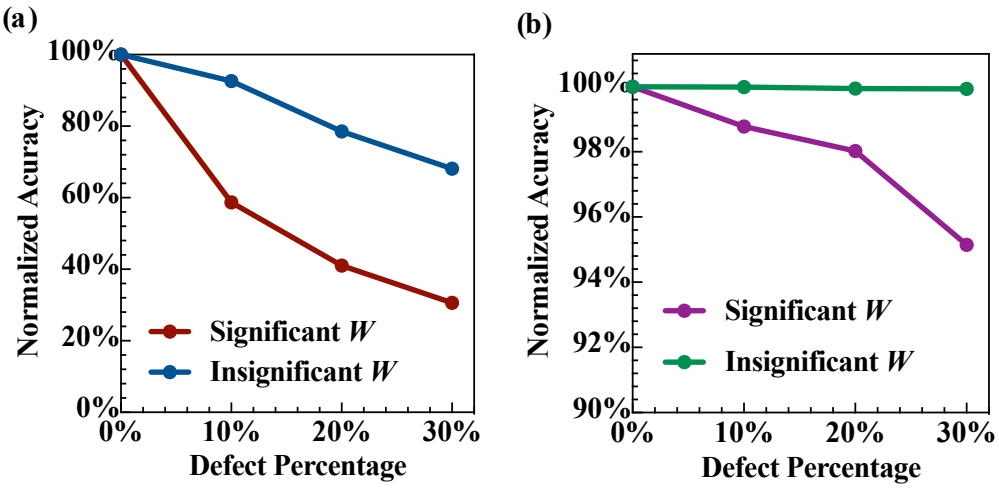

Figure 66: (a) The accuracy degradation in the significant and insignificant weights; (b) The ability to recover accuracy for defects at significant and insignificant weights through retraining.

6.3.3 Redundancy Memristor Design

The above evaluations prove that most of the accuracy, e.g., 98.1%, can be recovered by the retraining algorithm even with 20% SBF defects. In this work, a remapping process assisted with an efficient redundancy design is also developed to address the case where SBF defects arise at many significant weight locations that cannot be fully recovered by retraining. Figure 67(a) shows the illustration of the redundancy scheme: columns that are heavily polluted by defects will be replaced by additional memristor columns with a remapping algorithm. And outputs resulting from the new columns will be selected and utilized for the next step computation.

The results in Figure 66(b) prove that the defects with low significance have better resilience on retraining. Therefore, only the most significant defects weights are considered to be remapped to redundancy columns to decrease the design cost while improving the accuracy efficiently. Figure 67(b) shows the results when $0\% \sim 5\%$ of the most significant defects are remapped to the redundancy columns that without significant defects at 20% SBF defects, again taking the two-layer network as the example. The results show that 99.3% accuracy can be obtained with 5% significant defects being remapped, increasing from the 98.1% that was restored by retraining only. It is also observed that the accuracy improvement flattens out going from 4% to 5%, as the defects remapped are increasingly less significant. Hence, our proposed redundancy scheme is able to improve accuracy with minimal redundancy and design cost.

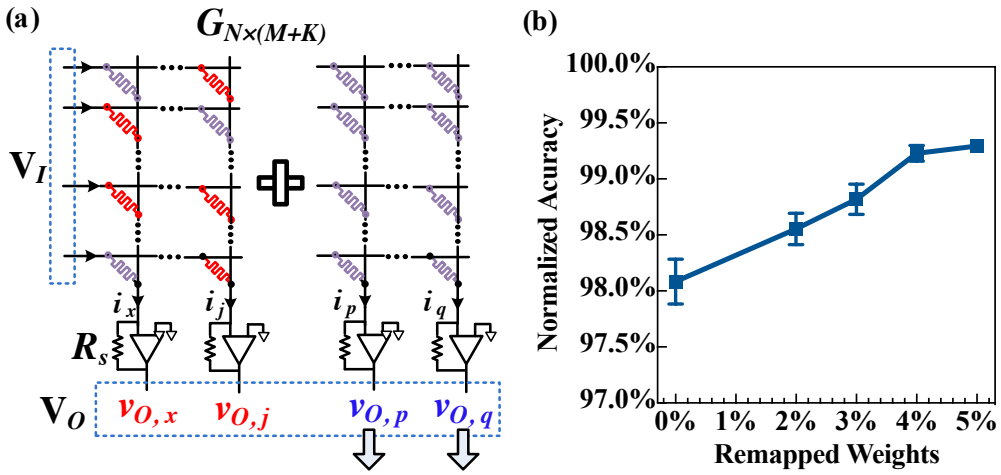

Figure 67: (a) A simple redundancy scheme; (b) Recovered accuracy with significant defects remapping at 20% SBF defects.

6.4 SUMMARY

The computation accuracy for memristor-based neuromorphic systems can be degraded significantly by random defects across memristor arrays. In this work, we proposed a defect rescuing design to effectively restore the accuracy. The proposed design has three major aspects including a weight significance categorization, a robust retraining algorithm, and an efficient remapping process. Basing on experimental device testing data in memristor arrays, the rescuing ability of our proposed design was evaluated in feed-forward neural networks for MNIST digit recognition. Considering 20% random single-bit defects, our proposed retraining process recovered the recognition accuracy to 98.1% and 94.5% from 42.5% and 10% in the two-layer and three-layer feed-forward networks, respectively. Additionally combining this with a remapping process, 99.3% accuracy can be achieved overall by remapping only 5% of the most significant defects to redundant columns in the two-layer network having 20% defects.

7.0 CONCLUSION

In my PhD career, my research work started with memristor devices modeling and related circuits design in resistive memory (ReRAM) technology by investigating their physical mechanism, statistical analysis, and intrinsic challenges. I successfully fulfilled neuromorphic computing systems by leveraging memristor devices and algorithm scaling in neural network and machine learning algorithms based on the similarity between memristive effect and biological synaptic behavior. At a higher level, I worked on the application-specific optimizations for further reliability improvement of the developed neuromorphic systems. From my work on the bran-inspired computing, I acquired a deep understanding of emerging technologies and the non-von Neumann computer. As well, I gained extensive experience in VLSI design by leading several Neuro-chips Tape-outs and implementing the on-chip designs.

BIBLIOGRAPHY

[1] A. Flocke and T. G. Noll, "Fundamental Analysis of Resistive Nano-Crossbars for the Use in Hybrid Nano/CMOS-Memory," in *European Solid State Circuits Conference (ESSCIRC)*, September 2007, pp. 328–331.

[2] C. Liu *et al.*, "A Spiking Neuromorphic Design with Resistive Crossbar," in *DAC*, July 2015, pp. 1–6.

[3] M. Hu *et al.*, "Hardware realization of BSB recall function using memristor crossbar arrays," in *DAC*, 2012, pp. 498–503.

[4] P. Simon, *Too Big to Ignore: The Business Case for Big Data*. Wiley, 2013.

[5] D. C. Ciresan *et al.*, "Convolutional Neural Network Committees for Handwritten Character Classification," in *2011 International Conference on Document Analysis and Recognition*, Sept 2011, pp. 1135–1139.

[6] V. Vanhoucke *et al.*, "Improving the speed of neural networks on CPUs," in *Deep Learning and Unsupervised Feature Learning Workshop*, 2011.

[7] S. M. Qasim *et al.*, "FPGA design and implementation of dense matrix-vector multiplication for image processing application," in *IJCSNS*, October 2010.

[8] W. A. Wulf and S. A. McKee, "Hitting the Memory Wall: Implications of the Obvious," *ACM SIGARCH computer architecture news*, vol. 23, no. 1, pp. 20–24, 1995.

[9] C. Mead and M. Ismail, *Analog VLSI implementation of neural systems*. Springer, 1989.

[10] J. Schemmel *et al.*, "A wafer-scale neuromorphic hardware system for large-scale neural modeling," in *ISCAS*, May 2010, pp. 1947–1950.

[11] S. Mitra *et al.*, "Real-time classification of complex patterns using spike-based learning in neuromorphic VLSI," *TBioCAS*, vol. 3, no. 1, pp. 32–42, 2009.

[12] A. Trafton, "Mimicking the brain, in silicon," 2011. [Online]. Available: http://news.mit.edu/2011/brain-chip-1115

[13] C. Schlottmann *et al.*, "A Digitally Enhanced Dynamically Reconfigurable Analog Platform for Low-Power Signal Processing," *JSSC*, vol. 47, no. 9, pp. 2174–2184, September 2012.

[14] P. Merolla *et al.*, "A digital neurosynaptic core using embedded crossbar memory with 45pJ per spike in 45nm," in *CICC*. IEEE, 2011, pp. 1–4.

[15] P. A. Merolla *et al.*, "A million spiking-neuron integrated circuit with a scalable communication network and interface," *Science*, vol. 345, no. 6197, pp. 668–673, 2014.

[16] Y. Chen, X. Wang, H. Li, and et al., "Design Margin Exploration of Spin-Transfer Torque RAM (STT-RAM) in Scaled Technologies," in *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 18, December 2010, pp. 1724–1734.

[17] S. S. Sheu, M. F. Chang, K. F. Lin, and et al., "A 4Mb Embedded SLC Resistive-RAM Macro with 7.2ns Read-write Random-access Time and 160ns MLC-access Capability," in *International Solid-State Circuits Conference (ISSCC)*, Febuary 2011, pp. 200–202.

[18] J. Y. Wu, M. Breitwisch, S. Kim, and et al., "A Low Power Phase Change Memory Using Thermally Confined TaN/TiN Bottom Electrode," in *International Electron Device Meeting (IEDM)*, 2011, pp. 3.2.1–3.2.4.

[19] H. Akinaga and H. Shima, "Resistive Random Access Memory (ReRAM) based on Metal Oxide," *Proceedings of the IEEE*, vol. 98, no. 12, pp. 2237–2251, December 2010.

[20] R. Waser *et al.*, "Redox-based Resistive Switching Memories–Nanoionic Mechanisms, Prospects, and Challenges," *Advanced Materials*, vol. 21, no. 25/26, pp. 2632–2663, 2009.

[21] M. Sharad *et al.*, "Proposal for neuromorphic hardware using spin devices," 2012. [Online]. Available: http://arXiv.org/abs/1206.3227

[22] J. J. Yang *et al.*, "High switching endurance in TaOx memristive devices," *APL*, vol. 97, pp. 1–3, December 2010.

[23] J. F. Gibbons and W. E. Beadle, "Switching Properties of Thin NiO Films," *Solid-State Electronics*, vol. 7, pp. 785–790, November 1964.

[24] T. W. Hickmott, "Low-frequency Negative Resistance in Thin Anodic Oxide Films," *Journal of Applied Physics*, vol. 33, pp. 2669–2682, September 1962.

[25] J. G. Simmons and R. R. Verderber, "New Conduction and Reversible Memory Phenomena in Thin Insulating Films," *Proceedings of Royal Society A*, vol. 301, no. 1464, pp. 77–102, October 1967.

[26] W. W. Zhuang, W. Pan, B. D. Ulrich, and et al., "Novell Colossal Magnetoresistive Thin Film Nonvolatile Resistance Random Access Memory (RRAM)," in *International Electon Devices Meeting (IEDM)*, December 2002, pp. 193–196.

[27] I. G. Baek, M. Lee, S. Seo, and et al., "Highly Scalable Nonvolatile Resistive Memory Using Simple Binary Oxide Driven by Asymmetric Unipolar Voltage Pulses," in *International Electron Devices Meeting (IEDM)*, December 2004, pp. 587–590.

[28] D. B. Strukov *et al.*, "The Missing Memristor Found," *Nature*, vol. 453, no. 12, pp. 28–35, 2008.

[29] L. O. Chua, "Memristor-the missing circuit element," in *IEEE Transactions on Circuit Theory*, vol. 18, September 1971, pp. 507–519.

[30] Y.-C. Chen, H. Li, and W. Zhang, "A Novel Peripheral Circuit for RRAM-based LUT," in *ISCAS*, May 2012, pp. 1811–1814.

[31] M. Hu, H. Li, Q. Wu, and et al., "Hardware Realization of BSB Recall Function Using Memristor Crossbar Arrays," in *Design Automation Conference (DAC)*, June 2012, pp. 498–503.

[32] J. Liang and H.-S. P. Wong, "Cross-Point Memory Array Without Cell Selectors–Device Characteristics and Data Storage Pattern Dependencies," *TED*, vol. 57, no. 10, pp. 2531–2538, October 2010.

[33] M.-J. Lee, Y. Park, B.-S. Kang, and et al., "2-stack 1D-1R Cross-point Structure with Oxide Diodes as Switch Elements for High Density Resistance RAM Applications," in *IEEE International Electron Device Meeting (IEDM)*, December 2007, pp. 771–774.

[34] V. S. S. Srinivasan and et al., "Punchthrough-Diode-Based Bipolar Punchthrough-Diode-Based Bipolar RRAM Selector by Si Epitaxy," in *Electron Device Letters*, vol. 33, no. 10, October 2012, pp. 1396–1398.

[35] R. Rosezin, E. Linn, L. Nielen, and et al., "Integrated complementary resistive switches for passive high-density nanocrossbar arrays complementary resistive switches for passive high-density nanocrossbar arrays," *IEEE Electron Device Letters*, vol. 32, pp. 191–193, 2011.

[36] M. S. Qureshi, W. Yi, and R. S. Williams, "AC Sense Technique for Memristor Crossbar," *Electronics Letters*, vol. 48, pp. 757–758, June 2012.

[37] M. S. Qureshi *et al.*, "CMOS Interface Circuits for Reading and Writing Memristor Crossbar Array," in *ISCAS*, May 2011, pp. 2954–2957.

[38] S. Kannan *et al.*, "Detection, Diagnosis, and Repair of Faults in Memristor-based Memories," in *VTS*, April 2014, pp. 1–6.

[39] S. Yu and H.-S. P. Wong, "A Phenomenological Model for the Reset Mechanism of Metal Oxide RRAM," *IEEE Electron Device Letters*, vol. 31, no. 12, pp. 1455–1457, December 2010.

[40] T.-Y. Liu, "A 130.7mm^2 2-layer 32Gb ReRAM Memory Device in 24nm Technology," in *IEEE International Solid-State Circuits Conference (ISSCC)*, Febuary 2013, pp. 210–211.

[41] R. Sameer, A. N. Mohieldin, and H. M. Eissa, "An Automated Design Methodology for Stress Avoidance in Analog & Mixed Signal Designs," in *International Design and Test Workshop (IDT)*, 2010, pp. 3–7.

[42] C. Gamrat, "Challenges and perspectives of computer architecture at the nano scale," in *ISVLSI*, 2010, pp. 8–10.

[43] S. Yu *et al.*, "Investigating the switching dynamics and multilevel capability of bipolar metal oxide resistive switching memory," *Applied Physics Letters*, vol. 98, no. 10, p. 103514, 2011.

[44] T.-W. Lee and J. Nickel, "Memristor resistance modulation for analog applications," *Electron Device Letters*, vol. 33, no. 10, pp. 1456–1458, Oct 2012.

[45] F. Clermidy *et al.*, "Advanced technologies for brain-inspired computing," in *ASP-DAC*, Jan 2014, pp. 563–569.

[46] M. Sharad *et al.*, "Ultra low power associative computing with spin neurons and resistive crossbar memory," in *DAC*. ACM, 2013, pp. 107:1–107:6.

[47] M. Chu *et al.*, "Neuromorphic Hardware System for Visual Pattern Recognition with Memristor Array and CMOS Neuron," *Industrial Electronics*, vol. PP, no. 99, pp. 1–1, 2014.

[48] W. Gerstner and W. M. Kistler, *Spiking Neuron Models*. Cambridge University Press, 2002.

[49] C. Liu and H. Li, "A Weighted Sensing Scheme for ReRAM-Based Cross-Point Memory Array," in *ISVLSI*, 2014, pp. 65–70.

[50] A. Flocke and T. Noll, "Fundamental analysis of resistive nano-crossbars for the use in hybrid Nano/CMOS-memory," in *Solid State Circuits Conference*, Sept 2007, pp. 328–331.

[51] J.-J. Huang *et al.*, "One Selector-one Resistor (1S1R) Crossbar Array for High-density Flexible Memory Applications," in *Electron Devices Meeting*, Dec 2011, pp. 31.7.1–31.7.4.

[52] S.-S. Sheu *et al.*, "A 5ns Fast Write Multi-level Non-volatile 1 K Bits RRAM Memory with Advance Write Scheme," in *VLSI Circuits*, June 2009, pp. 82–83.

[53] S.-H. Jo *et al.*, "High-Density Crossbar Arrays Based on a Si Memristive System," *Nano Letters*, vol. 9, no. 2, pp. 870–874, 2009.

[54] A. Joubert *et al.*, "Hardware spiking neurons design: Analog or digital?" in *IJCNN*, June 2012, pp. 1–5.

[55] B. Li *et al.*, "Merging the Interface: Power, Area and Accuracy Co-optimization for RRAM Crossbar-based Mixed-signal Computing System," in *DAC*. ACM, June 2015, pp. 1–6.

[56] M. Hu *et al.*, "Memristor Crossbar-Based Neuromorphic Computing System: A Case Study," *TNNLS*, vol. 25, no. 10, pp. 1864–1878, Oct 2014.

[57] D. Roclin *et al.*, "Sneak Paths Effects in CBRAM Memristive Devices Arrays for Spiking Neural Networks," in *Nanoscale Architectures*. ACM, July 2014, pp. 13–18.

[58] C. Y. Leung *et al.*, "An integrated cmos current-sensing circuit for low-voltage current-mode buck regulator," *IEEE Transactions on Circuits and Systems II: Express Briefs*, vol. 52:7, July 2005.

[59] Y. LeCun *et al.* The MNIST DATABASE of handwritten digits.

[60] M. Hu *et al.*, "Dot-Product Engine for Neuromorphic Computing: Programming 1T1M Crossbar to Accelerate Matrix-Vector Multiplication," in *DAC*, 2016, pp. 1–6.

[61] E. J. Merced-Grafals *et al.*, "Repeatable, Accurate, and High Speed Multi-level Programming of Memristor 1T1R Arrays for Power Efficient Analog Computing Applications," *Nanotechnology*, vol. 27, no. 36, p. 365202, 2016. [Online]. Available: http://stacks.iop.org/0957-4484/27/i=36/a=365202

[62] N. Srivastava *et al.*, "Dropout: A Simple Way to Prevent Neural Networks from Overfitting," *Journal of Machine Learning Research*, vol. 15, pp. 1929–1958, 2014. [Online]. Available: http://jmlr.org/papers/v15/srivastava14a.html

[63] S. Han *et al.*, "Learning both Weights and Connections for Efficient Neural Network," in *Advances in Neural Information Processing Systems*, 2015, pp. 1135–1143. [Online]. Available: http://papers.nips.cc/paper/5784-learning-both-weights-and-connections-for-efficient-neural-network.pdf

[64] ——, "EIE: Efficient Inference Engine on Compressed Deep Neural Network," in *ISCA*, 2016, pp. 243–254. [Online]. Available: http://dx.doi.org/10.1109/ISCA.2016.30

[65] ——, "DSD: Regularizing Deep Neural Networks with Dense-Sparse-Dense Training Flow," *arXiv preprint arXiv:1607.04381*, 2016.

[66] T. M. Mitchell, *Machine Learning*. McGraw-Hill Science, 1997.

[67] I. Kataeva, F. Merrikh-Bayat, E. Zamanidoost, and D. Strukov, "Efficient training algorithms for neural networks based on memristive crossbar circuits," in *Neural Networks (IJCNN), 2015 International Joint Conference on*. IEEE, 2015, pp. 1–8.