MEMRISTOR DEVICE MODELING AND CIRCUIT DESIGN FOR READ OUT INTEGRATED CIRCUITS, MEMORY ARCHITECTURES, AND NEUROMORPHIC SYSTEMS

Dissertation

Submitted to

The School of Engineering of the

UNIVERSITY OF DAYTON

In Partial Fulfillment of the Requirements for

The Degree of

Doctor of Philosophy in Electrical Engineering

By

Chris Yakopcic, M.S.

UNIVERSITY OF DAYTON

Dayton, Ohio

May, 2014

MEMRISTOR DEVICE MODELING AND CIRCUIT DESIGN FOR READ OUT INTEGRATED CIRCUITS, MEMORY ARCHITECTURES, AND NEUROMORPHIC SYSTEMS

Name: Yakopcic, Chris

APPROVED BY:

Tarek M. Taha, Ph.D. Advisory Committee Chairman Associate Professor Electrical and Computer Engineering	Guru Subramanyam, Ph.D. Committee Member Chair and Professor Electrical and Computer Engineering
Andrew Sarangan, Ph.D. Committee Member Associate Professor Electrical and Computer Engineering	Muhammad Usman, Ph.D. Committee Member Associate Professor Mathematics
John G. Weber, Ph.D. Associate Dean School of Engineering	Tony E. Saliba, Ph.D. Dean, School of Engineering & Wilke Distinguished Professor

ABSTRACT

MEMRISTOR DEVICE MODELING AND CIRCUIT DESIGN FOR READ OUT INTEGRATED CIRCUITS, MEMORY ARCHITECTURES, AND NEUROMORPHIC SYSTEMS

Name: Yakopcic, Chris
University of Dayton

Advisor: Dr. Tarek M. Taha

Significant interest has been placed on developing systems based on the memristor, which was physically recognized in 2008. The memristor is a nanoscale non-volatile device with a large varying resistance range. Voltage pulses can be applied to the memristor to change its resistance, and the last programed resistance remains until another voltage pulse is applied. The unique properties present in this device give it the potential to further the advancement of many electronic systems, such as read out integrated circuits (ROICs) for digital cameras, high-speed on-chip memory circuits, and neuromorphic circuits capable of parallel analog computation.

This work first describes the different memristor modeling techniques that have been proposed, followed by a new memristor model that this capable of reproducing the I-V curves and switching characteristics of many physical memristor characterizations very accurately. Circuits are then designed using this model for the three applications

previously mentioned (ROICs, memory arrays, and neuromorphic computation). It is demonstrated that a memristor based ROIC circuit can significantly reduce the area of a unit cell given that a very small memristor element is used to replace a large integrating capacitor. It is also shown that the memristor can be used to reduce the total area of on-chip microprocessor memory. Given the nonvolatile property of memristors, they can also be used to store information without consuming power to hold their memory state. Lastly, memristors can be used to implement neuromorphic circuits where parallel computations are performed in the analog domain. Just as chemical pulses alter synaptic weights in brain tissue, voltage pulses can be applied to memristors to alter their conductivity.

This work is completed in SPICE, which handles many low level circuit details using a very detailed memristor model. Novel circuit designs for three different applications are not only presented in this work, they are also simulated with a level of accuracy not accounted for in the current literature.

ACKNOWLEDGMENTS

My thanks are in order to Dr. Tarek M. Taha, my advisor, for providing the time and equipment necessary for the work contained herein. I would also like to express my appreciation to everyone who has helped me with this work, including Dr. Guru Subramanyam, Dr. Robinson Pino, Douglas Palmer, and Mark McLean.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER I

INTRODUCTION

1.1 Background

For nearly 180 years, it has been accepted that there are three fundamental passive circuit elements, the resistor (1827), the capacitor (1745), and the inductor (1831). Based on the symmetry of the equations that govern the resistor, capacitor and inductor, Dr. Chua hypothesized that fourth passive device should exist that holds a relationship between magnetic flux and charge. This would complete the circle where the resistor holds the relation between current and voltage, the inductor holds the relationship between current and flux, and the capacitor holds the relationship between voltage and charge. Dr. Chua proved that the abilities of the memristor could not be duplicated by any of the other three passive elements, and that an active circuit that mimics the functionality of the theorized device would require approximately 25 transistors.

The memristor was predicted to be a two-terminal device with a dynamic resistance that is determined by the integral of current flowing through it. Since this is an integral relationship, applying zero current would result in a constant charge, thus leaving the resistance constant. Therefore, the theorized memristor possesses the ability to retain a resistance value even after the power source is removed from the device. Unlike the capacitor and inductor, this is not an energy storage device, so the voltage must equal zero whenever the current goes to zero. This causes the I-V curve of the device to

produce a pinched hysteresis loop, revealing that the device has a memory effect associated with it.

The first fabrication of a memristor device was developed by a research team at HP Labs [2, 3]. The research team at HP Labs was formed in 1995, and they initially started working on ways to further increase computing power once transistors reach their minimum size constraint. They decided the solution should be a nanoscale crossbar series of switches, which would be designed so the system could still function even if some of the components were defective.

The memristor first fabricated at HP Labs was a thin-film titanium oxide device. The device structure comprised of a stoichiometric (TiO_2) and an oxygen deficient (TiO_{2-x}) layer sandwiched between two platinum electrodes. Applying a voltage across a memristor causes the oxygen deficiencies in the TiO_{2-x} layer to migrate, and this changes the thickness of the oxygen deficient layer. Likewise, this changes the resistance of the memristor device. Since the oxygen vacancies have a low mobility, they tend to stay in the same position after the voltage source is removed [2]. This phenomenon shows that the memristor can be used as a non-volatile memory element.

The memristor can be seen as a dynamic resistor where the programmed resistance is dependent on the sum of the voltage applied. Memristors can be fabricated in crossbar arrays, which are very dense grids of memristors formed between rows and columns of intersecting wires.

Given the nanoscale dimensions of memristors and their unique properties, several innovative applications for memristors have been proposed. One of the more interesting applications for the memristor involves using the device to mimic the

functionality of a synapse in brain tissue [3, 4]. Just as neural spikes are applied to a synapse to change the weight, voltage pulses can be applied to a memristor to change the resistance state. These nanoscale dimensions allow a high density of memristors to be packed onto a small chip area and thus allow a large number of synapses to be implemented. Another application area is the use of memristors as memory devices [5]. This also allows for the development of a memory system with a much higher bit density when compared with most alternative technologies. Overall, the memristor has the potential to reduce energy consumption and footprint area of a large number of electronic systems.

This thesis examines the modeling of memristors and examines the design of several memristor circuits. The circuits developed include a Readout Integrated Circuit (ROIC), a memory circuit, and several neuromorphic computing circuits.

1.2 Thesis Overview

Chapter II provides a review of publications related to the work presented. This includes modeling [6-22] of memristor devices [23-30] for SPICE circuit design [31-35], and mechanisms for memristor operation [36-37]. Furthermore, existing read out integrated circuit [38-43] are discussed. Alternative non-volatile devices [44-56] are discussed for their use in memory systems [57-60]. Lastly a review of work completed in memristor based neuromorphic architecture [61-91] is presented.

The projects in Chapters III and IV discuss memristor modeling techniques [6-10]. Chapter III discusses several memristor models [11-22] that have been published and the SPICE code required to simulate these models in circuits. Some of these models

correspond to the theory that was originally proposed in [1], and some are developed to simulate physical device characteristics published by a number of different institutions [23-30]. The SPICE code for each of these models is presented and simulations are performed that show the effectiveness of each model.

Chapter IV discusses a generalized memristor model [7-10] that is capable of simulating several different devices. This is done by altering the set of fitting parameters developed to correspond to the large variety in characteristics observed in published memristor devices [3, 4, 26-30]. This model can reproduce memristor switching characteristics for many different input voltage waveforms and can be used to determine accurate switching energies [92]. This chapter also discusses how to use the model to simulate variation [93-102] in memristor characteristics that may occur between devices [103-105] in a memristor grid.

Chapter V discusses a memristor-based read out integrated circuit (ROIC) [31] that was designed and simulated in SPICE using the previously mentioned generalized memristor model. A memristor can be used to replace the large integrating capacitors in detector readout circuits [38-43, 106] to increase the fill factor, and thus increase the number of pixels that can be placed on a chip (for single chip designs). This circuit was simulated in SPICE where the memristor model was matched to different published memristor device characterizations to determine which memristor would provide the strongest results.

Chapter VI discusses using the memristor as digital memory [107-112]. A partitioned memristor memory system is presented that can be used for on-chip memory. Partial transistor isolation is used to divide a large memristor array into smaller sections

to limit the amount of unwanted current paths present in the system. This design provides an increase in areal density when compared to alternative non-volatile memory architectures. It also provides a decrease in energy consumption and error likelihood when compared to a non-isolated high density resistive grid [113-115]. The proposed memory (using two different memristors [103, 116]) is compared to alternative systems [117] in terms of area, energy and timing.

Chapter VII discusses circuit designs for memristor based neuromorphic circuits [32-35] capable of low power highly parallel analog computation. An input data pattern is represented as a set of binary voltages connected to a number of neuron outputs through a grid of memristors. In this chapter these circuits display the ability to perform character recognition and to learn complex logic functions. The iterative algorithms [118] used to train these circuits are capable correcting for unwanted current paths that lead to errors within a memristor crossbar. This was determined by simulating the crossbar in SPICE to ensure that all current paths are accounted for. To determine the maximum size at which these neuromorphic circuits can be designed, this system was analyzed in terms of wire resistance, wire capacitance [119], and minimum difference between comparator input voltages. Lastly Chapter VIII provides a conclusion of the results obtained.

1.3 Novel Contributions

1.3.1 Memristor Modeling

Since the initial memristor fabrications and characterizations by HP Labs [2, 3] (as well as other research groups [4, 5]), a strong interest has been placed on memristor

SPICE modeling. This will allow for accurate circuit simulations as new memristor circuits are developed.

The wide variety in memristor structure and composition has led to the development of many different memristor modeling techniques [6-22]. Several compact models have been proposed that present modeling equations that approximate the functionality of published memristor devices [23-30]. Many of these models [11, 14, 16-19] are based on the memristor equations first proposed by HP Labs [2]. While these model results are similar to physical device behaviors reported in the literature for sinusoidal inputs, they are quite off for inputs that produce intermediate resistance states (such as repetitive DC sweeps or pulsed inputs).

Given that the dynamics of a memristor closely model those of a synapse [3, 4], memristors are considered ideal for spiking input based neuromorphic systems. Thus the simulation of memristors for spiking inputs is essential to the design of memristor based neuromorphic systems. Similarly, memristor based digital memory systems are based on applying a very short pulse to switch the states of memristors. Since repetitive sweeping inputs are more closely related in shape to pulsed and spiking inputs, a more precise SPICE model was needed.

The SPICE model [7-10] presented in Chapter IV is a generalized memristor model that uses fitting parameters to match the characterization data of many different memristor models. To the best of my knowledge, this is the first memristor SPICE model that has been quantitatively correlated to multiple devices for sinusoidal, repetitive sweeping, and pulsed inputs. In many cases the models was able to match the target characterization data with less than 7% error.

Many digital memory and neuromorphic circuit applications require large grids of memristors. To accurately represent these circuits in SPICE, a large number of memristors must be simulated. The model in Chapter IV is capable of simulating at least 3000 memristors in a single circuit. This number of memristors is not a hard limit, although simulating larger circuits within a reasonable time frame became infeasible. Several existing memristor models crashed when simulating a circuit with only 16 memristors when they were set to have a switching time in the nanosecond range.

The memristor model presented in Chapter IV was also used to complete a detailed analysis that simulates the possible effects of process variation within a memristor crossbar [7]. Since memristors are fabricated with a significant amount of variation, problems may occur when developing large memristor arrays, or when using memristors for multi-bit operation. Previous work has been completed that uses memristor modeling techniques to study device variation. Many of these studies have been completed based on the memristor model first proposed by HP Labs. To the best of my knowledge these results represent the first device variation analysis based on a memristor model correlated to actual device fabrication data. In addition to memristor thickness, variation in state variable dynamics is also studied. For example, a sputter deposition of TiO_2 may lead to variation in the Ti:O ratio across the wafer. This can be handed by adding variation to the state variable fitting parameters of the model presented in Chapter IV. However, this cannot be captured simply by adding variation to just the parameter representing device thickness as completed in previous work.

1.3.2 Memristor ROIC Circuit

A Read Out Integrated Circuit (ROIC) is the part of a digital imaging system that

converts photon energy into an analog voltage level that can then be processed by an analog-to-digital converter. A common way to develop these circuits is to place the photodetector on the same chip as the ROIC circuit. The drawback of this is that it limits the pixel resolution to the size of the ROIC unit cell. Each unit cell contains a very large integrating capacitor which is responsible for "counting" the number of incoming photons present at each photodetector.

The memristor is an alternative device that is capable of integrating a current generated by a photodetector and requires a much smaller chip area. Replacing every capacitor in the ROIC with a memristor will significantly reduce the unit cell area and allow for a much higher pixel density or photodetector fill factor.

Chapter V discusses a novel ROIC unit cell circuit that requires only a memristor and three transistors. This circuit requires a significantly smaller chip area than existing ROIC unit cells. This will lead to the desired increase in pixel density or photodetector fill factor.

This circuit was simulated in SPICE using the memristor model that was described in Chapter IV. Since memristors that have been published thus far vary widely in there electronic properties (such as resistance range and operating voltage), two different promising memristor devices were used to provide simulations of the unit cell operating cycle. This simulation technique allows for future memristor devices to be easily tested for use in this circuit to determine which memristor would provide the strongest results.

1.3.3 Memristor Memory System

Over the last few decades transistors have become smaller and smaller, which has

in turn provided a steady increase in microprocessor performance. However in recent years, transistors have been approaching their size limit as they are now developed in the nanometer scale. Due to the transistor scaling limit, microprocessor improvement has alternatively been achieved through increasing the number of processing cores on a chip (multicore processors). These systems are primarily limited in performance by the amount of on chip memory (the memory wall problem). Existing commercial processors dedicate 50% of their chip area to memory arrays, which limits the space available for additional processing cores. To provide a significant increase in the performance of these systems, an increase in both the number of processing cores and the amount of on-chip memory is required.

The dominant sources of on-chip energy consumption are the processing cores and the memory hierarchy. The increase in processing cores on a chip has significantly increased the overall energy consumption. Level two and lower level caches have low utilizations, thus making their energy consumption dominated by leakage power. As an alternative to traditional SRAM, memristors are a promising candidate for these caches because of their higher densities and lower leakage power due to their non-volatile behavior.

Similarly to memristors, other types of non-volatile memory that fall into the category of Resistive Random Access Memories (RRAM) have been proposed for use in on-chip memory. The three main types of RRAM include memristors, Phase Change Random Access Memory (PCRAM), and Spin-Torque Transfer Magnetic Random Access Memory (STT-MRAM). Each of these memories work based on different resistive switching mechanisms where a dynamic resistance value determines the

memory state of the device. However, the memristor has the potential to achieve the highest bit density.

Memristor crossbar arrays have been proposed as the potential building block of an ultra-high density memory system. The problem with these high density crossbar arrays is that the power consumption will increase dramatically with the size of the crossbar. This is due to the many alternate current paths lowering the effective resistance of the array. Additionally, read errors are much more likely due to these alternate current paths. To solve this, a 1 transistor-1 memristor (1T1M) bit cell can be used which is commonplace in STT-MRAM architectures. Unfortunately, this will lower the density of the memristor memory system to that of a single transistor array.

Chapter VI discusses a solution to this problem where a memristor crossbar is partitioned into smaller sub-arrays. This is done by limiting the size of a crossbar using transistors to isolate blocks containing either 16 or 64 memristors. This design has less than 10% of the write energy consumption of a simple un-isolated memristor crossbar. Also, it has up to 4 times the bit density of an STT-MRAM system and up to 11 times the bit density of an SRAM architecture.

This design was simulated in SPICE using the memristor model described in Chapter IV. Since this highly-scalable, more stable model was used, very long simulations were performed (more than 6400 read/write cycles in some cases). The SPICE circuit allowed for the consideration of wire and transistor resistances for a more realistic simulation. When large memory systems are approximated in faster, higher-level situations, much of the details in circuit operation are lost. The simulation of a memristor memory system in SPICE that operates over many read/write cycles has not previously

been published.

1.3.4 Memristor Neuromorphic System

Interest in neuromorphic computing has been increasing significantly over the past few years. Several groups have been exploring the simulation of large scale neural networks on high performance compute clusters in order to achieve powerful data processing capabilities [120-122]. These large compute clusters consume massive amounts of electrical power, take up significant space, and are expensive. This makes these clusters unsuitable for portable applications, thus significantly limiting the applications of the large neural algorithms.

The design of small and efficient specialized neural processing chips can solve these problems. They will have a large variety of applications, including acting as accelerators for more traditional computing platforms. A recent study has shown that the class of recognition, mining, and synthesis (RMS) applications can be simulated through neural networks [123]. Intel has identified RMS applications as a key driver of future computing systems [124]. Esmaeilzadeh et al. [125, 126] show that several key application kernels (such as FFT and JPEG) can be approximated using neural networks. They make the case for specialized neural network accelerators on general purpose CPUs.

The memristor [1] has received significant attention as a potential building block for neuromorphic systems. The memristor is a nanoscale non-volatile device with a large varying resistance range. Just as electrochemical pulses alter synaptic weights in brain tissue, voltage pulses can be applied to memristors to alter their conductivity. This means

that high density memristor crossbars can be used to perform parallel analog computations between a set of highly connected inputs and outputs.

We have shown previously [34] that a specialized memristor based neuromorphic architecture is capable of achieving an efficiency greater than 200,000 times that of a general purpose microprocessor. The work in Chapter VII describes the circuit designs that will act as building blocks for a memristor based neuromorphic system.

The circuits in Chapter VII provide the first results where a high-density memristor crossbar (without transistor or diode isolation) is capable of successfully implementing a learning algorithm in SPICE. It is shown that dense memristor crossbar can be trained with neural network algorithms despite the alternate current paths inherent in the circuits to carry out pattern recognition tasks. Since a very high connectivity between the input and output neurons is required in a very small area footprint alternate current paths are present in the resistive grid. However, the iterative learning techniques used to train the memristors can account for the errors due to alternate current paths and quickly train around them.

A two layer neural network was developed that is capable of learning non-linearly separable functions. This was completed using a supervised learning algorithm known as the Concurrent Learning Algorithm (CLA). Detailed SPICE level simulations show that the two serially connected memristor crossbars are capable of learning all possible 16 2-input logic functions. Furthermore a large singe-layer crossbar is used to demonstrate correct classification of handwritten digit from the set of MNIST images.

Chapter VII also examines the scalability and power consumption of our crossbar circuits and compares them to existing approaches presented in the literature. Analysis is

presented that shows how large the crossbar circuit may become before wire resistances and capacitances have an impact on the circuit. The driver power constraints on the scalability of the crossbar were also examined.

The overall impact of this study is the demonstration through low level circuit simulations that dense memristor crossbars can be effectively utilized to build neuromorphic processors. A neural network is trained to demonstrate how non-linearly separable functions can also be learned through our low power circuits proving that these neuromorphic circuits can learn any logic function.

CHAPTER II

REVIEW OF RELATED WORK

This section provides a literature review of research completed by others that is related to the work presented in the following chapters.

2.1 Memristor Modeling

2.1.1 Theoretical Models

Many of the previously published SPICE models [11, 14, 16-19] are based on the memristor equations proposed by HP Labs [2]. The main differences are in how the state variable equation was implemented, especially at the boundaries (the points of minimum and maximum conductance). SPICE implementations of two different models [11, 14] based on the equations proposed in [2] can be seen in Figures 3.1 and 3.2. Several discrepancies are present when comparing the output of these models to published characterization data.

When consecutive zero-to-positive DC sweeps are applied, the devices in [4, 26] show that the size of the hysteresis loops decreases as the conductivity of the device increases. Models based on equations in [2] show the opposite effect, with a very large loop near maximum conductivity (see Figures 3.1 and 3.2).

Characterization data has been published [36] where the motion of the state variable depends on both its value, and the polarity of the applied current. Many of the

previous models show the motion of the state variable to be equivalent, whether it is moving in the positive or negative direction. Many published memristor devices [3, 4, 26, 28] display a threshold voltage where hysteresis is not seen unless the voltage across the memristor exceeds that threshold. The models based on the equations in [2] do not provide this effect, as the change in state is directly proportional to the charge applied with no non-linear dependence on voltage magnitude.

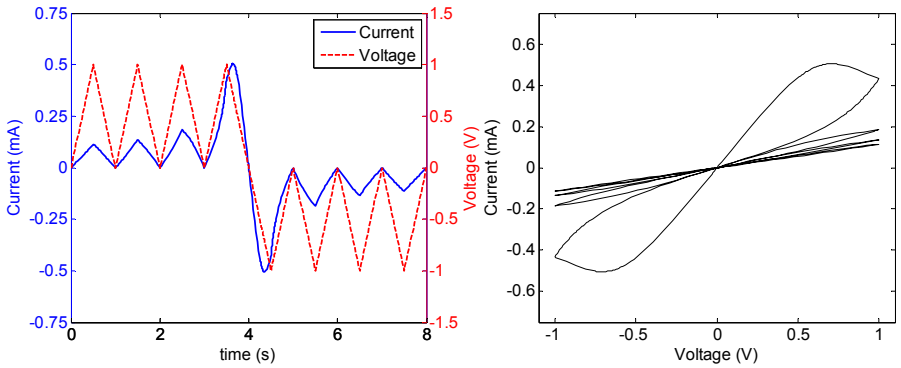


Figure 2.1. Simulated I-V characteristic using the memristor model in [11] where the plots show the voltage and current waveforms, and the corresponding I-V characteristic.

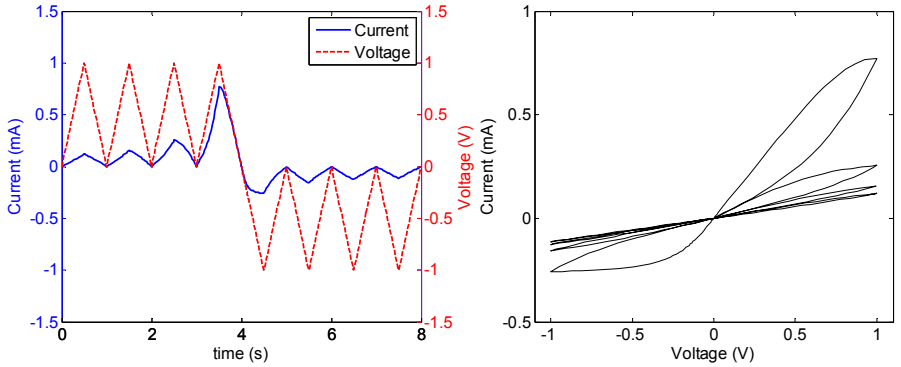


Figure 2.2. Simulated result using the memristor model in [14] where the plots show the voltage and current waveforms, and the corresponding I-V curve.

2.1.2 Hardware Correlated Models

A more complex model was proposed by Drs. Hisham Abdalla and Matthew D. Pickett at HP Labs [21]. This model appears to match the characterization data very well. Additionally, the model is supported by a very strong connection to the physical mechanisms within the device. This model was based on the assumption that the

memristor acts as a metal-insulator-metal (MIM) tunnel barrier [37]. It is assumed that the insulating tunnel barrier is represented by the TiO_2 layer, and the TiO_{2-x} layer acts as a low resistivity metallic layer. As voltage is applied, the thickness of the tunnel barrier is said to modulate due to the position of the oxygen deficiencies in the device.

The simulation results for the SPICE model can be seen in Figure 2.3. The top left plot shows the voltage across the memristor along with the current through the memristor. The voltage signal from the input source can be seen in the bottom left plot. The input voltage differs from the memristor voltage because probe resistance was taken into consideration in the SPICE simulation as a series resistor. The I-V curve closely matches the experimental characterization displayed in [21].

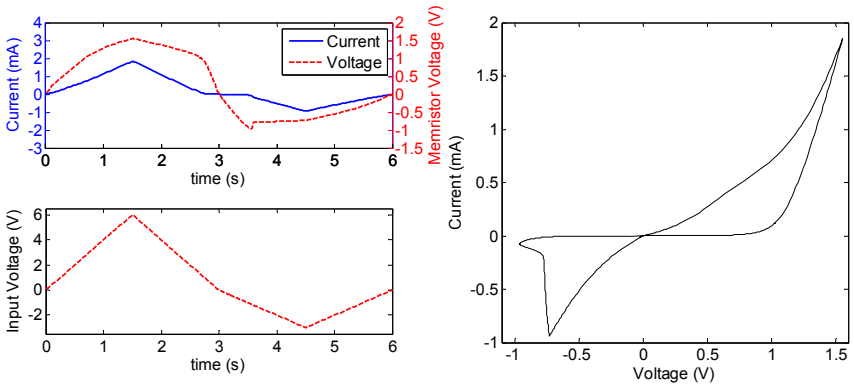


Figure 2.3. Simulated I-V characteristic using the memristor model in [21] where the plots show the memristor voltage and current waveforms, the input voltage waveform, and the corresponding I-V characteristic.

2.1.3 Hyperbolic Sine Models

The hyperbolic sine shape has been proposed for several memristor models [7-10, 12, 15, 20]. This is because the hyperbolic sine function can be used to approximate the I-V relationship of an MIM junction. Using a hyperbolic sine function in the I-V

relationship appears to provide a significantly better result when using a repetitive sweeping input compared to the results in Figures 3.1 and 3.2.

The simulation result for the hyperbolic sine model in Figure 2.4 shows an improvement in the shape of the I-V curve. This model also considered an ionic drift component that caused an unwanted decay in the resistance of the device. This provided an explanation for the overlap in hysteresis that was only present when the applied voltage was in the positive regime.

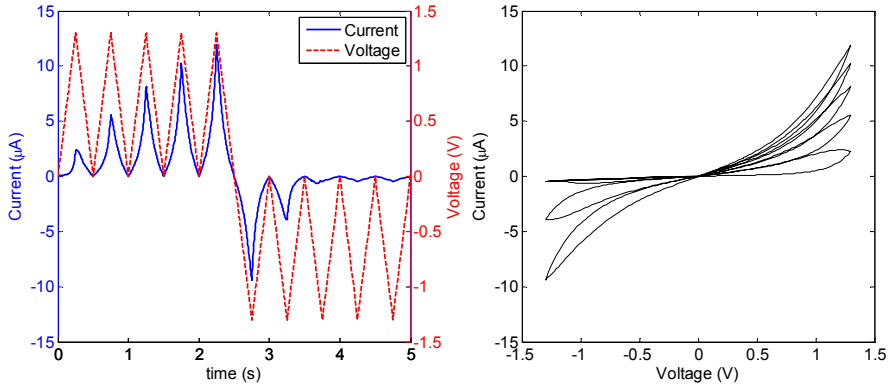


Figure 2.4. Simulated current and voltage waveforms and I-V characteristic using the memristor model in [20].

Summary: Several SPICE models have been proposed, but until now, no single SPICE model has yet been correlated to multiple different memristor characterizations from different sources. The SPICE model presented in Chapter IV has been correlated to characterization data for both cyclic and repetitive DC sweeping voltage inputs. Therefore, the model proposed in Chapter IV that has both a higher degree of accuracy and generality when compared to the models described in this section.

2.2 Read Out Integrated Circuits

Arrays of photodetectors rely on readout integrated circuits (ROICs) to accurately convert the amount of light present at a detector into an electrical signal. This is primarily

done by constructing an array of unit cells that separately integrate the current generated by each photodetector, which is proportional to the number of photons at each detector. The unit cells generally contain an integration capacitor surrounded by some control circuitry, and are either operated in voltage mode [38] or current mode [39, 40]. Other techniques for reading detectors have also been proposed, such as using a counter and pulse generator where frequency is dependent on the light input at the detector [41]. Circuits have been fabricated where the photodetector array is on a separate chip as the readout circuit [42], or in an all-inclusive single chip design [41].

When considering the physical layout of a readout circuit, the integrating capacitor requires a significant amount of area [38], [43]. When developing photodetector arrays and readout circuits on a single chip, the large capacitor area limits the amount of space that can be dedicated to the sensitive area of the photodetector. The memristor has been successfully fabricated with a cross-sectional area as small as 30 by 30 nm [3], which is nearly negligible compared to capacitor sizes on the micrometer scale. Replacing the integrating capacitor with a memristor could greatly reduce the size of the readout unit cell circuitry and increase the fill factor of the photodetector cell as shown in Chapter V.

2.3 Non-Volatile Memory Devices

Previous research suggests that STT-MRAM [44-49] is the most promising candidate for the future of high-density, non-volatile, resistance switching memories. The R_{OFF}/R_{ON} ratio of STT-MRAM is typically only about 2.5 [50], so it is not likely that an STT-MRAM memory system would work without an access transistor for each

individual memory device. This creates a problem where the maximum areal density of this type of memory system is limited by the size of an access transistor and not the nanoscale magnetic switching element.

PCRAM [51-52] is another promising new memory technology. It has the lowest endurance in terms of switching cycles before failure, and generally has a longer switching time (50 to 100ns) [52] when compared to memristors and STT-MRAM. For these reasons, the majority of the PCRAM based memory systems are proposed as a replacement for DRAM as opposed to SRAM. However, PCRAM has the advantage of unipolar switching [51], so diodes can be used to limit unwanted current paths in a higher density design.

Research completed in [53,54] presents memory based on devices called conductive bridge random access memory (CBRAM). While CBRAM devices have very favorable write voltages, low current, and high resistance ratios, the switching time can be significantly longer with times in the range of 10-100µs [55] and 1µs [56]. Another analysis shows that these devices can switch in as little as 50ns [57], which is the longest switching time of memristor devices that is considered in Chapter VI.

Table 2.1. Comparison of Non-Volatile Memory Devices

	STT-MRAM	**PCRAM**	**Memristors**
Data Density (Gb/cm^2)	0.975	1.48	10, 12
Switching Time (ns)	2.6-7	60	0.105 – 0.120
Bit Density (b/cell)	1	2 – 4	> 2
Endurance (cycles/cell)	10^{16}	10^{9}	10^{15}

2.4 Memristor Based Memory Systems

Different research groups have proposed various high density memristor crossbars for use in memory systems [58,59,60]. Results presented in [58] show how a

resistive ram system can be used along with traditional memories (SRAM/DRAM) to reduce leakage power.

Other research [59] shows how an analytical model for a resistive memory system was developed. In this study the analytical model is first matched to SPICE data for 8 specific cases (that differ in crossbar size and input patterns). This analytical model is then used for producing results. This is an effective approach for simulating large systems quickly, although some detail is lost as SPICE is not used to determine the real time current flowing through each device in the memory array.

Furthermore, [60] suggests a memory design where a diode is used in series with a memristor, or that a memristor can be used that has diode like properties. Using a diode like device will significantly reduce power and increase noise margin since the alternate current paths will be eliminated. However, putting a diode in series with the memory array will only allow for a write pulse in one direction resulting in a one-time programmable memory at best (unless a bidirectional MIM diode is used). Memristor devices with an inherent diode property [61] appear to have a switching time that is too large for on-chip memory applications (100μs).

2.5 Memristor Based Neuromorphic Systems

Several studies have been performed that describe memristor based neuromorphic architectures. Some of these studies use single memristors in neural circuits and some use memristor crossbars that achieve higher synaptic density and connectivity.

2.5.1 Neuromorphic Circuits Using Isolated Memristors

Single memristor single neuron circuits have been developed for supervised learning [62,63] and self-training [64] with very low operating power. Neuromorphic

systems based on CNN cell blocks have also been proposed [65]. In [66], a simple memristor based neural network was implemented by simulating 2 memristors and 3 CMOS neurons using microcontrollers to demonstrate associative memory with a basic memristor circuit. The study in [67] uses a SPICE simulation to show that a single-synapse circuit is capable of mimicking biological learning. These circuits show the potential of using a memristor as a synaptic weight, although these papers do not discuss how large memristor based multi-neuron systems can be used to solve problems.

2.5.2 Mathematical Studies of Memristor Based Neuromorphic Systems

In a more mathematical approach, instar and outstar synaptic models have also been used to develop memristor-based neuromorphic structures [68]. Additionally, work has been completed in developing a software platform that eventual memristor-based systems will utilize [69]. In this work several learning rules have been studied in terms of stability in a memristor based system. Other studies have been completed that show how the dynamics of the memristor may fit a Lotka-Volterra recurrent neural network based on computer simulation [70]. Other mathematical studies on memristive cellular neural networks [71] and chaotic memristive neural networks [72, 73] have also been presented. These are mathematical studies that analyze memristor neural systems at a higher level, and do not account for low level memristor device dynamics.

2.5.3 Memristor Bridge Circuits

Memristor bridge circuits have recently been proposed [74, 75] where small groups of memristors (either 4 or 5) are used to store a synaptic weight. One of the advantages of these bridge circuits is that either a positive or negative weight can be stored based on the sensed voltage. The use of these circuits in a crossbar structure for

large scale pattern recognition has not yet been studied. Furthermore, the study proposed in Chapter VII achieves both positive and negative weights with 2 memristors per synapse instead of 4 or 5.

2.5.4 Neuromorphic Crossbar Arrays

Crossbar arrays have been proposed as an efficient way to develop a large number of artificial synapses with high connectivity. Initial work in memristor-based neuromorphic designs involves using a memristor crossbar array that acts as a grid connecting rows of post and pre-synaptic neurons [3, 76]. It has also been shown that memristors can implement STDP learning similarly to biological synapses [3, 4, 76-78]. Work in [79] also proposes how a memristor grid can be used to implement a highly dense neural network that can be used in visual image processing. Work has been completed that shows how memristors can be used in neural logic blocks in a Field Trainable Neural Array (FTNA) [80]. Each neural block contains a memristor crossbar array and CMOS learning cells to program the memristive weights. Work in [81] examined implementation of Boolean functions in Neural Logic Blocks (NLB). Memristor crossbar based neuromorphic systems have been proposed where a synaptic grid could perform efficient optical character recognition [82], and for use in a competitive control system that can perform biologically inspired parallel analog computation [83]. In each of these cases, a low level circuit simulator such as SPICE is not used to demonstrate the evaluation of a learning function. Some of these studies use SPICE simulations to demonstrate small scale circuit operation but this does not account for how circuit properties such as alternate current paths impact learning.

Lastly, work in [84] demonstrated pattern classification using a single-layer perceptron network implemented with a memristor crossbar circuit and trained using the perceptron learning rule by ex situ and in situ methods. Non-linearly separable problems are not studied in [84]. This paper also shows how alternate current paths can be reduced by using virtual ground mode operational amplifiers to allow for ex situ training. Chapter VII shows that this approach requires about 3 times the circuit area and increases power consumption by about 2 orders of magnitude.

2.5.5 Neuromorphic Memristor Crossbar Simulation Techniques

A limited number of studies have been completed that simulate neuromorphic memristor crossbar circuits. The study in [85] proposes a circuit that similar to the one proposed in Chapter VII, although not every column in the crossbar is utilized every clock cycle. This circuit was simulated using python, which does not provide the ability to simulate a crossbar with a great level of detail when compared to SPICE.

The work described in [86] does use SPICE for the circuit simulations, although the circuit described requires transistor isolation within the memristor crossbar and also uses transistor threshold gate arrays in the training circuit. Similarly, the study in [87] places diodes in series with each memristor to limit current flow to one direction. However, this will not work for many iterative learning algorithms where both positive and negative weight adjustments are required.

The study in [88] describes learning in memristive connections using multilayer neural networks, although a specific circuit design is not strongly defined. The simulations are performed using a more mathematical approach and higher level

simulation tools. Work in [89] states that alternate current paths do not impact their results, although they do not appear to be handled in their simulation approach.

The memristor crossbar study in [90] assumes that the memristor crossbar performs an exact matrix multiply, which is not always the case when alternate current paths are considered. Furthermore, the memristor model used in this simulation is simplified compared to published memristor device characteristics. The studies performed in [87, 91] also use a simplified memristor model.

CHAPTER III

MEMRISTOR SPICE MODELING

3.1 Introduction

This chapter provides a review of many of the different memristor modeling techniques [11-22]. The memristor models chosen to be discussed in this chapter were selected to show a wide variety of different modeling techniques while minimizing redundancy. Some models have been designed to represent a specific device very accurately, and other models aim to reproduce the functionality of a wider range of devices in a more generalized manner.

The model results in this chapter are discussed in terms of their resulting I-V characteristics and how well they model the I-V characteristics of physical devices. The voltage inputs studied are either sinusoidal or triangular pulses. The triangular input pulses are applied multiple times with the same polarity to study how each model switches to intermediate levels between the maximum and minimum resistance. The SPICE simulations were performed in LTspice, and the subcircuit code is provided for each model. This allowed for a one-to-one comparison of many different memristor models to show the advantages and disadvantages of each.

This chapter is organized as follows: Section 2.2 describes the how the memristor modeling equations were developed based on the initial memristor fabrication at HP Labs. Section 2.3 shows how these initial equations were modified to develop SPICE

models. Section 2.4 describes two alternative SPICE modeling techniques where the model output of each correlates very closely to the characterization data of a specific device. Section 2.5 describes models that have been developed based on a hyperbolic sinusoid current-voltage relationship. This appears to improve the model result when using repetitively pulsed inputs.

3.2 Memristor Model Proposed by HP Labs

In 2008, HP Labs published results that described the memristor device according to equations (3.1) through (3.3). The current voltage relationship is described in equation (3.1). The value of $I(t)$ represents the current through the memristor, and the voltage $V(t)$ represents the voltage generated at the input source. These definitions for $I(t)$ and $V(t)$ are used consistently throughout this chapter. The constants R_{OFF} and R_{ON} represent the maximum and minimum resistances of the device respectively. The actual resistance of the device is dependent on the ratio between the value of the dynamic state variable $w(t)$ and the device thickness D. The state variable $w(t)$ represents the thickness of the oxygen deficient titanium dioxide layer (TiO_{2-x}). As the value of $w(t)$ increases, it can be seen that the overall device resistance lowers since $R_{OFF} > R_{ON}$.

The dynamic value of the state variable can be determined using equation (3.2) where dw/dt is described as the drift velocity of the oxygen deficiencies (v_D) in the device. The value for $w(t)$ can be determined by integrating equation (3.2), and the result can be seen in equation (3.3). After integration, it can be seen that the value for $w(t)$ is proportional to the charge on the device. Since the charge is the integral of the current, this provides an explanation for the non-volatile effect of the memristors: when no

current is flowing through the device, the charge is constant and thus, the resistance remains unchanged.

$$V(t) = \left[R_{ON} \frac{w(t)}{D} + R_{OFF}\left(1 - \frac{w(t)}{D}\right)\right] I(t) \tag{3.1}$$

$$v_D = \frac{dw}{dt} = \frac{\mu_D R_{ON}}{D} I(t) \tag{3.2}$$

$$w(t) = \frac{\mu_D R_{ON}}{D} q(t) \tag{3.3}$$

Figure 3.1 shows how this memristor model reacts to a simple sinusoidal voltage input. The I-V curve displays a pinched hysteresis loop that is characteristic of memristors. The hysteresis shows that the conductivity in a memristor is not only related to the voltage applied, but also to the previous value of the state variable $w(t)$, as more than one current value can be correlated to a single voltage. Figure 3.2 shows the simulation results of the model when several triangular voltage pulses are applied to the device. The first 4 voltage pulses, occurring in the time between 0 and 4 seconds, correspond to the right half of the I-V curve. Since the charge applied is always positive, the state variable continually increases. This results in an increase in the conductivity of the device as each pulse in applied. For the simulation time between 4 and 8 seconds, the current is always negative, and the opposite trend can be seen. These simulations were performed in MATLAB assuming that the voltage signal was directly applied to a memristor device with no additional circuit elements or added resistances. The next section discusses how these equations were used to generate a SPICE subcircuit.

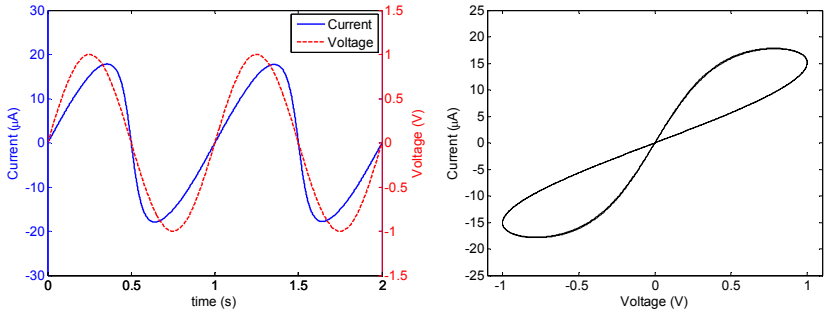


Figure 3.1. Simulation results for the HP Labs memristor with a sinusoidal input. In this simulation: R_{ON}=10kΩ, R_{OFF}=100kΩ, μ_D=10^{-14}m^{2}s^{-1}V^{-1}, D=27nm, x_0=0.1D, and $V(t)$=sin(2πt).

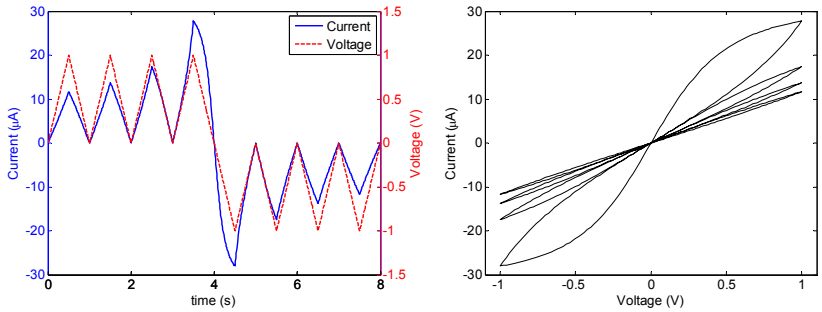


Figure 3.2. Simulation results for the HP Labs memristor with a pulsed input. In this simulation: R_{ON}=10kΩ, R_{OFF}=100kΩ, μ_D=10^{-14}m^{2}s^{-1}V^{-1}, D=27nm, x_0=0.1D, and $V(t)$=sin(2πt). Triangular pulses have magnitude of 1V and 1 second pulse width with rise and fall time of 0.5 seconds.

3.3 Initial Memristor SPICE Modeling

It is of great benefit to circuit designers to be able to model the memristor in SPICE simulators, therefore equations (3.1) and (3.2) were used to develop SPICE subcircuits [14-21] for memristors. Modifications to these initial equations were made to develop a robust technique for simulating memristors based on these initial equations.

First, it should be noted that the state variable equations were updated using the variable substitution $x(t)=w(t)/D$. The state variable is now a normalized quantity between 0 and 1. When $x(t)=0$ the memristor device is in the least conductive state, and the most conductive state occurs when $x(t)=1$. Memristor devices have been proposed using several different material structures [3, 4, 24-30], so the resistance switching mechanism is not always due to the change in thickness of a titanium oxide layer. This change in

state variable represents a generalization of the model so that it can represent more than just titanium oxide devices.

Next, the state variable boundaries were defined. The modeling equations in Section 2.2 do not account for the state variable boundaries: $0 \leq x(t) \leq 1$ (or $0 \leq w(t) \leq D$). If sufficient charge is applied to the memristor, then the value of $x(t)$ will become larger than 1 and the result of the model will become unstable and thus incorrect. The state variable motion was limited by using two different windowing functions [11, 14], and SPICE models were developed based on these modified equations.

3.3.1 Joglekar Modifications

In a publication by Yogesh N. Joglekar and Stephen J. Wolf [11], modifications were made to the initial equations proposed by HP Labs [2]. The parameter η was added so that memristors could be modeled where state variable motion could be in either direction relative to the input voltage. If a positive voltage signal applied to a memristor increases the value of the state variable, then the memristor device should be modeled where η=1. If the state variable decreases with the application of a positive voltage signal, then that memristor should be modeled with η=–1.

Additionally, the windowing function in equation (3.4) was added to the equation for state variable motion. This was done to ensure that the state variable will always fall in the range $0 \leq x(t) \leq 1$. Figure 3.3 displays plots for the Joglekar and Biolek (see Section 2.3.2) window functions for all acceptable values of $x(t)$. When looking at the left plot in Figure 3.3, it can be seen that the Joglekar window function forces the state variable motion to be zero at $x(t)$=0 or $x(t)$=1, thus defining the boundaries. Depending on the value of the parameter p, the window function can provide a harder boundary effect

(where p=100), or provide a smoother non-linearity in the motion of the state variable (where p=1). The state variable equation in its modified form can be seen in equation (3.5). It should be noted that the parameter D has been squared due to the substitution $x(t)=w(t)/D$.

$$F(x(t))=1-(2x(t)-1)^{2p} \tag{3.4}$$

$$\frac{dx}{dt}=\frac{\eta\mu_D R_{ON}}{D^2} I(t)F(x(t)) \tag{3.5}$$

3.3.2 Biolek Modifications

An alternative window function was proposed in a publication by Zdenek Biolek et al. [14]. When using the windowing function proposed by Joglekar, the motion of the state variable is reduced near the boundary whether it is traveling toward, or away from it. Alternatively, Biolek's window only reduces velocity at the boundary the state variable motion is tending toward. This appears to be a more accurate assumption based on the data collected by HP Labs that was presented in [36]. The Biolek window function is described in equations (3.6) and (3.7).

$$F(x(t))=1-(x(t)-stp(-I(t)))^{2p} \tag{3.6}$$

$$stp(I(t))=\begin{cases}1\ if\ I(t)>0\\0, if\ I(t)<0\end{cases} \tag{3.7}$$

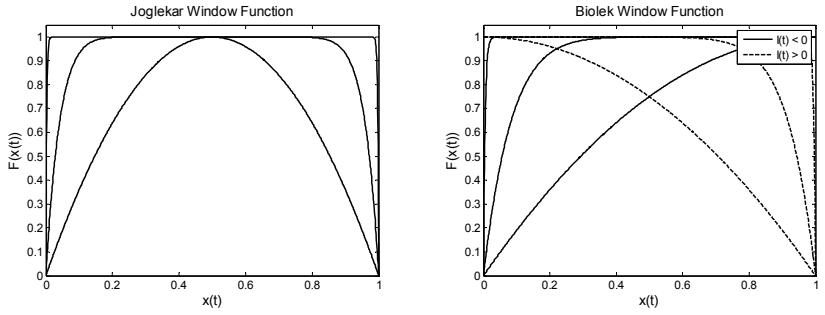


Figure 3.3. Both the Joglekar (left) and the Biolek (right) window functions are plotted where p=1, p=6, and p=100. It can be seen that if a hard limit effect at the borders is desired, then this can be implemented by setting p to a large number in each windowing function.

3.3.3 SPICE Model

The circuit layout for a SPICE model based on equations (3.1) through (3.7) is shown in Figure 3.4. The two terminals *TE* and *BE* represent the top and bottom electrodes of the memristor. The current source *Gm* generates a current based on equation (3.1). The value of the state variable is determined using a current source and an integrating capacitor. The output of the current source is set equal to the right hand side of equation (3.5), and the value $x(t)$ is determined using the integrating capacitor *Cx*. Memristor SPICE models have been previously proposed using a similar setup [14, 19]. The port *XSV* was created to provide a convenient method for plotting the state variable during a simulation. This is a helpful tool for debugging, but *XSV* should not be used as a voltage output in a circuit design. The circuit schematic in Figure 3.5 shows how the simulations were performed in LTspice. The memristor was simply connected to an input voltage source. Unless otherwise noted, all simulations in this chapter were performed using this arrangement.

Figure 3.6 displays code for the memristor subcircuit using the Joglekar window function, and Figure 3.7 contains the code for the subcircuit using the Biolek window

function. The code in Figures 3.6 and 3.7 is based on the subcircuit proposed in [14], although significant modifications have been made.

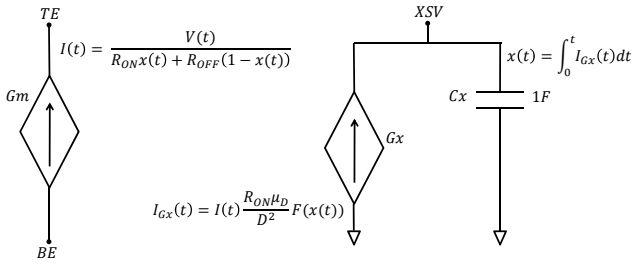


Figure 3.4. Circuit schematic for the memristor SPICE subcircuit based on [2,11,14].

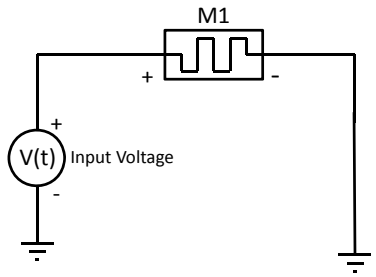


Figure 3.5. Circuit used to carry out SPICE simulations.

```
* HP Memristor SPICE Model Using Joglekar Window

* Connections:
* TE:  Top electrode
* BE:  Bottom electrode
* XSV: External connection to plot state variable
*      that is not used otherwise

.SUBCKT MEM_JOGLEKAR TE BE XSV

* Ron: Minimum device resistance
* Roff: Maximum device resistance
* D:   Width of the thin film
* uv:  Dopant mobility
* p:   Parameter for window function
* x0:   State variable initial value

.params Ron=1K Roff=100K x0=.5 D=10N uv=10F p=1

* Joglekar Window Function
.func f(V1) = 1-pow((2*V1-1),(2*p))

* Memristor I-V Relationship
.func IVRel(V1,V2) = V1/(Ron*V2 + Roff*(1-V2))

* Circuit to determine state variable
Gx 0 XSV value={ I(Gmem)*Ron*uv*f(V(XSV,0))/pow(D,2)
}
Cx XSV 0 {1}
.ic V(XSV) = x0

* Current source representing memristor
Gmem TE BE value={IVRel(V(TE,BE),V(XSV,0))}

.ENDS MEM_JOGLEKAR
```

Figure 3.6. SPICE subcircuit for the memristor model developed using the Joglekar window function.

```
* HP Memristor SPICE Model Using Biolek Window

* Connections:
* TE:  Top electrode
* BE:  Bottom electrode
* XSV: External connection to plot state variable
*      that is not used otherwise

.SUBCKT MEM_BIOLEK TE BE XSV

* Ron:  Minimum device resistance
* Roff: Maximum device resistance
* D:    Width of the thin film
* uv:   Dopant mobility
* p:    Parameter for window function
* x0:   State varaible initial value

.params Ron=1K Roff=100K x0=.5 D=10N uv=10F p=1

* Biolek Window Function
.func f(V1,I1)={1-pow((V1-stp(-I1)),(2*p))}

* Memristor I-V Relationship
.func IVRel(V1,V2) = V1/(Ron*V2 + Roff*(1-V2))

* Circuit to determine state variable
Gx 0 XSV
value={I(Gmem)*Ron*uv*f(V(XSV,0),I(Gmem))/pow(D,2)}
Cx XSV 0 {1}
.ic V(XSV) = x0

* Current source representing memristor
Gmem TE BE value={IVRel(V(TE,BE),V(XSV,0))}

.ENDS MEM_BIOLEK
```

Figure 3.7. SPICE subcircuit for the memristor model developed using the Joglekar window function.

3.3.4 Simulation Results with Joglekar Window

Figures 3.8 and 3.9 show the simulation results for the memristor SPICE model when using the Joglekar window function with a sinusoidal input of two different amplitudes. Figure 3.8 shows the result when the model is driven with the voltage input $v(t)=0.9\sin(2\pi ft)$. The result looks very similar to the model results displayed in Figure 3.1 because the state variable did not reach the boundaries where the window function has the strongest effect. Figure 3.8 shows the model response to the voltage input $v(t)=\sin(2\pi ft)$. This slightly larger voltage input forces the value of the state variable into the region affected by the windowing function and harder switching result can be seen.

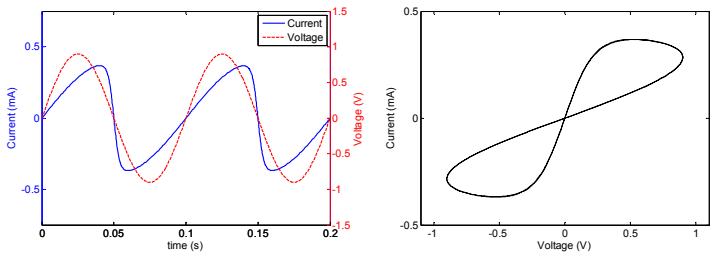


Figure 3.8. LTspice simulation results for the memristor model with the Joglekar window function. In this simulation: R_{ON}=100Ω, R_{OFF}=10kΩ, μ_D=5(10^{-14})$m^2s^{-1}V^{-1}$, D=12nm, x_0=0.56, p=7, and $V(t)$=0.9sin(2π10t).

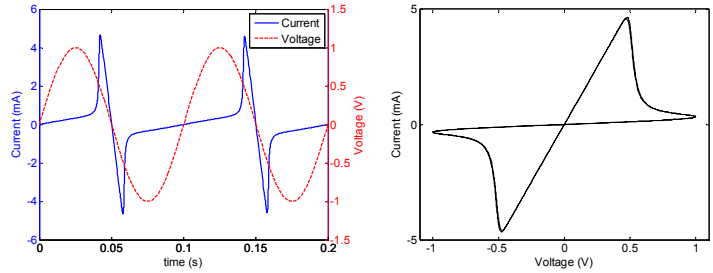


Figure 3.9. LTspice simulation results for the memristor model with the Joglekar window function and a large input voltage magnitude. In this simulation: R_{ON}=100Ω, R_{OFF}=10kΩ, μ_D=5(10^{-14})$m^2s^{-1}V^{-1}$, D=12nm, x_0=0.56, p=7, and $V(t)$=sin(2π10t).

Figure 3.10 shows the result using the Joglekar window function when applying several triangular voltage pulses. This result is similar to the one seen in Figure 3.2, although the most conductive hysteresis loop in this simulation is significantly larger.

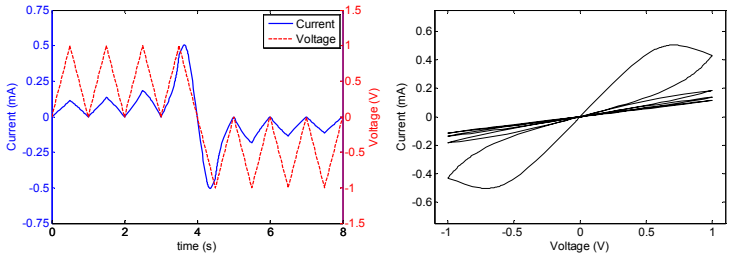


Figure 3.10. Results when simulating the HP Labs memristor with a triangular pulsed input. In this simulation: R_{ON}=1kΩ, R_{OFF}=10kΩ, μ_D=2(10^{-14})$m^2s^{-1}V^{-1}$, D=85nm, x_0=0.093, and p=2. Triangular pulses have magnitude of 1V and 1 second pulse width with rise and fall time of 0.5 seconds.

3.3.5 Simulation Results with Biolek Window

Simulations using the Biolek window function were also performed and the results can be seen in Figures 3.11 and 3.12. Figure 3.11 shows the model result with a sinusoidal voltage input with large enough amplitude to drive the state variable into the boundary where the window function has a larger impact. Figure 3.12 displays the result

when a triangular pulse input is applied. As opposed to the previous results, this model shows an asymmetric hysteresis with respect to voltage polarity for each of the simulations.

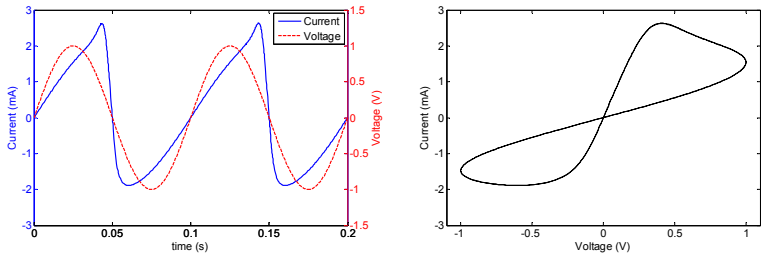


Figure 3.11. LTspice simulation results for the memristor model with the Joglekar window function. In this simulation: R_{ON}=100Ω, R_{OFF}=1kΩ, μ_D=4(10^{-14})$m^2s^{-1}V^{-1}$, D=16nm, x_0=0.076, p=7, and $V(t)$=sin(2π10t).

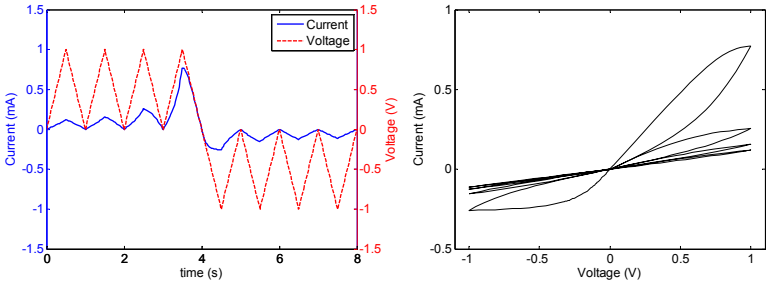


Figure 3.12. Results when simulating the HP Labs memristor with a triangular pulsed input. In this simulation: R_{ON}=1kΩ, R_{OFF}=10kΩ, μ_D=2(10^{-14})$m^2s^{-1}V^{-1}$, D=80nm, x_0=0.1, and p=2. Triangular pulses have magnitude of 1V and 1 second pulse width with rise and fall time of 0.5 seconds.

3.3.6 Discussion

There are several discrepancies when comparing the results obtained from these models to published physical characterization data. When applying a pulsed waveform, the results of these models show that the size of the hysteresis loops in the positive regime increases as conductivity increases. When looking at the published characterization data [3,4,26], an opposite trend is present.

Also, physical memristor devices show a threshold voltage where hysteresis is not seen unless the voltage across the memristor exceeds the threshold. No threshold voltage is present in these models. Lastly, Matthew D. Pickett et al. at HP Labs published characterization data where the motion of the state variable depends on both its value and

the polarity of the applied current [36]. This may suggest that the compression of oxygen vacancies has slightly different dynamics than the oxygen vacancy expansion, and that they are not perfectly mirrored. These models show the motion of the state variable to be equivalent, whether it is moving in the positive or negative direction. The remainder of this chapter discusses the different techniques that have been used to develop memristor SPICE models that provide a closer match to published characterization data.

3.4 Hardware Correlated Models

Alternative memristor models have been proposed that correlate more closely to physical characterization data. This section discusses two different techniques for modeling the memristors [13, 21] where the input voltage is sinusoidal (as opposed to using a repetitive pulse input). Each of these models produces a result that is matched very closely to a specific memristor device.

3.4.1 Air Force Research Lab Model

The first hardware correlated model was developed by Dr. Robinson E. Pino et al. at the Air Force Research Lab (AFRL) to match the I-V characteristic of a device developed at Boise State University [26]. This model matches the IV characteristic very well because it correlates voltage amplitude and slope to a piecewise function matched to the characterization of a physical device. Instead of defining a state variable for this model, the rate of change of the resistance was directly defined using equations (3.8) and (3.9). When the voltage across the memristor is greater than T_h, equation (3.8) is used to determine the change in resistance of the memristor device. When the voltage across the memristor is less than T_l, equation (3.9) is used to determine the change in device

resistance. Lastly, when the voltage across the memristor is in the range $T_l \geq V(t) \leq T_h$, there is no change in resistance.

$$\frac{dR}{dt} = \begin{cases} -K_{h1} e^{K_{h2}(V(t)-T_h)}, & R(t) > R_{ON} \\ 0, & R(t) \leq R_{ON} \end{cases} \tag{3.8}$$

$$\frac{dR}{dt} = \begin{cases} K_{l1} e^{K_{l2}(V(t)-T_l)}, & R(t) < R_{OFF} \\ 0, & R(t) \geq R_{OFF} \end{cases} \tag{3.9}$$

The equations for this model were first proposed in [13], and the LTspice code was developed for use in this chapter. The subcircuit can be seen in Figure 3.13. The model has three terminals, again with two representing the top and bottom electrodes of the device and a third terminal to plot the time integral of the rate equation. In this case, the change in resistance is output at the terminal *RSV* since there is no state variable defined other than the resistance itself. It can also be seen that the current through the memristor is determined through a division of two voltages. Even though the value of *V*(*RSV*) is a voltage in the simulation, it actually represents the resistance of the memristor.

```
* Code for memristor model proposed by Dr. Pino et al.

* Connections:
* TE:  Top electrode
* BE:  Bottom electrode
* RSV: External connection to plot resistance
*      that is not used otherwise

.SUBCKT MEM_PINO TE BE RSV

* Ron:      Minimum device resistance
* Roff:     Maximum device resistance
* Th:       Positive voltage threshold
* Tl:        Negative voltage threshold
* Kh1, Kh2: Fitting params for pos voltage
* Kl1, Kl2: Fitting params for neg voltage

.params Ron=160 Roff=1200 Th=0.2 Tl=-0.35 Kh1=5.5e6 Kh2=-20
+Kl1=4e6 Kl2=20

* Fits the change in resistance to characterization data
.func Rt(V1, V2) = IF( V1 <= Th, IF(V1 >= Tl, 0, IF(V2 <
+Roff, Kl1*exp(Kl2*(V1-Tl)), 0) ), IF(V2 > Ron, -
+Kh1*exp(Kh2*(V1-Th)), 0) )

* Circuit to integrate to find resistance
Gx 0 RSV value={Rt(V(TE,BE),V(RSV))}
Cx RSV 0 {1}
.ic V(RSV) = Roff

* Current source representing memristor
Gmem TE BE value = {V(TE,BE)/V(RSV)}

.ENDS MEM_PINO
```

Figure 3.13. LTspice subcircuit developed based on the AFRL memristor model equations in [13].

The model result can be seen in Figure 3.14. When comparing this result to the characterization data in [13], the model matches the both the current output and the I-V curve very closely.

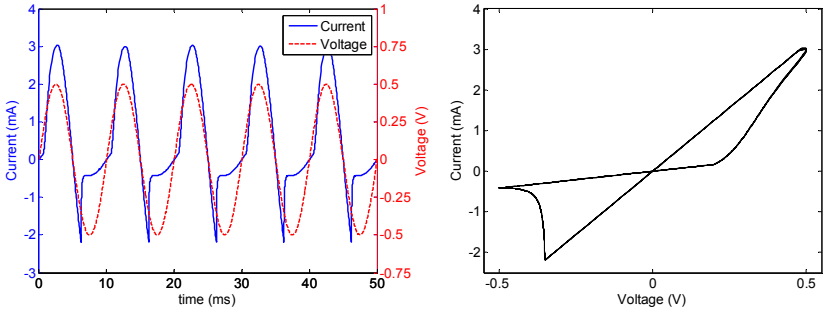


Figure 3.14. Simulation results using the model presented in [13]. In this simulation: R_{ON}=160, R_{OFF}=1200, T_h=0.2, T_l=-0.35, K_{h1}=5.5(10^6), K_{h2}=-20, K_{l1}=4(10^6), and K_{l2}=20.

3.4.2 HP Labs MIM Model

A more complex model was proposed by Drs. Hisham Abdalla and Matthew D. Pickett at HP Labs [21]. This model was based on the assumption that the memristor acts

as a metal-insulator-metal (MIM) tunnel barrier [37]. The insulating tunnel barrier is represented by the TiO_2 layer, and the TiO_{2-x} layer acts as a low resistivity metallic layer. As voltage is applied, the thickness of the tunnel barrier is said to modulate due to the position of the oxygen vacancies in the device. This model appears to match the characterization data very well. Additionally, the model is supported by a very strong connection to the physical mechanisms within the device.

The model equations can be seen in (3.10) through (3.14). In these equations the dynamic state variable is defined as $w(t)$. Although, in this model $w(t)$ represents the thickness of the TiO_2 layer as opposed to the TiO_{2-x} layer as seen in equations (3.1) through (3.3). Equations (3.10) and (3.11) govern the state variable dynamics. These equations were formed based the data presented in [36]. When $I(t) > 0$, the state variable motion is described by equation (3.10), otherwise, the state variable motion is described by equation (3.11). The fitting parameters in the model were defined as follows: f_{off}=3.5 μs, i_{off}=115 μA, a_{off}=1.2 nm, f_{on}=40 μs, i_{on}=8.9 μA, a_{on}=1.8 nm, b=500 μA, and w_c=107 pm.

$$\frac{dw}{dt} = f_{off} \sinh\left(\frac{|I(t)|}{i_{off}}\right) \exp\left(-\exp\left(\frac{w(t) - a_{off}}{wc} - \frac{|I(t)|}{b}\right) - \frac{w(t)}{w_c}\right) \tag{3.10}$$

$$\frac{dw}{dt} = -f_{on} \sinh\left(\frac{|I(t)|}{i_{on}}\right) \exp\left(-\exp\left(\frac{a_{on} - w(t)}{wc} - \frac{|I(t)|}{b}\right) - \frac{w(t)}{w_c}\right) \tag{3.11}$$

Equations (3.12) through (3.14) were developed by modifying the MIM tunnel barrier equations first proposed in [37] to account for a variable barrier width. In these equations: ϕ_0=0.95 V, w_1=0.1261 nm, B=10.24634Δw, λ=0.0998/$w(t)$, and $\Delta w = w_2 - w_1$. The variable v_g represents the voltage across the TiO_2 layer of the memristor. The total

voltage across the device is equal to the sum of v_g and v_r, where v_r is equal to the voltage across the TiO_{2-x} layer.

$$I(t) = \frac{0.0617}{\Delta w^2}\left\{\phi_I e^{-B\sqrt{\phi_I}} - \left(\phi_I + |v_g|\right)e^{-B\sqrt{\phi_I + |v_g|}}\right\} \tag{3.12}$$

$$\phi_I = \phi_0 - |v_g|\left(\frac{w_1 + w_2}{w(t)}\right) - \left(\frac{0.1148}{\Delta w}\right)\ln\left(\frac{w_2(w(t) - w_1)}{w_1(w(t) - w_2)}\right) \tag{3.13}$$

$$w_2 = w_1 + w(t)\left(1 - \frac{9.2\lambda}{\left(2.85 + 4\lambda - 2|v_g|\right)}\right) \tag{3.14}$$

The circuit used to test the model differed slightly compared to the original circuit in Figure 3.7. The circuit in Figure 3.15 shows how this memristor model was tested. The 2.4 kΩ resistor was added to model the resistance of the electrodes used to characterize the memristor device.

The LTspice code for the MIM memristor SPICE model can be seen in Figure 3.16. The code was taken from [21], except small changes were made so that the model would operate correctly in LTspice. Additionally, the node *WSV* was added as a terminal so that the state variable motion could be plotted easily.

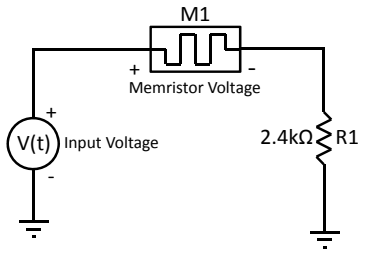


Figure 3.15. Schematic for testing the MIM memristor model [21].

```
* HP Labs MIM Model proposed by Drs. Abdalla and Pickett

* Connections:
* TE:  Top electrode
* BE:  Bottom electrode
* WSV: External connection to plot state variable
*      that is not used otherwise

.SUBCKT MEM_ABDALLA TE BE WSV

.params phio=0.95 Lm=0.0998 w1=0.1261 foff=3.5e-6 ioff=115e-6
+aoff=1.2 fon=40e-6 ion=8.9e-6 aon=1.8 b=500e-6 wc=107e-3
+winit = 1.2

* MIM IV Relationship
G1 TE internal
value={sgn(V(x))*pow((1/V(dw)),2)*0.0617*(V(phiI)*exp(-
+V(B)*V(sr))-(V(phiI)+abs(V(x)))*exp(-V(B)*V(sr2)))}
Esr sr 0 value={sqrt(V(phiI))}
Esr2 sr2 0 value={sqrt(V(phiI)+abs(V(x)))}
* Series resistance of TiO2-x Layer
Rs internal BE 215
Eg x 0 value={V(TE)-V(internal)}
Elamda Lmda 0 value={Lm/V(WSV)}
Ew2 w2 0 value={w1+V(WSV)-(0.9183/(2.85+4*V(Lmda)-
+2*abs(V(x))))}
EDw dw 0 value={V(w2)-w1}
EB B 0 value={10.246*V(dw)}
ER R 0 value={(V(w2)/w1)*(V(WSV)-w1)/(V(WSV)-V(w2))}
EphiI phiI 0 value={phio-abs(V(x))*((w1+V(w2))/(2*V(WSV)))-
+1.15*V(Lmda)*V(WSV)*log(V(R))/V(dw)}
* State variable integrating capactior
C1 WSV 0 1e-9
.ic V(WSV)=winit
R WSV 0 1e8MEG
Ec c 0 value={abs(V(internal)-V(BE))/215}
Emon1 mon1 0 value={((V(WSV)-aoff)/wc)-(V(c)/b)}
Emon2 mon2 0 value={(aon-V(WSV))/wc-(V(c)/b)}
Goff 0 WSV value={foff*sinh(stp(V(x))*V(c)/ioff)*exp(-
+exp(V(mon1))-V(WSV)/wc)}
Gon WSV 0 value={fon*sinh(stp(-V(x))*V(c)/ion)*exp(-
+exp(V(mon2))-V(WSV)/wc)}

.ENDS MEM_ABDALLA
```

Figure 3.16. LTspice code for the HP Labs MIM memristor model [21].

The simulation results for the SPICE model can be seen in Figure 3.17. The top left plot shows the voltage across the memristor along with the current through the memristor. The voltage signal from the input source can be seen in the bottom left plot. The simulated I-V curve closely matches the data displayed in [21].

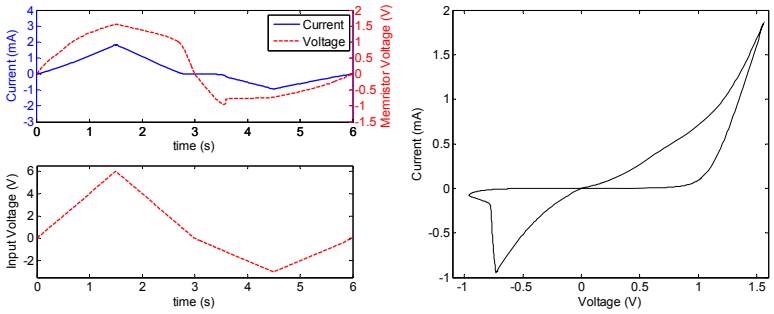


Figure 3.17. Simulation results using the HP Labs MIM model.

3.4.3 Discussion

The models presented in this section show a very close correlation to the characterizations that they were designed to match. The disadvantage is that it is not known how closely these models will match the characterization data for alternative voltage inputs such as repetitive triangular pulses. Also, these models are very specific to a single fabricated device. Given the wide variety in the current voltage characterization of memristors, these models will most likely have to be updated significantly for use with different device structures.

3.5 Hyperbolic Sine Models

The hyperbolic sinusoid shape has been proposed for several memristor models [12, 15, 20]. This is because the hyperbolic sinusoid function can be used to approximate the I-V relationship of an MIM junction [37]. Since thinfilm memristors are commonly fabricated by sandwiching an oxide between two metal electrodes, modeling a memristor as an MIM device seems reasonable. Section 2.4.2 has already demonstrated a model [21] based on MIM tunneling equations. Using a hyperbolic sine function in the I-V relationship appears provide a significantly better result when using a repetitive voltage pulse input.

3.5.1 General Hyperbolic Sine Model

A general hyperbolic sinusoid model was proposed by Dr. Mika Laiho et al. [12] and is described in equations (3.15) and (3.16). The current voltage relationship is represented by a hyperbolic sinusoid modulated by the value of the state variable $x(t)$. The parameters a_1, a_2, b_1, and b_2 are used to adjust the I-V response of the model. The state variable is also modeled using a hyperbolic sine function (see equation (3.16)). Several memristor characterizations show that the state of the device will not change unless the voltage applied exceeds a threshold [3, 4, 26], and the hyperbolic sinusoid based state variable is one option that achieves this effect. The constants c_1, c_2, d_1, and d_2, are used to shape the threshold and intensity of the state variable dynamics.

$$I(t) = \begin{cases} a_1 x(t) \sinh\big(b_1 V(t)\big), & V(t) \geq 0 \\ a_2 x(t) \sinh\big(b_2 V(t)\big), & V(t) < 0 \end{cases} \tag{3.15}$$

$$\frac{dx}{dt} = \begin{cases} c_1 \sinh\big(d_1 V(t)\big), & V(t) \geq 0 \\ c_2 \sinh\big(d_2 V(t)\big), & V(t) < 0 \end{cases} \tag{3.16}$$

This model was originally proposed in [12] without boundary conditions to stop the state variable from exceeding the range $0 \leq x(t) \leq 1$. To address this issue, the model has been modified by using the Biolek window function [14] to define the device boundaries. The updated state variable equation can be seen in (3.17) where $F(x(t))$ represents the Biolek window function. The SPICE code for the model with and without the boundary addition has been developed and can be seen in Figures 3.18 and 3.19 respectively.

$$\frac{dx}{dt} = \begin{cases} c_1 \sinh\big(d_1 V(t)\big) F\big(x(t)\big), & V(t) \geq 0 \\ c_2 \sinh\big(d_2 V(t)\big) F\big(x(t)\big), & V(t) < 0 \end{cases} \tag{3.17}$$

```
* SPICE model for equations proposed by Dr. Mika Laiho et al.

* Connections:
* TE:  Top electrode
* BE:  Bottom electrode
* XSV: External connection to plot state variable
*      that is not used otherwise

.subckt MEM_LAIHO TE BE XSV

.param a1=4e-8 b1=1.2 a2=1.25e-7 b2=1.2 c1=6e-4 d1=2
+c2=6.6e-4 d2=3.8 x0=0.001

* Hyperbolic sine IV relationship
.func IVRel(V1,V2) = IF(V1 >= 0, a1*V2*sinh(b1*V1),
+a2*V2*sinh(b2*V1))

* Equation for state variable
.func SV(V1) = IF(V1 >= 0, c1*sinh(d1*V1), c2*sinh(d2*V1))

* Current source representing memristor
Gmem TE BE value = {IVRel(V(TE,BE),V(XSV,0))}

* Circuit to determine value of state variable
Gxsv 0 XSV value = {SV(V(TE,BE))}
Cx XSV 0 {1}
.ic V(XSV) = x0

.ends MEM_LAIHO
```

Figure 3.18. LTspice code that was developed for the generalized hyperbolic sinusoid model proposed by Laiho et al.

Figures 3.20 and 3.21 show the simulation results for this model with and without the addition of the Biolek window function. Figure 3.20 shows results of the model as it was presented in [12]. Modeling the boundary is not an issue since not enough charge was applied to let the state variable move outside the boundaries. Figure 3.21 shows the results of the model where the Biolek window function was added to set boundaries on the value of the state variable. It can be seen that current through the device increases with each voltage pulse until the upper limit of the state variable motion is achieved. At this point the peak of each current pulse no longer changes until the polarity of the input voltage is reversed.

```
* SPICE model for equations proposed by Dr. Mika Laiho et al.
* with Biolek window for memristor boundaries

* Connections:
* TE:  Top electrode
* BE:  Bottom electrode
* XSV: External connection to plot state variable
*      that is not used otherwise

.subckt MEM_LAIHO_WINDOW TE BE XSV

.param a1=4e-8 b1=1.2 a2=1.25e-7 b2=1.2 c1=6e-4 d1=2
+c2=6.6e-4 d2=3.8 x0=0.001 p=1

* Hyperbolic sine IV relationship
.func IVRel(V1,V2) = IF(V1 >= 0, a1*V2*sinh(b1*V1),
+a2*V2*sinh(b2*V1))

* Equation for state variable
.func SV(V1) = IF(V1 >= 0, c1*sinh(d1*V1), c2*sinh(d2*V1))

* Biolek window function
.func f(V1,I1)={1-pow((V1-stp(-I1)),(2*p))}

* Current source representing memristor
Gmem TE BE value = {IVRel(V(TE,BE),V(XSV,0))}

* Circuit to determine value of state variable
Gxsv 0 XSV value = {SV(V(TE,BE))*f(V(XSV,0),I(Gmem))}
Cx XSV 0 {1}
.ic V(XSV) = x0

.ends MEM_LAIHO_WINDOW
```

Figure 3.19. LTspice code that was developed for the generalized hyperbolic sinusoid model with the addition of the Biolek windowing function.

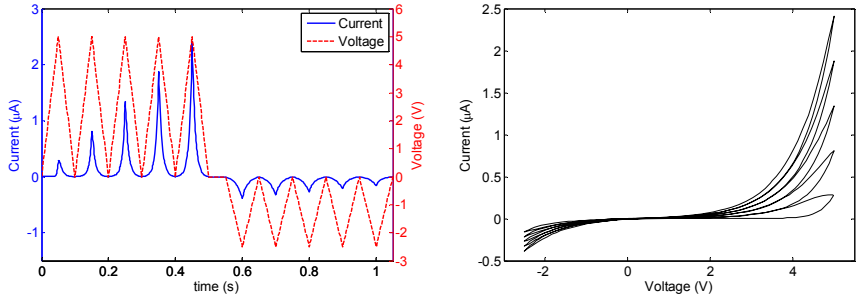


Figure 3.20. Simulation results for the hyperbolic sinusoid model proposed by Laiho et al. In this simulation: a_1=4(10^{-8}), b_1=1.2, a_2=1.25(10^{-7}), b_2=1.2, c_1=6(10^{-4}), d_1=2, c_2=6.6(10^{-4}), d_2=3.8, and x_0=0.001. Triangular pulses have magnitude of +5V/-2.5V and a 0.1 second pulse width with rise and fall time of 0.05 seconds.

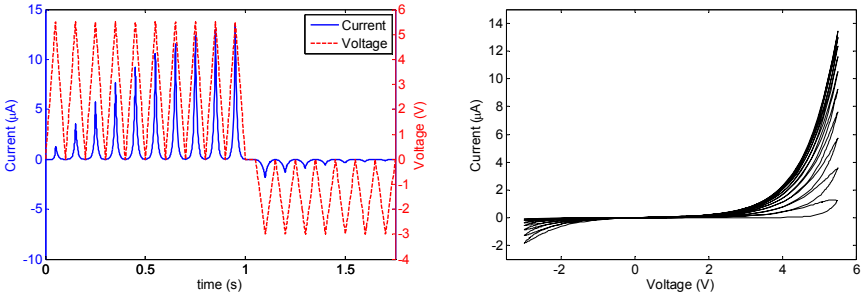


Figure 3.21. Simulation results for the hyperbolic sinusoid model proposed by Laiho et al. with the addition of the Biolek window function. In this simulation: a_1=4(10^{-8}), b_1=1.2, a_2=1.25(10^{-7}), b_2=1.2, c_1=6(10^{-4}), d_1=2, c_2=6.6(10^{-4}), d_2=3.8, and p=1, x_0=0.001. Triangular pulses have magnitude of +5.5V/-3V and a 0.1 second pulse width with rise and fall time of 0.05 seconds.

3.5.2 University of Michigan Model

An alternative model based on the hyperbolic sine I-V relationship was developed by Ting Chang et al. in [20], and the corresponding SPICE code can be seen in Figure 3.22. The I-V relationship can be seen in (3.18) where the first term is due to a Schottky barrier between the oxide layer and the bottom electrode, and the second term is due to the tunneling through the MIM junction. The state variable $x(t)$ is a value between 0 and 1 that represents the ion migrations which determine the conductivity of the device. The motion of the state variable is described by equation (3.19) which is similar to equation (3.16). The fitting parameters η_1, η_2, and λ are used shape the dynamics of the state variable equation. In each of the simulations in Figures 3.23 through 3.24, the constants in the equations were defined as follows: α=5(10^{-7}), β=0.5, γ=4(10^{-6}), δ=2, λ=4.5, η_1=0.004, η_2=4, and τ=10.

$$I(t) = (1 - x(t))\alpha\left[1 - e^{\beta V(t)}\right] + x(t)\gamma \sinh(\delta V(t)) \tag{3.18}$$

$$\frac{dx}{dt} = \lambda\left[\eta_1 \sinh(\eta_2 V(t))\right] \tag{3.19}$$

An alternative function for the motion of the state variable was proposed in [20] to account for the overlapping of multiple hysteresis loops where the device was tested with a repetitive pulse input (see Figure 3.25). The overlap in hysteresis was said to be caused by diffusion of ions within the device. Equation (3.20) shows the modified state variable equation with the diffusion term added. The LTspice code for the model was obtained from [20], and was then modified so that either state variable equation could be used with the change of a binary variable. This allows for the drift component to be turned on or off from the simulation that is using the subcircuit. The resulting SPICE

subcircuit can be seen in Figure 3.22. It should be noted that equations (3.19) and (3.20) were taken from the SPICE code in [20], not from the text in [20]. The equations in the text differed slightly so precedence was given to the equations used to develop the model.

$$\frac{dx}{dt} = \lambda\left[\eta_1 \sinh(\eta_2 V(t)) - \frac{x(t)}{\tau}\right] \tag{3.20}$$

```
* Memristor subcircuit developed by Chang et al.

* Connections:
* TE:  Top electrode
* BE:  Bottom electrode
* XSV: External connection to plot state variable
*      that is not used otherwise

.SUBCKT MEM_CHANG TE BE XSV

* Parameters:
* alpha:      Prefactor for Schottky barrier
* beta:       Exponent for Schottky barrier
* gamma:      Prefactor for tunneling
* delta:      Exponent for tunneling
* xmax:       Maximum value of state variable
* xmin:       Minimum value of state variable
* drift_bit:  Binary value to switch the ionic drift in (1)
*             or out (0) of the equation
* lambda:     State variable multiplier
* eta1, eta2: State variable exponential rates
* tau:        Diffusion coefficient

.param alpha=0.5e-6 beta=0.5 gamma=4e-6 delta=2 xmax=1 xmin=0
+drift_bit = 0 lambda=4.5 eta1=0.004 eta2=4 tau=10

.param cp={1}
Cpvar XSV 0 {cp}

* Rate equation for state variable
Gx 0 XSV value={
trunc(V(TE,BE),cp*V(XSV))*lambda*(eta1*sinh(eta2*V(TE,BE))-
+drift_bit*cp*V(XSV)/tau) }

.ic V(XSV) = 0.0

* Auxiliary functions to limit the range of x
.func sign2(var) {(sgn(var)+1)/2}
.func trunc(var1,var2) {sign2(var1)*sign2(xmax-var2)+sign2(-
+var1)*sign2(var2-xmin)}

* Memristor IV Relationship
Gm TE BE value={(1-cp*V(XSV))*alpha*(1-exp(-
+beta*V(TE,BE)))+(cp*V(XSV))*gamma*sinh(delta*V(TE,BE))}

.ENDS MEM_CHANG
```

Figure 3.22. LTspice code for the for the memristor model proposed by Chang et al. in [20].

The simulation in Figure 3.23 shows the model results when a voltage signal with zero net charge is applied to the memristor device. The result is similar to previous

simulations where a sinusoidal input is applied. In this simulation a prominent curvature can be seen due to the hyperbolic sine term in the I-V relationship. Figure 3.24 shows the model results when repetitive pulses are applied to the device. The multiple-hysteresis pattern looks similar to the result in Figure 3.20 although this model shows a higher conductivity when negatively biased. Figure 3.25 shows the model result when the ion diffusion term was added to the state variable equation. This result appears to be a better match to the characterization data in [20]. The hysteresis loops overlap in the positively biased area, but a larger gap can be seen between the loops when negative pulses are applied. Ion diffusion appears to be a logical explanation for this effect.

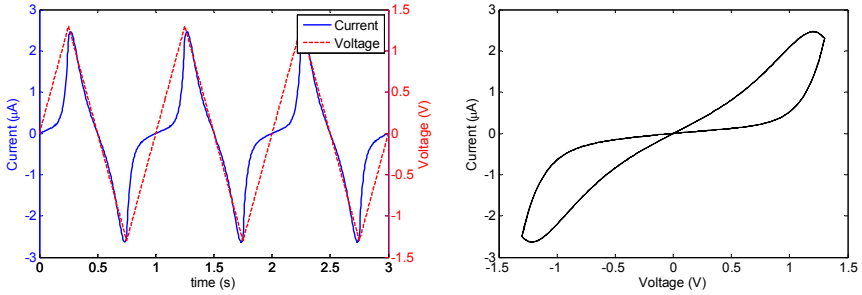


Figure 3.23. Simulation results for the memristor SPICE model proposed by Chang et al. without the ion diffusion term included in the state variable equation. The triangular pulses have a magnitude of 1.25V and a pulse width of 0.5 seconds with a 0.25 second rise/fall time.

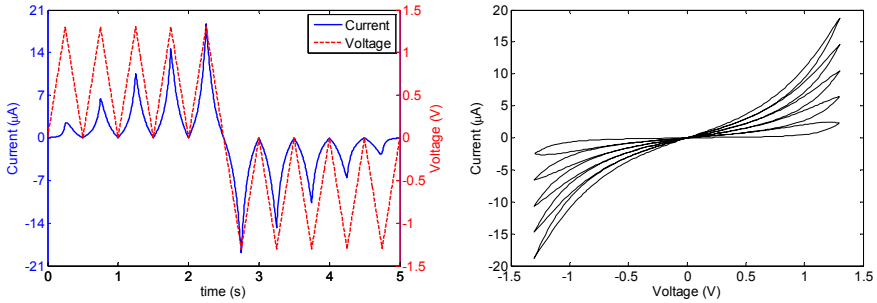


Figure 3.24. Simulation results for the memristor SPICE model proposed by Chang et al. without the ion diffusion term included in the state variable equation. The triangular pulses have a magnitude of 1.25V and a pulse width of 0.5 seconds with a 0.25 second rise/fall time.

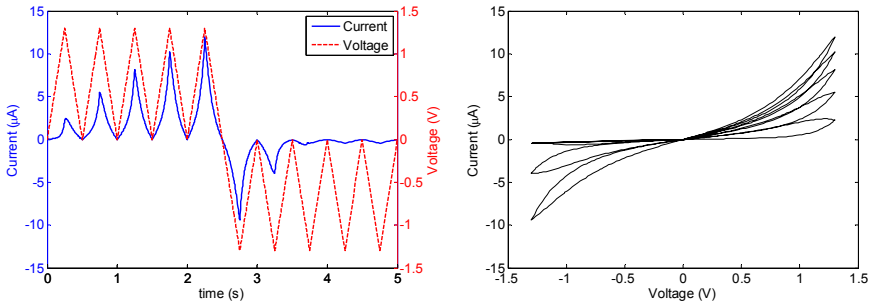


Figure 3.25. Simulation results for the memristor SPICE model proposed by Chang et al. with the ion diffusion term included in the state variable equation. The triangular pulses have a magnitude of 1.25V and a pulse width of 0.5 seconds with a 0.25 second rise/fall time.

3.5.3 Discussion

The models described in this section use a hyperbolic sine function in the I-V relationship which models the characteristics of a memristor well for both single sweep and repetitive pulse inputs. Additional properties were also modeled such as the Schottky barrier at a metal-oxide interface, and the diffusion of ions. This resulted in a stronger correlation to physical memristor characterization data. These models have the potential to describe the functionality of a memristor in a more generalized way which also appears to be quite accurate. The drawback is that these models do not correlate to physical hardware as closely as the models in Section 2.4. The next section describes a generalized model that quantitatively matches published characterization data for a variety of different memristor devices for a variety of different voltage inputs.

CHAPTER IV

GENERALIZED MEMRISTOR MODEL

4.1 Introduction

This chapter presents a SPICE model based on a set of equations that was previously proposed in [8]. To the best of my knowledge, this is the first memristor SPICE model that has been quantitatively correlated to multiple devices for both sinusoidal and repetitive sweeping inputs. The model was correlated to reported current-voltage data of memristor devices published by HP Labs [3, 30] (TiO_2, TaO_x), the University of Michigan [4] (a-Si and Ag), Boise State University [26] (Ag chalcogenide), and Iowa State University [28, 29] (TiO_2). The SPICE model was also matched to Resistive Random Access Memory (RRAM) characterizations [5]. RRAM is a type of non-volatile memory that can be classified as a memristive device, and it also has a dynamic resistance component [5, 93]. Additionally, a simulation was completed to accurately reproduce the experimental pulse switching results of the 40nm a-Si RRAM device in [5]. An accurate representation of memristive I-V characteristics provides the potential for more accurate power and energy calculations in memristor based systems. Studies have been performed that show this SPICE model can be used in circuits containing up to 256 memristors, and when compared with other models, this model is less likely to have convergence problems.

Following the description and verification of the memristor model, device variation studies were completed. Since memristors are fabricated with a significant amount of variation, problems may occur when developing large memristor arrays, or when using memristors for multi-bit operation [3, 4, 26]. Previous work has been completed that uses memristor modeling techniques to study device variation [94-101]. Many of these studies have been completed based on the memristor model first proposed by HP Labs [2]. I believe this chapter presents the first device variation analysis based on a memristor model correlated to actual device fabrication data. The memristor model was also used in a neuromorphic read/write circuit design proposed in [102] to determine how much wafer variation could be tolerated before incorrect reading of data occurs. In addition to memristor thickness, variation in state variable dynamics is also studied.

This chapter is organized as follows: Section 3.2 describes the memristor model and equations, and Section 3.3 displays the memristor and RRAM device simulation results and energy calculations. In Section 3.4, a study is performed that shows how much variation in each fitting parameter can be tolerated before the I-V curve significantly changes shape. Section 3.5 describes a memristor based read/write circuit and shows how much variation can be tolerated in a memristor array before read errors occur.

4.2 Memristor SPICE Model

4.2.1 Memristor Model Equations

The proposed model is matched to memristor current-voltage data based on three different characteristics observed in memristors: a metal-insulator-metal junction in eq.

(4.1), a voltage threshold for state variable motion in eq. (4.2), and a non-linear velocity function for oxygen vacancy or dopant drift in eq. (4.3) and (4.4).

The generalized I-V relationship for the proposed memristor SPICE model can be seen in equation (4.1), and was previously proposed in [12]. This equation is based on a hyperbolic sinusoid to account for the metal insulator metal junction observed in memristor operation. The hyperbolic sine shape causes the device to have an increase in conductivity beyond a certain voltage threshold. The parameters a_1, a_2, and b are used to fit equation (4.1) to the different device structures of the memristors studied in this chapter. Based on existing memristor characterization data, the devices appear to be more conductive in the positive region. To account for this, a different amplitude parameter is required depending on the polarity of the input voltage. The fitting parameter b was used to control the intensity of the threshold function relating conductivity to input voltage magnitude.

The I-V relationship also depends on the state variable $x(t)$, which provides the change in resistance based on the dynamics in each device. In this model, the state variable is a value between 0 and 1 that directly impacts conductivity.

$$I(t) = \begin{cases} a_1\, x(t)\sinh\big(bV(t)\big), & V(t) \geq 0 \\ a_2\, x(t)\sinh\big(bV(t)\big), & V(t) < 0 \end{cases} \tag{4.1}$$

The change in the state variable is based on two different functions, namely, $g(V(t))$ and $f(x(t))$. The function $g(V(t))$ in equation (4.2) is responsible for implementing the threshold voltage that must be surpassed to induce a change in the value of the state variable. This was done because each of the published memristor devices [3, 4, 26-30] show that there is no state change unless a certain voltage threshold is exceeded. These state changes represent either the motion of low mobility ions or dopants [3, 4, 27-30], or

the state change in a chalcogenide device [26]. As opposed to the hyperbolic sinusoid programming threshold implemented in [12], the method in equation (4.2) provides the possibility of having different thresholds based on the polarity of the input voltage. This is required to provide a better fit to the characterization data, since several memristors show different threshold values depending on whether the input voltage is positive or negative. The exponential value subtracted in (4.2) is a constant term during simulations and ensures that the value of the function *g(V(t))* starts at 0 once either voltage threshold is surpassed.

In addition to the positive and negative thresholds (V_p and V_n), the magnitude of the exponentials (A_p and A_n) can be adjusted. The magnitude of the exponential represents how quickly the state changes once the threshold is surpassed. In Section IV, the simulation result for the chalcogenide device [26] displays a very fast change in resistance once the threshold is surpassed. This is modeled using a large value for the magnitude coefficients. Alternatively, the device based on the motion of silver dopants [4] requires a much lower magnitude coefficient, as this appears to be a slower phenomenon.

$$g(V(t)) = \begin{cases} A_p\left(e^{V(t)} - e^{V_p}\right), & V(t) > V_p \\ -A_n\left(e^{-V(t)} - e^{V_n}\right), & V(t) < -V_n \\ 0, & -V_n \leq V(t) \leq V_p \end{cases} \tag{4.2}$$

The second function used to model the state variable *f(x(t))*, can be seen in (4.3) and (4.4). This function models the non-linear ion motion, as it becomes harder to change the state of the devices when the state variable approaches the boundaries. This idea was theorized in [11, 14], and determined experimentally in [36]. This is known as non-linear

dopant drift in the state variable motion, and it is the third memristor effect accounted for in this model.

Also, this function provides the possibility of modeling the motion of the state variable differently depending on the polarity of the input voltage. This is a necessary addition as it has been experimentally verified that the state variable motion is not equivalent in both directions [36]. The memristor device model published in [21] also uses a switching state variable where the motion varies depending on the polarity of the current through the device. One possible explanation for this may be that it is more difficult to put ions back in their original position after they have been previously moved.

When $\eta V(t)>0$, the state variable motion is described by equation (4.3), otherwise the motion is described by (4.4). The term η was introduced to represent the direction of the motion of the state variable relative to the voltage polarity. When $\eta=1$, a positive voltage above the threshold will increase the value of the state variable, and when $\eta=-1$, a positive voltage results in a decrease in state variable (as in [11]).

The function $f(x(t))$ is used to divide the state variable motion into two different regions depending on the existing state of the device. The state variable motion is constant up until the point x_p or x_n. At this point the motion of the state variable is limited by an exponential function decaying with a rate of either α_p or α_n. The parameters (x_p, x_n, α_p, and α_n) are required so that this model is able to match dynamics of several types of devices. These differences may be due to the fact that the motion of the state change in a chalcogenide device is very different than the motion of ions or dopants.

$$f(x)=\begin{cases} e^{-\alpha_p(x-x_p)}w_p(x,x_p), & x \geq x_p \\ 1, & x < x_p \end{cases} \tag{4.3}$$

$$f(x)=\begin{cases} e^{\alpha_n(x+x_n-1)}w_n(x,x_n), & x \le 1\text{-}x_n \\ 1, & x > 1\text{-}x_n \end{cases} \quad (4.4)$$

In equation (4.5), $w_p(x,x_p)$ is a windowing function that ensures $f(x)$ equals zero when $x(t)$=1. In (4.6), $w_n(x,x_n)$ keeps $x(t)$ from becoming less than 0 when the current flow is reversed.

$$w_p(x,x_p)=\frac{x_p-x}{1-x_p}+1 \quad (4.5)$$

$$w_n(x,x_n)=\frac{x}{1-x_n} \quad (4.6)$$

Equation (4.7) is used to model state variable motion in each of the memristor devices. Since the modeled state variable must match devices with many different physical structures, this equation is very different than the equation in [2] that was used to model only TiO_2 devices. The term η is also used in (4.7) to determine the direction of the state variable motion.

$$\frac{dx}{dt}=\eta g(V(t))f(x(t)) \quad (4.7)$$

4.2.2 Summary of Device Model and Relation to Physical Mechanisms

The device model described above is based on three main characteristics namely, electron tunneling, non-linear drift, and a voltage threshold. The I-V relationship in (4.1) uses a hyperbolic sine function to approximate the effect of a tunnel barrier. Equation (4.2) induces a threshold so that the resistance of the device is not changed unless the threshold voltage is surpassed. Lastly, the function $f(x(t))$ in (4.3) and (4.4) is used to provide non-linear drift where state variable motion is slowed near the maximum and minimum resistance boundaries. Table 4.1 shows how each of the parameters in these

equations has a relation to the physical properties that govern the resistance switching mechanism.

Table 4.1. Description of Model Parameters

Parameter	**Relation to Physical Behaviors**
a_1 and a_2	These parameters closely relate to the thickness of the dielectric layer in a memristor device, as more electrons can tunnel through a thinner barrier leading to an increase in conductivity.
b	This parameter determines how much curvature is seen in the I-V curve relative to the applied voltage. This relates to how much of the conduction in the device is Ohmic and how much is due to the tunnel barrier.
A_p and A_n	These parameters control the speed of ion (or filament) motion. This could be related to the dielectric material used since oxygen vacancies have a different mobility depending which metal-oxide they are contained in.
V_p and V_n	These parameters represent the threshold voltages. These may be related to the number of oxygen vacancies in a device in its initial state. A device with more oxygen vacancies should have a larger current draw which may lead to a lower switching threshold if switching is assumed to be based on the total charge applied.
α_p, α_n, x_p and x_n	These parameters determine where state variable motion is no longer linear, and they determine the degree to which state variable motion is dampened. This could be related to the electrode metal used on either side of the dielectric film since the metals chosen may react to the oxygen vacancies differently.

4.2.3 SPICE Model and Equivalent Circuit for the Memristor

The SPICE code seen in Figure 4.1 was written for use in LTSpice, and was used to generate the result seen in Figure 4.3. The parameter sets for modeling different devices can be seen in the description of each corresponding figure.

The SPICE model was developed according to the layout in Figure 4.2. The current source *Gm* in Figure 4.2 (a) holds the I-V relationship for the device, which was presented in equation (4.1). The terminals *TE* and *BE* represent the top and bottom electrodes of the memristor where the model would be connected in a circuit schematic.

Figure 4.2 (b) shows how the value of the state variable is determined using another current source and a capacitor. The current source *Gx* produces a current determined by the multiplication of the functions *g(V(t))* and *f(x(t))*. The capacitor *Cx* is used to integrate the current generated by *Gx* to produce the value of the state variable. This technique is similar to those proposed in other memristor SPICE models [14, 19]. The port *XSV* was created to provide a convenient method for plotting the internal state variable.

```
* SPICE model for memristive devices
* Created by Chris Yakopcic
* Last Update: 12/21/2011
*
* Connections:
* TE - top electrode
* BE - bottom electrode
* XSV - External connection to plot state variable
* that is not used otherwise

.subckt mem_dev TE BE XSV

* Fitting parameters to model different devices
* a1, a2, b:      Parameters for IV relationship
* Vp, Vn:         Pos. and neg. voltage thresholds
* Ap, An:         Multiplier for SV motion intensity
* xp, xn:         Points where SV motion is reduced
* alphap, alphan: Rate at which SV motion decays
* xo:             Initial value of SV
* eta:            SV direction relative to voltage

.params a1=0.17 a2=0.17 b=0.05 Vp=0.16 Vn=0.15
+Ap=4000 An=4000 xp=0.3 xn=0.5 alphap=1 alphan=5
+xo=0.11 eta=1

* Multiplicative functions to ensure zero state
* variable motion at memristor boundaries
.func wp(V) = (xp-V)/(1-xp)+1
.func wn(V) = V/(1-xn)

* Function G(V(t)) - Describes the device threshold
.func G(V) = IF(V <= Vp, IF(V >= -Vn, 0, -An*(exp(-
+V)-exp(Vn))), Ap*(exp(V)-exp(Vp)))

* Function F(V(t),x(t)) - Describes the SV motion
.func F(V1,V2) = IF(eta*V1 >= 0, IF(V2 >= xp, exp(-
+alphap*(V2-xp))*wp(V2) ,1), IF(V2 <= (1-xn),
+exp(alphan*(V2+xn-1))*wn(V2) ,1))

* IV Response - Hyperbolic sine due to MIM structure
.func IVRel(V1,V2) = IF(V1 >= 0, a1*V2*sinh(b*V1),
+a2*V2*sinh(b*V1) )

* Circuit to determine state variable
* dx/dt = F(V(t),x(t))*G(V(t))
Cx XSV 0 {1}
.ic V(XSV) = xo
Gx 0 XSV
+value={eta*F(V(TE,BE),V(XSV,0))*G(V(TE,BE))}
* Current source for memristor IV response
Gm TE BE value = {IVRel(V(TE,BE),V(XSV,0))}

.ends mem_dev
```

Figure 4.1. Memristive device subcircuit for use in LTSpice.

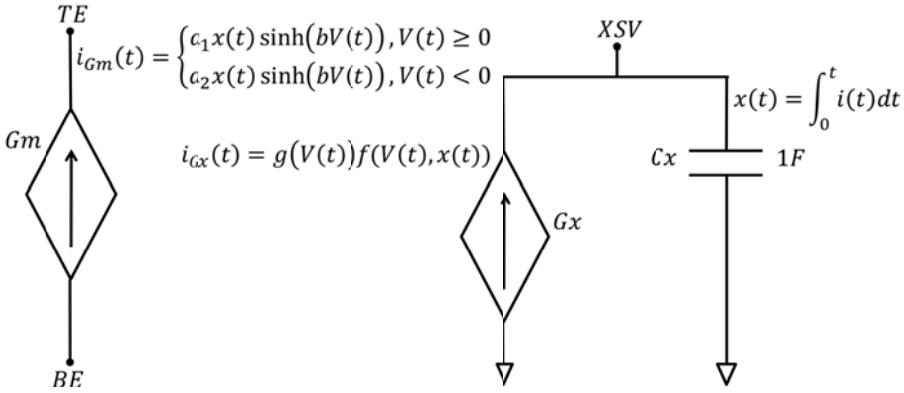


Figure 4.2. Diagrams that describe the functionality of the SPICE model where the circuit in (a) determines the I-V relationship of the memristor model and (b) shows how the value of the state variable is determined.

4.2.4 SPICE Model Convergence

To study the convergence of this SPICE model, a 4×4 crossbar (16 memristors) with wire resistance and access transistors bordering the rows and columns was utilized. A write iteration is performed across an entire row in parallel. A single read iteration is considered to be a sequential read of each row in the crossbar to ensure the correct data is present after each write. The simulations used 10ns write and erase pulses along with 0.3ns read pulses to test high speed switching in the model. In this case the model was matched to the 10ns switching device in [103].

The model performed very consistently without error for a number of simulations for at least 400 read/write iterations (1600 writes and 6400 reads). An 8×8 crossbar (64 memristors) produced a similar result where a convergence error was rarely obtained within 400 read/write iterations (3200 writes and 25600 reads). In a 16×16 crossbar (256 memristors) convergence errors started to become more common, although many simulations were performed where 200 read/write iterations (3200 writes and 51200 reads) were completed without error.

A number of other SPICE models [11, 12, 14, 20] were used to simulate the 4×4 crossbar setup to compare them in terms of convergence rate (see Table 4.2). In cases where they previously did not exist, SPICE subcircuits were made for these models that could be used in LTSpice. Some models [11, 14, 20] suffered from convergence errors, especially when several devices in the crossbar were switched to their minimum resistance state. This could be due to the fact that the model parameters had to be changed significantly to achieve nanosecond switching, and these models were not designed for this. The model in [12] (where the Biolek window [14] was added to define boundaries) was able to successfully complete the simulation of the 4×4 crossbar, although the model presented in this chapter correlates to actual device data much more closely. The models [11, 14] can be used to model many devices, but they have a low correlation to physical characterization data. The models [12, 20] show a stronger correlation to physical devices, but they have not been matched quantitatively. The proposed model has been quantitatively matched to several different memristors.

Table 4.2. Time before convergence error of different SPICE models for a nanosecond switching 4×4 crossbar.

Measurement	**Memristor Models Included in the Study**				
	Biolek [14]	***Joglekar [11]***	***Laiho [12]***	***Chang [20]***	***Proposed Model***
Time Before Error (μS)	0.517	0.517	15 (Finished)	0.135	15 (Finished)
Accuracy	Low	Low	Medium	Medium	High
Generality	High	High	Medium	Medium	High

4.3 I-V Simulation Results

4.3.1 Memristor Results

To display the functionality of the SPICE model described in the previous section, each of the published memristor devices [3, 4, 26-30] was modeled in an I-V simulation.

Figure 4.3 displays the first simulation result where the device in [26] was modeled with a sinusoidal input both at 100 Hz and 100 kHz. The hysteresis in the model diminished when the frequency was increased to 100 kHz just as it did in [26]. The simulated I-V characteristic was matched to the 100 Hz data provided in [26] using selected data points (dots in Figure 4.3 (b)) with an average error of 84.8μA (6.64%). The percent error was determined by using the sum of the error divided by the absolute sum of the currents from the device characterizations at each of the tested points.

Figure 4.4 shows the result when matching the model to the repetitive DC sweeping input also provided in [26]. The model was able match the characterization data with an average error of 32.7μA (6.66%). The characterization data provided for this simulation was only available for when positive voltage sweeps were applied. It was assumed that the parameters for the negative regime would closely match the parameters used in the sinusoidal simulation in Figure 4.3. The one exception was that the negative conductivity parameter a_2 was set to the value decided for a_1 in the repetitive sweeping simulation.

When comparing the parameters used to model the device characterized in [26], it can be seen that very few adjustments were necessary when switching between the two modes of operation. Out of the total 12 parameters, 9 of them remained the same between the simulations in Figures 4.3 and 4.4. The three parameters that were changed include the conductivity parameters a_1 and a_2, and the initial position of the state variable x_0. Each of these parameters is highly related to the device thickness. These differences could have been caused by non-uniformities in the wafer if the characterizations in [26] were conducted on different devices.

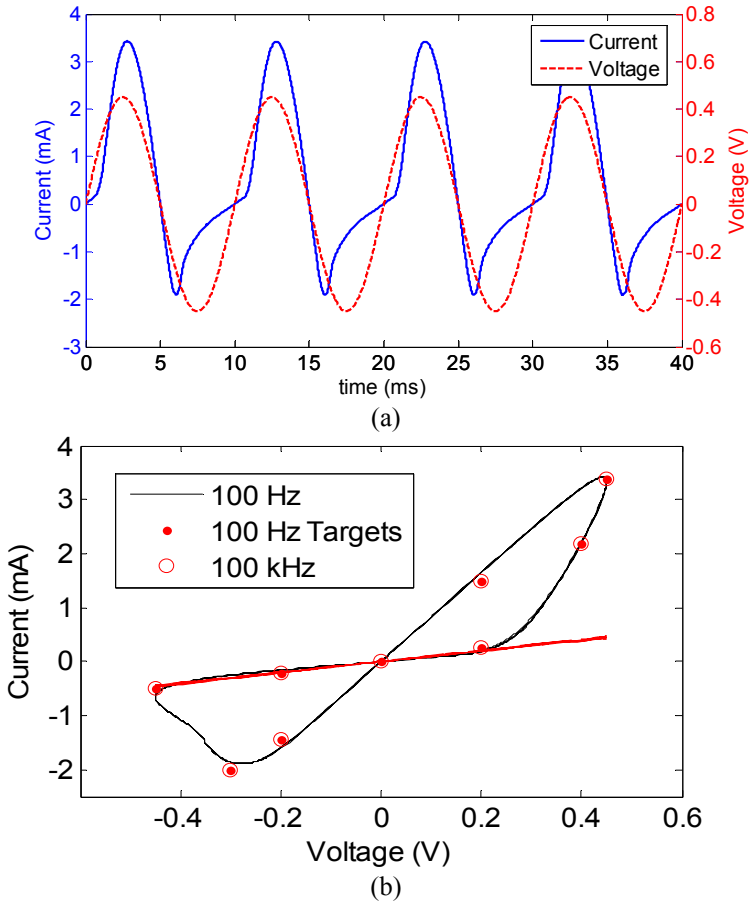
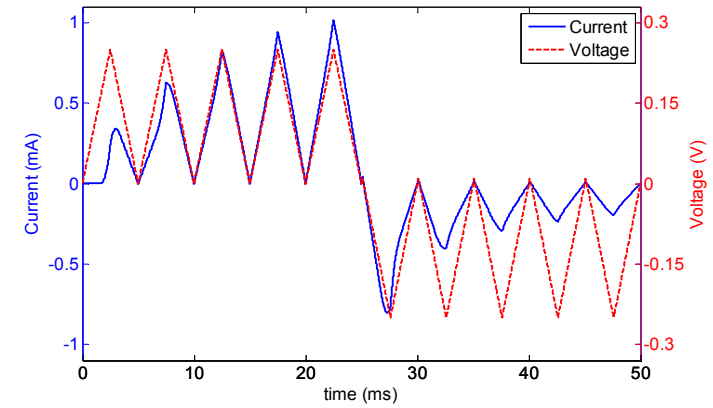
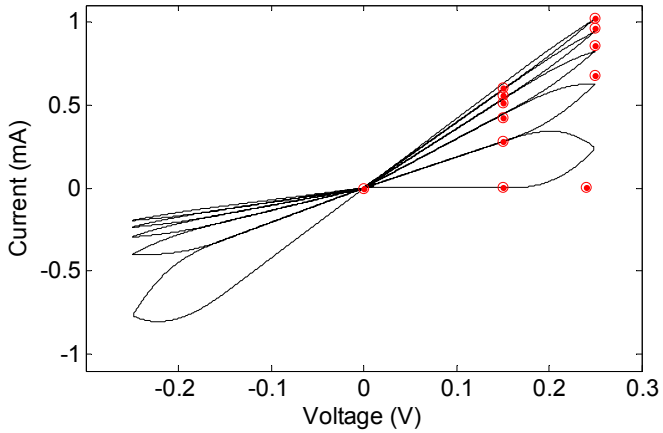


Figure 4.3. Simulation results when modeling the device in [26] for a sinusoidal input (Dots in (b) show target data points). The plots show (a) the current and voltage waveforms and (b) the I-V curve. The plot in (b) also shows the I-V curve for a high frequency input where the device behaves as a linear resistor. In this simulation: V_p=0.16V, V_n=0.15V, A_p=4000, A_n=4000, x_p=0.3, x_n=0.5, α_p=1, α_n=5, a_1=0.17, a_2=0.17, b=0.05, x_0=0.11, η=1.

Figure 4.5 displays the simulation results of a device developed by HP Labs. The simulation in Figure 4.5 was based on the characterization data provided in [30] where a cyclic voltage sweep was applied to a TaO_x memristor device. The average error in this case was determined to be 81.1 µA (4.60%).



(a)



(b)

Figure 4.4. Results matching the characterization data in [26]. Dots in (b) show the points from [26]. In this simulation: V_p=0.16V, V_n=0.15V, A_p=4000, A_n=4000, x_p=0.3, x_n=0.5, α_p=1, α_n=5, a_1=0.097, a_2=0.097, b=0.05, x_0=0.001, η=1.

The simulation in Figure 4.6 was based on the device characterized in [4]. This device characterization required about 20 seconds to complete. The simulation matches each target data point with an average error of 20.0nA (6.21%).

The plots in Figure 4.7 correspond to the memristor developed in [28, 29]. The simulated I-V characteristic was matched to target data points with an average error of 5.97%. The input voltage waveform was replicated based on the data provided in [29]. The I-V characteristic in [28, 29] shows 3 sequential cyclic voltage sweeps. Figure 4.7 shows the results when modeling target data from the third sweep since the decay in the first two sweeps is most likely due to initial forming and would not be present over a

large number of cycles. Contrary to the previous simulations, this device was characterized so that the device conductivity decreases as positive voltage is applied. To accommodate for this, the variable η was set to –1.

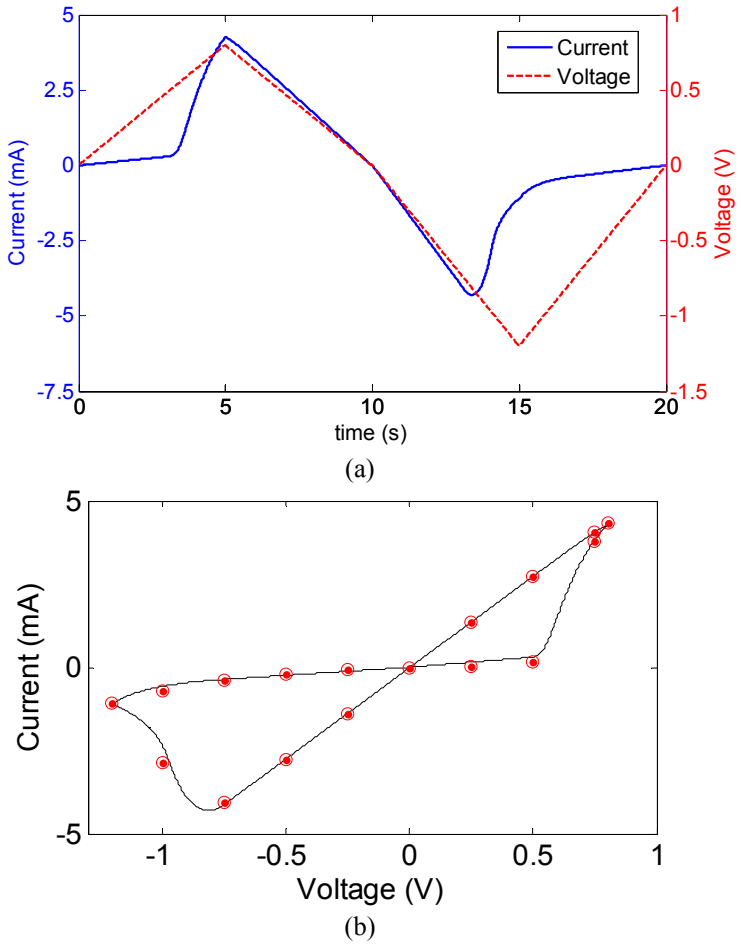
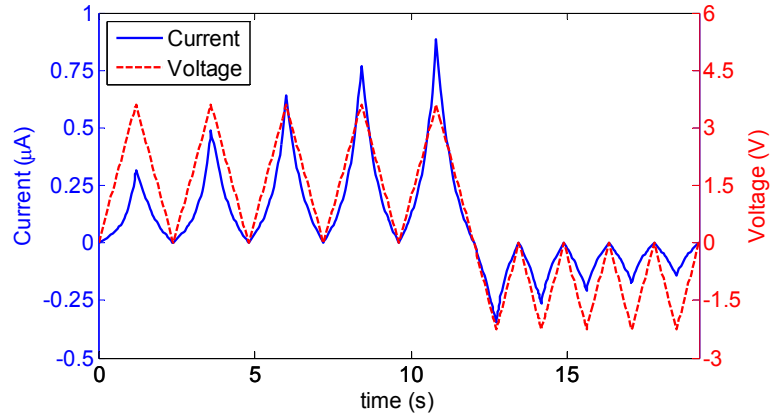
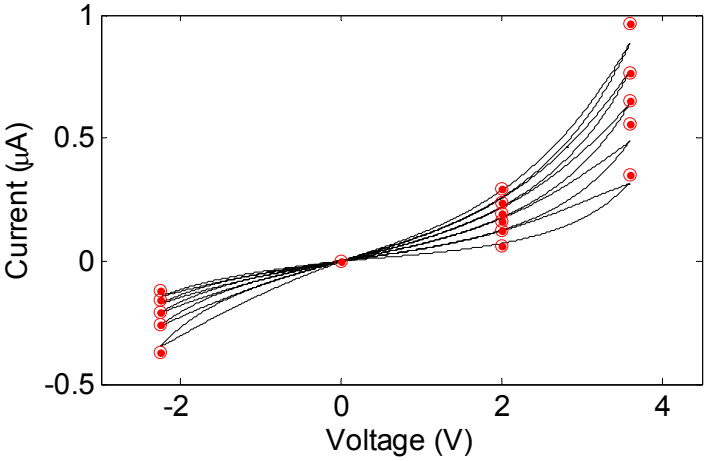


Figure 4.5. Results when modeling the device in [30] for a cyclical DC sweep (dots in (b) show target data points). V_p=0.5V, V_n=0.75V, A_p=7.5, A_n=2, x_p=0.3, x_n=0.5, α_p=1, α_n=5, a_1=0.11, a_2=0.11, b=0.5, x_0=0.11, η=1.

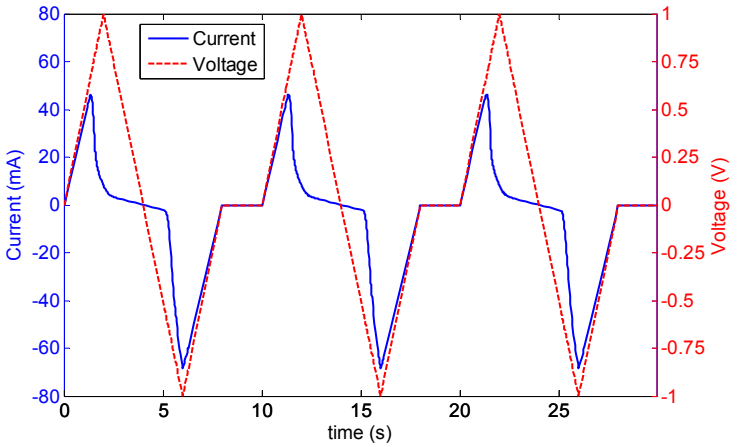


(a)

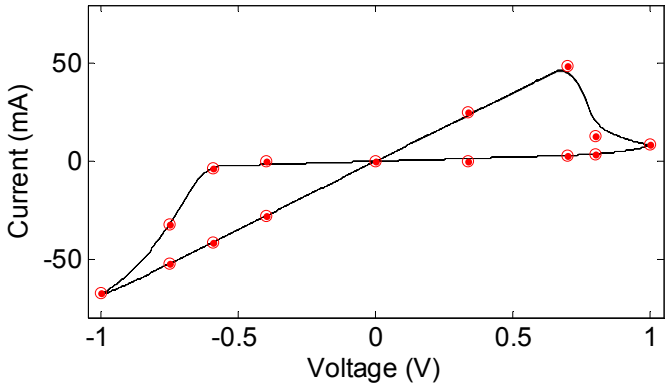


(b)

Figure 4.6. Results obtained for matching characterization in [7]. Dots show the points from [4]. In this simulation: V_p=1.5V, V_n=0.5V, A_p=0.005, A_n=0.08, x_p=0.2, x_n=0.5, α_p=1.2, α_n=3, a_1=3.7(10^{-7}), a_2=4.35(10^{-7}), b=0.7, x_0=0.1, η=1.



(a)



(b)

Figure 4.7. Results obtained for matching the characterization in [28, 29]. Plots show (a) the voltage and current waveforms and (b) the I-V curve. Dots in (b) show the target date points. In this simulation: V_p=0.65V, V_n=0.56V, A_p=16, A_n=11, x_p=0.3, x_n=0.5, α_p=1.1, α_n=6.2, a_1=1.4, a_2=1.4, b=0.05, x_0=0.99, η= –1.

4.3.2 RRAM IV Characteristic

The following simulations show the how this device model is also able to simulate the effects of resistive random access memories (RRAMs), a more general class of memristive devices. A simulation was completed that matches the SPICE model to the characterization data for the device in [5] with a 40nm thick a-Si layer. The simulation results in Figure 4.8 show where the model is matched to the target points in the characterization with an average error of 206µA (6.15%). The characterization data in [5] shows that the current in the I-V curve for the 40nm a-Si device was limited to 10mA. This was not done in the simulation since this limiting effect is most likely due to the

system used to characterize the device, and is not a property of the device itself. The frequency of the voltage input in this simulation was assumed to be 100 MHz based on the characterization results [5] that show the device switching using 5 and 10 ns pulses.

The state variable parameters for this device can be described intuitively. The voltage thresholds (V_p and V_n) are set at points where device leaves its zero conductivity state, and the magnitudes of the exponential (A_p and A_n) were set very high due to the fast switching. The limits of free state variable motion (x_p and x_n) were set very close to the boundaries, and the decay rates (α_p and α_n) were set high, as the state variable appears to have a more linear motion in this device. In general, this device appears to have a much more symmetric operation than the memristor devices in [26-30].

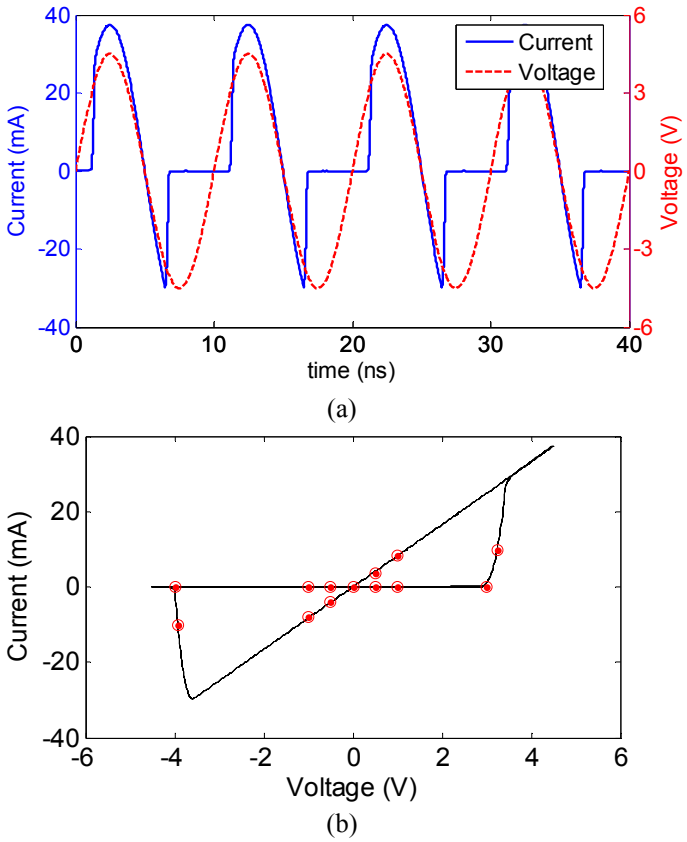


Figure 4.8. Results obtained when matching the RRAM characterization in [28]. The plots display (a) the voltage and current waveforms, and (b) the I-V curve where the dots in (b) show target points from the data in [28]. In this simulation: V_p=2.86V, V_n=3.56V, A_p=5.5(10^8), A_n=4(10^8), x_p=0.9, x_n=0.9, α_p=20, α_n=20, a_1=0.165, a_2=0.165, b=0.05, x_0=0.01, η=1.

4.3.3 RRAM Switching Results

The device in [5] with the 40 nm thick a-Si layer was characterized using pulses to switch the device on and off. This result was simulated to further validate the SPICE model. The simulation was completed by placing the RRAM model in series with a 90Ω resistor and applying a voltage waveform similar to the one in the published characterization [5] (see Figure 4.9). It was stated in [5] that a pulse width of 5 ns was used to switch the device on, and a 10 ns pulse was used to switch the device off. Also, the peak voltages of the write pulses were 6.5 and -6.5 V. The voltage input in the simulation alternated the positive and negative write and erase pulses every 100μs. Wherever a write pulse was not applied, a 2V bias with additive noise was used to read the device (see Figure 4.10 (a)). The other two plots in Figure 4.10 (a) show the voltage drop across the 90Ω resistor and the change in the state variable. The voltage across the resistor is seen as the read voltage, and it changes due to the voltage division between the resistor and the memristor. The read voltage can be seen switching between 0 and about 0.8 V just as it did in the device characterization [5]. The state variable plot shows the device fully switching between 0 and 1, the minimum and maximum values. This confirms that the 5 and 10 ns pulses are capable of fully switching the device model.

Figure 10 (b) shows a zoomed in section of the simulation across one of the write pulses. In the state variable plot, much more detail can be seen in the switching of the device. Given the Gaussian shape of this pulse (as in [5]), the voltage is only above the programming threshold of the device for a short amount of time. This could explain why a 5 ns pulse is needed to switch the device when the actual switching effect occurs in about 1 ns.

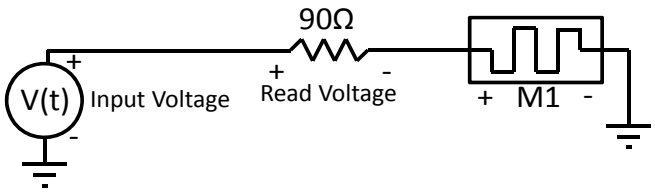


Figure 4.9. Circuit used to perform the resistance switching simulation based on the data published in [5].

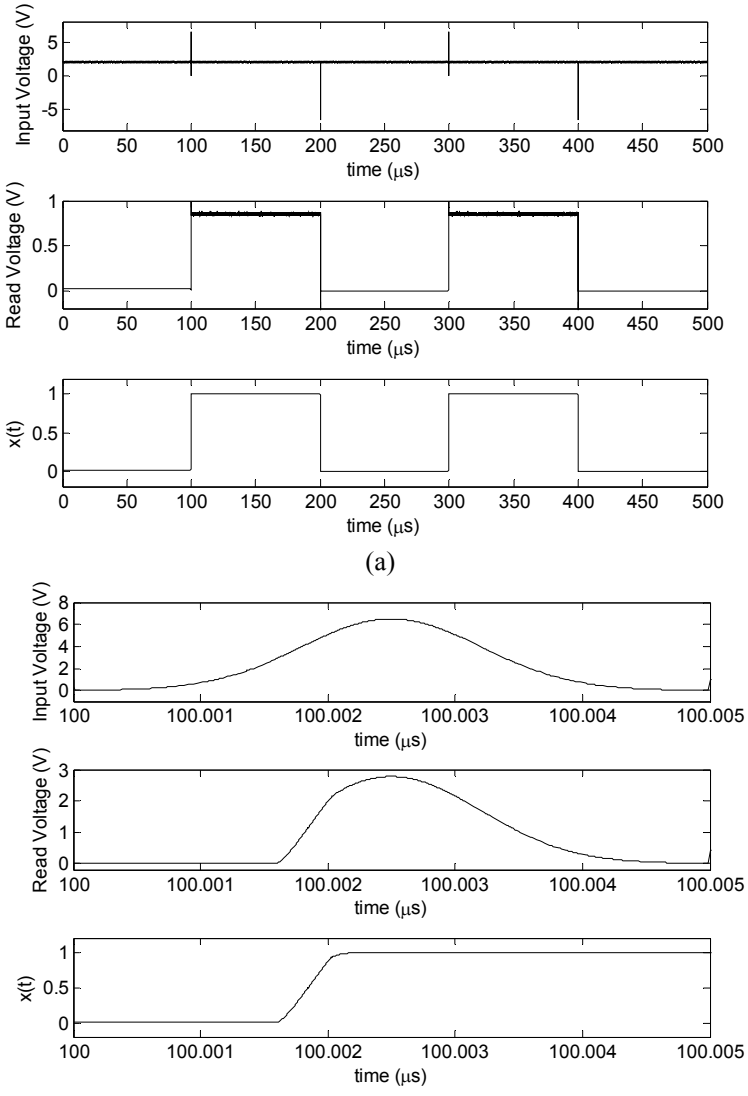


Figure 4.10. Results obtained for modeling the read write characterization published in [5]. The simulation results are plotted displaying (a) the input voltage, the read voltage, and the state variable motion for the entire simulation, and (b) the same data on a time scale relative to the width of a write pulse.

4.3.4 Simulation to Match Published High-Speed Switching Characteristics

Figures 4.11 and 4.12 display results when modeling high-speed switching observed in some memristor devices. The simulation result seen in Figure 4.11 was also

performed to match a TaO_x device characterization in [30]. This device had a different geometry than the one modeled in Figure 4.5, and the characterization in [30] was performed at a much higher speed. Figure 4.11 (a) displays a string of alternating write and erase pulses with a Gaussian shape, each followed by a lower magnitude read pulse. The read pulse is applied below the voltage threshold of the memristor device so data is not corrupted during a read operation. The Gaussian write pulses were chosen so that the write pulses in [30] could be matched as closely as possible where the full width at half maximum of the write and erase pulses was 1.5ns and 1.91ns respectively.

The published data [30] shows that the on state of the physical memristor that has been modeled in this section has a resistance of about 100 Ω. Also, the off state of this memristor has a slight amount of variance, but is centered at about 10^5 Ω. This correlates closely to the simulation data presented where the on and off state resistances of the model are 100.08 Ω and 93,360 Ω respectively.

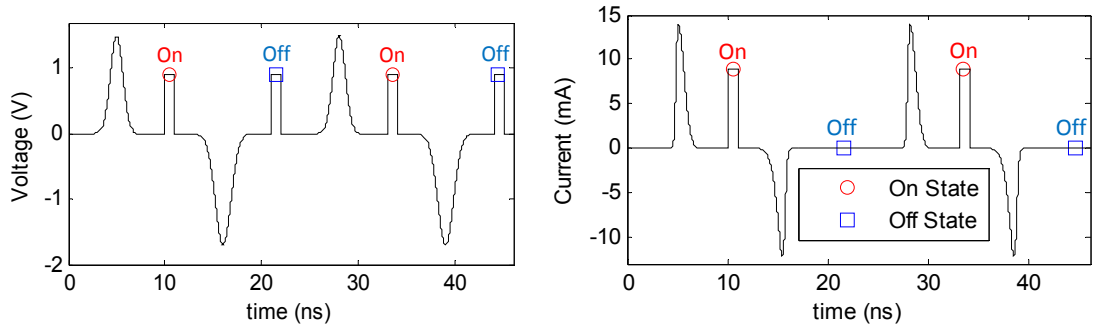


Figure 4.11. Voltage and current waveforms simulated to match the 10ns switching memristor device described in [30]. The following fitting parameters were used in the model to obtain this result: V_p=1.1V, V_n=1.1V, A_p=1.9(10^9), A_n=1.9(10^9), x_p=0.675, x_n=0.675, α_p=0.01, α_n=0.01, a_1=0.2, a_2=0.2, b=0.05, x_0=0.001.

The result in Figure 4.12 is a similar simulation where a different published device [103] is modeled based on switching characteristics. When compared to the device simulated in Figure 4.11, this device has a slightly longer switching time (10ns) and a larger off to on ratio (10^6). Additionally, this device operates at a much lower power with

an on state resistance of about 125kΩ (determined by the 8μA current from a 1V read pulse). This correlates closely to the simulation data presented where the on and off state resistances of the model are 124.95kΩ and 125.79×10^{9}Ω respectively.

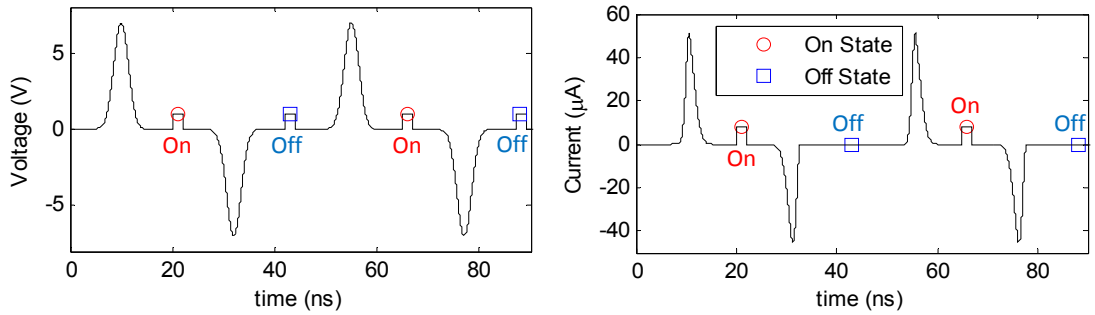


Figure 4.12. Voltage and current waveforms simulated to match the 10ns switching memristor device described in [103]. The following fitting parameters were used in the model to obtain this result: V_p=1.088V, V_n=1.088V, A_p=816000, A_n=816000, x_p=0.985, x_n=0.985, α_p=0.1, α_n=0.1, a_1=1.6(10-4), a_2=1.6(10-4), b=0.05, x_0=0.01.

4.3.5 Memristor Switching Energies

This section shows how the results of the simulations in Figures 4.13 and 4.14 can be used to calculate the switching energy of a memristor device. Both a set and reset pulse was applied to the model which has been matched to the devices in [30, 103]. The applied voltage was then used to determine the total set and reset energies for the devices. The energy calculation in this section is based on the memristor model and the data present in the published device data [30, 103]. This energy calculation does not consider the potential snapback effect present in some resistive switching devices [104, 105].

The waveforms obtained from the simulation include the input voltage applied across the memristor, the memristor current, and the memristor state variable. The memristor state variable shows that the voltage applied is sufficient to fully switch the device as desired. The time varying power in the memristor device is determined by multiplying the current and voltage waveforms. Lastly, the power is integrated to determine the energy. It can be seen that there are two steady energy levels other than the initial value

of zero. The on switching is equivalent to the energy value at Level 1, and the off switching energy is determined by subtracting the energy at Level 1 from Level 2. The on and off switching energies for the device simulated in Figure 4.13 were determined to be 15.6pJ and 12.97pJ respectively. These relatively high values are mainly due to the low on state resistance of 100Ω. The device simulated in Figure 4.14 was determined to have a much lower switching energy with values of 0.54pJ (on) and 0.46pJ (off).

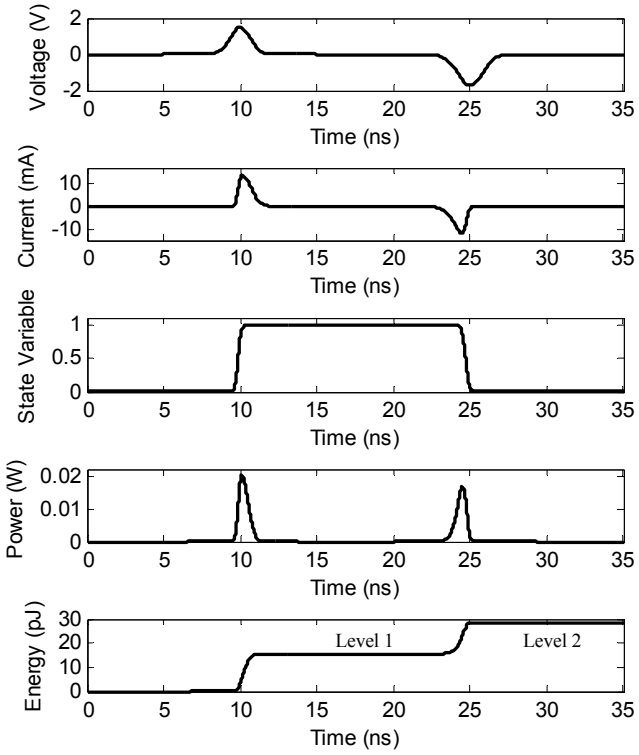


Figure 4.13. Resulting waveforms and energy calculation for the device in [30].

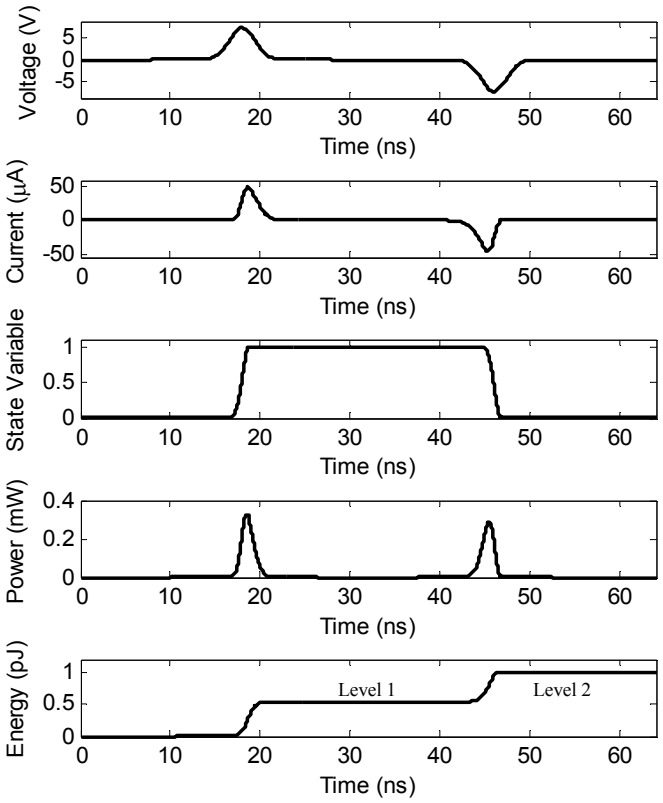


Figure 4.14. Resulting waveforms and energy calculation for the device in [103].

4.4 Memristor Parameter Variation

Given the nanoscale sizes of memristors, it is difficult to fabricate a memristor wafer without some amount of variation between the devices. Several studies have been presented previously that describe an analysis of the variation in thickness across an array of memristors [94-101]. This is then related to the variation in resistance values attainable within an array of memristors. However, more complex variation in memristor wafer composition may have an impact on state variable dynamics. For example, a reactive sputter technique could produce a memristor wafer with a uniform thickness, but with a large variation in the stoichiometric ratio of the dielectric layer. Previous memristor wafer variation studies based layer thickness alone would not be able to address this issue. This section provides an alternate memristor study based on a variation in the state

variable dynamics, which is more likely related to variations in composition such as the amount of oxygen deficiencies present across the wafer.

There are four different pairs of fitting parameters in the model used in equations $f(x(t))$ and $g(V(t))$ that relate to the switching within a device. The equation $f(x(t))$ in (4.3) and (4.4) is used to shape the non-linear motion due to the boundaries within the device and contains parameters x_p, x_n, α_p, and α_n. The equation $g(V(t))$ in (4.2) induces a threshold in the memristor device where the resistance state will not change unless a certain voltage is surpassed. This equation contains the parameters V_p, V_n, A_p, and A_n. The parameters V_p and V_n represent the positive and negative threshold voltages that must be surpassed to decrease or increase the resistance of the device respectively. The parameters A_p and A_n are amplification factors that determine the magnitude of resistance change once the threshold voltage has been surpassed. The parameters x_p and x_n represent the points where state variable motion is no longer linear and is reduced due to the effect of the minimum and maximum resistance boundaries. Lastly, the parameters α_p and α_n are used to control how rapid the decay in state variable motion occurs as the resistance reaches the boundary values.

4.4.1 Process Variation and Change in Memristor Resistance States

It is likely that variations in a memristor wafer will lead to inconsistencies in how the devices operate. In this study each pair of parameters in the equations (4.1) through (4.4) was varied to show the changes in both the shape of the I-V curve, and the change in position of 6 resistance states. This was completed by setting the model to match the device in [26] developed at Boise State University. The resistance states are defined as S_1 through S_6 in Figure 4.15 on either side of each of the 5 hysteresis loops. These resistance

states are determined by the state variable equation (4.7), which is meant to model either the motion of oxygen vacancies or the formation of metallic filaments within the devices.

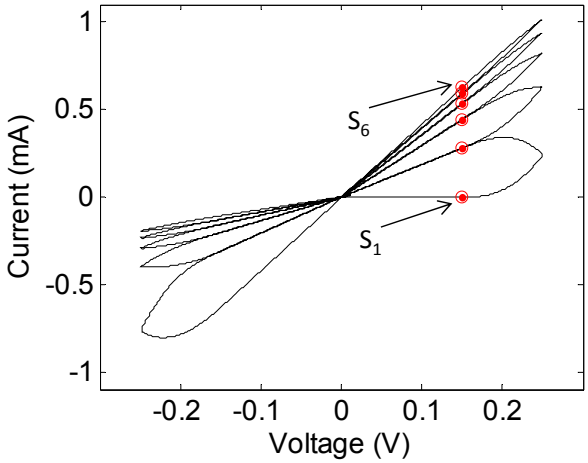


Figure 4.15. Model output with fitting parameters optimized for matching the I-V characteristic in [26]. V_p=0.16V, V_n=0.15V, A_p=4000, A_n=4000, x_p=0.3, x_n=0.5, α_p=1, α_n=5, a_1=0.097, a_2=0.097, b=0.05, x_0=0.001, η=1.

Figure 4.16 displays the result obtained when varying the parameters A_p and A_n. The top plot in Figure 4.16 shows how the position of the resistance states changes as the parameter A_p is varied. The dot-dashed line denotes the original value of A_p, and the dashed lines to the left and right correspond to the values of A_p for the I-V curves in Figure 4.16. As the values of A_p and A_n increase, the initial hysteresis loop becomes larger since more resistance change can now occur within a single voltage sweep. Hysteresis shrinks as the point of maximum conductivity is reached. As the values for A_p and A_n decrease the opposite occurs, and less change is induced with each repetitive DC sweep.

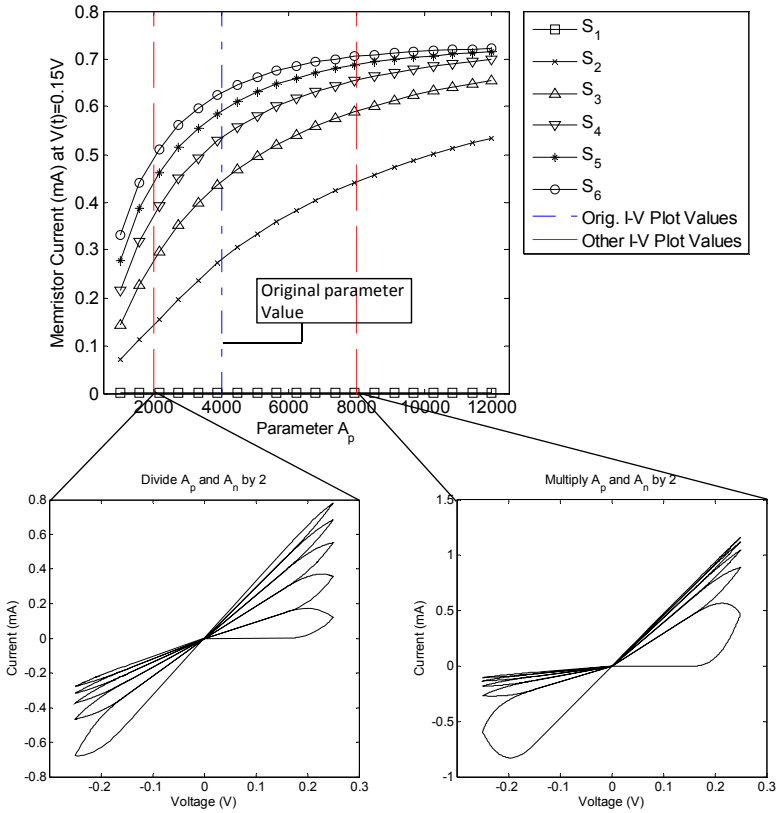


Figure 4.16. Plots displaying the 6 resistance states in the memristor as a function of the parameter A_p, and the modified memristor model output where $A_p = A_n = 2000$ and $A_p = A_n = 8000$. In this simulation: $V_p = 0.16$V, $V_n = 0.15$V, $x_p = 0.3$, $x_n = 0.5$, $\alpha_p = 1$, $\alpha_n = 5$, $a_1 = 0.097$, $a_2 = 0.097$, $b = 0.05$, $x_0 = 0.001$, $\eta = 1$.

Figure 4.17 displays a similar result, except the parameters V_p and V_n are varied, which represent the voltage thresholds required to induce a resistance state change. The plots in Figure 4.17 show a much more significant change in resistance states and the shape of the I-V characteristic as V_p and V_n are varied. Once the threshold voltage V_p became greater than maximum input voltage, the memristor device behaved as a linear resistor. The last 2 pairs of parameters (the pair α_p and α_n and the pair x_p and x_n) are used to shape the function $f(x(t))$, and Figures 4.18 and 4.19 display the effects of varying each of these pairs of parameters. These two parameter variations have less impact on the change in the resistance states when compared to the parameter variations in Figures 4.16 and 4.17.

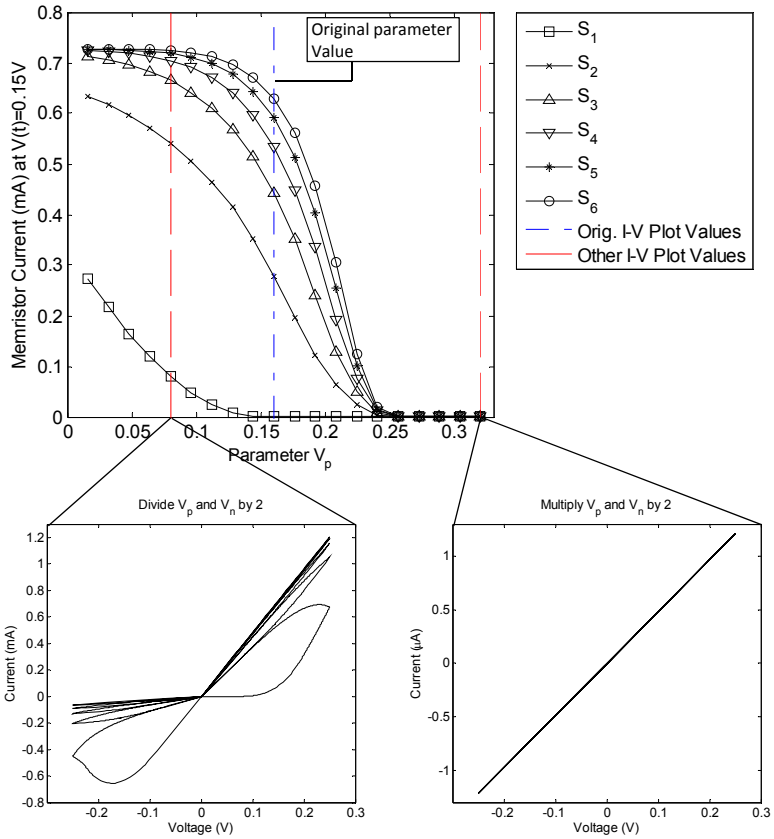


Figure 4.17. Plots displaying the 6 resistance states in the memristor as a function of V_p, and the modified model output where V_p=0.08V and V_n=0.075V and V_p=0.32V and V_n=0.3V. In this simulation: A_p=4000, A_n=4000, x_p=0.3, x_n=0.5, α_p=1, α_n=5, a_1=0.097, a_2=0.097, b=0.05, x_0=0.001, η=1.

When relating these results to variation in physical device properties, it is most likely that a change in the parameters A_p and A_n will be produced by the amount of oxygen vacancies present in a device. When comparing the two I-V plots in Figure 4.16 it can be seen that the parameters A_p and A_n have a stronger relation to device conductivity than any of the other 8 state variable parameters. The amount of oxygen vacancies present would also provide a significant change in resistance of the device. The parameters α_p, $\alpha_{n,}$ x_p and x_n may be subject to variation depending on the quality of the thinfilm. If sections of a thinfilm become more polycrystalline as opposed to amorphous, then this may shape the paths for the oxygen deficiencies to form conductive filaments. Lastly, the parameters V_p and V_n may also change depending on the number of oxygen

deficiencies present within a specific device. If more oxygen vacancies are present, then the device will draw a larger current and most likely have a lower switching voltage. This is assuming that switching is based on the amount of energy applied. In general, this study shows how variation in the state variable parameters affects the conductivity of memristor devices.

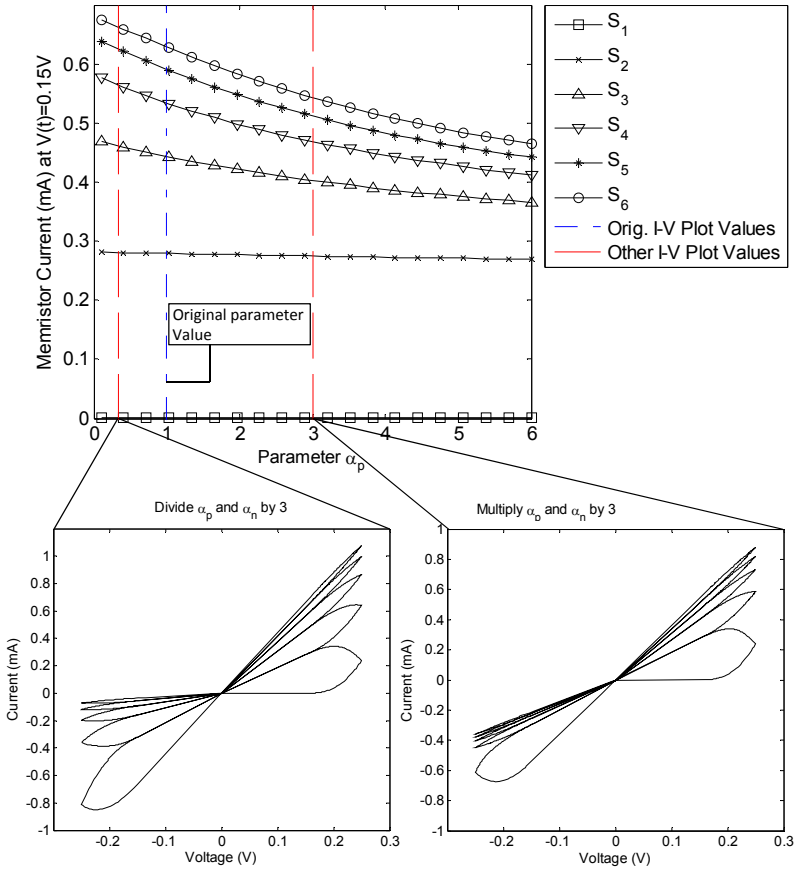


Figure 4.18. Plots displaying the 6 resistance states in the memristor as a function of α_p, and the modified model output from where α_p=.333 and α_n=1.667 and α_p=3 and α_n=15. In this simulation: V_p=0.16V, V_n=0.15V, A_p=4000, A_n=4000, x_p=0.3, x_n=0.5, a_1=0.097, a_2=0.097, b=0.05, x_0=0.001, and η=1.

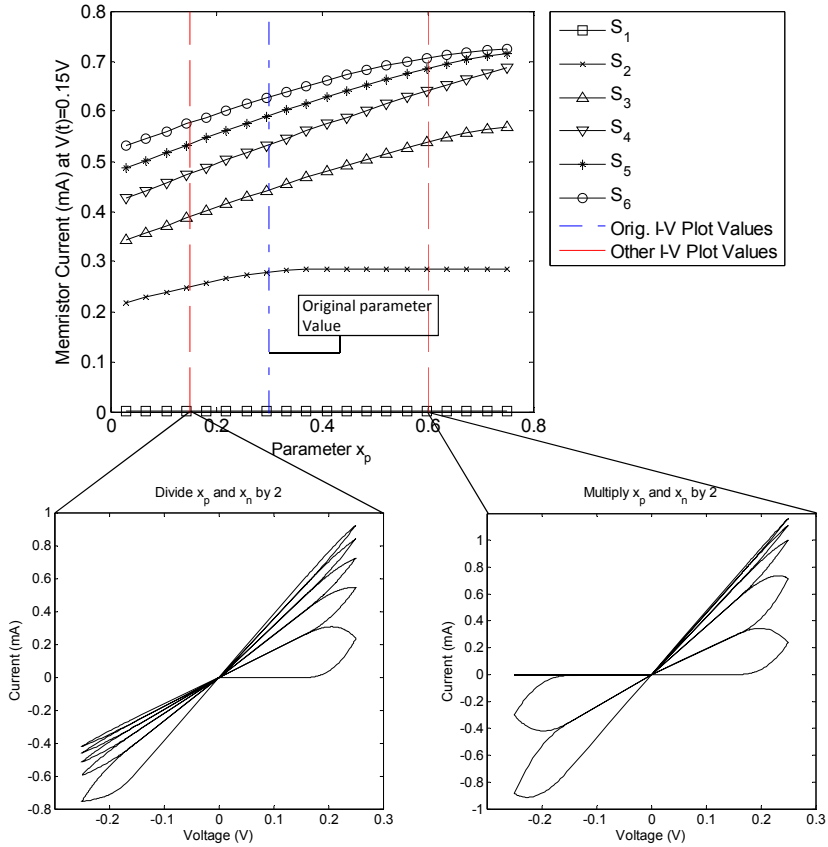


Figure 4.19. Plots displaying the 6 resistance states in the memristor as a function of x_p, and the modified model output where x_p=0.15 and x_n=0.25 and x_p=0.6 and x_n=0.999. In this simulation: V_p=0.16V, V_n=0.15V, A_p=4000, A_n=4000, x_p=0.3, x_n=0.5, α_p=1, α_n=5, a_1=0.097, a_2=0.097, b=0.05, x_0=0.001, and η=1.

4.4.2 Modeling Variation through Change in Model Parameters

The memristor model was used to determine how much variation in a memristor array can be tolerated for each of the different devices characterized in Section IV. This was done by independently adjusting each of the parameters in the model and observing the amount of change in the I-V curve.

The devices considered in this study were characterized in [3, 4, 26-30]. The simulated I-V characteristic for many of these devices can be seen in Figures 4.3 through 4.8. For each device that was modeled, each of the fitting parameters was adjusted until the change in the I-V characteristic exceeded 10%. The 10% change in each I-V characteristic is measured using the total average difference between the altered I-V

curve and the initial for each simulated I-V characteristic. Tables 4.3 and 4.4 display the percentages that each pair of fitting parameters can be increased and decreased respectively.

The memristor device that stands out as the most sensitive to parameter variation is the HP Labs titanium oxide device [3]. This device could tolerate the least amount of change in each of the parameters with the exception of the pair x_p and x_n. The RRAM device is least tolerant to change in the parameters that control the function $g(V(t))$, but most tolerant to change in the parameters that control the function $f(x(t))$. The parameters in the function $f(x(t))$ relate to the non-linear drift in the state variable. Since the change of state is so large once the threshold voltage is surpassed, the function $f(x(t))$ has much less impact on the I-V characteristic compared to the other devices. Since the threshold function $g(V(t))$ is mainly responsible for the state change, much less variation can be tolerated in the corresponding fitting parameters.

In nearly all the devices, the parameters that relate to the I-V equation (a_1, a_2, and b) had a very linear effect on the output of the IV curve. A 10% change in the parameter had exactly a 10% change in the IV curve. The exception to this was when the starting value for b was larger (with a value of 1.5 or 3). A higher b value induces more non-linearity in the output of the model, and thus a more non-linear pattern in the scaling of b is observed. The parameters a_1 and a_2 induced a linear change in all cases because these are directly multiplied in the I-V characteristic equation.

The state variable parameters that could tolerate the most amount of change were the damping parameters (α_p and α_n). In many cases this parameter pair reached zero before a

10% change could be observed in the IV characteristic. This is the reason for the 0.00% values in Table 4.3.

The state variable parameter that was most sensitive to change was the voltage threshold for state change (V_p and V_n). For the cases where the model was set to match the HP Labs titanium oxide device [3] and the RRAM device (Figure 4.8) the threshold could only be altered by 1 to 2% before a 10% change in the I-V characteristic was observed.

Table 4.3. Data collected that shows how much each parameter in the model can be decreased for each device before a 10% change in the output I-V characteristic is observed.

	x_p and x_n	α_p and α_n	V_p and V_n	A_p and A_n	a_1 and a_2	b
Boise	61.54%	14.50%	65.53%	67.00%	90.00%	90.00%
Boise DC	50.45%	0.00%	92.57%	79.44%	90.00%	90.00%
HP TaOx	69.03%	0.00%	85.07%	63.81%	90.00%	90.00%
HP [3]	48.65%	75.55%	98.15%	88.40%	90.00%	96.81%
U of M	62.46%	0.00%	52.50%	79.08%	90.00%	95.15%
Iowa	71.30%	0.00%	91.41%	61.80%	90.00%	90.00%
RRAM	77.40%	0.00%	99.19%	86.35%	90.00%	90.00%

The data collected in this study is useful for modeling realistic physical arrays of memristor devices. The sensitivity to threshold voltage variation displayed in these results is important to consider, since threshold has been known to vary even within the same device over multiple voltage sweeps [28, 29]. The parameter pair α_p and α_n is most tolerant to variability, so memristor arrays may be able to tolerate more variation in the properties that impact the path of the state variable motion such as the electron mobility or the consistency of the crystallinity of the material.

Table 4.4. Data collected that shows how much each parameter in the model can be increased for each device before a 10% change in the output I-V characteristic is observed.

	x_p and x_n	α_p and α_n	V_p and V_n	A_p and A_n	a_1 and a_2	b
Boise	141.21%	211.50%	136.06%	148.10%	110.00%	110.00%
Boise DC	157.90%	278.50%	106.24%	127.50%	110.00%	110.00%
HP TaOx	190.00%	203.90%	110.58%	223.50%	110.00%	110.00%

HP [3]	152.00%	123.40%	101.93%	111.73%	110.00%	102.89%
U of M	133.50%	316.00%	130.75%	122.85%	110.00%	104.40%
Iowa	193.01%	180.50%	107.60%	201.00%	110.00%	110.00%
RRAM	110.85%	661.00%	100.85%	113.65%	110.00%	110.00%

CHAPTER V

MEMRISTOR BASED READ OUT INTEGRATED CIRCUITS

5.1 Introduction

It has been shown that memristors have the ability to be programmed with a range of resistance values between a set maximum and minimum resistance defined by the device [3, 4, 26]. Since the resistance of the memristor is proportional to the charge applied, it can be used to determine the number of photons that are present on a photodetector over a span of time. This chapter presents the circuit design and simulation results for a unit cell in a memristor based readout circuit. The memristor model used [8] has the ability to accurately fit several different memristor I-V characteristics [3, 4, 26, 27]. Since memristors have been published using a large variety of materials and structures, the simulations show how different memristor devices [4, 26] behave in the unit cell design.

This chapter is organized as follows: Section 4.2 describes the circuit design for the memristor based unit cell. Section 4.3 briefly describes the simulation setup, and subsections 4.3.1 through 4.3.3 describe each of the three simulations. Section 4.4 provides a discussion of the results.

5.2 Memristor-based ROIC Unit Cell Design

The circuit design for the memristor-based unit cell can be seen in Figure 5.1. This circuit has three modes of operation (integrate, read, and erase), and the equivalent circuit for each of them can be seen in Figure 5.2. It is assumed that each pixel in a photodetector array will have its own dedicated unit cell. During the integrate mode, all photodetectors are exposed and a current is generated in each unit cell proportional to the number of photons present on each corresponding detector. Each memristor will then have a variable resistance dependent on the amount of charge generated by each photodetector. For a single unit cell, this operation is described by the equivalent circuit in Figure 5.2 (a), where $R_{T2,\,ON}$ represents resistance of transistor T2 when it is switched on. Since the memristor model contains a programming threshold, the voltage drop across transistor T2, photodetector D1, and memristor M1 must be divided so that a voltage drop exceeding the programming threshold is applied to M1 for integration to occur. The voltage Vb represents the negative bias applied to the photodetector, and is the total voltage drop across D1, M1, and T2 during integration. For optimal circuit operation, a photodetector should be selected that has a low series resistance. Also, the transistor T2 should have minimal resistance when switched on.

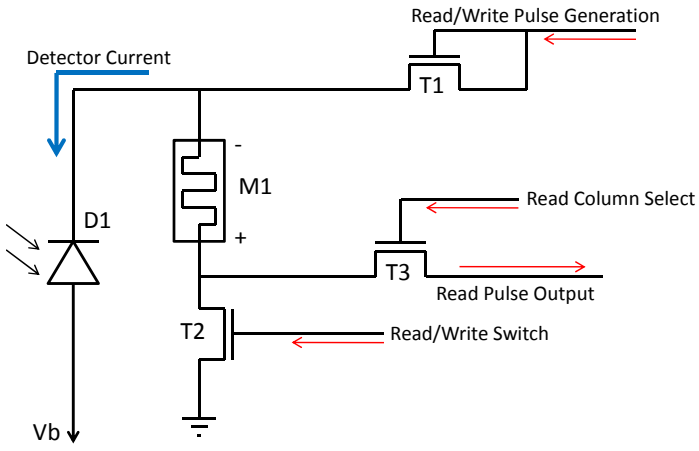


Figure 5.1. Circuit design for the memristor based ROIC unit cell.

After integration, the value of the memristor is read by applying a pulse to the memristor with known amplitude, and then reading the amplitude at the read pulse output. This will generally be done either sequentially by pixel or by row, as opposed to the integration which occurs at the same time for all pixels. The equivalent circuit for this mode of operation can be seen in Figure 5.2 (b). The output pulse will have variable amplitude based on a voltage division between the memristor and transistor T2. In this mode, the gate voltage on T2 is very low so that the resistance between the source and drain will be increased. Since there will be a larger voltage drop across T2, the voltage applied to the memristor will be below the programming threshold and the read pulse will not destroy the information contained in the memristor. This method for reading eliminates the need for two pulse generators where one would be for reading and other for erasing.

After the memristor in each unit cell is read, an erase pulse will be applied to all memristors simultaneously to reset the memristor back to a low resistance state. It can be seen in Figure 5.2 (c) that this is completed by using a pulse applied from T1. The erase time will be dependent on the pulse amplitude and the voltage drop across the memristor, so it is again important that the transistor T2 has a minimal resistance when switched on for the erase mode.

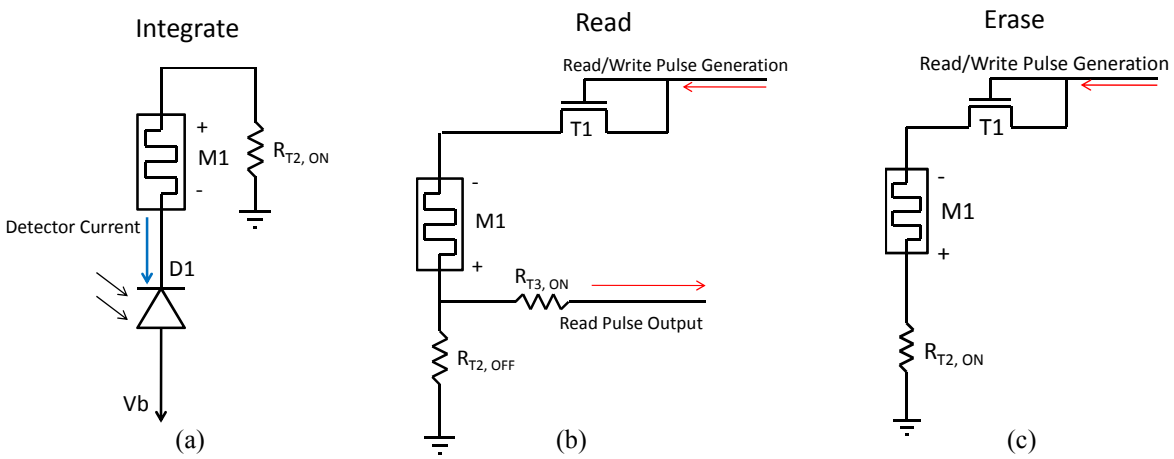


Figure 5.2. Equivalent circuits that describe three different modes of operations. The equivalent circuits represent (a) the integrate mode, (b) the read mode, and (c) the erase mode.

5.3 Unit Cell Simulations

The unit cell circuit described in the previous section was simulated using LTSpice. The memristor was modeled using a SPICE subcircuit based on the model in Section 2, and the photodetector was modeled based on the SFH2400 datasheet. The SFH2400 photodetector has a maximum current output of 10 μA and a sensitive area of 1 mm^2. The simulations were performed to show how the circuit reacts when different memristors were simulated and different detector currents were applied.

5.3.1 Simulation 1: low resistance memristor

The first unit cell simulation was completed with the memristor model matched to the characteriz the first 5 ms of the frame were used to perform integration on all of the cells in parallel, the next 20 ms were left to read each of the unit cells (either individually or by row), and the last 5 ms were dedicated to erasing all of the cells in parallel.

Figure 5.3 (a) shows the current generated by a single photodetector during the integration period (0 – 5 ms). The current had to be set at a large value (500 μA) so that the voltage drop across the photodetector was as small as possible. This caused a problem because the current generated by the photodetector model was greater than the limit of

the actual device according to the SFH2400 data sheet. This problem did not occur in the following simulation results (Sections 4.2 and 4.3) since a higher resistance memristor was used.

The next portion of the frame cycle was dedicated to reading each of the unit cells. It can be seen in Figure 5.3 (b) that a short read pulse was applied 5.5 ms into the simulation. The read pulse in these simulations was meant only to represent the read pulse applied to the simulated unit cell, and space was left where all of the other cells were to be read between 5 and 25 ms. Figure 5.3 (c) shows that when the unit cell was being read, the Read/Write Switch (transistor T2) was switched into a mode with a higher resistance by lowering the gate voltage to 1.2 V. This was done to lower the voltage drop across the memristor below the programming threshold voltage so that the value stored in the memristor would not be corrupted. Figure 5.3 (d) shows that the column select was switched on at 5.5 ms to signify a read from the column in which this particular unit cell resides. The output read pulse can be seen in Figure 5.3 (e) where the amplitude is determined by a voltage division between the memristor and the transistor T2.

The last step in the frame cycle was to reset the memristor to its high resistance state, which was done in the interval between 25 and 30 ms. Unlike a capacitor that can simply be discharged by shorting its terminals, the memristor undergoes physical state changes that must be reset back to the initial device structure. This was done by applying a long pulse (seen in Figure 5.3 (b)) to the memristor while the transistor T2 was set to have a low resistance. Figure 5.3 (f) shows the memristor state variable, which is related to device conductivity. It can be seen that the state variable increased during the integration period signifying a drop in resistance. The state variable remained constant

over the read period before dropping due to the erase pulse, which signified the return to a high resistivity state.

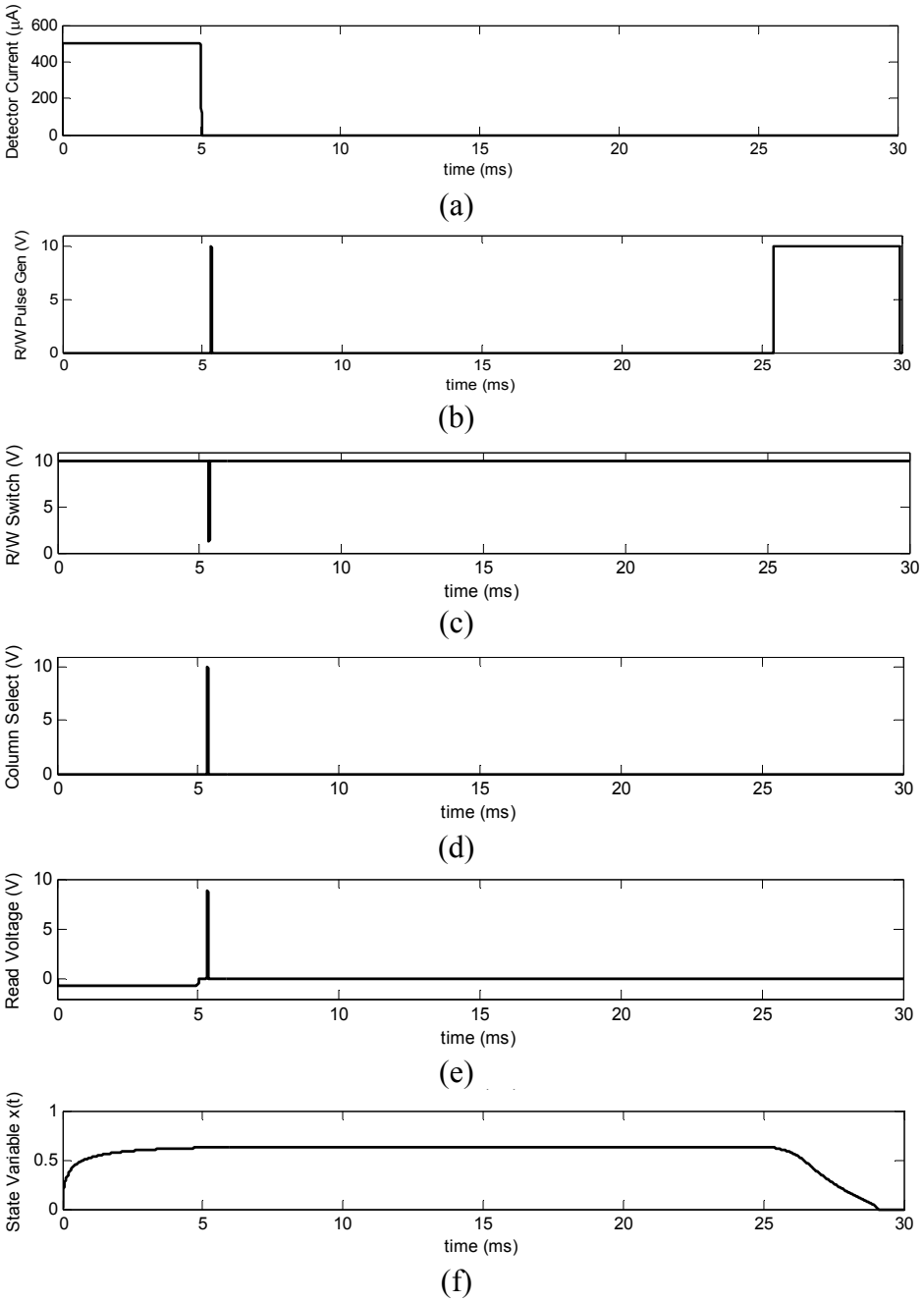


Figure 5.3. Simulation results for the memristor based unit cell (Vb = –5 V) with the memristor in [26]. The plots display the port inputs and outputs including (a) the current generated by the photodetector, (b) the pulses applied for reading and erasing, (c) the gate voltage supplied to the Read/Write Switch, (d) the gate voltage at the column select, and (e) the read pulse output, as well as (f) the value of the memristor state variable.

5.3.2 Simulation 2: high resistance memristor

This simulation was performed in Figure 5.4 with the memristor model matched to the device published in [4]. Since this memristor has a lower conductivity, the voltage drop across the memristor was greater in this simulation. The higher resistance led to a larger memristor voltage drop which is an advantage because the photodetector was able to perform at a lower current.

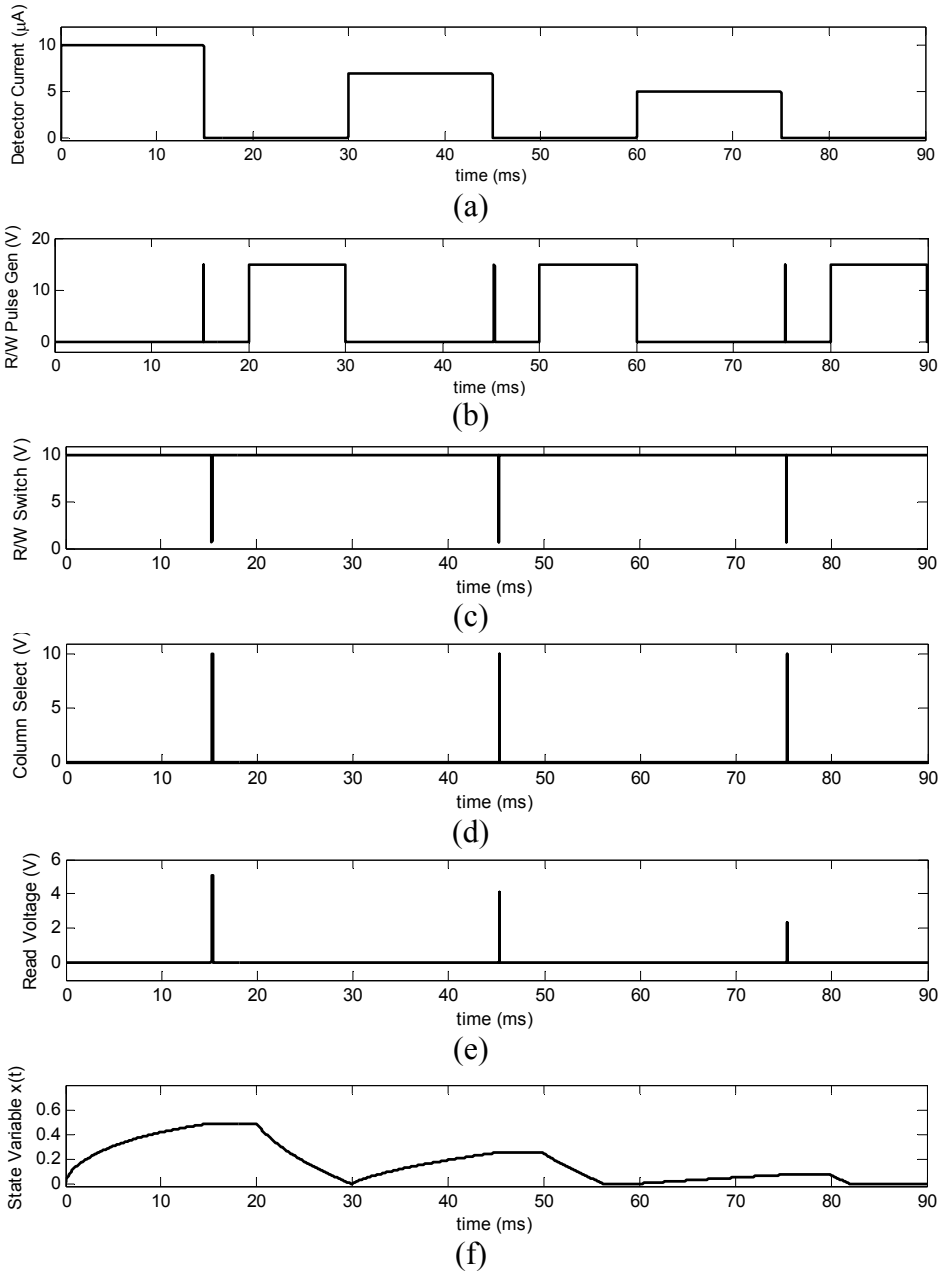


Figure 5.4. Simulation results for the memristor based unit cell using a higher resistance memristor [4] (Vb = –10 V). The plots display the port inputs and outputs including (a) the current generated by the photodetector, (b) the pulses applied for reading and erasing, (c) the gate voltage supplied to the Read/Write Switch, (d) the gate voltage at the column select, and (e) the read pulse output, as well as (f) the value of the memristor state variable.

The programming threshold of this memristor was much higher than the threshold of the memristor in the previous simulation, so this led to the necessity of larger pulse voltages. Also, this memristor had such a high resistance in its off state that the current leaked through transistor T1 during the integration period, reducing the amount charge applied to the memristor. This was fixed by placing a resistor between transistor T1 and the Read/Write Pulse Generator. This resistor also limited the current coming out of the pulse generator, thus lengthening the erase time, so the resistance value of this resistor had to be optimized. A value of 900 kΩ was chosen.

In this simulation, the integration and erase times were increased because of the slower state variable motion present in this memristor model. The integration time was increased to 15 ms and the erase time was set to 10 ms. The remaining 5 ms were used to read each of the unit cells. Depending on the array size, the pixels may have to be read several at a time due to the short read interval for each frame.

Figure 5.4 (a) shows that the photodetector current in this simulation is within the range of what can be produced by the SFH2400 photodetector according to the datasheet. This simulation demonstrates how a single unit cell would operate over three complete frame cycles. In each cycle, the detector current was reduced to show how the state variable changes relative to the amount of charge applied to the memristor.

Figure 5.4 (b) shows the pulses that were used to read and erase the memristor. It can be seen that in each cycle (Figure 5.4 (c)), the Read/Write Switch has a low gate voltage (0.6 V) during the read and high gate voltage during the erase. This is again so

that a single amplitude pulse can be used to change the memristor state (when $V_{G,T2}$ = 10 V), or perform a non-destructive read (when $V_{G,T2}$ = 0.6 V). The gate voltage of T2 for reading is lower in this simulation so that the resistance of the transistor T2 is closer to the average resistance on the memristor in [4].

The column select was performed as expected in Figure 5.4 (d) where the transistor T3 was on for each read, and the output read pulses can be seen in Figure 5.4 (e). Since there is more than one cycle in this simulation, the read voltage amplitudes can be compared where the difference is dependent on the resistance of the memristor during each read. It can be seen in Figure 5.4 (f) that a difference in the memristor state variable from 0.1 to 0.5 corresponds to a dynamic range of almost 2 V when comparing the read pulses in Figure 5.4 (e).

5.3.3 Simulation 3: high resistance memristor in low light environment

The following simulation shows how the circuit would react in a low light environment. Since less light is assumed to be present at the detectors, the frame rate was lowered to 10 fps so the integration time could be lengthened. The current generated by the photodetector was set to be applied for 80 ms (Figure 5.5 (a)), and a read pulse was applied right before and slightly after the integration period (Figure 5.5 (b)). Although the motion of the state variable is small (from 0 to 2%), the resistance change is quite large. This is because the resistance drops very significantly when initially pushing the memristor device out of its maximum resistance state. This can be seen in the I-V curves in Figures 5.2 and 5.3, as the bottom hysteresis loop is significantly larger than the proceeding ones (which is also the case in the characterization data [4, 26]).

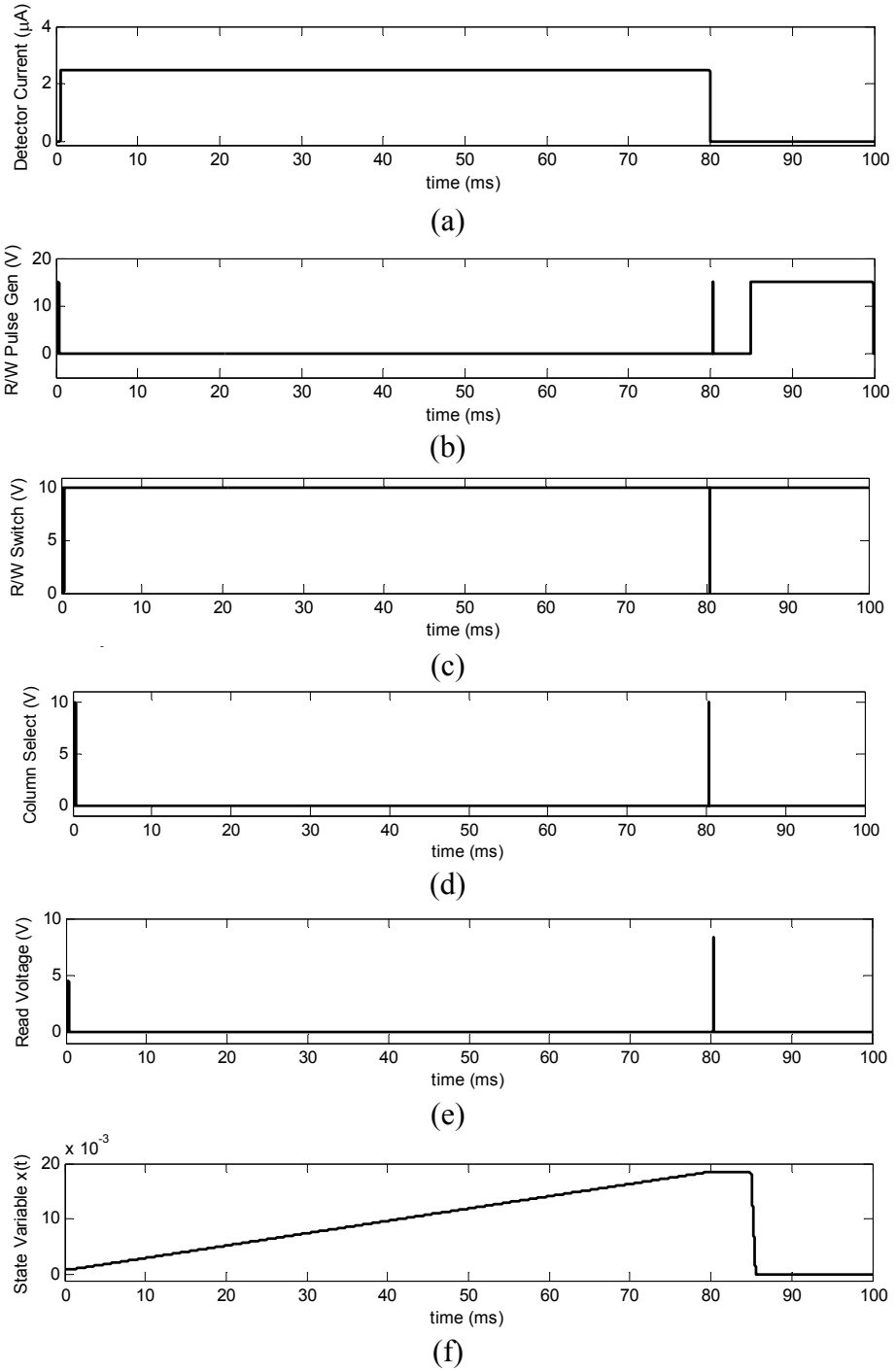


Figure 5.5. Simulation results for the memristor based unit cell for a low light operation when modeling the memristor in [4] (Vb = –10 V). The plots display the port inputs and outputs including (a) the current generated by the photodetector, (b) the pulses applied for reading and erasing, (c) the gate voltage supplied to the Read/Write Switch, (d) the gate voltage at the column select, and (e) the read pulse output, as well as (f) the value of the memristor state variable.

It can be seen in Figure 5.5 (c) that the gate voltage of transistor T2 was lowered when the read pulses were applied by a larger amount ($V_{G,T2}$ = 0.1 V) when compared to the previous simulations. This helps to increase the dynamic range of the read pulse voltages, as the higher transistor resistance will correspond more closely to a resistance range closer to the maximum resistance or the memristor device. A reduction of frame rate and a slight change in the control signal may lead to the option of a low light operation mode for the detection system. Figure 5.5 (e) shows that this small change in state produces a significant voltage difference of about 4 V. The state variable motion can be seen in Figure 5.5 (f). This simulation shows that this memristor circuit may be advantageous for use in low light situations, although more physical memristor characterizations should be performed to ensure the device is stable with such a small change in state variable.

5.4 Discussion

When comparing the unit cell circuit operation for each of the memristors, it can be seen that it was more beneficial to have a memristor with a larger overall resistance. This is contrary to existing unit cell designs where low input impedance is desired for maximum signal transfer between the photodetector and the readout circuit [39]. The high resistance was a requirement because sufficient voltage needed to be applied to the memristor to change the state of the device. This may also be solved by using a photodetector with a lower series resistance.

In both cases, the circuits required large pulse voltages. When considering the low resistance device, this was due to the fact that the memristors have a programming threshold, and that the transistors in the simulations had a larger on state resistance (from

source to drain) than the minimum resistance of the memristors. When considering the high resistance device, high voltage pulses were needed to supply sufficient energy to the device since the state variable was more resistant to change and had a higher programming threshold.

When considering the total read time available for the 30 fps simulations to be either 5 or 20 ms, the total time per pixel can be easily determined (5 and 20 ms were chosen based on the simulation results in subsections 4.1 and 4.2). Assuming the detector array size is 640 by 480, the read time per pixel would be about 65 ns for a 20 ms total read time, but only 16 ns for a 5 ms total read time. Assuming that the high speed readout circuits published have a pixel read speed of about 20 ns [106], multiple pixels may need to be read at the same time when using 5 ms as the total read time.

A comparison of the area requirement for several unit cells, including the proposed design, can be seen in Table 5.1. When comparing the total area requirement of this circuit to a voltage mode circuit [38], this circuit has a significantly smaller area due to the absence of the large integrating capacitor. Additionally, the circuit in [38] contains two operational amplifiers (op amps), which also require a significant amount of area. The circuit in [40] shows that the integrating capacitor can be removed, but additional circuitry was required at each column in addition to the 3 transistors required at each pixel. Lastly, the circuit in [41] uses a counter and a frequency generator to count pulses relating to the number of photons present at a detector. This technique circuit requires a large amount of CMOS circuitry including an op amp and a CMOS counter. The unit cell design presented in this paper only requires three transistors and one memristor. When considering the small area requirement when using the memristor based approach, it may

be a promising option for developing single chip photodetector and readout circuits with a large photodetector fill factor.

Also, when looking at the simulation results, a logarithmic response can be seen in the state variable integration as it approaches the low resistance boundary. Some existing circuits use a logarithmic response during integration to increase the integration time and dynamic range [43]. Based on the simulation results, this effect may also be achievable using the memristor based circuit discussed in this paper.

Table 5.1. Chip area requirement of the circuit presented in this paper compared to existing alternatives.

This Circuit	Voltage Mode [38]	Current Mode [40]	Frequency Counter [41]
1 Memristor 3 Transistors	2 Op Amps 2 Capacitors 3 Transistors	3 Transistors Additional read circuitry at each column	1 Op Amp 1 Counter 1 Transistor

CHAPTER VI

MEMRISTOR MEMORY ARCHITECTURE

6.1 Introduction

This chapter presents a memristor based memory system that is capable of achieving more than 3 times the density of a typical STT-MRAM array. Additionally, it has dramatically reduced power consumption when compared to a high-density transistor-less memristor crossbars.

The analysis of this memory design was performed through detailed SPICE simulations. To model the memristors, a previously published device model [7-10] was utilized that is capable of reproducing memristor characteristics very accurately. Two different memristor devices were studied for use in the proposed memory system that differ in their resistance range and switching time. Both the wire resistance and the isolating transistors in the array were simulated to provide a more complete crossbar analysis. This allowed for a very precise platform to determine energy consumption and when alternate current paths lead to read errors. As opposed to the related research, this work utilized SPICE as much as possible to increase the accuracy of the results.

Our SPICE analysis shows that both 4×4 and 8×8 memristor crossbars can handle a large number of write/read cycles with zero error. In 16×16 crossbar arrays, read errors start to become prevalent, thus making them unsuitable for memory applications. The work proposed in this chapter suggests breaking a larger memristor grid into smaller tiles

of 4×4 or 8×8 memristors surrounded by isolating transistors. These smaller arrays with transistors are then stitched together to make a large memory array where alternate current paths are not a significant problem. This will reduce the power consumption and improve the noise margin of the system, while still enabling much higher memory densities. Only a few memristor crossbar simulations [58, 59, 107-109] account for wire resistance, and these were completed with less accurate device models.

Resistive memories have generally been studied as replacements for level two and lower level caches because of their low leakage energies. Their longer write latencies compared to SRAM caches have generally prevented them from being used in level one caches. Sun et al. [48] propose a modified L1 replacement policy and use of write buffers to compensate for long write latency of STT-MRAM caches.

In this chapter we evaluate the use of the proposed memristor based circuits as level two caches. Memristor devices typically have short read times (comparable to SRAM), but have longer write latencies compared to STT-MRAM. As level two cache access is not very frequent, the use of write buffer reduce long write latency penalties of memristor based caches.

6.2 Scaling Memristor Arrays

6.2.1 Crossbar Array Designs

Many studies [2, 4, 103, 107] have proposed large memristor crossbar arrays as a high density memory design. The layout and circuit diagram typically proposed for this type of memory is seen in Figures 6.1 (a) and (b). In this design each nanoscale memory element will consume an area of just $4F^2$ [108] where F is the feature size of the

fabrication technique. In contrast, a typical STT-MRAM memory array has a transistor present with each memory element which increases the cell size to $40F^2$ [110].

The schematic in Figure 6.1 (b) illustrates a potential problem with high density memristor crossbars. To read or write memory element $m_{1,1}$ voltages are applied to the wires a_1 and b_1. As there are no access transistors in this design, nothing is stopping current from flowing through other memristor devices. This can lead to potential read errors in a large crossbar because the current sensed at b_1 could be due to a chain of devices in a low resistance state when the selected memristor is in a high resistance state. The high R_{OFF}/R_{ON} in memristor devices help alleviate the impact of alternate current paths. The high R_{OFF}/R_{ON} enables a much larger possible voltage range to be observed at the output of the crossbar.

A common solution to eliminate alternate current paths in a resistive memory system is to place an access transistor alongside each memory element (see Figure 6.2). This technique is known as a 1T1R memory system. It is essential when the R_{OFF}/R_{ON} of the memory cell is low, such as with STT-MRAM devices (where $R_{OFF}/R_{ON} \approx 2.5$). It greatly reduces the chance of a read error and limits the power consumption since greater control is placed on the path of the current flow. The disadvantage of this type memory system is that the areal density of the system is now limited by the area of the transistors and not the area of the memory devices.

In the following subsections we show that in spite of the high $R_{OFF}/R_{ON,}$ there is a limit to how large the transistors-less crossbar array can be scaled before read errors occur. We use accurate SPICE models [7-10] of memristors and modeled wire resistance in the crossbars to capture detailed behaviors of alternate current paths. We simulated

multiple read and write cycles for memristor crossbars of varying sizes. Our results indicate that crossbars of 16×16 memristors have errors when reading the stored data. Our results also show that alternate current paths lead to an increase in the total current drawn within larger crossbars, thus increasing the power consumption.

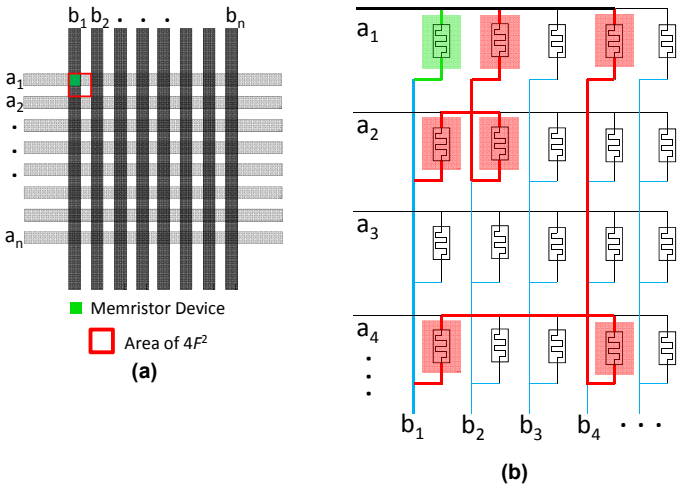


Figure 6.1. Memristor crossbar diagrams displaying (a) the layout and (b) the schematic for a high density unconstrained crossbar. The schematic (b) displays the target memristor path (green) as well as two possible alternate current paths (red).

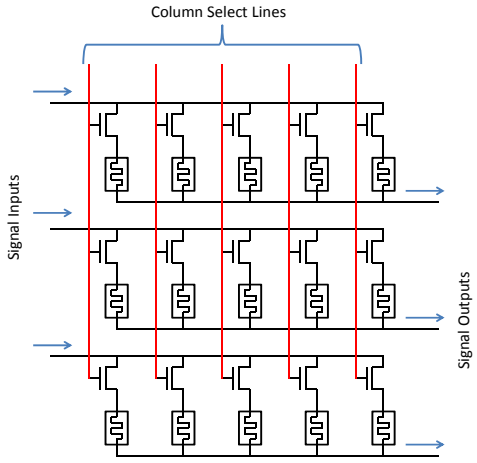


Figure 6.2. Schematic for one possible 1T1R memristor memory array.

6.2.2 Read and Write Operations

In a memristor crossbar, a write operation is a two-step process [111] that writes to an entire row of memristors. The voltage across a memristor typically needs to surpass

a threshold voltage in order to change the device resistance. This write method works assuming that the write voltage V_w is greater than the memristor threshold voltage, and $V_w/2$ is less than the threshold voltage.

Figure 6.3 demonstrates the write operation in a 4×4 crossbar. The first step is to apply a voltage of $V_w/2$ to a selected row (grounding the other 3), and to apply a voltage of $-V_w/2$ to all columns where a 1 (low resistance state) should be written (see Figure 6.3 (a)). Furthermore, a voltage of $V_w/2$ should be applied to all global column wires where a 0 (high resistance state) should be written. This will result in writing a 1 to only the memristors in the selected row that need to be set to 1. This provides a voltage drop of about V_w across all memristors to be written, and a maximum absolute voltage drop of about $V_w/2$ across all others. A small amount of voltage will be applied to surrounding wire resistance.

During the second step in the write process, a voltage of $-V_w/2$ is applied to the selected row, and all column wires are set as they were in step one. This will result in writing a 0 to the rest of the memristors in the row.

A parallel read operation is performed by setting a selected row of memristors to a voltage below the switching threshold, and sensing the voltage at each crossbar column output. This voltage will then be applied to a comparator that will convert the analog voltage level to a binary value.

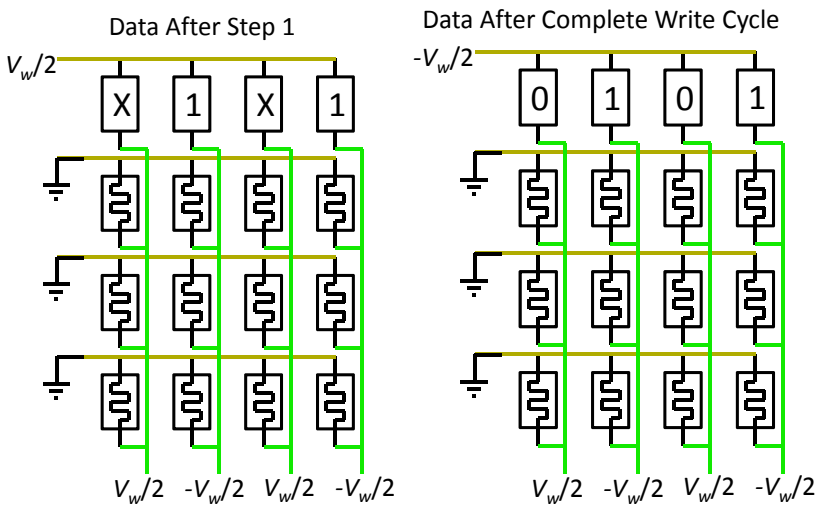


Figure 6.3. Demonstration of the write operation in a crossbar tile.

6.2.3 Memristor Device Model

To perform an accurate device level analysis of a memristor crossbar memory system, a SPICE equivalent of the memristor model first proposed in [8] was utilized. This model was set to match the characterization data of one of the memristor devices published in [103] (see Figure 6.4). This device was chosen for use in the proposed memory design because it had a large R_{OFF}/R_{ON} ratio (10^6) while still retaining a relatively low switching time (about 10 ns). It also had a large on state resistance of about 125kΩ (determined by the 8μA current from a 1V read pulse).

The simulation result in Figure 6.4 shows the minimum and maximum resistances of the model to be 124.95kΩ and 125.79×10^9Ω respectively, which correlates very closely to the characterization [103]. Applying a +7V pulse successfully switches the device into a low resistance state, and applying a -7V pulse drives the model into a high resistance state.

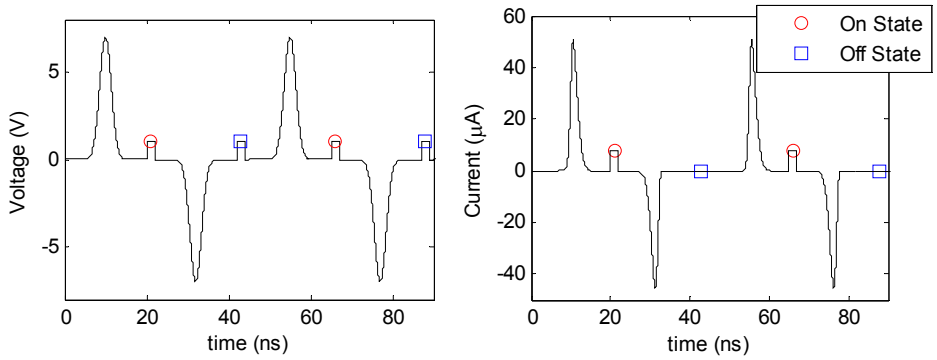


Figure 6.4. Simulation results displaying the input voltage and current waveforms for the memristor model [15] that was based on the device in [103]. The following parameter values were used in the model to obtain this result: V_p=4V, V_n=4V, A_p=816000, A_n=816000, x_p=0.985, x_n=0.985, α_p=0.1, α_n=0.1, a_1=1.6×10^{-4}, a_2=1.6×10^{-4}, b=0.05, x_0=0.01.

6.2.4 SPICE Circuit Simulation

The noise margin for memory operations within a memristor crossbar was determined using LTSpice to simulate a large number of read/write operations. Crossbars of size 4×4, 8×8 and 16×16 were simulated. The schematic for the 4×4 crossbar can be seen in Figure 6.5. Both read and write signals are applied from the voltage sources V_{R1} through V_{R4}. The circuit has two possible output signal paths controlled by the access transistors at each column. When a read is performed, the signal is measured across the sense resistor R_S to determine the output voltage. When a write is performed, the read enable is turned off and the column voltage sources V_{C1} through V_{C4} are used complete the write process described in Figure 6.3.

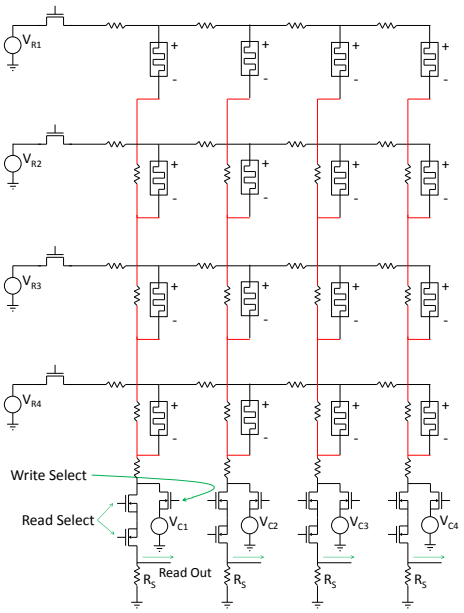


Figure 6.5. Crossbar circuit used to carry out noise margin and energy experiments.

A text file with up to 400 random input data sequences for the crossbar was generated, with each sequence targeted to a randomly selected row within the crossbar. In each write cycle, one sequence from the text file was written to the entire targeted row in the crossbar. After each write cycle, every row within the crossbar was read and the read output voltages were recorded. Upon completing the simulation the read output voltage signals were compared to the write data to determine if any read errors had occurred. If no read errors were present, the noise margin was calculated. The noise margin is defined as the voltage range between the largest voltage required to represent a 0 (V_{0H}), and the smallest voltage required to represent a 1 (V_{1L}), as shown in Figure 6.6. This experiment was repeated for different values of R_S until the noise margin was maximized.

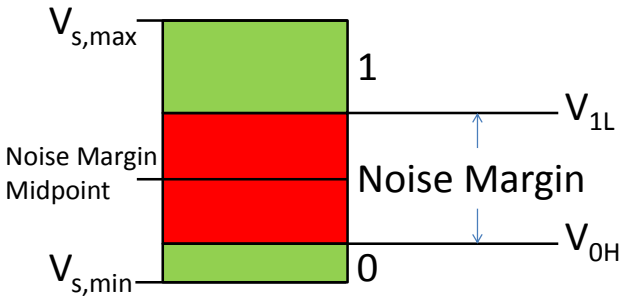


Figure 6.6. Diagram displaying the noise margin present in a memristor crossbar.

To determine the write energy, an experiment similar to that used to determine noise margin was also completed in SPICE. In this case the write sequences were applied with no read cycles. The power dissipated in each device in the circuit was integrated over the simulation time to determine the total energy consumption in the crossbar. The total energy consumption in the circuit was then divided by the number of writes to determine the average write energy per bit.

To determine the read energy, the half of the memristors in the SPICE circuit were initialized to their high state and the other half were initialized to the low state. Each row in the memristor crossbar was read and the total power in the circuit was integrated and the divided by the total number of memristors in the circuit to determine the average read energy of a single bit.

6.2.5 Crossbar Error Analysis

This section demonstrates the drawbacks of using a single large high density memristor crossbar. The crossbar experiments described in the previous section were used to determine how large of a crossbar could be simulated before read errors occur (using the memristor model in Figure 6.4). Crossbars with dimensions of 4×4 and 8×8 were simulated successfully producing zero errors. The noise margins determined for

these crossbars were 418mV and 74mV respectively. The simulations of the 16×16 crossbars showed read errors, due to a larger number of alternate current paths.

A physical memristor array was developed [112] (using a device with properties similar to the one we modeled) that shows successful data storage and reading in an 8×8 crossbar. This correlates closely to our result, as we simulated 8×8 crossbars without obtaining read errors, while the 16×16 generated read errors within a few cycles.

The result presented in Figure 6.7 shows the errors that were obtained when attempting to read and write to a 16×16 crossbar array for 100 write/read cycles. Each write operation writes to all the memristors within a single row of the array. Within each read operation, all 16 rows of memristors are read to guarantee that every error present in the system is accounted for.

Figure 6.7 (a) shows how many errors are present in the crossbar depending on the voltage level selected for the noise margin midpoint. A much larger voltage range tends to be required to represent a 1, so a noise margin midpoint of $(V_{S,max}-V_{S,min})/2$ is not usually optimal for reducing errors. Figure 6.7 (a) shows that the optimal noise margin midpoint for this simulation occurs at about 0.1V for this simulation of the 16×16 array. The optimal noise margin midpoint was about 0.4V in the 4×4 crossbar and about 0.22V for the 8×8 crossbar.

Figures 6.7 (b) and (c) show the error associated with the 16×16 crossbar when using the noise margin midpoint that produces the minimum amount of errors. Figure 6.7 (b) shows that 5 cycles were successfully written before a single error occurred. Errors increase more quickly as the number of cycles is increased. This is because the chances

of alternate current paths within the resistive grid becomes more prominent as more memristors are switched to a low resistance state.

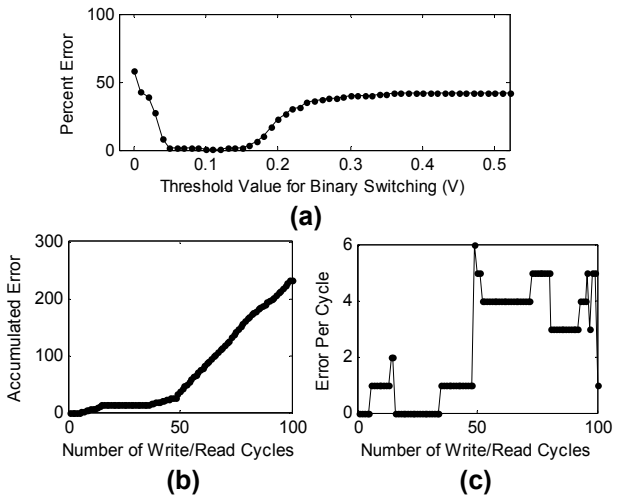


Figure 6.7. Error analysis for a 16×16 crossbar simulation carried out in SPICE showing (a) the optimal point to sense the threshold between 0 and 1, (b) the total number of errors accumulated over 100 write/read cycles, and (c) the number of errors present at each write/read cycle.

Figure 6.7 (c) displays how many errors were present in each read cycle for the 16×16 array. There are several areas in this plot where the number of errors remains constant. This is because some data pattern in the memristors led to a read error. That read error remains until a future write changes the resistance of memristor(s) that are producing a critical alternate current path as the memristor resistance values are being read.

Additionally our simulation results show how the energy consumption of a high density crossbar increases with crossbar size in Figure 6.8. The value plotted is the energy required to write a single bit. The three data points collected are based on the write energy per bit required in a 4×4, 8×8, and 16×16 crossbar. If assumed to be linear, the data can be extrapolated to show that large crossbars quickly reach a level of energy consumption that that would be unrealistic for a competitive memory technology.

As a possible solution to this problem, one publication [61] presents a memristor

device with current suppression in one direction. This reduces the problem of alternate current paths, although the switching time of this device is too large for an on-chip memory application (100µs).

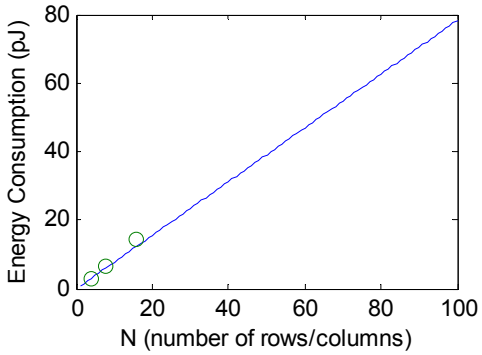


Figure 6.8: Energy required to switch a single bit as crossbar size increases. This data is based on two write cycles per row (one for writing 1s and one for writing 0s as shown in Figure 6.3).

6.3 Proposed Hybrid Crossbar Memory

6.3.1 Circuit Design

Based on the results in the previous section, high density memristor crossbars start to become problematic at relatively small sizes. As a solution to this problem, a large memory array composed of many smaller memristor crossbar arrays is proposed in this chapter. Figure 6.9 displays a portion of the circuit design for the hybrid memory system. In this design, transistors are used to isolate small crossbars within a larger array. These smaller memristor arrays will be referred to as tiles. The isolating transistors surrounding each tile limit alternate current paths to the group of memristors within a tile and prevents larger alternate paths from forming between tiles.

In this example, 4 memristor tiles from two rows of a larger memory array are displayed, with each tile consisting of 16 memristors arranged into a 4×4 crossbar.

The memory is accessed by activating all the tiles within a row simultaneously. Within all of these activated tiles, the same row of memristors is either read or written to in a

specific cycle. The row decoder in Figure 6.9 containing the row select signals (S_1 through S_N) is designed to turn on only one row of tiles during a parallel read or write operation.

The top of the circuit displays a pulse generator block, which is responsible for sending data to a single row of memristors in each activated tile (through either D_{R1}, D_{R2}, D_{R3}, or D_{R4}). The rest of the rows of memristors within the active tiles are grounded. During a read or write operation, data from the selected row of tiles will be processed by the column circuits. For a read operation, the analog output voltage across the sense resistor R_S is converted to a binary signal using a comparator and the constant threshold resistance R_T. For a write, the path to the sense resistor is shut off, and the column pulse generator is used to carry out the write process described in Figure 6.3.

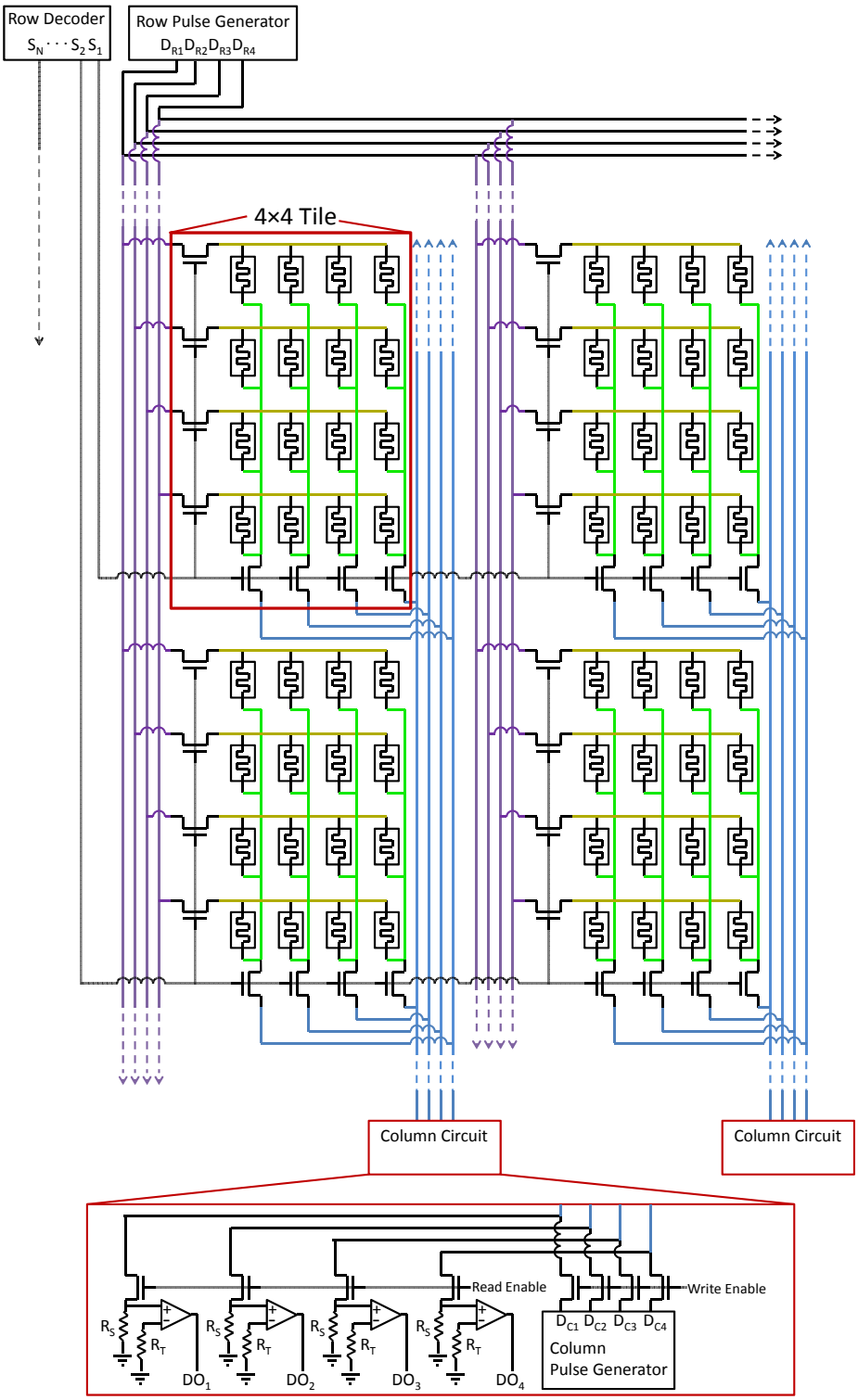


Figure 6.9: Circuit diagram for the tiled crossbar memory system.

6.3.2 Area Analysis

Each tile has transistors along two edges – one for each row access wire and

column access wire. Thus 4×4 tiles require 8 transistors to drive the 16 memristors within (an average of 2 memristors per transistor). An 8×8 tile would require 16 transistors to drive the 64 memristors within (an average of 4 memristors per transistor). This allows the memristive memories to have multiple bits stored per transistor.

In these designs the size of the transistors is still the limiting factor in memory density. However, the 4×4 and 8×8 tile systems provide an increase in density over a 1T1R system by a factor of 2 and 4 respectively. The layouts for the 4×4 and 8×8 tiles can be seen in Figure 6.10, where the red squares represent memristors.

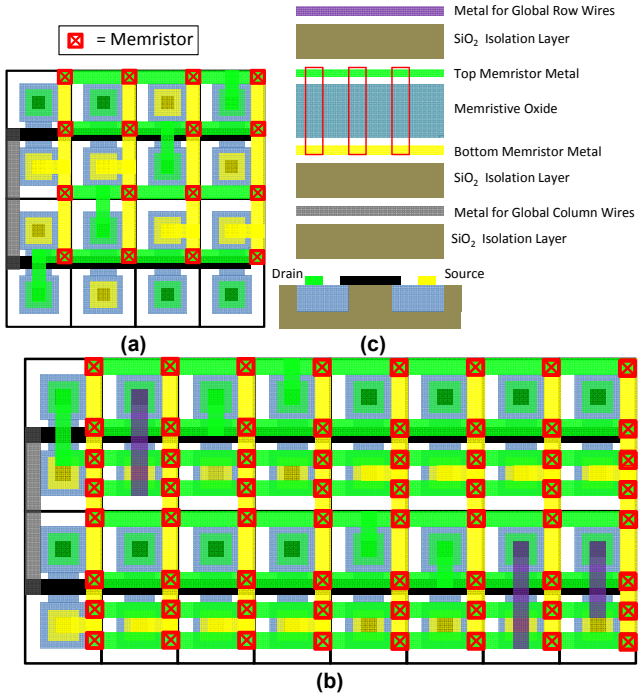


Figure 6.10: Layout for (a) the 4×4 tile, (b) the 8×8 tile, and (c) the layer organization for each of the tiles.

6.3.3 Transistor Sizing

To keep the density of the memristor tiles high, the access transistors must be made as small as possible. Using memristors in the proposed memory system with very low values for R_{ON} will increase the drain current in the access transistors. This will cause the transistors to become very large. Using memristor devices with the high R_{ON} value

such as the one modeled in Figure 6.4 (R_{ON}≈125kΩ) will help minimize the worst case current requirement of the crossbar. According to the crossbar simulations in section 3.4, the absolute maximum current spike through a transistor in the 4×4 tile was about 175μA (and about 225μA in the 8×8 tile).

Equation (6.1) is used to determine the minimum possible width of the transistors based on the drain currents required for the crossbars. A similar method for determining transistor size was presented in [111].

$$I_D = \frac{\mu_n C_{ox}}{2}\frac{W}{L}(V_{GS} - V_{th})^2(1 + \lambda(V_{DS} - V_{DSsat})) \quad (6.1)$$

Using a value of 225μA for I_D, equation (6.1) was solved for W to determine the minimum transistor size. It was determined that a transistor area of $50F^2$ was required for this system.

These results also correlate to the general trend for transistor sizing in STT-MRAM [45,47], where a $40F^2$ device is required for driving a 196μA current [45].

Table 6.1 shows the minimum value for R_{ON} that is required so that the bit density of the system can exceed either the bit density of SRAM or STT-MRAM. To complete this study, an approximated equation to determine the maximum drain current can be seen in equation (6.2), where V_w is the write voltage in the system and M is equal to the number of columns in the crossbar. Equation (6.2) shows that a higher drain current is required when using a device with a lower value for R_{ON}. Additionally the maximum drain current in each transistor increases as more columns are added to the crossbar. From this analysis it was determined that many memristor devices [30, 26, 113-115] have a value for R_{ON} that is too low for the proposed memory system.

$$I_D = M\frac{V_w}{R_{ON}} \tag{6.2}$$

Table 6.1. Minimum device resistance values required to that the drive current of the access transistors in the system will cause a decrease in bit density beyond either SRAM or STT-MRAM.

Tile Size	Minimum R_{ON} to exceed SRAM bit density ($146F^2$)	Minimum R_{ON} to beat STT-MRAM bit density ($40F^2$)
4×4	8.1kΩ	37kΩ
8×8	7.8kΩ	32kΩ

6.3.4 Memristor Devices Used in Simulation

In addition to the memristor device [103] modeled in Figure 6.4, a second device was modeled which was first developed at Arizona State University [116]. The characterization result for this model can be seen in Figure 6.11. Compared to the first device [103], the switching characteristic of the second device showed a lower write voltage (1V instead of about 7V), a higher device resistance, but a longer write time (50ns instead of 10ns). From this point onward, the device developed at the University of Michigan will be referred to as device X, and the device developed at Arizona State University will be referred to as device Y. Table 6.2 compares the key characteristics inherent in each of these devices.

When simulating the crossbar tiles with device Y in SPICE, the 4×4 crossbar exhibited 0 errors with a noise margin of 32mV. However, the 8×8 crossbar simulation based on device Y produced errors similarly to the 16×16 crossbar based on device X. This is most likely due to the low threshold voltage requirement of device Y. This helps keep the write voltage low, but limits the read voltage to about 0.5V, as opposed to the 3V read voltage capable in device X.

Table 6.2. Devices used for the simulations in this chapter.

Device	R_{ON} (Ω)	R_{OFF} (Ω)	R_{OFF}/R_{ON}	Threshold Voltage (V)	Switching Time (ns)
X: U of M	125k	125G	10^6	4	10
Y: ASU	500k	1G	2000	0.6	50

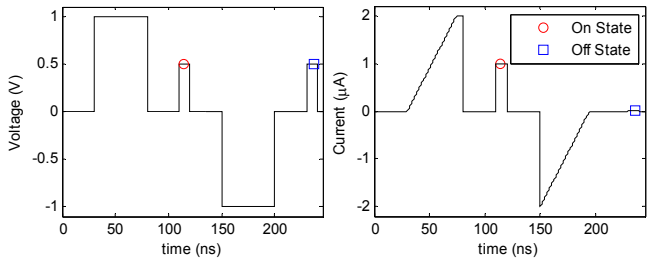


Figure 6.11. Simulation results displaying the input voltage and current waveforms when modeling the device presented in [116]. The following parameter values were used in the model [7-10] to obtain this result: V_p=0.6V, V_n=0.6V, A_p=2.5*10^7, A_n=2.5*10^7, x_p=0.97, x_n=0.97, α_p=0.9, α_n=0.9, a_1=4×10^{-5}, a_2=4×10^{-5}, b=0.05, x_0=0.001.

6.3.5 Crossbar Memory Analysis

A large number of simulations were completed to determine the optimal sense resistance and write voltage that would maximize the noise margin (see Table 6.3) for each of the crossbar designs. These include the 4×4 and 8×8 crossbar tiles based on device X, and the 4×4 tile based on device Y. When comparing devices X and Y, a significant drop in switching energy is observed. This is because device Y operates at a lower current and write voltage, so even though switching time is increased, energy is still reduced. The read energy shows a similar trend because the read voltage must be lowered to 0.5V so that data stored in the device is not disrupted by the read pulse.

Table 6.3. Comparison of the 4×4 and 8×8 tiles.

Measurement	***4×4 (Dev. X)***	***8×8 (Dev. X)***	***4×4 (Dev. Y)***
Write/Erase Voltage (V)	7	7.5	1
Read Voltage (V)	3	3	0.5
Write Energy/Bit (fJ)	3265	6536	117
Read Energy/Bit (fJ)	4.953	6.180	0.0441
Opt. Sense Resistance (Ω)	1M	75k	380k
Max. Noise Margin (mV)	418	74	32
Wire Resistance (Ω)	5	5	5
Largest Current Spike (μA)	175	225	7

Table 6.4 compares the proposed memory array designs to other existing memory architectures. The values for bit density were calculated assuming 45nm technology. The 4×4 and 8×8 tiled systems consume less than 5% and 10% of the energy consumed in the unconstrained 1kB crossbar respectively. Furthermore, the read energy in the tiled

systems is less than 1/3 of the read energy in the unconstrained crossbar.

The proposed architecture has a higher density when compared to SRAM, although it has a longer write time. Also, SRAM has a large amount of leakage energy [117] that is not present in the memristor or STT-MRAM based systems. This is because these devices do not require power to retain their memory state, and the transistors in the crossbar tiles are only active when a read or write pulse is present.

The SRAM leakage current was assumed to be 29μA/Mb with a 1V operating voltage [117]. To obtain leakage energy of a single-bit access, it was assumed that an average of 1000 accesses per bit would be performed in one second. It should be noted that the leakage energy will be consumed by all memory cells within an SRAM array even if they are not being accessed, so this will lead to larger total energy consumption. The read energy in the STT-MRAM system was found to be larger than that of the memristor systems. It is assumed that STT-MRAM cells have a lower resistance range, and thus draw a larger current during a read.

Table 6.4. Performance comparison of different memory cell designs.

Memory Architecture	***SRAM Active***	***SRAM Leakage***	***STT-MRAM***	***Dev X (4×4)***	***Dev X (8×8)***	***Dev Y (4×4)***	***1kB Crossbar (Based on Dev X)***
Bit Density ($Gbits/cm^2$)	0.338		1.23	1.98	3.95	1.98	12.35
Read Energy (fJ/bit)	0.7	27.7	60.4	4.593	6.180	0.0441	21
Write Energy (fJ/bit)	0.7	27.7	1177	3265	6536	117	70000
Read Time (ns)	0.3		0.3	0.3	0.3	0.3	0.3
Write Time (ns)	0.3		0.57	10 (20/row)	10 (20/row)	50 (100/row)	10 (20/row)

CHAPTER VII

NEUROMORPHIC MEMRISTOR CIRCUITS

7.1 Introduction

To correctly study the capabilities and behaviors of these circuits, it is necessary to carry out detailed SPICE level simulations where alternate current paths are accounted for. This chapter presents the first study examining the pattern recognition capability of memristor crossbars at the SPICE level, taking alternate current paths into account. The study presents a circuit level training approach for memristor crossbars and examines the scalability of the crossbars. It also compares our low power approach to a virtual-ground circuit technique [84] that eliminates alternate current paths, but significantly increases power consumption. To the best of our knowledge, no other study has been completed where a learning algorithm is applied to a memristor-based crossbar neural circuit in SPICE.

The first set of contributions in this chapter relate to showing that a dense memristor crossbar can be trained with neural network algorithms despite the sneak paths inherent in the circuits to carry out pattern recognition tasks. We develop novel circuit level techniques to pulse the memristors within the crossbar to update their resistances through a neural network training algorithm. This circuit mimics a single layer neural network. Two such circuits are then coupled together to mimic a two layer neural network that is capable of learning non-linearly separable functions. We show that using

the Concurrent Learning Algorithm (CLA) [118], a supervised learning algorithm, that the training circuit would have a minimal hardware footprint. Our detailed SPICE level simulations show that the two serially connected memristor crossbars are capable of learning all possible 16 2-input logic functions and also demonstrate correct classification of a reduced set of MNIST images.

The second set of contributions in this chapter examine the scalability and power consumption of our crossbar circuits, and compare them to existing approaches presented in the literature. Analysis is presented that shows how large the crossbar circuit may become before wire resistances and capacitances have an impact on the circuit. The driver power constraints on the scalability of the crossbar have also been examined.

We have quantified the impact of adding virtual ground op-amps to the circuit which eliminate alternate current paths and compared that circuit to our simpler approach. Existing studies [84] have also shown that using virtual ground mode operational amplifiers at each column in the memristor crossbars eliminates alternate current paths. Therefore, higher level simulation tools can be used to analyze and train the neural networks. However, as we show in this study, the op-amps add significant area and energy requirements to the circuit compared to the circuit approach presented in this chapter. This makes the virtual ground approach less desirable for lower power applications.

A comparison study has also been completed in this chapter where the proposed crossbar circuits were trained using only MATLAB before applying the obtained resistance values directly to the circuit in SPICE. As a result, errors were present in the crossbar circuit because alternate current paths were not accounted for during training.

The overall impact of this study is the demonstration through low level circuit simulations that dense memristor crossbars can be effectively utilized to build neuromorphic processors. A neural network is trained to demonstrate how non-linearly separable functions can also be learned through our low power circuits proving that these neuromorphic circuits can learn any logic function.

7.2 Memristor SPICE Modeling

As opposed to carrying out the simulation in MATLAB, evaluating the actual memristor circuit in SPICE allows for more accurate modelling of the memristor grid. The SPICE simulation captures the alternate current paths and wire resistance within the crossbars. Our results show that training a crossbar without modeling the presence of alternate current paths leads to improper operation when the final weights are simply written to the memristors after off-line training. The simulations in this chapter were completed by writing the memristors after each data pattern was applied, allowing for the crossbar to essentially train around the alternate current paths.

Additionally, SPICE is used in this chapter to accurately model memristor devices. The related work in this area [87, 91] sometimes uses memristor models [2, 11, 14] that are very good representations of the theoretical memristors proposed in [1], but they do not match published memristor device characteristics [3, 4, 103] very accurately.

To model the memristor device in SPICE in this chapter, a previously published memristor model [7-10] was utilized that is capable of reproducing switching characteristics very accurately when compared to other existing models [11-22]. The memristor device simulated for this circuit was published in [103] and the switching

characteristics for the model are displayed in Figure 7.1. This device was chosen for its high minimum resistance value, large resistance ratio, and fast switching time. According to the data presented in [103] this device has a minimum resistance of 125kΩ, a resistance ratio of 10^6, and the full resistance range of the device can be switched in 10ns.

Using this model it was possible to simulate at least 3000 memristors in a single crossbar. The simulations suggested that larger crossbars could be handled, but simulation time became unrealistically long when simulating circuits this large in SPICE. Other research [7] shows that some other memristor models fail in larger simulations, especially when they are set up to produce nanosecond switching times.

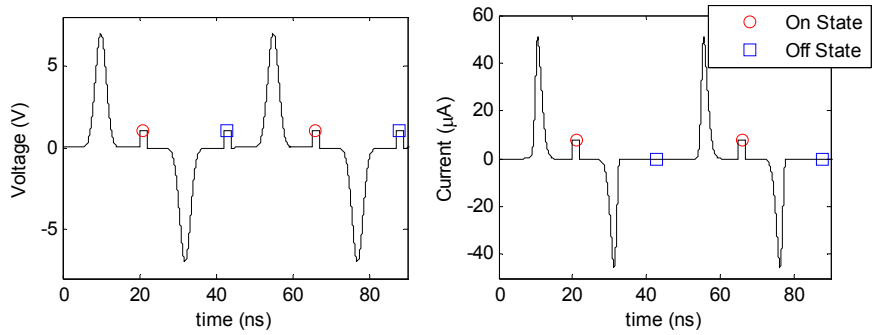


Figure 7.1. Simulation results displaying the input voltage and current waveforms for the memristor model [7-10] that was based on the device in [103]. The following parameter values were used in the model to obtain this result: V_p=4V, V_n=4V, A_p=816000, A_n=816000, x_p=0.985, x_n=0.985, α_p=0.1, α_n=0.1, a_1=1.6×10-4, a_2=1.6×10-4, b=0.05, x_0=0.01.

7.3 Memristor Based Neuromorphic Circuits

7.3.1 Single Layer Perceptron

The schematic in Figure 7.2 shows our approach for developing a memristor based neuron circuit. This example shows two data inputs in addition to a high and low bias input. The neuron outputs in this study are thresholded, and thus a comparator is used. Each signal input is connected to the positive and negative comparator input through a grid of memristors. Two memristors are required to represent a single synaptic

weight, as this allows for each data input to have either a positive or negative effect on the total dot product. Figure 7.3 displays an example of how each pair of memristors is used to determine the total synaptic weight value. If σ_A^+ is greater than σ_A^- then the net synaptic weight will be positive, otherwise it will have a negative impact on the total dot product.

Figure 7.4 displays a neural crossbar circuit where many single neuron circuits are connected together. Each of the comparator outputs corresponds to a different predetermined function output.

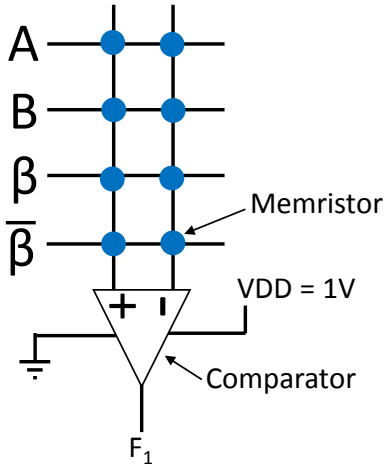


Figure 7.2. Neuron circuit that was trained to recognize linearly separable two-input logic functions.

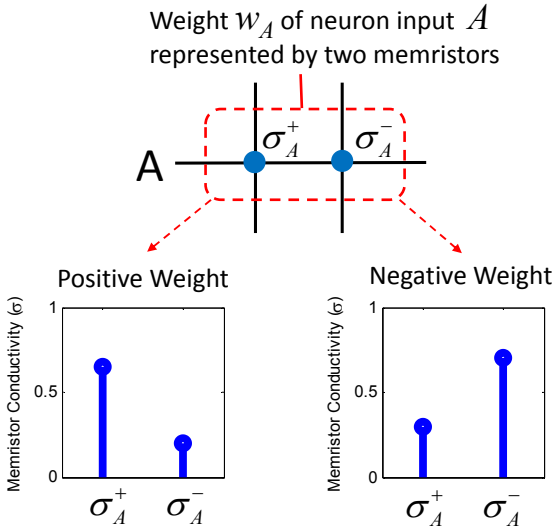


Figure 7.3. The difference in conductance of two memristors determines weight value and polarity for each of the weights in the crossbar.

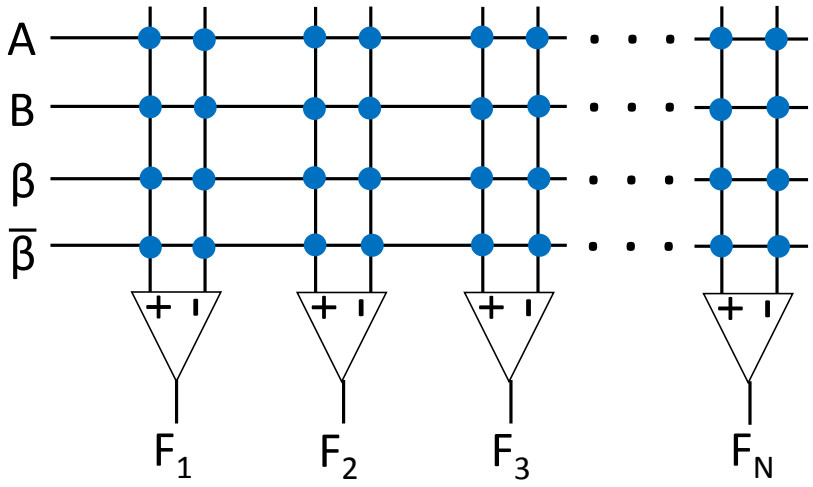


Figure 7.4. Single layer perceptron circuit that is expanded horizontally.

7.3.2 Single Layer Perceptron Training

MATLAB and SPICE were used to develop an accurate system for applying the perceptron learning algorithm to a multi-neuron circuit (see Figure 7.5). The learning function is implemented in MATLAB, and weight updates and evaluations are performed in the crossbar circuit in SPICE.

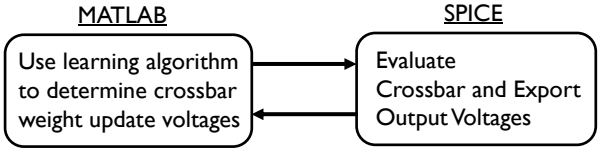


Figure 7.5. Block diagram displaying how the MATLAB and SPICE simulation platform was used to train and evaluate the neuron circuit.

Given that two-input logic functions are being trained, the four data patterns applied to the A and B inputs are 00, 01, 10, and 11 (additionally, 1 is always applied to the β input and 0 is always applied to the $\overline{\beta}$ input as a bias). During each training iteration one of the data patterns is applied as a set of voltages to the crossbar inputs. The crossbar circuit is evaluated in SPICE and the output of each comparator in the circuit (see Figure 7.4) is recorded. Pulses are then applied to the circuit to update the weights according to the learning algorithm. This process is described in the following steps:

1) Initialize the memristors with high random resistances.

2) Apply an input vector x to crossbar circuit in SPICE and evaluate hidden neuron and output neuron values. Record the comparator outputs F_1 through F_N.

3) Calculate the difference, del_j, between the comparator outputs (F_j) and the target outputs (D_j) using equation (7.1) in MATLAB. The parameter del_j represents learning direction and has a value of either +1, -1, or 0 (corresponding to either a weight increase, a weight decrease, or no change respectively).

$$del_j = D_j - F_j \tag{7.1}$$

4) Determine the direction of Δw, for each memristor that requires a conductance change using equation (2) where η is a constant learning rate, and x is the input data pattern.

$$\Delta w_{i,j} = \eta \times del_j \times \left(2x_{i,j} - 1 \right) \tag{7.2}$$

5) Apply write pulses to the crossbar in SPICE with pulse widths proportional to $\Delta w_{i,j}$ to update each memristor conductance.

6) Go to step 2 and apply the next input pattern x in the sequence.

It is assumed that this entire training process will eventually be performed in hardware surrounding the crossbar circuit. The actual hardware design that will record the comparator inputs (or dot products) will be saved for future work.

7.3.3 Multi-Layer Network for Training Non-Linearly Separable Functions

The previous section presented a single layer neural circuit that has the potential to learn linearly separable functions. This section describes a multilayer neural circuit that can learn non-linearly separable functions.

The diagram in Figure 7.6 (b) shows the neural network that was constructed in this study. Each neuron is implemented using the circuit described earlier (reproduced in Figure 7.6 (a)). The circuit diagram for the two layer neural network that was used in this study is displayed in Figure 7.7. This circuit has 3 data inputs and 2 bias inputs. There are 7 neurons in the hidden layer and 2 neurons in the output layer.

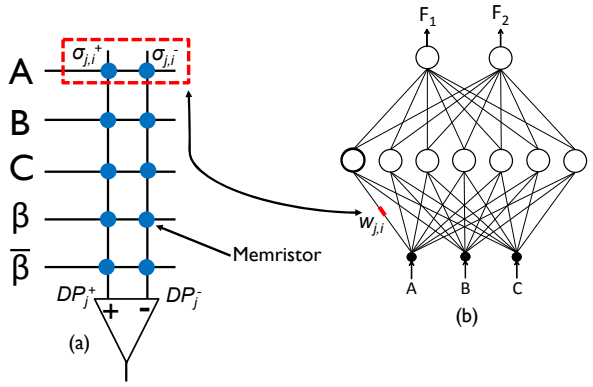


Figure 7.6. (a) Circuit diagram for a single memristor-based neuron and how it fits into (b) the neural network that was constructed in this study.

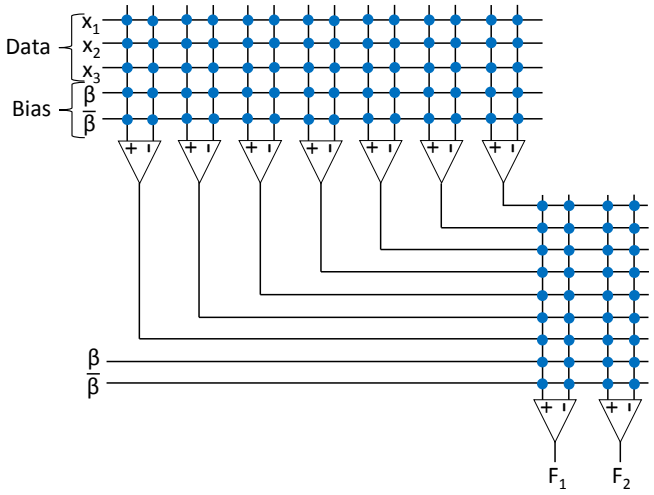


Figure 7.7. Schematic for the neural network circuit that was simulated.

7.3.4 Multilayer Training Algorithm

The Concurrent Learning Algorithm (CLA) [118] was utilized in training these circuits in this study. CLA was chosen because it can learn non-linearly separable problems by training multiple layers of threshold perceptrons and its implementation in hardware can be very efficient. This learning algorithm is similar to the one presented in

the previous section, although there are some slight differences. The steps in this algorithm to train the crossbar for a set of inputs x are:

1) Initialize the memristors with high random resistances.

2) Apply an input vector x to crossbar circuit in SPICE and evaluate hidden neuron and output neuron values. Record comparator inputs DP_j^+ and DP_j^-, where j is the comparator within a given layer.

3) Calculate the difference, *del,* between the comparator outputs (F_j) and the target outputs (D_j) using equation (7.3) in MATLAB.

$$del = \sum_j D_j - F_j \tag{7.3}$$

4) Determine the amount, Δw, that each memristor's conductance should be changed (η is the learning rate) using equations (7.4) and (7.5).

$$DP_j = DP_j{}^+ - DP_j{}^- \tag{7.4}$$

$$\Delta w_{j,i} = \eta \times del \times \frac{1}{1+DP_j{}^2} \times x_{j,i} \tag{7.5}$$

5) Apply write pulses to the crossbar in SPICE with pulse widths proportional to $\Delta w_{j,i}$ to update each memristor conductance.

6) Go to step 2 and apply the next input pattern x in the sequence.

Using this method for updating the weights replaces two steps in the CLA training algorithm with analog operations in the crossbar circuits (simulated in SPICE). First the dot products (DP_j) would normally be calculated according to equations (7.6) and (7.7). Furthermore, the weight update equations (7.8) and (7.9) are replaced by a series of pulses that alter the weights proportional to $\Delta w_{j,i}$.

$$DP_j{}^{+} = \sum_i x_i\left(w_{j,i}{}^{+}\right) \tag{7.6}$$

$$DP_j{}^{-} = \sum_i x_i\left(w_{j,i}{}^{-}\right) \tag{7.7}$$

$$w_{j,i,new}{}^{+} = w_{j,i,old}{}^{+} + \Delta w_{j,i} \tag{7.8}$$

$$w_{j,i,new}{}^{-} = w_{j,i,old}{}^{-} - \Delta w_{j,i} \tag{7.9}$$

7.4 Simulation Results

7.4.1 Single Layer Linearly Separable Logic Functions

To test the functionality of the neural circuit in Figure 7.4, a circuit with 14 comparator outputs was trained so that each neuron represented one of the 14 linearly separable two-input logic functions. The result in Figure 7.8 shows absolute error that was present during the training process. The system was successfully trained after 5 epochs.

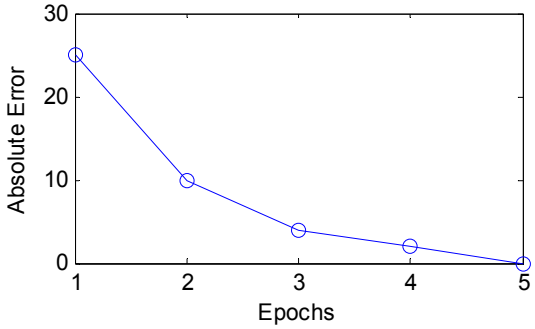


Figure 7.8. Absolute error present at each epoch when trained a neural circuit with 14 outputs to learn each of the 14 linearly separable logic functions.

Furthermore, the single layer perceptron circuit was studied to see how many two-input digital logic functions could be trained within a single crossbar. Multiple simulations were completed where the width of the crossbar was progressively increased (see Figure 7.4) to determine how many functions could be trained. This was process was

completed until 128 functions were successfully trained using a single crossbar comprised of 1024 memristors within a SPICE circuit. It is assumed that larger crossbars could also be trained successfully, but enlarging the SPICE circuit further became infeasible. Therefore, the next section describes how wire resistance was increased to simulate the effect of a much larger crossbar.

7.4.2 Multilayer Non-Linearly Separable Logic Functions

The 2 output neurons in the circuit in Figure 7.7 were trained to recognize the non-linearly separable functions displayed in equations (7.10) and (7.11). There is no plane in a three dimensional space which can separate the different outputs of these functions. Therefore, at least a two layer neural network with non-linear thresholding is required.

$$F_1 = A \oplus B \oplus C \tag{7.10}$$

$$F_2 = ABC + \bar{A}\bar{B}\bar{C} \tag{7.11}$$

Figure 7.9 shows the absolute error present throughout the training process when learning functions F_1 and F_2. Learning continues until the total error reaches 0, and the results show that it took about 130 epochs to learn both functions. This is a significantly larger number of epochs when compared to the result in Figure 7.8. This increase in required epochs is due to the complex learning algorithm and the complex target functions that were learned. This increase in epochs is due to the increase in complexity of the problem and is not specific to the proposed circuit design.

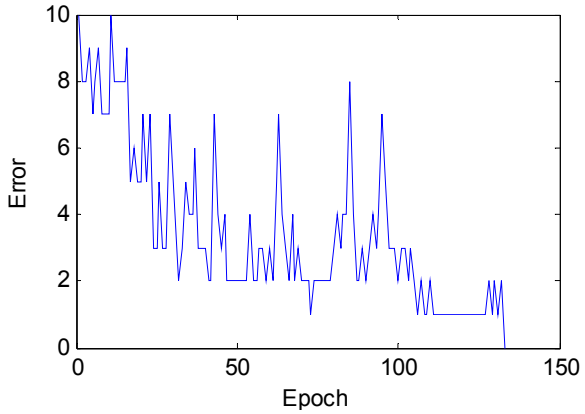


Figure 7.9. Result that displays the training error reaching zero after about 130 epochs.

7.4.3 MNIST Image Recognition

This section shows how the proposed single-layer circuit in Figure 7.4 (with a significant increase in the number of inputs) is able to successfully classify MNIST characters when all the low lever properties in the proposed neuromorphic circuit are considered. This study used a limited number of images from the MNIST data set to show that a single layer memristor crossbar is capable of image recognition. In this example 12 of the 28 by 28 images in the MNIST database (4 zeros, 4 ones, and 4 twos) were reduced to 14 by 14 pixel images for training. The system trained to zero error after three epochs as displayed in Figure 7.10. This was done with an 1188 memristor crossbar in SPICE.

After training, the system was evaluated with a separate set of 3 testing images which are displayed in Figure 7.11 along with the comparator outputs in each case. The comparator outputs show that each digit was correctly identified.

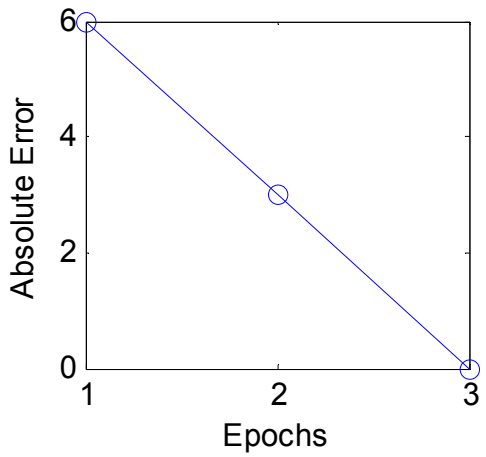
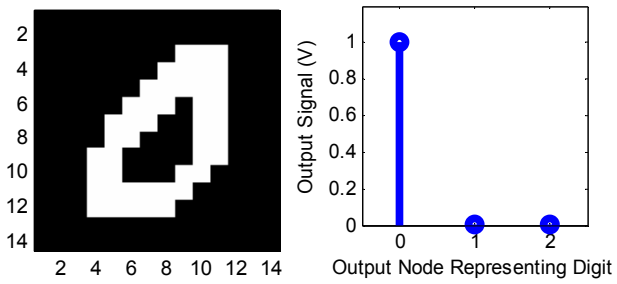
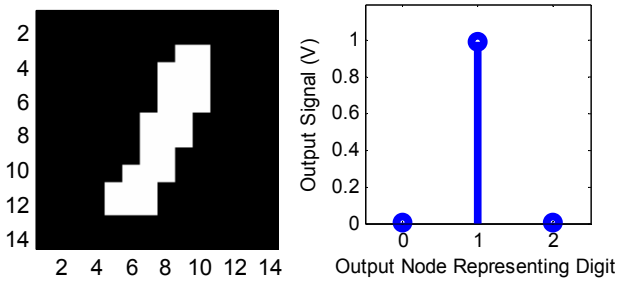


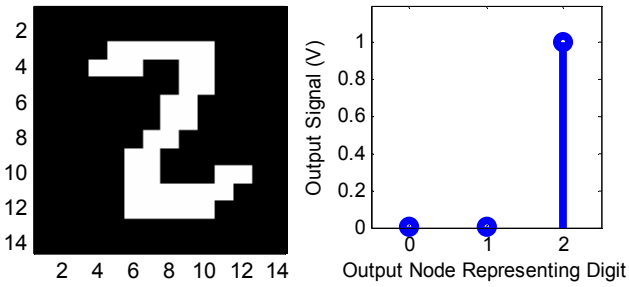
Figure 7.10. Absolute error during training using the 12-image data set from MNIST.



(a)



(b)



(c)

Figure 7.11. Testing image and resulting circuit outputs for the testing images representing (a) zero, (b) one, and (c) two.

It has been shown that these circuits are capable of learning complex logic functions. The following sections describe analyses that were performed to determine how large a single neural crossbar layer could become. Training multilayer circuits according to a learning algorithm in SPICE is a very time consuming process, so the analysis performed in the following sections is based on expanding a single layer of neurons. A multilayer system may perform differently, but isolating a single layer is the best way to provide a clearer result that can be adjusted to account for multiple layers.

7.5 Minimum Comparator Voltage Difference and Alternate Current Paths

The studies in this section demonstrate how considering alternate current paths alters the functionality of neuromorphic memristor circuits. The first study shows that errors occur if this circuit was trained off-chip with a higher level simulation tool without considering alternate current paths. The second study examines the minimum difference in comparator input voltage that is possible when considering and not considering alternate current paths.

7.5.1 Impact of Alternate Current Paths on Training

This study demonstrates how training a system using approximate circuits in MATLAB that do not handle all possible current paths can lead to errors. The experiments were completed first considering alternate current paths by analyzing the circuit in Figure 7.12 (a) in SPICE. Also, the circuit with alternate current paths removed (see Figure 7.12 (b)) was evaluated analytically in MATLAB. The circuit in Figure 7.12 (b) is the type of circuit that would be required when assuming a memristor crossbar is capable of doing a direct matrix multiply as in [90].

The circuit in Figure 7.12 (b) was solved analytically using superposition equations in MATLAB (and alternate current paths were ignored). This circuit was trained until zero error occurred according to the learning algorithm applied to the approximated circuit.

The memristor resistances that were obtained using this training method were then applied directly to the SPICE circuit in Figure 7.12 (a) and errors resulted during the evaluation of 2-input logic functions. Figure 7.13 shows the number of errors in the circuit that were obtained (due to the added presence of alternate current paths) by directly importing resistance values for a varying number functions. This shows that the neuromorphic circuit will not operate correctly unless alternate current paths are handled during the learning process.

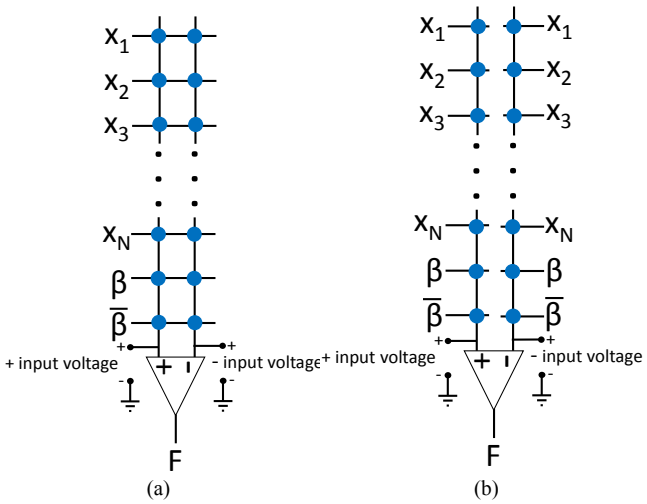


Figure 7.12. Circuit for determining minimum difference between comparator input voltages (a) with and (b) without alternate current paths.

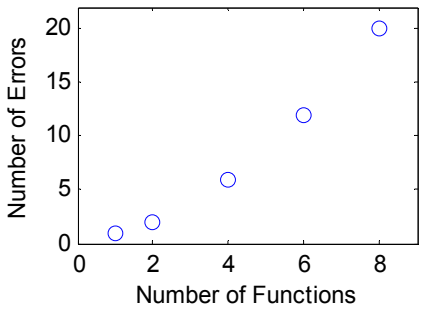


Figure 7.13. Number of errors present in the system when weights obtained through ex situ training are applied to the circuit in Figure 7.12.

7.5.2 Impact of Alternate Current Paths on Comparator Voltage

The resolution of the comparator that is used in the circuit can contribute to how large a single crossbar can be, as this will impact the maximum number of inputs. In this experiment, the minimum comparator resolution is defined as the smallest voltage difference between the + and – inputs that will allow correct switching in the comparator to occur. To complete this experiment, the circuits in Figure 7.12 were used, and the number of rows (or data inputs) the circuits possessed was progressively increased.

Figure 7.14 displays the resulting difference between the comparator input voltages as a function of the number of rows in the crossbar circuit. The results in Figure 7.14 are based on setting all of the inputs to 1, and setting all of the data input memristors to their max value (R_{OFF}) except for single a memristor corresponding to an input on the positive side of the comparator (which was set to the minimum resistance value R_{ON}). One option for setting the bias memristors that provided strong results was to set the memristors corresponding to the β input to R_{MIN} on the left and $2R_{ON}$ on the right and setting the memristors corresponding to the $\bar{\beta}$ input to R_{OFF}.

The results in Figure 7.14 show that the voltage difference between the + and – comparator inputs is reduced as more inputs are added to the neuron circuit. This data can be used to determine an upper limit on the number of rows in the neural crossbar based on the minimum voltage difference required in the comparators.

When comparing the results in Figure 7.14, the voltage differences obtained when considering alternate current paths (using the circuit in Figure 7.12 (a)) are slightly lower than those obtained analytically (using the circuit in Figure 7.12 (b)). This is because the

alternate current paths cause the voltage differences to be slightly smaller due to a few more paths to ground through the bias memristors. Figure 7.15 shows the result from a similar simulation where the only difference is a change in the data input set. In this case a 1 is applied to the row where the memristor set to R_{ON} is connected to the positive comparator input, and a zero is applied to all other inputs. In this case there is a much larger difference between the two results. This is because the applied data pattern has man more zeros in it, and thus many more paths to ground. Therefore, less current is sampled to determine the comparator voltages when alternate current paths are considered. This result shows that it is more important to consider alternate current paths with a sparse input data set when using this circuit.

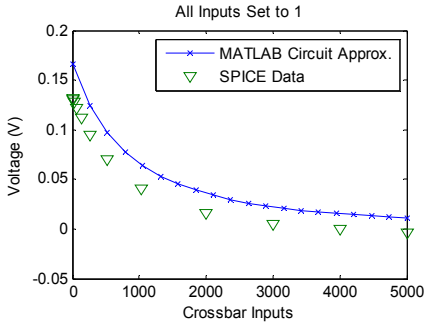


Figure 7.14. Difference in voltage between the positive and negative comparator inputs as a function of the number of inputs in the crossbar where all data inputs are set to 1.

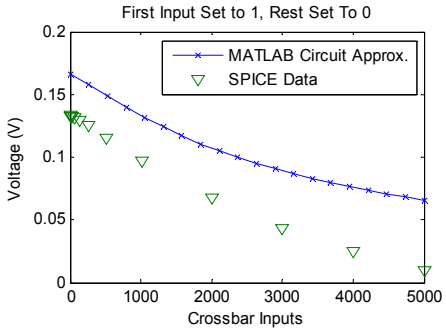


Figure 7.15. Difference in voltage between the positive and negative comparator inputs as a function of the number of inputs in the crossbar where only a single data input is set to 1.

7.6 Crossbar Wire Resistance

In larger crossbars, it is possible that the wire resistance of the horizontal and vertical wires may consume enough voltage drop to disrupt a crossbar write operation. This may lead to difficulty or an inability to train memristors in the crossbar that are located further from the voltage sources.

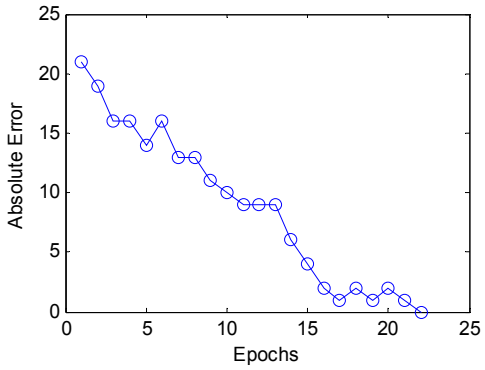
A 5Ω wire resistance between each memristor has been assumed, and the minimum resistance of the memristors studied in these simulations was 125KΩ. Simulating a crossbar large enough for wire resistance to have an impact on this design is infeasible, so the wire resistance value was increased for the simulations in this section.

Table 7.1 describes two experiments that were completed to test the impact of wire resistance. The wire resistance was increased to 20KΩ between each memristor. This circuit can be understood to represent a much larger crossbar where only select columns with a lot of distance between them are being trained. The result in Figure 7.16 shows a general trend where the functions closest to the voltage sources (see functions 1 and 2 in Figure 7.16 (b)) require the least epochs to train, and the functions furthest from the voltage sources require the most epochs to train (see function 8 in Figure 7.16 (b)). The entire system was able to train to zero error in less than 25 epochs. Figure 7.17 shows a nearly identical experiment where the number of function outputs in the system was increased to 12. In this case the 6 functions furthest from the voltage source were not able to train. This is because enough voltage was lost on the wires, so that the write threshold was not exceeded on the memristors corresponding to these functions. Using this data, it can be assumed that a crossbar with a 5Ω wire resistance may have write errors if more than 24,000 functions are present in the crossbar. The next sections

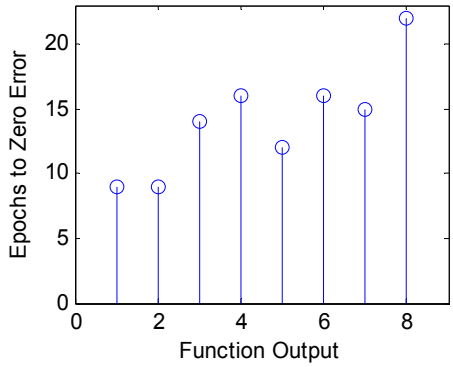
describe how driver power requirements and wire capacitance may impact these larger crossbars.

Table 7.1. Simulation setup for the 2 wire resistance experiments.

Measurement	Exp. 1	Exp. 2
Function Outputs	8	12
Wire Resistance	20KΩ	
V_w	4.5V	
Data Inputs	2	
Function Type	All OR Functions	

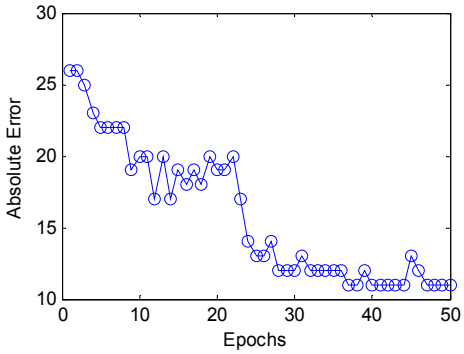


(a)

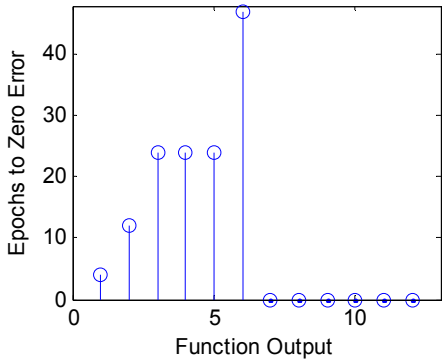


(b)

Figure 7.16. Simulation results for Exp. 1 displaying (a) the absolute error and (b) how long it took each function to train depending on its position relative to the input voltage sources.



(a)



(b)

Figure 7.17. Simulation results for Exp. 2 displaying (a) the absolute error and (b) how long it took each function to train (if it was able to train) depending on its position relative to the input voltage sources. The zeroes for functions 6 through 12 in (b) shows that these functions were unable to train.

7.7 Driver power requirements of larger crossbars

As the crossbar circuit in Figure 7.4 is expanded horizontally and more functions are added, this will increase the amount of power consumption at the data inputs, which will impact the driver circuitry. Our simulation results show that increasing the size of the crossbar by a single function will increase the peak power by about 25nW and the evaluation energy by 400aJ. The maximum number of function output in the crossbar will depend on the size of the input current drivers used. The plots in Figure 7.18 show how the energy and power requirements increase as the number of function outputs in the crossbar is increased.

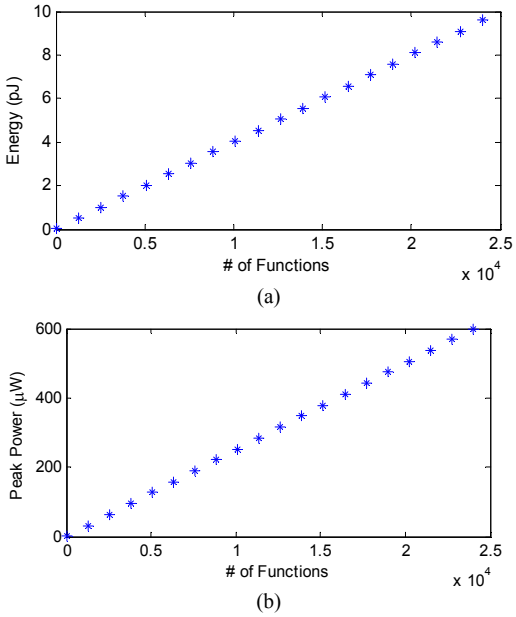


Figure 7.18. Plots displaying (a) the increase in evaluation energy and (b) the peak power for a single evaluation input as the number of functions is increased.

7.8 Crossbar Wire Capacitance

To calculate memristor capacitance within a crossbar, the parallel plate of the memristor electrodes was considered in addition to the capacitance between the horizontal and vertical wires. The parallel plate capacitance was calculated according to equation (7.12), where the plate area was assumed to be 50nm×50nm, and the thickness set to 50nm. Since the device studied in [103] had a thinfilm comprised of a-Si, the dielectric constant ε_r was set to 11.8. The resulting parallel plate capacitance was determined to be 5.224aF.

$$C = \frac{\varepsilon_0 \varepsilon_r A}{d} \tag{7.12}$$

To determine the capacitance of the longer parallel crossbar wires the method described in [119] was used which can be seen in equations (7.13) through (7.18). The parameters *g*, *h*, and *s* represent half the spacing between the wires (25nm), the dielectric

thickness (50nm), and the width of the strips (50nm) respectively. The function K represents the elliptic integral of the first kind.

$$C = \varepsilon_0 \varepsilon_{eff} \frac{K(k_0')}{K(k_0)} \tag{7.13}$$

$$\varepsilon_{eff} = 1 + (\varepsilon_r - 1)q \tag{7.14}$$

$$q = \frac{1}{2} \frac{K(k_0')}{K(k_0)} \frac{K(k_0)}{K(k_0')} \tag{7.15}$$

$$k_0 = \frac{g}{s+g} \tag{7.16}$$

$$k_0' = \sqrt{1 - k_0^2} \tag{7.17}$$

$$k = \frac{\tanh\left(\frac{\pi g}{2h}\right)}{\tanh\left(\frac{\pi(s+g)}{2h}\right)} \tag{7.18}$$

The value determined for the capacitance in F/m per each set of parallel wires was determined to be 6.51×10^{-11}. This value was then multiplied by 100nm, the length of the wire segments associated with each memristor device in the crossbar based on a 25% packing density to obtain 6.507aF per memristor. To obtain the total capacitance associated with each memristor the parallel plate capacitance was added to double the wire segment capacitance (to account for both the horizontal and vertical wires) for a total capacitance of 18.238aF per memristor. Upon adding these capacitances to the SPICE simulation, it was determined that this was not enough capacitance to significantly alter the operation of the crossbar. This is most likely because the maximum resistance of a memristor observed in the crossbar circuits is typically about 120MΩ, leading to an RC constant of about 2.16ns. Read and write operations were set at 50ns in our simulations.

Therefore, the capacitance will not have an impact unless memristor resistance exceeds 2.8GΩ.

7.9 Comparison to a Virtual Ground System

It has been suggested in [84] that alternate current paths can be eliminated by using virtual ground mode operational amplifiers at the output of each column in the crossbar. This design is displayed in Figure 7.19. In this case the comparator is switched between -1 and 1 V to allow for switching in the negative region (which is necessary due to the inverting op amps).

The advantage of the op-amp approach is that alternate current paths are no longer present, so circuits can be trained off-line and programmed more quickly. This is because the elimination of alternate current paths allows for weights to be programmed using an ex situ method.

The data in Table 7.2 shows a comparison of the two approaches in terms of area and energy. The approach that uses op-amps requires over three times as many transistors per neuron and about 88 times more energy during an evaluation of an input data pattern. However, the addition of virtual ground op-amps allows for the possibility of ex situ training. In applications where ex situ training is not desired, the circuits proposed in this chapter can be used for in situ training at a fraction of the area and energy cost.

As more synaptic inputs are applied to a single comparator output, the number of transistors per input is reduced. Figs. 21 (a) and (b) show how the number of transistors and circuit area per synaptic input are reduced as the number of synaptic inputs increases. Overall, the op-amp method requires about 3 times more transistors than the proposed method.

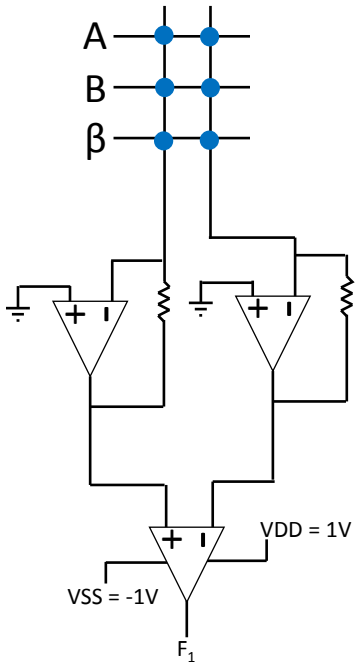
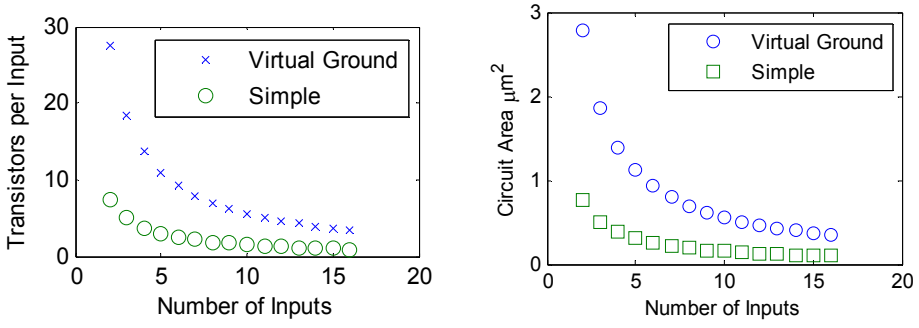


Figure 7.19. Circuit design for a neuron circuit that uses operation amplifiers to elimiate alternate current paths.

Table 7.2. Area and energy comparison of the circuit designs proposed in this chapter and a virtual ground approach.

	Sneak Paths	Evaluation Energy per function operation (fJ)	Transistors to Implement a single function
Simple Approach	Yes	415	15
Virtual Ground Approach	No	36800	55



(a) (b)

Figure 7.20. Area comparison of the proposed system and the virtual ground appraoch in terms of (a) number of transistors and (b) circuit area per synaptic input.

Figure 7.21 displays how power consumption increases as a function of the number of neurons in a circuit. Since the op-amps consume a considerable amount of energy, total energy consumption increases dramatically as the number of neurons is increased when using the virtual ground approach.

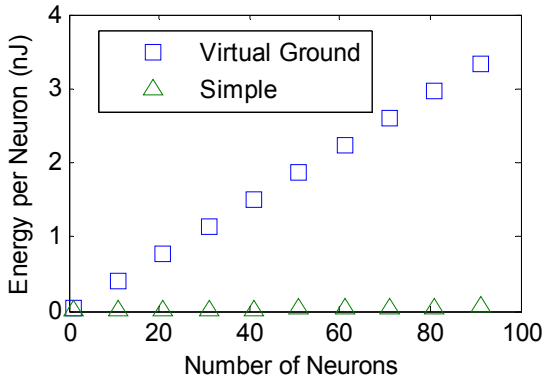


Figure 7.21. Plot displaying energy consumption as the number of neurons in the system is increased.

7.10 Memristor Fault Tolerance

The following section describes how the neuromorphic circuit described in Figure 7.22 reacts to defective memristor devices that are stuck at either a low or high resistance value. In this section N is set to 14 and each function output F_1 through F_{14} is set to a different linearly separable two-input logic function.

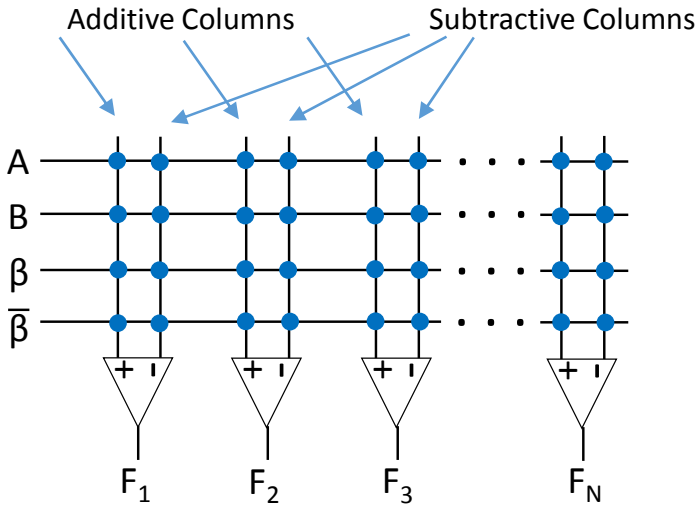


Figure 7.22. Circuit used to perform the fault tolerance experiments.

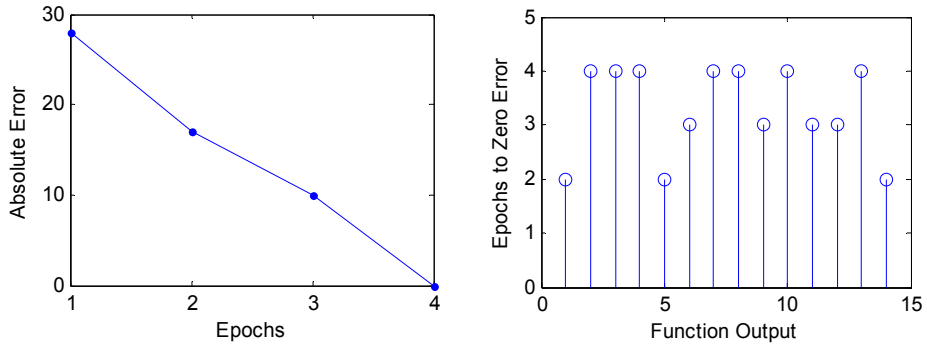
7.10.1 Defective Memristors Stuck at a High Resistance State

In the first experiment, randomly selected memristors were set so that they were unable to change from their initial high resistance state. In this case the random defective memristors were all selected to be in either only in the additive or only the subtractive columns. This eliminated the possibility having a defective pair of memristors that corresponded to a single synapse. The results are displayed in Table 7.3. The circuit appears to be very resilient defective memristors and is able to train when half of the memristors in the circuit are unable to switch. Out of the 30 simulations performed, only 5 failed to train after 50 epochs.

Table 7.3. Number of epochs required to train the circuit for varying numbers of defective memristors.

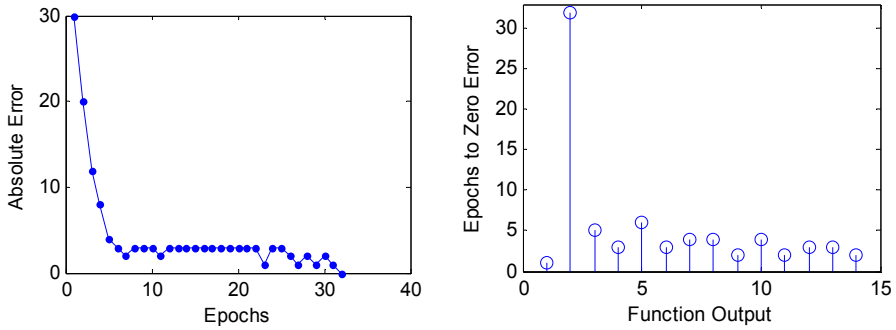
	Percentage of Defective Memristors					
Trial	0%	5%	10%	30%	40%	50%
1	5	24	14	**50**	6	32
2	4	4	**50**	8	**50**	4
3	5	20	14	10	5	6
4	4	22	47	**50**	27	5
5	5	5	5	**50**	7	5

The simulation result in Figure 7.23 shows that in one case all functions were able to train in 4 epochs when 50% of the memristors were stuck at a high resistance state. Figure 7.24 displays the results of an identical experiment where 32 epochs were required to train the circuit. The difference was most likely due to the random initial values of the memristors. Figure 7.24 (b) shows that all but one of the functions was able to train in less than 10 epochs, and the extra training time was all due to training a single function. This is a common occurrence in these experiments where a single situation acts as a bottleneck in training time for the entire circuit.



(a) (b)

Figure 7.23. Training result displaying (a) errors per epoch and (b) the number of epochs it took to train each function when 50% of the memristors are stuck at a high resistance state where all functions were trained in 4 epochs.



(a) (b)

Figure 7.24. Training result displaying (a) errors per epoch and (b) the number of epochs it took to train each function when 50% of the memristors are stuck at a high resistance state where all functions were trained in 32 epochs.

Table 7.4 shows how the results change when defective memristors are placed in either memristor column with respect to function output, although it was ensured that at least one memristor was functional within each synapse. In this case it was much more difficult to train all functions in the circuit, where 3 out 5 simulations were unable to learn in 50 epochs. Even though there are no defective pairs of memristors corresponding to the same synapse, there are a lot more possible combinations of defective memristors that can lead to error.

Table 7.5 displays a result where defective memristor positions are completely randomized across the circuit, although only 10% of the memristors are set to be

defective. The comparison in Table 7.5 shows a fairly similar result for each experiment, although completely randomizing the position of the defective memristors seems to increase the average number of epochs required to train. Although, the completely randomized case is much harder to train when the percentage of defective memristors is increased. This is because this will increase the likelihood of a defective pair of memristors that correspond to a single synapse.

Table 7.4. Number of epochs required to train with 50% defective memristors when errors can either only appear within the left or the right column compared to randomized errors still not allowing pairs.

Trial	Errors Occur in Either Column	Errors Occur in A Single Column
1	26	32
2	22	4
3	**50**	6
4	**50**	5
5	**50**	5

Table 7.5. Number of epochs required to train with 10% defective memristors when errors are completely randomized compared to randomized errors still not allowing pairs.

Trial	No Error Pairs	Completely Random Errors
1	14	6
2	50	50
3	14	9
4	47	5
5	5	6

7.10.2 Comparison to Alternative Learning Circuit

Experiments similar to those explained in this section have previously been performed on a similar circuit [80]. One of these experiments was repeated using the circuit in Figure 7.25 to compare these circuits in terms of their ability to overcome defective memristors.

The results in Table 7.6 show that there are 3 cases when the circuit in [80] was unable to train in 50 epochs. However, the circuit in Figure 7.25 was able to train for every defective memristor position that was tested in [80]. Although, the number of epochs has to be increased when the subtractive side bias memristor was stuck at a high resistance state.

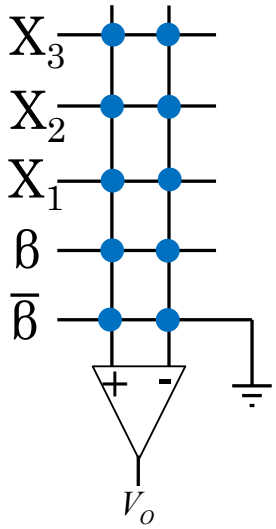


Figure 7.25. Circuit used to compare to alternative defective memristor experiments [80].

Table 7.6. Comparison in resilience to defective memristors between the circuit described in this chapter and the circuit described in [80].

	Our System	Other System []
Memristor(s) Stuck in High Resistance State	Epochs	Epochs
None	3	3
X3+	4	2
X3-	2	2
X2+	19	**50**
X2-	3	4
X1+	35	**50**
X1-	4	4
BH+	2	4
BH-	75	**50**
BL+	2	[redacted]
BL-	2	[redacted]
X3+, X3-, X2-, X1-, BH+	2	2

The main difference between this circuit and the one presented in [80] is that there is only one row of bias memristors in the circuit [80]. The circuit in 7.25 has two rows of bias memristors for greater redundancy, and this is most likely the reason for the increase in tolerance to defective memristors.

7.10.3 Memristors Stuck at Low Resistance State

As an alternative experiment, the defective memristors in this study were instead set to their minimum resistance value. Based on the results of preliminary experiments, it is much more difficult to train around a memristor that is stuck at its minimum value. Instead of trying to train a circuit with a random selection of defective memristors, the two-input circuit in Figure 7.22 was trained with the same memristor position in each function was selected to be stuck. This process was repeated 8 times until each possibility for a defective memristor position was tested. This led to a result that showed which functions could handle which defective memristor positions for all possible linearly separable functions. The results in Figure 7.26 show depending on the placement of the defective memristor, between 1 and 6 functions can be successfully trained in less than 50 epochs. In the plots in Figure 7.26 a value of zero epochs represents a function that could not be trained.

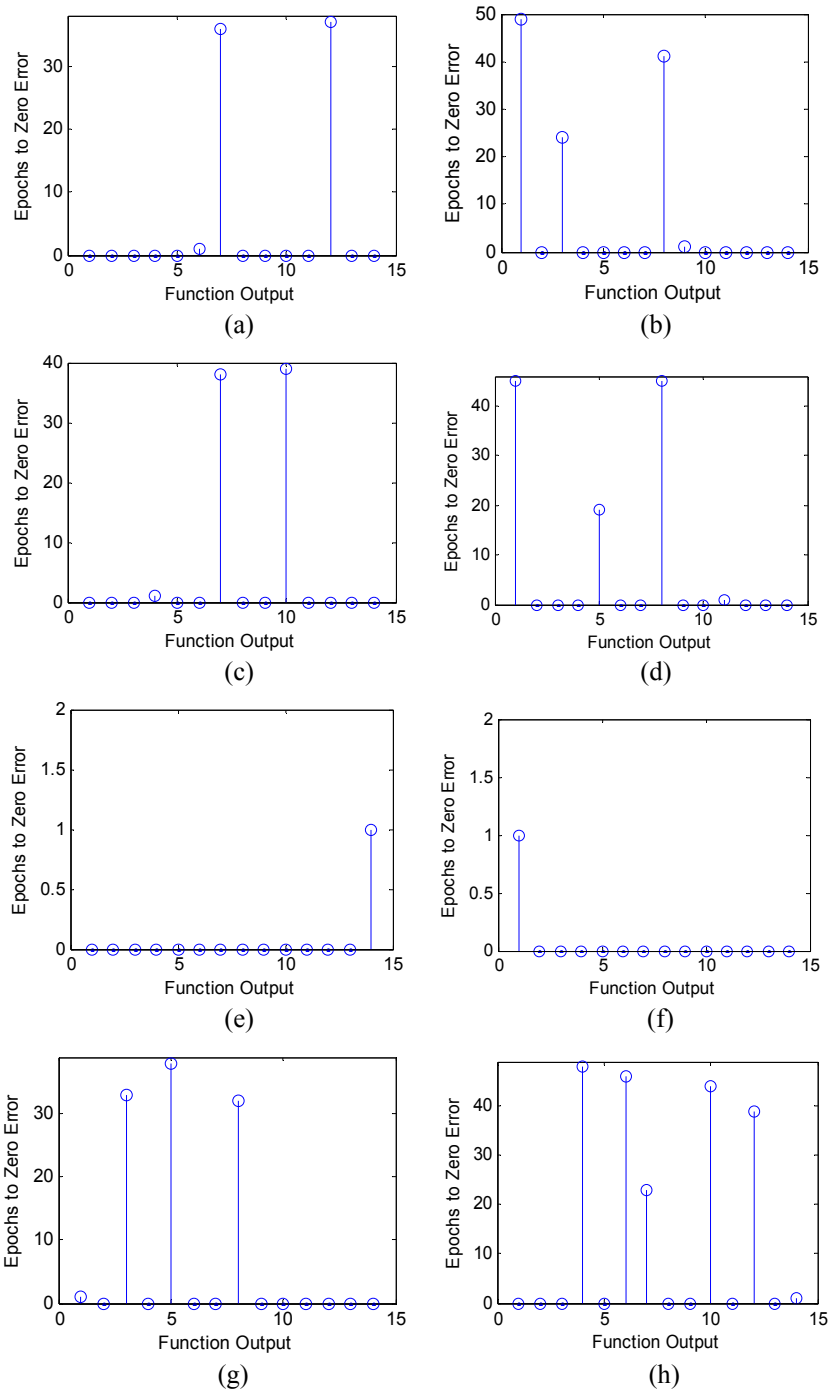


Figure 7.26. Number of epochs required to train each of the functions in Figure 7.22 when a defective memristor stuck at its minimum resistance was placed at (a) A^+, (b) A^-, (c) B^+, (d) B^-, (e) β_H^+, (f) β_H^-, (g) β_L^+, and (h) β_L^-.

CHAPTER VIII

CONCLUSION

The chapters within this document describe a memristor model as well as several novel circuit designs. The memristor model is accurate to characterized memristor devices and is highly scalable. This allowed for very detailed simulation of circuit designs for a variety of applications. The memristor device is relatively new and fabrication process are still being developed using a variety of materials and deposition techniques. The work described shows that even limited by the current state of memristor technology these devices yield strong results. Overall this document shows how the memristor has the potential to vastly improve electronic systems in terms of area and energy. The following sections describe the conclusions drawn from each of the projects.

8.1 Memristor Modeling

The SPICE model presented in Chapter IV can be used to provide accurate circuit simulations for a wide range of memristor devices and voltage inputs with a low likelihood of convergence errors. Adjusting the fitting parameters provided the capability of simulating devices with different physical structures. Furthermore, high-speed switching devices were modeled using nanosecond pulse simulations which led to accurate values for switching energy. A study was completed that determined how much parameter variation could be tolerated before a significant change in the I-V characteristic

occurred. The SPICE model was also used to determine how much variation can be present in a memristor array before read errors occur.

The generalized memristive SPICE model presented in this paper should be capable of modeling future devices, as it has already been shown to successfully model the large amount of variety seen in many different existing devices. This model should alleviate the necessity of developing a specific SPICE subcircuit for each published memristor characterization.

To improve this memristor model, access to a large amount of physical device data would be beneficial. At this point the memristor model is based on the very limited amount of characterization data that is available in the literature. Collecting more characterization data would also provide more detail into how much variation is present across a memristor wafer. Then the techniques in Section 3.4 could be used to match the memristor parameter variation to a realistic number according to the collected data.

Also, physical memristor characterizations show a non-uniformity in the amount of resistance change incurred by a given pulse width. To study this it would be beneficial to repetitively apply several short pulses thought to partially change the state of a memristor over several switching cycles. This data could be used to randomize the state change mechanism in the model, as extremely specific resistance values may not be as attainable using physical devices.

8.2 Memristor-Based ROIC Circuit Design

A memristor based unit cell for a photodetector readout circuit has been presented in Chapter V. The circuit was simulated using SPICE, where the memristor was modeled based on published characterization data of two different devices. When simulating the

circuit with a low resistance memristor [26], it required unrealistic power requirements to operate properly. When modeling a memristor with a higher resistance [4], the detector current could be made much smaller, although the voltages remained high due to the programming threshold of the memristor. The simulations also show that the circuit may work well in low light situations with a longer integration time.

In the future, different models for the MOSFETs will be used that more closely model specific devices. Specific MOSFETs should be chosen with resistances that are better matched to the memristor's resistance values. This will cause less leakage through the MOSFETs and lower pulse voltage requirements.

Also, different photodetectors will be inspected that have lower series resistance. Further future work could include a study the logarithmic response of the change in the state variable as it approaches the low resistance boundary. This may be advantageous as the memristor does not linearly saturate at the resistance limit, leading to a longer and more useful integration range.

Further improvements to the circuit may include an alternative circuit design that does not allow a variable voltage drop across the detector-memristor current path (Figure 5.2 (a)). A variable voltage drop is not typically seen across the photodetector in a read-out integrated circuit and it limits the dynamic range of the memristor based on its programming threshold. Also, higher resistance memristors will have to be studied to further lower the required alter memristor resistance. This will allow for finer integration as a lower number of photons will need to be present to alter memristor resistance.

8.3 Memristor Memory Architectures

The results in Chapter VI indicate that the memristor based cache proposed, along with the hybrid cache structure can improve the performance of processors and reduce power consumption. As new and better devices are produced, improved performance from memristor based caches can be expected. For instance the 2012 memristor device from HP [23] has a lower write time of about 2ns. However this device has a resistance that is still too low for use in a cache.

As future work we will examine the use of multiple bits per memristor cell. A similar study for STT-MRAM cells [47] shows that this can increase the areal density of the memories. Also, the use of MIM diodes in series with memristors (or non-linear memristors that possess this property) may reduce power consumption and result in a larger minimum crossbar partition size. More possible improvements to this memory system may become apparent as faster, higher resistance memristor devices are developed. Minimizing programming threshold voltage would also be beneficial for on-chip applications.

8.4 Memristor Neuromorphic Systems

Chapter VII clearly shows that a passive memristive crossbar can be used as the synapses in larger neuromorphic processors. Training was completed in SPICE to ensure that alternate current paths within the memristor crossbar were accounted for. Simulations in SPICE showed that learning algorithms could properly train in the presence of these alternate current paths and that the size of the crossbar can be quite large. Many different circuit properties were considered including wire resistance, wire capacitance, input driver current, and minimum voltage difference between the voltage

comparators. Each of these properties was used to predict limits to how large a realistic neuromorphic memristor crossbar may be.

The proposed neuron circuit was used in a multilayer system to successfully train non-linearly separable functions using the CLA algorithm [9]. This shows that a neuromorphic processing tile based on our design can learn any digital logic function. Finally, it was shown that this feedback learning system could learn without virtual grounds, greatly reducing the power and size the processing tile would require (but requiring in-situ training).

These analog neuromorphic circuits have the capability of quickly solving pattern recognition problems. These high-density low power networks provide the opportunity to perform the computations in a much more portable fashion (as opposed to computing clusters). Future work will involve the study of larger multilayer neural networks that can solve more complex problems. Once sufficient simulation results have been obtained, development of a neuromorphic processing chip can begin.

BIBLIOGRAPHY

[1] L. O. Chua, Leon O, "Memristor—The Missing Circuit Element," *IEEE Transactions on Circuit Theory*, 18(5), 507–519 (1971).

[2] D. B. Strukov, G. S. Snider, D. R. Stewart, and R. S. Williams, "The missing Memristor found," *Nature*, 453, 80–83 (2008).

[3] G. S. Snider, "Cortical Computing with Memristive Nanodevices," *SciDAC Review*, (2008).

[4] S. H. Jo, T. Chang, I. Ebong, B. B. Bhadviya, P. Mazumder, and W. Lu, "Nanoscale Memristor Device as Synapse in Neuromorphic Systems," *Nano Letters*, 10 (2010).

[5] S. H. Jo, and W. Lu, "CMOS Compatible Nanoscale Nonvolatile Resistance Switching Memory," *Nano Letters*, 8(2), 392-397 (2008).

[6] C. Yakopcic, T. M. Taha, G. Subramanyam, E. Shin, P. T. Murray, and S. Rogers, "Memristor-Based Pattern Recognition for Image Processing: an Adaptive Coded Aperture Imaging and Sensing Opportunity," *SPIE Adaptive Coded Aperture Imaging and Non-Imaging Sensors*. (2010).

[7] C. Yakopcic, T. M. Taha, G. Subramanyam, and R. E. Pino, "Generalized Memristive Device SPICE Model and its Application in Circuit Design," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 32(8) August, 2013 pp. 1201-1214.

[8] C. Yakopcic, T. M. Taha, G. Subramanyam, R. E. Pino, S. Rogers, "A Memristor Device Model," *IEEE Electron Device Letters*, 30(10), 1436-1438, October 2011.

[9] C. Yakopcic, T. M. Taha, G. Subramanyam, and R. E. Pino, "Memristor SPICE Model and Crossbar Simulation Based on Devices with Nanosecond Switching Time," IEEE International Joint Conference on Neural Networks (IJCNN), August 2013.

[10] C. Yakopcic, T. M. Taha, G. Subramanyam, R. Pino, and S. Rogers, "Memristor SPICE Modeling," in *Advances in Neuromorphic Memristor Science and Applications*, R. Kozma, R. Pino, and G. Pazienza (eds), Springer. July 2012.

[11] Y. N. Joglekar and S. J. Wolf, "The elusive memristor: properties of basic electrical circuits," *European Journal of Physics*, 30(661), (2009).

[12] M. Laiho, E. Lehtonen, A. Russel, and P. Dudek, “Memristive synapses are becoming a reality,” *The Neuromorhic Engineer*, (2010).

[13] R. E. Pino, J. W. Bohl, N. McDonald, B. Wysocki, P. Rozwood, K. A. Campbell, A. Oblea, and A. Timilsina, “Compact method for modeling and simulation of memristor devices: Ion conductor chalcogenide-based memristor devices,” IEEE/ACM International Symposium on Nanoscale Architectures, pp. 1 – 4, (2010).

[14] Z. Biolek, D. Biolek, V. Biolková, "Spice Model of Memristor with Nonlinear Dopant Drift," *Radioengineering*, 18(2), 210-214 (2009).

[15] E. Lehtonen and M. Laiho, “CNN using memristors for neighborhood connections,” feb. 2010, pp. 1 –4.

[16] D. Batas, H. Fiedler, “A Memristor SPICE Implementation and a New Approach for Magnetic Flux-Controlled Memristor Modeling,” *IEEE Transactions on Nanotechnology*, 10(2), 2011.

[17] A. Rak and G. Cserey, “Macromodeling of the memristor in spice,” *Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on*, vol. 29, no. 4, pp. 632 –636, Apr. 2010.

[18] S. Benderli and T. Wey Cserey, “Macromodeling of the memristor in spice,” *Computer-Aided Design of Integrated Circuits and*, “On SPICE macromodelling of TiO2 memristors,” *Electronics Letters*, vol. 45, no. 7, pp. 377–379, 2009.

[19] M. Mahvash and A. C. Parker, “A memristor SPICE model for designing memristor circuits,” Aug. 2010, pp. 989 –992.

[20] T. Chang, S. H. Jo, K. H. Kim, P. Sheridan, S. Gaba, W. Lu, “Synaptic behaviors and modeling of a metal oxide memristor device,” Applied Physics A, 102 pp. 857-863, (2011).

[21] H. Abdalla, and M. D. Pickett, “SPICE Modeling of Memristors,” *ISCAS* (2011).

[22] S. Shin, K. Kim, and S.-M. Kang, "Compact Models for Memristors Based on Charge-Flux Constitutive Relationships," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 29, no. 4, pp. 590-598, April 2010.

[23] C. Yakopcic, A. Sarangan, J. Gao, T. M. Taha, G. Subramanyam, and S. Rogers, “TiO_2 Memristor Devices,” IEEE National Aerospace & Electronics Conference, July 2011.

[24] C. Yakopcic, E. Shin, T. M. Taha, G. Subramanyam, P. T. Murray, and S. Rogers, “Fabrication and Testing of Memristive Devices,” *International Joint Conference on Neural Networks (IJCNN)*. (2010).

[25] C. Yakopcic, T. M. Taha, E. Shin, G. Subramanyam, P. T. Murray, and S. Rogers, "Memristor Fabrication and Characterization; An Adaptive Coded Aperture Imaging & Sensing Opportunity," *SPIE Adaptive Coded Aperture Imaging and Non-Imaging Sensors*. (2010).

[26] A. S. Oblea, A. Timilsina, D. Moore, and K. A. Campbell, "Silver Chalcogenide Based Memristor Devices," *IJCNN*, (2010).

[27] J. J. Yang, M. D. Pickett, X. Li, D. A. A. Ohlberg, D. R. Stewart and R. S. Williams, "Memristive switching mechanism for metal/oxide/metal nanodevices," *Nature Nanotechnology*, 3, 429–433 (2008).

[28] K. Miller, K. S. Nalwa, A. Bergerud, N. M. Neihart, and S. Chaudhary, "Memristive Behavior in Thin Anodic Titania," *IEEE Electron Device Letters* 31(7), (2010).

[29] K. Miller, "Fabrication and modeling of thin-film anodic titania memristors," *Master's Thesis*, Iowa State University, Electrical and Computer Engineering (VLSI), Ames, Iowa, (2010).

[30] F. Miao, J. P. Strachan, J. J. Yang, M.-X. Zhang, I. Goldfarb, A. C. Torrezan, P. Eschbach, R. D. Kelley, G. Medeiros-Ribeiro, R. S. Williams, "Anatomy of a Nanoscale Conduction Channel Reveals the Mechanism of a High-Performance Memristor," Advanced Materials, vol. 23, no. 47, pp. 5633-5640, Nov. 2011.

[31] C. Yakopcic, T. M. Taha, G. Subramanyam, and S. Rogers, "Memristor-based Readout Integrated Circuits," SPIE Adaptive Coded Aperture Imaging and Non-Imaging Sensors V, August 2011.

[32] C. Yakopcic, R. Hasan, T. M. Taha, M. R. McLean, and D. Palmer, "Efficacy of Memristive Crossbars for Neuromorphic Processors," IEEE International Joint Conference on Neural Networks (IJCNN), 2014.

[33] C. Yakopcic and T. M. Taha, "Energy Efficient Perceptron Pattern Recognition Using Segmented Memristor Crossbar Arrays," IEEE International Joint Conference on Neural Networks (IJCNN), August 2013.

[34] T. M. Taha, R. Hasan, C. Yakopcic, and M. R. McLean, "Exploring the Design Space of Specialized Multicore Neural Processors," IEEE International Joint Conference on Neural Networks (IJCNN), August 2013.

[35] C. Yakopcic, R. Hasan, T. M. Taha, M. R. McLean, D. Palmer, "Memristor-based neuron circuit and method for applying a learning algorithm in SPICE," *Electronics Letters*. (Accepted)

[36] M. D. Pickett, D. B. Strukov, J. L. Borghetti, J. J. Yang, G. S. Snider, D. R. Stewart, and R. S. Williams, "Switching dynamics in titanium dioxide memristive devices," *Journal of Applied Physics* 106(074508), (2009).

[37] J. G. Simmons, “Generalized Formula for the Electric Tunnel Effect between Similar Electrodes Separated by a Thin Insulating Film,” *Journal of Applied Physics*, 34(6), (1963).

[38] A. Khoshakhlagh, "Design of a Readout Integrated Circuit (ROIC) for Infrared Imaging Applications," Master's Thesis, University of New Mexico, (2010).

[39] P. M. Ferreira, J. G. R. C. Gomes, and A. Petraglia, “Current Mode Read-out Circuit for Infrared Photodiode Applications in 0.35µm CMOS,” SBCCT, (2008).

[40] R. M. Philipp, D. Orr, V. Gruev, J. Van der Spiegel, R. Etienne-Cummings, “Linear Current-Model Active Pixel Sensor,” IEEE Journal on Solid State Circuits 42(11),(2007).

[41] X. Wang, W. Wong, and R. Hornsey, “A High Dynamic Range CMOS Image Sensor With Inpixel Light-to-Frequency Conversion,” IEEE Transactions on Electron Devices 53(12), (2006).

[42] A. Hairston, J. Stobie, and R. Tinkler, “Advanced Readout Circuit Signal Processing,” Infrared Technology and Applications, Proc. of SPIE Vol. 6206, 62062Z, (2006).

[43] T. A. Chesler, "Design Of Pixel Level CMOS Readout Circuitry For Continuous Bias Uncooled Bolometric Long Wave Infrared Focal Plane Arrays," Master's Thesis University of Maryland, (2004).

[44] C.J. Lin, S.H. Kang, Y.J. Wang, K. Lee, X. Zhu, W.C. Chen, X. Li, W.N. Hsu, Y.C. Kao, M.T., Liu, W.C. Chen, Y. C. Lin, M. Nowak, and N. Yu, L. Tran., “45nm low power CMOS logic compatible embedded STT MRAM utilizing a reverse-connection 1T/1MTJ cell,” *IEDM*, 2009

[45] G. Sun, X. Dong, Y. Xie, J. Li, and Y. Chen, "A novel architecture of the 3D stacked MRAM L2 cache for CMPs," In Proc. HPCA 2009, pp. 239-249.

[46] W. Xu, H. Sun, X. Wang, Y. Chen, T. Zhang, "Design of Last-Level On-Chip Cache Using Spin-Torque Transfer RAM (STT RAM)," IEEE Transactions on Very Large Scale Integration (VLSI) Systems, vol.19, no.3, pp.483-493, March 2011.

[47] J. Wang, X. Dong, Y. Xie, "OAP: An obstruction-aware cache management policy for STT-RAM last-level caches," Design, Automation & Test in Europe Conference & Exhibition (DATE 2013), pp.847-852, March 2013.

[48] H. Sun, C. Liu, W. Xu, J. Zhao, N. Zheng, and T. Zhang, "Using Magnetic RAM to Build Low-Power and Soft Error-Resilient L1 Cache," IEEE Transactions on Very Large Scale Integration (VLSI) Systems, vol.20, no.1, pp.19-28, Jan. 2012.

[49] M. Rasquinha, D. Choudhary, S. Chatterjee, S. Mukhopadhyay, and S. Yalamanchili, "An energy efficient cache design using Spin Torque Transfer (STT) RAM," ACM/IEEE International Symposium on Low-Power Electronics and Design (ISLPED), pp.389-394, August 2010.

[50] Z. M. Zeng, P. Khalili Amiri, G. Rowlands, H. Zhao, I. N. Krivorotov, J.-P. Wang, J. A. Katine, J. Langer, K. Galatsis, K. L. Wang, and H. W. Jiang, "Effect of resistance-area product on spin-transfer switching in MgO-based magnetic tunnel junction memory cells," Appl. Phys. Lett. 98, 072512 (2011).

[51] H.-S. P. Wong, S. Raoux, S. Kim, J. Liang, J. P. Reifenberg, B. Rajendran, M. Asheghi, and K. E. Goodson, "Phase Change Memory," Proceedings of the IEEE, vol. 98, no. 12, pp. 2201-2207, Dec. 2010.

[52] M. Zhu, L. Wu, Z. Song1 F. Rao, D. Cai, C. Peng, X. Zhou, K. Ren, S. Song, B. liu, and S. Feng, "$Ti_{10}Sb_{60}Te_{30}$ for phase change memory with high-temperature data retention and rapid crystallization speed," Applied Physics Letters, 100(12), 2012.

[53] Y. Wang, C. Zhang, H. Yu, and W. Zhang, "Design of Low Power 3D Hybrid Memory by Non-volatile CBRAM-Crossbar with Block-level Data-retention," In Proc. ISLPED, 2012, pp. 197-202.

[54] Y. Wang, C. Zhang, R. Nadipalli, H. Yu, and R. Weerasekera, "Design Exploration of 3D Stacked Non-Volatile Memory by Conductive Bridge based Crossbar," IEEE International 3D Systems Integration Conference, Jan. 31 - Feb. 2, 2012.

[55] S. Yu and H.-S. P. Wong, "Compact Modeling of Conducting-Bridge Random-Access Memory (CBRAM)" IEEE Trans. on Electron Devices, 58(5), May, 2011.

[56] S. Yu, and H.-S. P. Wong, "A Phenomenological Model for the Reset Mechanism of Metal Oxide RRAM," IEEE Electron Device Letters, 31(12), Dec. 2010.

[57] S. Dietrich, M. Angerbauer, M. Ivanov, D. Gogl, H. Hoenigschmid, M. Kund, C. Liaw, M. Markert, R. Symanczyk, L. Altimime, S. Bournat, and G. Mueller, "A Nonvolatile 2-Mbit CBRAM Memory Core Featuring Advanced Read and Program Control," IEEE Journal of Solid-State Circuits, 42(4), April, 2007.

[58] D. Niu, C. Xu, N. Muralimanohar, N. P. Jouppi, and Y. Xie, "Design Trade-Offs for High Density Cross-Point Resistive Memory" In Proc. ISLPED, 2012, pp. 197-202.

[59] J. Liang, S. Yeh, S. S. Wong, and H. -S. P. Wong, "Scaling Challenges for the Cross-point Resistive Memory Array to Sub-10nm Node – An Interconnect Perspective," IEEE International Memory Workshop, May, 2012, pp. 1-4

[60] W. Fei, H. Yu, W. Zhang, and K. S. Yeo, "Design Exploration of Hybrid CMOS and Memristor Circuit by New Modified Nodal Analysis," IEEE Trans. on VLSI Systems, 20(6), June, 2012.

[61] K.-H. Kim, S. Gaba, D. Wheeler, J. M. Cruz-Albrecht, T. Hussain, N. Srinivasa, and W. Lu, “A Functional Hybrid Memristor Crossbar-Array/CMOS System for Data Storage and Neuromorphic Applications,” Nano Letters, 12(1), 2012, pp. 389-395.

[62] G. S. Rose, R. Pino, Q. Wu, “A low-power memristive neuromorphic circuit utilizing a global/local training mechanism,” in *Proc. IJCNN*, pp.2080-2086, 2011.

[63] G. S. Rose, R. Pino, Q. Wu, “Exploiting memristance for low-energy neuromorphic computing hardware,” in *Proc. ISCAS*, 2011, pp. 2942-2945.

[64] H. Wang and H. Li, "Memristor-based Synapse Design and Training Scheme for Neuromorphic Computing Architecture," IEEE World Congress on Computational Intelligence, June, 10-15, (2012).

[65] E. Lehtonen and M. Laiho, “CNN using memristors for neighborhood connections,” feb. 2010, pp. 1 –4.

[66] Y.V. Pershin, M. Di Ventra, “Experimental demonstration of associative memory with memristive neural networks,” *Neural Networks*, vol. 23, no. 7, pp. 881-886, Sep. 2010.

[67] K. D. Cantley, A. Subramaniam, H. J. Stiegler, R. A. Chapman, and E. M. Vogel, “Neural Learning Circuits Utilizing Nano-Crystalline Silicon Transistors and Memristors” IEEE Transactions on Neural Networks and Learning Systems 23(4), Apr. 2012.

[68] G. Snider, “Instar and outstar learning with memristive nanodevices,” *Nanotechnology*, vol. 22, no. 1, 2011.

[69] Z. Vasilkoski, H. Ames, B. Chandler, A. Gorchetchnikov, J. L´veill´, G. Livitz, E. Mingolla, and M. Versace, “Review of stability properties of neural plasticity rules for implementation on memristive neuromorphic hardware,” in *Proc IJCNN*, 2011, pp. 2563–2569.

[70] A. Wu and Z. Zeng, "Dynamic Behaviors of Hybrid Lotka-Volterra Recurrent Neural Networks with Memristor Characteristics," IEEE World Congress on Computational Intelligence, June, 10-15, (2012).

[71] Z. Guo, J. Wang, and Z. Yan, “Attractivity Analysis of Memristor-Based Cellular Neural Networks With Time-Varying Delays,” IEEE Transactions on Neural Networks and Learning Systems, (accepted).

[72] F. Corinto, A. Ascoli, and M. Gilli, "Memristor models for chaotic neural circuits," IEEE World Congress on Computational Intelligence, June, 10-15, (2012).

[73] G. Zhang and Y. Shen, “New Algebraic Criteria for Synchronization Stability of Chaotic Memristive Neural Networks With Time-Varying Delays,” IEEE Transactions on Neural Networks and Learning Systems, 24(10) Oct. 2013.

[74] M. Pd. Sah, C. Yang, H. Kim and L. O. Chua, "Memristor Circuit for Artificial Synaptic Weighting of Pulse Inputs," IEEE ISCAS 2012, pp. 1604-1607.

[75] S. P. Adhikari, C. Yang, H. Kim, and L. O. Chua, "Memristor Bridge Synapse-Based Neural Network and Its Learning," IEEE Transactions on Neural Networks and Learning System 23(9), Sept. 2012, pp. 1426-1435.

[76] S. H. Jo, T. Chang, I. Ebong, B. B. Bhadviya, P. Mazumder, and W. Lu, “Nanoscale Memristor Device as Synapse in Neuromorphic Systems,” *Nano Letters*, 10 (2010).

[77] J. A. Pérez-Carrasco, C. Zamarreño-Ramos, T. Serrano-Gotarredona, and B. Linares-Barranco, “On Neuromorphic Spiking Architectures for Asynchronous STDP Memristive Systems,” in *Proc. ISCAS*, 2010, pp.1659-1662.

[78] I. Ebong, and P. Mazumder, “Memristor based STDP learning network for position detection,” in *Proc. ICM,* 2010, pp. 292-295.

[79] C. Zamarreño-Ramos, L. A. Camuñas-Mesa, J. A. Pérez-Carrasco, T. Masquelier, T. Serrano-Gotarredona, and B. Linares-Barranco, “On b nspike-timing-dependent-plasticity, memristive devices, and building a self-learning visual cortex,” Frontiers in Neuroscience, Neuromorphic Engineering, vol. 5, pp. 1-22, Article 26, Mar. 2011.

[80] D. Chabi, W. Zhao, D. Querlioz, and J.-O. Klein, “Robust neural logic block (NLB) based on memristor crossbar array,” in *Proc. NANOARCH*, 2011, pp.137-143.

[81] M. Soltiz, D. Kudithipudi, C. Merkel, G. S. Rose, and R. E. Pino, “Memristor-Based Neural Logic Blocks for Nonlinearly Separable Functions,” IEEE Transactions on Computers, 62(8), Aug. 2013.

[82] M. Hu, H. Li, Q. Wu, G. S. Rose, and Y. Chen, "Memristor Crossbar Based Hardware Realization of BSB Recall Function," IEEE World Congress on Computational Intelligence, June, 10-15, (2012).

[83] S. Afshar, O. Kavehei, A. van Schaik, J. Tapson, S. Skafidas, and T. J. Hamilton, "Emergence of Competitive Control in a Memristor-Based Neuromorphic Circuit," IEEE World Congress on Computational Intelligence, June, 10-15, (2012).

[84] F. Alibart, E. Zamanidoost, and D.B. Strukov, "Pattern classification by memristive crossbar circuits with ex-situ and in-situ training", Nature Communications, 4:2072, Jun. 2013.

[85] A. M. Sheri, H. Hwang, M. Jeon, and B. Lee, "Neuromorphic Character Recognition System With Two PCMO Memristors as a Synapse," IEEE Transactions on Industrial Electronics, 61(6), Jun. 2014.

[86] H. Manem, J. Rajendran, and G. S. Rose, "Stochastic Gradient Descent Inspired Training Technique for a CMOS/Nano Memristive Trainable Threshold Gate Array," IEEE Transactions on Circuits and Systems I, 59(5), May 2012.

[87] F. Merrikh-Bayat, S. B. Shouraki, and A. Rohani, "Memristor Crossbar-Based Hardware Implementation of the IDS Method," IEEE Transactions on Fuzzy Systems, 19(6), Dec. 2011.

[88] G. Howard, E. Gale, L. Bull, B. de Lacy Costello, and A. Adamatzky, "Evolution of Plastic Learning in Spiking Networks via Memristive Connections," IEEE Transaction on Evolutionary Computation, 16(5) Oct. 2012.

[89] M. Hu, H. Li, Y. Chen, Q. Wu, G. S. Rose, and R. W. Linderman, "Memristor Crossbar-Based Neuromorphic Computing System - A Case Study," IEEE Transactions on Neural Networks and Learning Systems (accepted).

[90] F. Merrikh-Bayat and S. B. Shouraki, "Memristive Neuro Fuzzy System" IEEE Transactions on Cybernetics, 43(1), Feb. 2013.

[91] K. D. Cantley, A. Subramaniam, H. J. Stiegler, R. A. Chapman, and E. M. Vogel, "Hebbian Learning in Spiking Neural Networks With Nanocrystalline Silicon TFTs and Memristive Synapses" IEEE Transactions on Nanotechnology, 10(5), 2011.

[92] C. Yakopcic, T. M. Taha, G. Subramanyam, R. E. Pino, and S. Rogers, "Analysis of a Memristor based 1T1M Crossbar Achitecture," IEEE International Joint Conference on Neural Networks (IJCNN), August 2011 (Invited paper).

[93] K. W. Zhang, S. B. Long, Q. Liu, H. B. LU, Y. T. Li, Y. Wang, W. T. Lian, M. Wang, S. Zhang, and M. Liu, "Progress in rectifying-based RRAM passive crossbar array," *Science China Technological Sciences* 54, 1–8 (2011).

[94] D. Niu, Y. Chen, C. Xu, Y. Xie, "Impact of Process Variations on Emerging Memristor," in Proc. 47th ACM/IEEE Design Automation Conference, pp. 877-882, (2010).

[95] K-H. Jo, C-M. Jung K-S. Min, and S-M. Kang, "Self-adaptive write circuit for low-power and variation-tolerant memristors" IEEE Transactions on Nanotechnology, 9(6), pp. 675-678, Nov. 2010.

[96] H. Manem, G. S. Rose, X. He, and W. Wang, "Design Considerations for Variation Tolerant Multilevel CMOS/Nano Memristor Memory," in Proc. 20th Symposium on VLSI (GLSVLSI), (2010).

[97] J. Rajendran, Maenm, R. Karri, G. S. Rose, "An Approach to Tolerate Process Related Variations in Memristor-Based Applications," 24th international conference on VLSI design, pp. 18-23, (2011).

[98] G. S. Rose, Y. Yao, J. M. Tour, A. C. Cabe, N. Gergel-Hackett, N. Majumdar, J. C. Bean, L. R. Harriott, M. Stan. "Designing CMOS/Molecular Memories While Considering Device Parameter Variations" ACM Journal on Emerging Technologies in Computing Systems, 3(1) Apr. 2007, pp. 1-24.

[99] M. Hu, H. Li, Y. Chen, X. Wang, R. E. Pino, "Geometry Variations Analysis of TiO2 Thin-Film and Spintronic Memristors," In 16th Asia and South Pacific Design Automation Conference, pp. 25-30, (2011).

[100] D. Chabi, W. Zhao, D. Querlioz, J-O. Klein, "Robust Neural Logic Block (NLB) based on Memristor Crossbar Array," IEEE/ACM ISNA, pp. 137-143 (2011).

[101] D. Querlioz, O. Bichler, C. Gamrat, "Simulation of a Memristor-Based Spiking Neural Network Immune to Device Variations," International Joint Conference on Neural Networks, pp. 1775-1781 (2011).

[102] C. Yakopcic, T. M. Taha, G. Subramanyam, and S. Rogers, "Multiple Memristor Read and Write Circuit for Neuromorphic Applications," *International Joint Conference on Neural Networks (IJCNN)*, (2011).

[103] W. Lu, K.-H. Kim, T. Chang, S. Gaba, "Two-terminal resistive switches (memristors) for memory and logic applications," in Proc. 16th Asia and South Pacific Design Automation Conference, 2011, pp. 217-223.

[104] C.-N. Peng, C.-W. Wang, T.-C. Chan, W.-Y. Chang, Y.-C. Wang, H.-W. Tsai, W.-W. Wu, L.-J. Chen and Y.-L. Chueh, "Resistive switching of Au/ZnO/Au resistive memory: an in situ observation of conductive bridge formation," Nanoscale Research Letters, 7:559, 2012, pp. 1-6.

[105] Chen TP, M. Tse S, Sun CQ, Fung S, Lo KF, "Snapback behaviour and its similarity to the switching behaviour in ultra-thin silicon dioxide films after hard breakdown," J Phys D: Appl Phys 2001, 34:95–98.

[106] J. A. McNeill, J. Stander, and C. A. Raanes, "A CMOS analog integrated circuit for detector readout at a 50MHz pixel rate," 22nd International Congress on High Speed Photography and Photonics, SPIE vol. 2869 (1996).

[107] O. Kavehei, S. Al-Sarawi, K.-R. Cho, K. Eshraghian, and D. Abbott, "An Analytical Approach for Memristive Nanoarchitectures," IEEE Trans. on Nanotechnology 11(2), March 2012, 374-385.

[108] A. Heittmann and T. G. Noll, "Limits of Writing Multivalued Resistances in Passive Nanoelectronic Crossbars Used in Neuromorphic Circuits," GLSVLSI 2012 227-232.

[109] Y. Wang, C. Zhang, H. Yu, and W. Zhang, “Design of Low Power 3D Hybrid Memory by Non-volatile CBRAM-Crossbar with Block-level Data-retention,” In Proc. ISLPED, 2012, pp. 197-202.

[110] K. Eshraghian, K. –R. Cho, O. Kavehei, S. –K. Kang, D. Abbott, and S. –M. S. Kang, “Memristor MOS Content Addressable Memory (MCAM) Hybrid Architecture for Future High Performance Search Engines,” IEEE Trans. On VLSI Systems, vol. 19, no. 8, 1407–1417, Aug. 2011.

[111] X. Dong, C. Xu, S. Member, Y. Xie, and N. P. Jouppi, “NVSim: A Circuit-Level Performance, Energy, and Area Model for Emerging Nonvolatile Memory,” IEEE Trans. on Computer Aided Design of Integrated Circuits and Systems, 31(7), July, 2012, pp. 994-1007.

[112] S. H. Jo, K.-H. Kim, and W. Lu, "High-Density Crossbar Arrays Based on a Si Memristive System" Nano Letters, 9(2), 2009, pp. 870-874.

[113] A. M. Grishin, A. A. Velichko, and A. Jalalian, "Nb2O5 nanofiber memristor," Applied Physics Lett. 103, 053111, (2013).

[114] A. J. Lohn, J. E. Stevens, P. R. Mickel, and M. J. Marinella, "Optimizing TaOx memristor performance and consistency within the reactive sputtering forbidden region," Applied Physics Lett. 103, 063502, (2013).

[115] Y.-E. Syu, T.-C. Chang, J.-H. Lou, T.-M. Tsai, K.-C. Chang, M.-J. Tsai, Y.-L. Wang, M. Liu, and S. M. Sze, “Atomic-level quantized reaction of HfOx memristor,” Appl. Phys. Lett. 102, 172903 (2013).

[116] M.N. Kozicki, M. Balakrishnan, C. Gopalan, C. Ratnakumar, and M. Mitkova, "Programmable Metallization Cell Memory Based on Ag-Ge-S and Cu-Ge-S Solid Electrolytes," Non-Volatile Memory Technology Symposium, 2005, pp. 83-89.

[117] N. Maeda, S. Komatsu, M. Morimoto, K. Tanaka, Y. Tsukamoto, K. Nii, and Y. Shimazaki, "A 0.41 μA Standby Leakage 32 kb Embedded SRAM with Low-Voltage Resume-Standby Utilizing All Digital Current Comparator in 28 nm HKMG CMOS," IEEE Jounal of Solid State Physics, 48(4), April 2013.

[118] M. R. McLean, "Concurrent Learning Algorithm and the Importance Map", Network Science and Cybersecurity Advances in Information Security Volume 55, 2014, pp 239-250

[119] S. Gevorgian and H. Berg, "Line Capacitance and Impedance of Coplanar-Strip Waveguides on Substrates with Multiple Dielectric Layers," 31st European Microwave Conference, 2001.

[120] B. Han, and T. M. Taha, “Acceleration of spiking neural network based pattern recognition on NVIDIA graphics processors,” Journal of Applied Optics, 49(101), pp. 83-91, 2010.

[121] J. M. Nageswaran, N. Dutt, J. L. Krichmar, A. Nicolau, and A.Veidenbaum, "Efficient simulation of large-scale spiking neural networks using CUDA graphics processors," In Proceedings of the 2009 international joint conference on Neural Networks (IJCNN). IEEE Press, Piscataway, NJ, USA, 3201-3208, 2009.

[122] T. M. Taha, P. Yalamanchili, M. Bhuiyan, R. Jalasutram, C. Chen, and R. Linderman, "Neuromorphic algorithms on clusters of PlayStation 3s," International Joint Conference on Neural Networks (IJCNN), pp.1-10, 18-23 July 2010.

[123] T. Chen, Y. Chen, M. Duranton, Q. Guo, A. Hashmi, M. Lipasti, A. Nere, S. Qiu, M. Sebag, O. Temam, "BenchNN: On the Broad Potential Application Scope of Hardware Neural Network Accelerators," IEEE International Symposium on Workload Characterization (IISWC), November 2012.

[124] P. Dubey, "Recognition, mining and synthesis moves computers to the era of tera," Technology@Intel Magazine, Feb. 2005.

[125] H. Esmaeilzadeh, A. Sampson, L. Ceze, and D. Burger, "Neural Acceleration for General-Purpose Approximate Programs," International Symposium on Microarchitecture (MICRO), 2012.

[126] H. Esmaeilzadeh, A. Sampson, L. Ceze, and D. Burger, "Towards Neural Acceleration for General-Purpose Approximate Computing," Workshop on Energy Efficient Design (WEED), 2012.