Titre: Title:	Design and Characterization of a CMOS-Based Long Retention Time Memristive Cell
Auteur: Author:	Hussein Assaf
Date:	2022
Type:	Mémoire ou thèse / Dissertation or Thesis
Référence: Citation:	Assaf, H. (2022). Design and Characterization of a CMOS-Based Long Retention Time Memristive Cell [Thèse de doctorat, Polytechnique Montréal]. PolyPublie. https://publications.polymtl.ca/10479/

POLYTECHNIQUE MONTRÉAL

affiliée à l'Université de Montréal

Design and characterization of a CMOS-based long retention time memristive cell

HUSSEIN ASSAF

Département de génie électrique

Thèse présentée en vue de l'obtention du diplôme de *Philosophiæ Doctor*

Génie électrique

Août 2022

POLYTECHNIQUE MONTRÉAL

affiliée à l'Université de Montréal

Cette thèse intitulée :

Design and characterization of a CMOS-based long retention time memristive cell

présentée par **Hussein ASSAF**

en vue de l'obtention du diplôme de *Philosophiæ Doctor*

a été dûment acceptée par le jury d'examen constitué de :

Yves AUDET, président

Yvon SAVARIA, membre et directeur de recherche

Mohamad SAWAN, membre et codirecteur de recherche

François LEDUC-PRIMEAU, membre

Ricardo IZQUIERDO, membre externe

DEDICATION

À Mes chers parents,

À Toute ma famille,

À Tous ceux qui me sont chers,

À Tous mes amis,

Je dédie le fruit de mes 30 ans d'études.

ACKNOWLEDGEMENTS

I would to thank my dear professor Mohamad Sawan who gave me the chance to pursue my Ph.D. at Polytechnique Montreal. I would like to thank, as well, professor Yvon Savaria who believed in me during the most critical moments of my doctorate. In addition, I am grateful to IVADO for their continuous financial support. I am also grateful to CMC Microsystems for their help in chips fabrication and testing. Moreover, I would like to acknowledge my family's support and patience until I earned this achievement. I do not want to forget thanking my close friends and colleagues, particularly Dr. Abbas Hammoud.

RÉSUMÉ

Malgré les promesses de la dernière décennie, les memristors ne sont pas encore des dispositifs utilisables en pratique pour les applications récentes qui nécessitent une immense puissance de calcul. Le dispositif, que Leon Chua a découvert en 1971 et que Hewlett Packard a fabriqué pour la première fois en 2008, manque encore de robustesse et de fiabilité dans son processus de fabrication. Chaque memristor peut être caractérisé par deux caractéristiques principales: (a) la memristance qui est la fenêtre de résistance et (b) la durabilité qui est souvent appelée temps de rétention. Pourtant, les memristors, qui sont également connus comme des dispositifs de commutation résistifs, sont confrontés à de multiples défis qui empêchent les chercheurs de les utiliser dans leurs applications finales.

Les défis les plus courants pour un memristor sont la contrôlabilité de la fenêtre de résistance, le dopage des ions, les caractéristiques physiques des matériaux constitutifs, la nano-échelle et quelques autres défis. Heureusement, les émulateurs de memristors peuvent remplacer le dispositif réel dans plusieurs applications modernes. Ces émulateurs sont basés sur un processus de fabrication CMOS éprouvé. Cependant, l'inconvénient commun de ces émulateurs est leur incapacité à conserver leurs états internes pendant de longues périodes.

Dans ce travail, nous présentons une cellule memristive qui non seulement possède les caractéristiques du memristor comme la fameuse courbe d'hystérésis I-V pincée, mais qui possède également un temps de rétention de 10 ans. Pour réaliser une telle cellule, nous avons utilisé des transistors à grille flottante pour la rétention des charges et un émulateur de memristor pour que le circuit présente un comportement similaire à celui des memristors.

Une puce de 1 mm^2, que nous avons fabriquée en utilisant un processus CMOS standard de 65 nm avec une fine épaisseur de diélectrique de grille, a été utilisée pour caractériser les mécanismes de charge et décharge des dispositifs à grille flottante. Nos expérimentations en laboratoire ont permis de découvrir des faits intéressants sur les transistors à grille flottante réalisés avec un procédé CMOS à 65 nm. Par exemple, nous avons découvert que l'effet tunnel des charges commence à environ 350 mV, et qu'à environ 650 mV, le dispositif perd

sa capacité à piéger les charges. Néanmoins, et malgré la sensibilité du processus, les charges qui ont été délicatement tunnelisées dans les dispositifs à grille flottante ont été (et sont toujours) piégées à l'intérieur de leurs circuits.

Dès lors, nous avons combiné en un seul circuit le modèle que nous avons découvert pour les dispositifs à grille flottante et un circuit d'émulation de memristor que nous avons adapté de la littérature pour correspondre aux conditions de fonctionnement des dispositifs à grille flottante. Le circuit résultant est notre circuit memristif qui a montré sa robustesse et sa fiabilité lors de la simulation du circuit.

Nous proposons ainsi une cellule memristive avec quelques architectures de circuit utiles pour le calcul résistif. En particulier, nous présentons le circuit dans un petit réseau 4x4 adapté à la multiplication vecteur-matrice. La simulation de ce réseau a montré l'autonomie de chaque cellule et la possibilité de la programmer à un autre état simplement en contrôlant les tensions de deux terminaux de la cellule memristive proposée.

ABSTRACT

Despite being actively researched in the past decade, memristors still lack practical realizations that could be used to implement usable devices. The device, which Leon Chua discovered in 1971 and Hewlett Packard fabricated for the first time in 2008, is still missing robustness and reliability in its manufacturing process. Any memristor can be characterized by two main features: (a) memristance, which is the device's resistance range and (b) durability, which is often referred to as retention time. Nonetheless, the memristors, also known as resistive switching devices, are facing multiple challenges that prevent researchers from using them in their end applications.

The most common challenges for a memristor are memristance window controllability, ion doping, physical characteristics of the composing material and nano-scalability. Luckily, memristor emulators are good substitutes, for the time being, in most modern applications. Many memristor (or resistive switching device) emulators are based on a mature CMOS process. However, the common setback for these emulators lies in them lacking the ability to retain their internal states for long periods.

In this dissertation we present a memristive cell that not only has the memristor's characteristics as the famous I-V pinched hysteresis curve but also has a 10-year retention time. To bring forward such a cell, we used floating gate transistors for charge trapping and a memristor emulator for a memristor-like behavior of the circuit.

We fabricated a 1 mm^2 chip using a standard 65 nm CMOS process with thin-gate dielectric thickness that was used to characterize both charging and discharging mechanisms of the floating-gate devices. Our in-laboratory experiments uncovered interesting facts about floating-gate transistors in the 65 nm process. For example, we found that charge tunneling starts at around 350 mV gate-source potential. However, at around 650 mV, the device loses its ability to trap charges. Nonetheless, and despite process sensitivity, the charges that were delicately tunneled into floating-gate devices continue to be trapped inside their circuits.

Henceforth, we integrated into one complete circuit a model we discovered for floating-gate

devices and a memristor emulating circuit which was adapted from literature to match the operating conditions of floating-gate devices. The resulting memristive circuit showed robustness and reliability during circuit simulations.

We propose a cell with some circuit architectures that are useful for resistive computing. Particularly, we present the circuit in a small 4×4 crossbar array suitable for vector-matrix multiplication. This simulated array showed the autonomy of every cell and the ability to program it to a different state only by controlling the terminal voltages of the proposed two-terminal memristive cell.

TABLE OF CONTENTS

DEDICATION iii

ACKNOWLEDGEMENTS iv

RÉSUMÉ v

ABSTRACT vii

TABLE OF CONTENTS ix

LIST OF TABLES xiii

LIST OF FIGURES xiv

LIST OF SYMBOLS AND ABBREVIATIONS xix

LIST OF APPENDICES xxi

CHAPTER 1 INTRODUCTION 1

1.1 From AI to Resistive Computing 1

1.2 Resistive Switching Device Characteristics 2

1.3 RSD Applications 3

1.4 The Need for RSD Emulating Circuits 4

1.5 Research Objectives 4

1.6 Research Contributions 5

1.7 Outline of the Thesis 6

CHAPTER 2 LITERATURE REVIEW 7

2.1 Resistive Switching Devices 7

2.1.1 Definition 7

2.1.2 Metrics 8

2.1.3 Comparing Existing RSDs . . . 10
2.1.4 RSD Emulating Circuits . . . 10
2.2 Resistive Computing . . . 11
2.2.1 Definition . . . 11
2.2.2 In-Memory Computing . . . 12
2.2.2.1 Computing with the Resistive Processing Unit . . . 13
2.2.2.2 The Dot Product Engine . . . 13
2.3 Data Retention in RSDs . . . 13
2.3.1 Floating Gate Transistors . . . 13
2.3.1.1 Definition . . . 13
2.3.1.2 FGT Charging and Discharging . . . 14
2.3.1.3 FGT Applications . . . 15
2.4 Summary . . . 16

CHAPTER 3 ARTICLE 1: VECTOR MATRIX MULTIPLICATION USING CROSS-BAR ARRAYS: A COMPARATIVE ANALYSIS . . . 17
3.1 Introduction . . . 18
3.2 Resistive Processing Unit . . . 19
3.3 The Dot Product Engine . . . 20
3.4 Results . . . 23
3.5 Discussion . . . 25
3.6 Conclusion . . . 26

CHAPTER 4 ARTICLE 2: MEMRISTOR EMULATORS FOR AN ADAPTIVE DPE ALGORITHM: COMPARATIVE STUDY . . . 27
4.1 Introduction . . . 28
4.2 The Adaptive Dot Product Engine . . . 30
4.3 Memristor Emulators . . . 33
4.4 Preliminary Simulation Results . . . 35
4.5 Conclusion . . . 36

CHAPTER 5 ARTICLE 3: IMPLEMENTING FLOATING GATE TRANSISTORS WITH A THIN GATE DIELECTRIC 65 nm CMOS TECHNOLOGY 38
5.1 Introduction 39
5.2 Methodology 40
5.2.1 Individual Transistor Cells 41
5.2.2 Capacitive Cells 43
5.3 Results 45
5.3.1 Individual Transistor Cells 46
5.3.1.1 Transistor Response 46
5.3.1.2 Tunneling Voltage for $1V$ model MOSFET 47
5.3.1.3 Substrate Break-Down Voltage 49
5.3.1.4 Oxide BreakDown Voltage 49
5.3.1.5 Tunneling Independence on MOSFET Width 51
5.3.2 Capacitive Cells 52
5.3.2.1 VCO-based Verification 52
5.3.2.2 Retention Time 54
5.4 Model 55
5.5 Behavioural Hypotheses 57
5.6 Conclusion 58

CHAPTER 6 THE PROPOSED MEMRISTIVE CELL 59
6.1 High-Level Architecture 59
6.2 The Memristor Emulator 60
6.3 FGT Model 62
6.4 The Control Circuitry 64
6.4.1 The Pulse Train Generator Circuit 64
6.4.2 The Comparator 65
6.4.3 Generating the Control Signals 68
6.5 Detailed Architecture of the Memristive Cell 70
6.6 Summary 73

CHAPTER 7 INTEGRATING THE PROPOSED MEMRISTIVE CELL IN ANALOG COMPUTING FOR ARTIFICIAL INTELLIGENCE 74
7.1 Vector Matrix Multiplication 74
7.1.1 VMM using Complete Memristive Cells 75
7.1.2 VMM using Shared Memristor Emulator 76
7.2 Configuration 77
7.2.1 Weight Update 77
7.2.2 Programming Procedure for Precomputed Weights 78
7.2.3 Programming Procedure for Weight Learning 78
7.2.4 Design Consideration for $V_{bias}(t)$ 79
7.3 A Proof of Concept 81
7.4 A Design Verification Chip 85
7.5 Summary 86

CHAPTER 8 GENERAL DISCUSSION 87

CHAPTER 9 CONCLUSIONS AND RECOMMENDATIONS 91
9.1 Summary of the Work Done 91
9.2 Summary of Contributions 92
9.2.1 Publications 92
9.3 Limitations 93
9.4 Future Research 93

REFERENCES 94

APPENDICES 107

LIST OF TABLES

Table 2.1	Comparison of some existing RSDs according to references [1, 2] . . .	12
Table 3.1	Comparison of Analog Approaches versus Ideal Software [3–5]	24
Table 4.1	Different window functions used for memristor	30
Table 4.2	Comparison of different memristor models	32
Table 4.3	Selected memristor circuit emulators for the ADPE algorithm since 2014 and their characteristics	34
Table 5.1	Characteristics of the transistors used in the fabricated circuits	40
Table 5.2	The extracted parameters to apply in eq. (5.2) for estimating the gate voltage of MOSFET $M2$ in Fig. 5.4	57
Table 6.1	Truth table for generating the control signals in Fig. 6.7	67

LIST OF FIGURES

Figure 1.1 Percentage approximation of RSD integration in different domains . . 3

Figure 2.1 An example of the RSD cross-talk noise in crossbar arrays 9

Figure 2.2 The memristor kit developed by KNOWM: (a) Three 8×1 memristor chips with dimensions 4.3 $mm \times 2.1$ mm per die (b) The memristor board for AHAH computing with dimensions 18.3 $mm \times 22.35$ mm (c) PC connecting board (d and e) Two programming boards 11

Figure 2.3 An example of a circuit containing floating gate transistors 14

Figure 3.1 Resistive Processing unit (a) RPU base circuit as designed in [3] (b) RPU System in [6] 19

Figure 3.2 The four circuit variables showing the memristor link [7] 20

Figure 3.3 A circuit that resembles the memristor in action [7] 21

Figure 3.4 A DPE grid as suggested by [4]. Different colors shows different memristance values according to their conductances. Peripheral circuitry is responsible for controlling the input voltages and output currents . . 22

Figure 3.5 Simulated Memristor Characteristics using MATLAB Simulink (a) The input sinusoidal voltage in V (b) The current response of the device in mA (c) The change in resistance value according to the input voltage in $k\Omega$ (d) The time vector applied to the experiment in s (e) Variation of doped area with input voltage in nm (f) Memristor R-V characteristic curve (g) Memristor I-V hysteresis curve [8–10] 23

Figure 4.1 The memristor 28

Figure 4.2 Response of SPICE simulation with PWL input voltage shows how the internal state of the memristor is preserved while voltage changes; with each new arriving signal the memristance starts varying from its previous value 29

Figure 4.3 An adapted version of the Dot Product Engine (DPE) to solve the problem of vector matrix multiplication (a) Sample Circuit: Voltages v1-v5 are the input voltages. The output voltage is measured at the resistors R1-R5 (b) The targeted output voltage to achieve by adding the voltages across R1-R5 (c) The response per column of the circuit 36

Figure 5.1 The schematics of: (a) The Individual Transistor Cell (ITC) used to characterize the floating transistor *nmos* $M1$ with 3 different width values (b) ITC with a *p-type* MOSFET, $M3$ 42

Figure 5.2 The predicted gate leakage current with respect to the source voltage for various dielectric thicknesses according to [11] 43

Figure 5.3 The schematic of a Capacitive Cell (CC) which is composed of one ITC connected to capacitor $C1$ and to the gate of $M2$ to indirectly measure $C1$'s voltage through V_g 43

Figure 5.4 The schematic of the CC cell connected to two VCOs for indirect measurement $C1$'s voltage through V_g performed by comparing the VCOs' output frequencies 44

Figure 5.5 The schematic of the implemented nine-stage Voltage Controlled Oscillator (VCO) used for voltage to frequency conversion 45

Figure 5.6 An image showing a healthy sample of the fabricated chip under the microscope with its dimensions 46

Figure 5.7 The simulated and measured drain current plots of the 2.5 V MOSFET observed when sweeping the gate voltage from 0 V to 1 V 47

Figure 5.8 Measured tunneled currents in the proposed ITCs when applying a 1 MHz pulse train: (a) $W = 8\ \mu m$ and VDD = 0.8 V, (b) $W = 8\ \mu m$ and VDD = 0.8 V, (c) $W = 2\ \mu m$, VDD = 0.5 V for Trials 1 and 4, and VDD = 0.8 V for Trials 2 and 3, and (d) $W = 4\ \mu m$ and VDD = 0.8 V 48

Figure 5.9 An example of one the damaged test chips before and after the stress test: (a) before (b) after 50

Figure 5.10 Variation of the tunneled currents interpolated from real-time measurements on all FGT variants vs the amplitude of the pulse train 52
Figure 5.11 Curves of measured and simulated VCO frequency output signals for CC in Fig. 5.4 with respect to voltage sweeps for (a) Vg , and (b) for amplitudes of the applied pulse-train 52
Figure 5.12 Picture of the test setup showing the drain current measured by ammeter $A1$ from Fig. 5.3 55
Figure 5.13 A general model representing the variation of the tunneled current vs the amplitude of the pulse train interpolated from physical measurements on floating gate transistors 55
Figure 6.1 The general architecture of the proposed memristive cell 59
Figure 6.2 The schematic of the used memristor emulator which is adapted from [12] into 65 nm CMOS process 60
Figure 6.3 Simulation results of the memristor emulator in Fig. 6.2 showing both the pinched and the cyclical hysteresis loops when 0 V was applied on terminal B and a 1 MHz sinusoidal signal on A with: (a and b) 200 mV amplitude, (c and d) 400 mV amplitude 61
Figure 6.4 The basic FGT cell in our chips: (a) schematic (b) behavioral model . 62
Figure 6.5 Computer simulation using VerilogAMS for the model in Fig. 6.4(b) . 62
Figure 6.6 Schematic of the FGT cell used in computer simulation where the dashed components are replaced from the original circuit with VerilogAMS code 63
Figure 6.7 The PT generation circuit where the signals $Ctrl\ \{1-4\}$ are generated by some logic circuitry that will be explained later in this chapter . . 64
Figure 6.8 The PT output of the circuit presented in Fig. 6.7 65
Figure 6.9 The comparator with hysteresis used in our chips 66
Figure 6.10 DC Response of the comparator with hysteresis used in our chips . . 68
Figure 6.11 Transient Response of the comparator with hysteresis used in our chips 69
Figure 6.12 Decision circuitry that is responsible for generating the control signals of the pulse generator circuitry presented in Fig. 6.7 69

Figure 6.13 The detailed architecture of the proposed memristive cell which includes all sub-circuits explained in this chapter where each comparator is a copy of that in Fig. 6.9 70
Figure 6.14 Transient response of the proposed memristive cell presented in Fig. 6.13 72
Figure 7.1 Schematic of one crossbar layer where at every crossbar intersection is the memristive cell of Fig. 6.13 75
Figure 7.2 Schematic of the VMM layer shown in Fig. 7.1 optimized for density 76
Figure 7.3 Transient current responses of the proposed memristive cell presented in Fig. 6.13 for different column bias-voltages swept between 0 V and 400 mV 80
Figure 7.4 Transient state responses of the proposed memristive cell presented in Fig. 6.13 for different column bias-voltages swept between 0 V and 400 mV 80
Figure 7.5 Transient input voltages, $V_i(t)$ and $V_{bias,j}(t)$, to the 4x4 crossbar array 82
Figure 7.6 Transient input voltages, V_{R_1} and V_{R_2} (see Fig. 6.13), to the comparators of the memristive cells located in the first column of the 4x4 array for a $V_{bias,1}$ of 0 V and different values of $V_i(t)$ as shown in Fig. 7.5 . 82
Figure 7.7 Transient FGT capacitor voltage responses of the memristive cells located in the first column of the 4x4 array for a $V_{bias,1}$ of 0 V and different $V_i(t)$ as shown in Fig. 7.5 83
Figure 7.8 Transient input voltages, V_{R_1} and V_{R_2} (see Fig. 6.13), to the comparators of the memristive cells located in the first row of the 4x4 array for the same $V_1(t)$ and different values of $V_{bias,j}$ as shown in Fig. 7.5 . . . 83
Figure 7.9 An image of the second fabricated chip in 65 nm: (a) A microscopic photograph after fabrication (b) CAD layout 85
Figure A.1 Chip Overview 107
Figure A.2 Single NMOS transistors 108
Figure A.3 Single PMOS transistors 109
Figure A.4 Matched transistors using common centroid method 109

Figure A.5 FG cells with an NMOS for the FG transistor and variable capacitance values 110
Figure A.6 FG cell with an NMOS for the FG transistor and a capacitance value of 50fF attached to a VCO to match the cells based on frequency . . 111
Figure A.7 FG cell with an NMOS for the FG transistor and a PMOS transistor to substitute a capacitor of capacitance value $10fF$ 112
Figure A.8 FG cell with a PMOS for the FG transistor and a capacitance value of 50 fF 112
Figure A.9 Different variants of the SX cells 113
Figure A.10 The used comparator in the chip 114
Figure A.11 The first memristive cell with a 50 fF capacitance 114
Figure A.12 The second memristive cell with a 50 fF capacitance for the SX emulator and a capacitance of 50 fF or 200 fF for the FG cell 115

LIST OF SYMBOLS AND ABBREVIATIONS

AI	Artificial Intelligence
CMOS	Complementary Metal-Oxide Semiconductor
FPGA	Field Programmable Gate Array
RSD	Resistive Switching Device
VMM	Vector Matrix Multiplication
FGT	Floating Gate Transistor
CAD	Computer Aided Design
$O(n)$	Order of n
NN	Neural Network
W	Set of weights in NN
ΔW	Weight update in NN
α	Learning rate in NN
RPU	Resistive Processing Unit
q	Charge (electron)
ϕ	Flux
CCVS	Current Controlled Voltage Source
VCCS	Voltage Controlled Current Source
DPE	Dot Product Engine
G	RSD conductance in crossbar array
1T1M	1 transistor 1 memristor
ADPE	Adaptive Dot Product Engine
$M(r)$	Memristance
PWL	Piece Wise Linear
EP	Equilibrium Propagation
CPU	Central Processing Unit
GPU	Graphical Processing Unit
EEPROM	Electrically Erasable Programmable Read Only Memory

FPAA	Field Programmable Analog Array
ITC	Individual Transistor Cell
CC	Capacitive Cell
VCO	Voltage Controlled Oscillator
GPIB	General Purpose Interface Bus
RAM	Random Access Memory
MOSFET	Metal–Oxide–Semiconductor Field-Effect Transistor
C_{ox}	Oxide capacitance
PT	Pulse Train
TCM	Thermal-Chemical Mechanism
VCM	Valance Change Mechanism
ECM	Electro-Chemical Mechanism

LIST OF APPENDICES

Appendix A Chip 1 Description: Support Material for Chapter 5 107

CHAPTER 1 INTRODUCTION

1.1 From AI to Resistive Computing

The quest for improving Artificial Intelligence (AI) algorithms has been an active research goal ever since Alan Turing proposed the Turing test in 1950 [13]. This test allows an investigator to distinguish a computer from a human through their responses to a given challenge [14].

Consequently, as the complexity of AI Algorithms increases, the corresponding computer and hardware architectures must complement each other in order to fulfill the required intelligence in their end applications. Unfortunately, improving hardware architectures is not a straightforward task with the successive down-scaling of technologies over the past decades [15]. Despite that technology down-scaling increases transistor density per unit area, the number of transistors per application is still limited by Moore's law [16, 17].

A brute force solution for some advanced and complex AI algorithms, like Equilibrium Propagation [18] and AlphaGo which may lead to as many as 10^{700} theoretical possible outcomes, is difficult using any recent computer in feasible time [19]. Such algorithms are usually solved via heuristics and implemented in digital environments; either by using programming languages like Python [20], R [21] and MATLAB [22] or by using digital hardware architectures such as Field Programmable Gate Arrays (FPGAs) [23].

Recently, some researchers shifted to implementing mixed analog and digital learning architectures in order to accelerate AI algorithms [24]. Resistive switching devices (RSDs) (such as memristors [9], Resistive Random Access Memories (RRAMs) [25], Resistive Processing Units (RPUs) [26], etc...) are fundamental for these analog architectures. Basically, these devices are arranged in a crossbar structure to increase parallelism. Each resistive device can be programmed to a certain resistance when placed in the crossbar structure, where this resistance (conductance) affects the current traversing the circuit, and consequently, might change the final outcome. It is important that every RSD be able to retain its conductance value during the inference phase, or even during the learning process that could last for sev-

eral hours. Every RSD should also be capable of holding to its value until (during and even after) later prediction and deployment phases. The maximum reported memristor's retention time is around 10 years [27].

1.2 Resistive Switching Device Characteristics

The main features that characterize RSDs are:

(i) **Retention time:** Defined by the maximum time to which an RSD can hold to its conductance value. This time can vary from a few milliseconds to a couple hours [27].

(ii) **Conductance range:** Defined by the resistance window that bounds the device's range of operation. It may vary from a couple ohms to mega ohms.

(iii) **Voltage range:** Every RSD has two phases of operation which are SET and READ. During the SET phase, the RSD is in the programming mode, during which it is programmed to a certain resistance. Whereas during the READ phase, the RSD resistance value is read indirectly through the current that traverses the RSD. Since most RSDs are bipolar, the voltage across the device's terminals determines its mode of operation. As a result, this voltage range is important to characterize.

(iv) **Dimensions:** Defined by an RSD's length and width. The smaller the device, the more scalable the crossbar structure is.

(v) **Robustness and Reliability:** Depend on the technology used in RSD fabrication or, in other words, on the physical structure of the device's material. If the fabrication process is reliable and robust, the resulting RSD can be used at a large scale.

Every manufacturing process has its own advantages and disadvantages. When considering nano-scale devices, overcoming their disadvantages is a more challenging task due to the required nano-scale dimensions of devices, particularly for miniature devices with new physical prototypes. Some RSDs, precisely memristors, are more challenging to fabricate than others [28]. These challenges include (but are not limited to): accuracy in reading and programming, low yield in crossbar arrays, doping level, device dimensions, large fan-in and fan-out

in arrayed structures, etc. [28–30]. As a result of the mentioned fabrication challenges, the RSD becomes less reliable, hence the need for RSD implementation in a robust and reliable technology is fundamental, particularly if dedicated for Vector Matrix Multiplication (VMM). Moreover, since the retention time is a valuable characteristic for RSDs, the device can be implemented in a reliable CMOS process using Floating Gate Transistors (FGTs) to achieve retention times that could last for 10 years.

1.3 RSD Applications

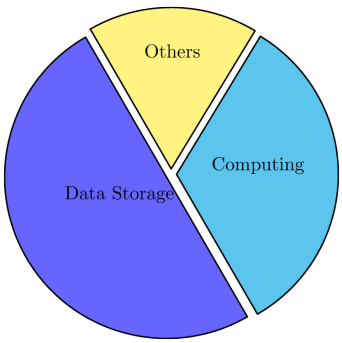


Figure 1.1 Percentage approximation of RSD integration in different domains

The concept of resistive switching elements is not new, as will be emphasized in later chapters, and so is the concept of non-volatile memories which started at a low scale in 1987 [31]. However, having both concepts merged into one single device was a novel step in 2008 [9]. Over the years, devices implementing both concepts were, and ought to remain, integrated into several applications which can be categorized into 3 main categories:

(a) Computing: Includes all categories of Neural Networks (NNs), resistive computing, in-memory computing, etc.

(b) Data storage: Includes storing the data passively or actively, such as in near-memory computing, and for either long or short periods of time.

(c) Others: such as the realization of logic gates, radio-frequency switches, etc.

Figure 1.1 shows an approximation of RSD percentage-usage in each of the three domains as inferred from references [32, 33]. The percentages reported in the figure resemble the number of RSD usages per domain of application with respect to the total number of RSD usages in all domains.

1.4 The Need for RSD Emulating Circuits

RSD fabrication processes are numerous, and researchers are still progressing towards fabricating the best possible robust RSD [34]. There are multiple challenges facing these fabrication processes, such as: (a) electrode placement and material selection, (b) RSD medium selection, (c) machine precision, and (d) cell size requirements per application. Until the majority of these challenges are overcome, other circuitry emulating the behavior of an RSD are necessary. Such circuitry is usually implemented using existing conventional robust and reliable devices that are fabricated in a recent standard technology. Advanced CMOS processes are the closest viable option, as they can fulfill most of the requirements for an RSD.

1.5 Research Objectives

In this research, we aimed at improving analog implementation of AI using a reliable RSD that is fabricated in a robust CMOS technology. Our main focus was overcoming two main challenges facing existing RSDs:

(i) Lack of an existing robust and reliable RSD defined by its I-V characteristic curve and by its long retention time.

(ii) Lack of an accurate model for computer simulation that matches the behavior of the physical devices.

1.6 Research Contributions

The main contributions of this research are:

(I) **Characterizing Current Tunneling in FGTs**

The purpose of this phase was to find a reliable CMOS process to implement and characterize current tunneling in FGT devices. Thus, we managed to characterize FGT's behavior in a standard 65 nm CMOS process with thin gate dielectric. In addition, we discovered a behavioral model that was common among all tested transistors. We also believe that the discovered model applies to more advanced technologies to which securing access is difficult for researchers.

(II) **Modeling and Simulating an Automated Memristive Cell**

The purpose of this phase is to establish an FGT model based on the measurement results from phase (I). Finding a simulation model for FGTs in CAD software is an important step towards designing and simulating a memristive cell that deploys both the aforementioned model and a memristor emulator adapted from the literature to match FGT's operating conditions. The resulting memristive cell ought to have the famous pinched hysteresis I-V curve in addition to a long retention time. The cell also includes additional circuitry responsible for: (a) connecting the FGT model to the memristor emulator and (b) controlling FGT's charging and discharging mechanisms.

(III) **Putting the Resulting Memristive Cell in Action**

In this phase, we aim at deploying the proposed memristive cell in the resistive computing domain. We demonstrate that the proposed memristive cell can be organized into crossbar arrays, which are essential structures for vector matrix multiplication. We show how our proposed memristive cell can be used in a 4x4 array with variable input voltages across each of the two-terminal cells. Each row of cells shares the same input voltage on one terminal, whereas the second terminal of each cell shares another voltage with other cells in the same column.

Published Articles In the process, we published two articles that are embedded in this dissertation and one journal paper that is submitted to IEEE Transactions on Electron Devices, and that passed the first round of reviews. Our three articles are:

(i) Assaf, H., Savaria, Y., & Sawan, M. (2018, December). Vector Matrix Multiplication Using Crossbar Arrays: A Comparative Analysis. In 2018 25th IEEE International Conference on Electronics, Circuits and Systems (ICECS) (pp. 609-612). IEEE.

(ii) Assaf, H., Savaria, Y., & Sawan, M. (2019, March). Memristor emulators for an adaptive dpe algorithm: Comparative study. In 2019 IEEE International Conference on Artificial Intelligence Circuits and Systems (AICAS) (pp. 13-17). IEEE.

(iii) Assaf, H., Savaria, Y., Ali, M., Nabavi, M., & Sawan, M. (2022, May). Implementing floating gate transistors with a thin gate dielectric 65 nm CMOS technology. Submitted to IEEE Transactions on Electron Devices.

Articles in Development Additionally, an article titled: *A memristive cell with high endurance and long retention time for vector matrix multiplication: From hardware to simulation* will be submitted to Nature Electronics.

1.7 Outline of the Thesis

Chapter 1 presented a brief introduction to the ultimate goal of this research, and the rest of this dissertation is organized as follows: In chapter 2, we introduce existing literature that inspired our work, focusing on key concepts such as vector matrix multiplication and resistive computing. Additionally, we present existing efforts towards achieving a memristor and a brief review on floating-gate devices. Chapters 3 and 4 present two published articles to enrich the reader's knowledge about different topics covered in this dissertation. In chapter 5, we present an article that contributes to the characterization of floating-gate devices implemented in a standard 65 nm CMOS process. Based on the lessons learned in this article, we introduce our proposed memristive cell in chapter 6 highlighting its major components. In chapter 7, we provide a proof of concept simulation for the memristive cell used in a 4×4 crossbar array that can be used in resistive computing. In the following chapter, chapter 8, we present a general discussion for the whole project before concluding the dissertation in chapter 9.

CHAPTER 2 LITERATURE REVIEW

It is clear, without a doubt, that researchers have paved the way for Resistive Switching Devices (RSDs) towards competing in Artificial Intelligence (AI) applications [35]. The concept of a programmable RSD was first introduced, in 1971, by Leon Chua as the *memristor* [36]. In 1976, the same concept was explained to the finest details [37].

2.1 Resistive Switching Devices

2.1.1 Definition

An RSD is a programmable resistor capable of adjusting its internal resistance based on the voltage applied across its terminals [15]. In theory, RSDs are either bipolar or unipolar devices that possess a resistance switching property. In other words, every device that is able to change its electric resistance is named an RSD [34].

RSDs are getting famous recently due to their capability of being deployed in multiple domains such as: (a) representation of binary states in digital logic, (b) information storage, (c) computation near memory, (d) computation in memory and (e) Deep neural networks (DNNs) [15, 25, 33, 34]. The flexibility that RSDs can add to the end application is directly related to their distinguishing features which are:

(a) Scalability: Related to RSD's dimensions which may vary in size from microscopic scales (in *nano* and *micro* meters) to macrosopic scales. For most applications, RSDs are supposed to fit into dense layers of crosspoint arrays [38].

(b) Fast transition between states: As the name suggests, an RSD belongs to the resistor family where the current-voltage relationship is linear, and should not be time consuming.

In this chapter, we present a literature review on different topics that are under study in this thesis. For certain topics, we will point the reader to some sections in chapters 3, 4, and 5. These chapters correspond to our three published/submitted articles mentioned in the introduction. We preferred to keep the mentioned sections as parts of their respective chapters and only point the reader to the relevant sections to avoid repetitions.

(c) Endurance: Defined by the device's lifetime. It can also be defined by the number of possible programming cycles for the RSD. This depends mainly on RSD's physical composition.

(d) State retention: Defined by the amount of time during which an RSD can hold to its value without losing its state.

(e) Switching mechanism: Although RSDs have fast transitions between states, yet, it is important to discover the switching mechanism for every RSD in order to obtain the fastest response possible. This is important because the response time is a critical metric for most applications, especially those that can exploit RSDs such as the DNNs where Vector Matrix Multiplications (VMMs) are expensive [34]. The most common RSD switching mechanisms are: (i) Thermal-Chemical Mechanism (TCM), (ii) Valance Change Mechanism (VCM) and (iii) Electrochemical Metallization (ECM) [25].

2.1.2 Metrics

When assessing and comparing different RSDs, several metrics should be taken into consideration. The significance of every metric varies from one RSD to another depending on RSD's physical formation and domain of interest. The main RSD assessment metrics are:

(a) State/Data Retention: As explained before.

(b) Endurance: As explained before.

(c) Disturbance/Noise: Several factors might affect the programming and reading processes of an RSD. Usually, RSDs are organized in crossbar arrays to increase their output efficiency. However, the cross-talk between neighboring devices is one of the most common noise sources for RSDs. Some RSDs can have up to eight neighboring cells, in which case the sum of currents due to cross-talk noise might be equivalent to the targeted current traversing the RSD. In addition, other noise factors as instrumental and environmental errors, circuit precision and parasitic capacitance are critical, and can cloud the read/write process in an RSD, especially in cases where the targeted currents are in *nano-amperes* [39].

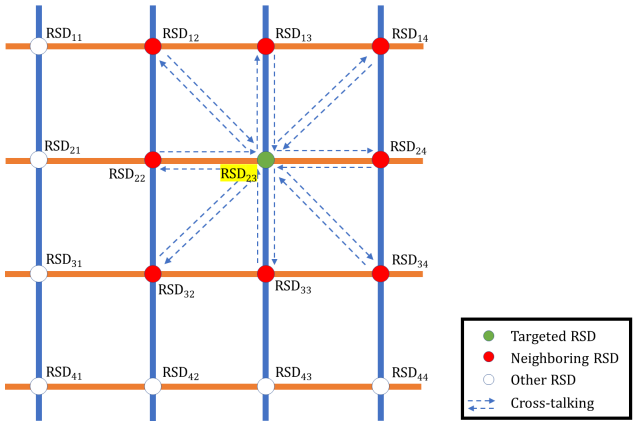


Figure 2.1 An example of the RSD cross-talk noise in crossbar arrays

(d) Precision: RSDs are multi-state devices, thus depending on RSD's precision, the number of states that an RSD can have for a certain application varies. For example, binary RSDs cannot be considered for analog memory applications, yet they are fit for digital applications. Not only does RSD precision depend on its resistance range (R_{on} and R_{off}) but also on other circuit components that will communicate with it. Thus, an RSD should (or has the flexibility to) match the precision of other circuit components.

(e) Size: With the recent advances in technology, having a macroscopic RSD device does not satisfy most applications requirements. Most crossbar structures demand nano-scaled devices for their final architectures [40, 41]. Consequently, the smaller an RSD can be, the more fitting it will be for modern applications.

(f) Power Consumption: The vast spread of RSDs along with their integration into dense and complicated circuits requires low power consumption per device [42]. If the current-behavior in end-applications is non-linear, it is more challenging to deploy RSDs in 1T1RSD (1-Transistor-1-RSD) cell configuration. In 1T1RSD cells, the transistors are usually used as a cell enable/disable switch which reduces crossbar's power consumption [34].

On the other hand, although an RSD has a linear I-V relationship similar to resistors, it cannot replace the latter yet. Usually, a conventional resistor dissipates more power than an RSD. This is mainly due to the material composition of each device. In conventional

resistors, the I-V curve is linear and non-cyclic since it has only one resistance state which renders the device passive. However, an RSD has a cyclic I-V curve which defines two modes of operation: active and passive. In the first mode (active), the RSD is programmed phase whereas during the second mode (passive), the RSD conducts with the programmed-resistance. In this respect, a conventional resistor is supposed to be made of materials that are more conducting than those composing an RSD whose behavior depends mostly on ion-doping in semi-conductors [43].

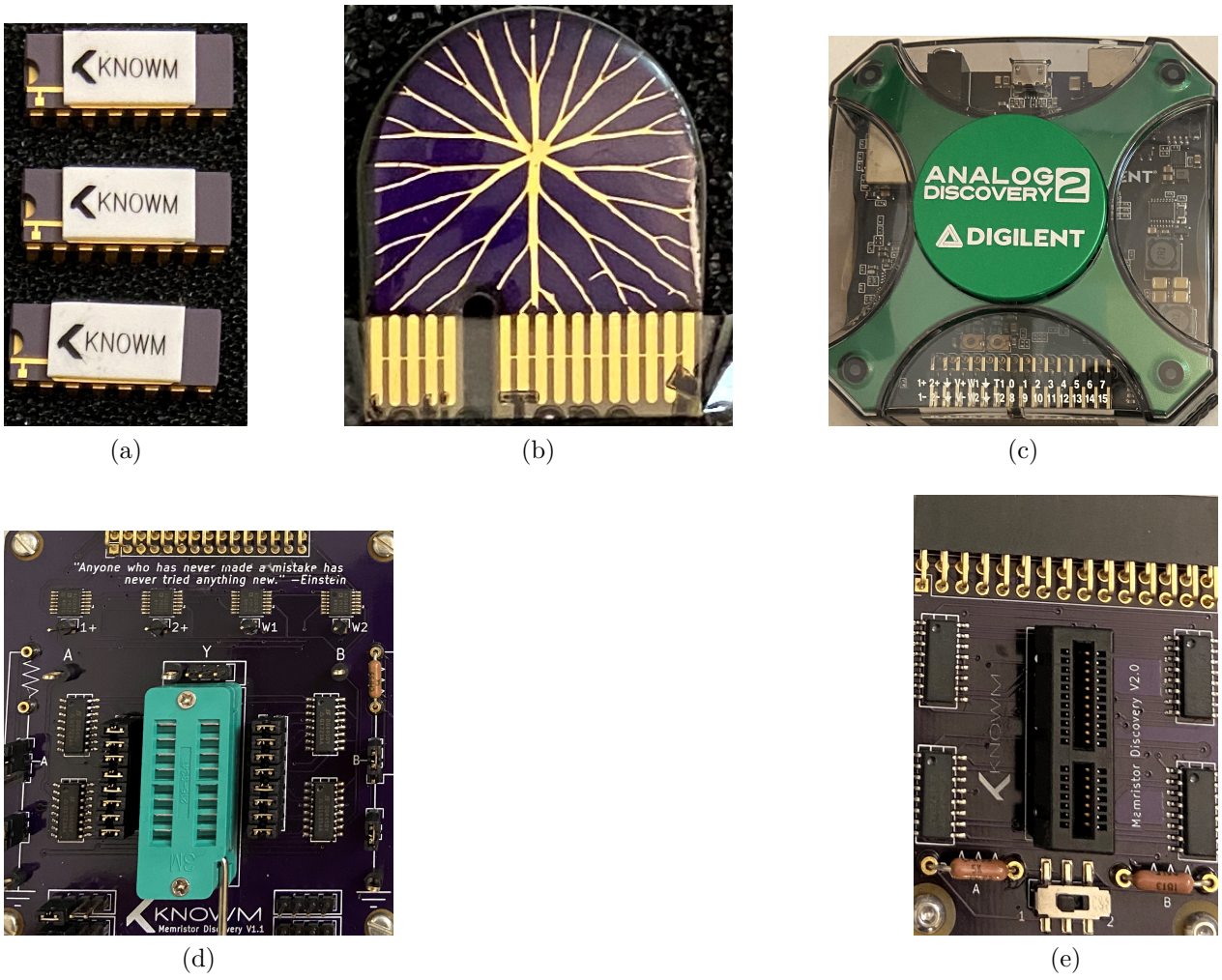
2.1.3 Comparing Existing RSDs

In the literature, RSDs can have different names based on either their end-application or on material composition. The most famous name for an RSD is the *memristor*, which stands for *memory-resistor*, as defined by Leon Chua in 1971 [36]. Later, after deploying RSDs in multi-bit memory applications the name was generalized into *Resistive Random Access Memory (RRAM)* [44]. RSDs can have other names such as: *Conductive-Bridging Random Access Memory (CBRAM)* [45] and *Phase-change Memory (PCM)* [46]. Chapter 4 contains a more elaborate explanation on memristors.

Table 2.1 compares some existing RSD devices based on the metrics mentioned before. For most RSDs of Table 2.1 partially reported the characteristics of their devices, particularly the size. Nonetheless, almost all these RSDs have nano-scale dimensions. In addition, Fig. 2.2 shows an example of the memristor kit developed by KNOWM.

2.1.4 RSD Emulating Circuits

Memristors are the most common existing RSDs. For an elaborate discussion on recent memristor emulating circuits, please refer to section 4.3 of this document.



(a) (b) (c)

(d) (e)

Figure 2.2 The memristor kit developed by KNOWM: (a) Three 8×1 memristor chips with dimensions 4.3 $mm \times 2.1$ mm per die (b) The memristor board for AHAH computing with dimensions 18.3 $mm \times 22.35$ mm (c) PC connecting board (d and e) Two programming boards

2.2 Resistive Computing

2.2.1 Definition

The simplest definition of resistive computing is the process of utilizing resistors or resistor-like devices in hardware-structures for executing machine instructions [15]. Currently, RSDs are the core devices for resistive computing which is divided into two main categories [32]:

(i) *In-memory computing*: In this type of computing, RSDs are usually organized on cross-

Table 2.1 Comparison of some existing RSDs according to references [1, 2]

Model	**Retention T.**	**Endurance**	R_{on}	R_{off}	**Size**	**Power Cons.**
[9, 47]	years	10^{12} cycles	$\frac{R_{off}}{160}$	$160R_{on}$	$4F^2$	0.1–3 pJ/bit
[48, 49]	≥ 6.4 $hours$	$\geq 10^6$ cycles	2 $k\Omega$	20 $k\Omega$	1 μm^2	1.67 μW
[50]	10 years	5000 cycles	20 $k\Omega$	200 $k\Omega$	-	1 nJ/bit
[51, 52]	-	10^3 cycles	200 Ω	550 Ω	0.25 μm^2	$\sim$ pJ
[53]	-	200 cycles	167 $M\Omega$	10 $G\Omega$	-	200 nJ/bit
[54]	-	$\geq 10^6$ cycles	20 $k\Omega$	283 $k\Omega$	0.1 μm^2	-
[55]	-	10^6 cycles	4 $k\Omega$	100 $k\Omega$	-	-
[56]	-	-	1 $k\Omega$	-	-	-
[57]	2.5 $hours$	10^3 cycles	4.5 $k\Omega$	8.5 $k\Omega$	-	11.7 nJ/bit
[58]	-	≥ 200 cycles	2 $k\Omega$	15 $k\Omega$	-	25 pJ
[59]	-	10^5 cycles	-	-	-	-
[60]	-	10^3 cycles	25 $k\Omega$	150 $k\Omega$	-	61.16 nJ
[61]	-	-	700 Ω	10 $M\Omega$	-	-
[62]	10^4 s	≥ 500 cycles	5 $k\Omega$	50 $k\Omega$	-	-
[63]	-	50 cycles	25 Ω		-	-
[64]	1.2×10^6 s	4000 cycles	10 $k\Omega$	-	-	-
[65]	10^5 s	10^4 cycles	100 $k\Omega$	-	-	-
[66]	-	10^4 cycles	≥ 100 $M\Omega$	≥ 500 $M\Omega$	1 μm^2	7.35 pJ/bit
[67]	-	10^{10} cycles	≥ 10 $k\Omega$	≥ 1 $M\Omega$	-	-

bar arrays as bottlenecks for the currents traversing every column in the array. RSD's conductance play a critical role by controlling the I-V relationship during this process which is often referred to as *Vector Matrix Multiplication (VMM)* (see chapter 3).

(ii) *Near-memory computing*: In this type of computing, RSDs are used for data storage and are placed near the computing devices to reduce latency.

2.2.2 In-Memory Computing

In this document, our main focus is on *in-memory computing*. Many systems were developed based on this concept, we mention hereby two recent systems developed by IBM and HP respectively:

2.2.2.1 Computing with the Resistive Processing Unit

Please refer to section 3.2 for a complete explanation of resistive computing using the Resistive Processing Unit.

2.2.2.2 The Dot Product Engine

Please refer to section 3.3 for a complete explanation of this system.

2.3 Data Retention in RSDs

In this document, we tackle two main setbacks for current RSDs which are device robustness and data retention. As previously mentioned, RSD fabrication processes are facing several challenges that makes the device unreliable. In order to address these challenges, researchers have proposed CMOS circuits to emulate RSD's behavior as discussed in section 2.1.4. Therefore, we believe that the best solution for overcoming RSD's robustness and reliability challenges, until further RSDs are fabricated, is to rely on emulating circuits that can be manufactured using robust and mature CMOS processes.

On the hand, it is now clear from Table 2.1 that existing RSDs vary in retention times, and in some cases, an RSD can preserve its state for years. Since most RSD emulating circuits cannot passively uphold their internal states for long periods, using these emulators demands the existence of a re-configurable CMOS charge trap which are implemented using *Floating Gate Transistors (FGTs)*.

2.3.1 Floating Gate Transistors

2.3.1.1 Definition

By definition, a floating gate transistor is a transistor whose gate is not connected to any active terminal of other circuit components. In other words, the sneak current path from the floating node to ground is blocked by devices with high impedance. For example, transistors M_1 and M_2 in Fig. 2.3 are FGTs, and the connection attaching the gates of both transistors

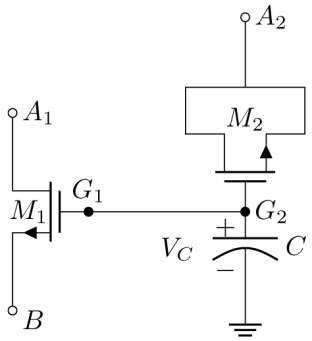


Figure 2.3 An example of a circuit containing floating gate transistors

to C, between nodes G_1 and G_2, is a charge trap whose maximum capacitance is $C+\epsilon$ Farads. A brief historical overview on FGTs can be found in section 5.1.

2.3.1.2 FGT Charging and Discharging

Charging and discharging the FGT's charge trap depend on the mechanism used to tunnel electrons from the outer world to inside the trap or vice versa [68]. This process is often referred to as *current tunneling.*

The potential difference across the MOSFET's gate plays a key role in current tunneling. For example, in Fig. 2.3, the voltages applied on nodes A_1 and B along with V_C are critical to tunnel charges through M_1's gate to G_1G_2. On the other hand, to tunnel charges through transistor M_2's gate, we need to know the voltage on node A_2 with V_C. It is important to notice that the internal FGT's voltage, V_C in Fig. 2.3, cannot be directly controlled from the external world.

The main charge tunneling mechanisms for FGT are:

(i) Channel Hot-Electron injection (CHE): CHE is the most used charge injection mechanism for forcing charges into (and out of) the trap. The lateral electric field created due to source and drain voltages heats-up electrons in the channel, and consequently, increases electron mobility until some electrons gain energy to jump into (or out of) the trap. The main drawbacks of this mechanism are power consumption and large currents [68].

(ii) CHannel Initiated Secondary ELectron current (CHISEL): CHISEL was proposed as an improvement for CHE energy consumption. In CHISEL, multiple electric fields are created, one between MOSFET's source and drain, and another between MOSFET's substrate and its floating gate. The latter is caused by the ionization impact on the drain-substrate contact by applying negative substrate voltage. Therefore, the energy required for an electron to travel across the floating gate in CHISEL is less than that in CHE [68, 69].

(iii) Fowler-Nordheim Tunneling (FNT) current: In FNT, electrons are carefully tunneled through the floating gate dielectric barrier using high voltages and low currents that rises exponentially with the increase in the electric field. The delicacy of FNT makes it more efficient to use than other charge injection mechanisms. However, it was reported that FNT can only be applied to MOSFETs with a gate-dielectric insulator thicker than 5 nm [68, 70].

2.3.1.3 FGT Applications

FGTs are mainly used in the following applications:

- Electrically Erasable Programmable Read Only Memory (EEPROM): EEPROM is one of the oldest architectures for *Non-Volatile Memories (NVM)*. EEPROMs are programmed and erased using CHE and FNT mechanisms, respectively [71].

- Flash memories: These memories are based on EEPROMs and can be divided into types: NOR and NAND. NAND flash arrays are slower, but easier to implement and consume less power. Most data storage applications use the NAND-based arrays whereas NOR arrays are used to store flash data that will be frequently accessed but not updated [71].

- Field Programmable Analog Array (FPAA): FPAAs were first proposed to explore the flexibility of analog circuits based on FGTs. Despite the advances in Field Programmable Gate Arrays (FPGAs) and their large scalability as compared to FPAAs,

the latter arrays make interesting solutions for applications that require long processing-times on FPGAs [72, 73]. However, FPAA manufacturing is not mature yet. In 2018, we ordered some FPAAs from a tier 1 manufacturer at a time they claimed device readiness. We waited for three years before giving up on the devices' arrival.

- Memristive devices: Recently some researchers proposed a memristive cell based on a Y-connected FGT flash cell [74].

2.4 Summary

The flexibility that RSDs add to any application has attracted researchers in both academia and industry. The power of these nanodevices lies first and foremost in their ability to switch resistance based on the applied terminal voltages, and secondly, in their long data retention times. Nonetheless, the progress of RSD integration into recent applications is evolving slowly due to the multiple challenges facing these devices' manufacturing processes. These challenges have led researchers to rely on circuits designed using robust CMOS processes in order to emulate the RSDs. Such emulating circuits succeeded in substituting RSDs in several domains, particularly in resistive computing systems such as in the Resistive Processing Unit (RPU) computing system and in the Dot Product Engine (DPE). However, one common setback remain in the inability to passively preserve states for extended periods of time.

Hence, in this chapter and with reference to material from chapters 3 and 4, we explained the need for RSDs along with their evolution in recent applications. Furthermore, the main concerns of this dissertation were elaborated justifying the need for a robust memristive cell with a long retention time.

CHAPTER 3 ARTICLE 1: VECTOR MATRIX MULTIPLICATION USING CROSSBAR ARRAYS: A COMPARATIVE ANALYSIS

Hussein Assaf[1], Yvon Savaria[1] and Mohamad Sawan[1]
[1]Department of Electrical Engineering, Polytechnique Montreal,
Montreal, QC, Canada
Email: *hussein.assaf@polymtl.ca*

This article discusses some popular approaches to integrate resistive switching devices into vector matrix multiplication focusing on the Resistive Processing Unit (developed by IBM) and on the Dot Product Engine (developed by HP). The article includes a comparison of the two designs highlighting the importance of resistive computing. It summarizes our point of view on analog circuit solutions for vector matrix multiplication at the time of publishing. Although the jury members (who evaluated this thesis) had comments on this article's content, we could not change any of the material presented since we are required to include it in its published version. The article was published at the 2018 25th IEEE International Conference on Electronics, Circuits and Systems (ICECS), pp. 609-612, doi: 10.1109/ICECS.2018.8617942.

Abstract

Vector Matrix Multiplication (VMM) is a demanding operation that exposes a weakness of current digital hardware when applied to Artificial Intelligence (AI) algorithms. Crossbar arrays were proposed as a means to get more effective analog implementations. These arrays are often composed of circuits based on memristors or other custom designs. The Dot Product Engine and the Resistive Processing Units are two proposed solutions. Either design can reduce the training time to $O(1)$ per VMM layer giving a boost to AI and making future requirements more possible.

3.1 Introduction

Recently, crossbar arrays have been proposed as a means to satisfy the needs of super computing blocks with analog domain implementations, since the digital domain shows its limitations when the conventional Moore's law; given by feature size scaling, is saturating [3].

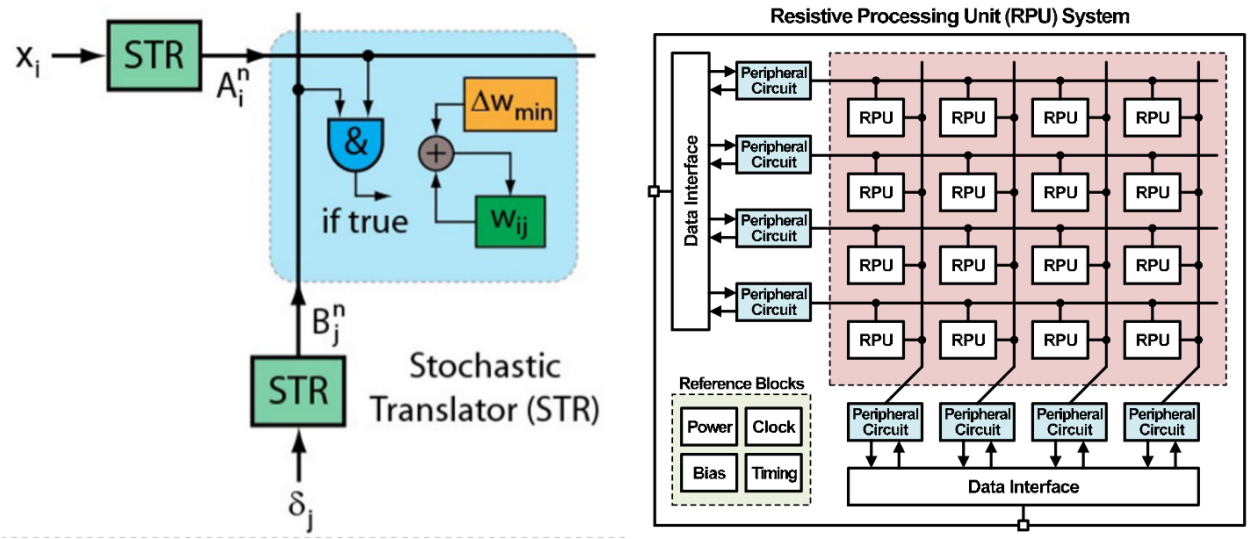
In machine learning, algorithmic functions are often expressed in terms of weights that typically reflect the strength of interconnections among neurons in neural-networks' (NNs) methods or in terms of features' weights (relative significance) in other techniques. These weights are usually represented by W for matrices and w for individual weights. They are indexed with subscript; w_{ij}, corresponding to the weight of each link between neuron i and neuron j. Training W is done by applying some algorithm; like back-propagation in NNs, to a data-set. The data-set is typically divided into three non equal parts: training, testing and validation sets. Each of these sets consists of several tuples; i.e. a vector of numbers or classes observed simultaneously and constitute one example for the subject under study at a certain instance.

Training requires matrix multiplication after each iteration to update W according to the following equation:

$$W = W + \alpha \Delta W \tag{3.1}$$

where ΔW is the weight update after the current iteration and α is the learning rate. The most expensive step in digital NNs is Vector Matrix Multiplication (VMM).

Circuit simulations differ from digital processor calculations where every number must be converted into voltage, current, resistance, capacitance or even inductance. Usually, designs are based on different assumptions. Recent studies have tried to simplify VMM through some analog designs that transform it from an operation of the order $O(n^2)$ to the order of $O(1)$, where in one single step VMM can be achieved. However different cross-point devices have been suggested over the past decade: Resistive Processing Units (RPUs), memristors or compound gates. Analog hardware implementation of VMM showed advantages over a digital based design. First, it provides faster results, although digital circuits can run at high clock rates (in the GHz range). Second, some analog circuit techniques consume less power than their digital competitors based on both: CMOS technology used and operating frequency [73].



(a) (b)

Figure 3.1 Resistive Processing unit (a) RPU base circuit as designed in [3] (b) RPU System in [6]

In this paper, we will compare two different approaches highlighting their respective advantages in simulating VMM. Sections 3.2 and 3.3 explain the RPU and DPE approaches respectively and section 3.4 provides some preliminary results. Our discussion and comparison results are reported in section 3.5 before presenting the conclusion in section 3.6.

3.2 Resistive Processing Unit

A resistive processing unit (RPU) is a small circuit that fits at the intersection of wires in a crossbar array. In [3], the weight update equation was simplified into a simple *and* gate operation with addition. This transformation from multiplication into a linear *and* operation leads to the circuit of Fig. 3.1(a) and is represented by equation (3.2)

$$w_{ij} = w_{ij} + \Delta w_{min} \sum_{n=1}^{BL} A_i^n \wedge B_j^n \tag{3.2}$$

where A and B are the two input bits and BL is the bit stream length, $\{i, j\}$ are row and column indices and n is the matrix dimension [3]. This approach assumes that the input tuples propagated to each layer are converted into voltages applied on both column and row

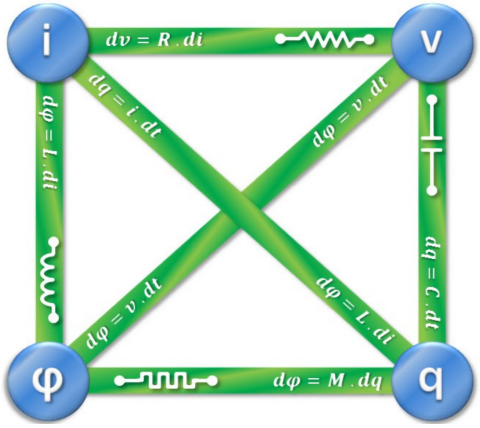


Figure 3.2 The four circuit variables showing the memristor link [7]

wires of the cross bar array. The output is then collected as a current on the column bars, then converted using peripheral circuitry into voltages applied to the next layer and so on. Every machine learning approach basically consists of two main cycles: training and weight update based on the error revealed. The former is a VMM operation and it requires the weights to be stored somewhere in the circuit for later participation in the multiplication process. This necessitates recognizing the difference between multiplication and update cycles. During update, the error at the output layer is propagated towards weights that are subjected to change based on the calculated error. In this case, the process consists of three cycles: forward and backward propagations then weights' update according to the back-propagation algorithm. Each of the operations occur at a different clock cycle, where the weight update is performed according to the relative amplitude of input signals with respect to a reference voltage; V_s. The pulsing at bit-lines is done in a manner to prevent coincidence of pulses, where voltage for each pulse varies between $-V_s/2$ and $V_s/2$, hence the *and* operation delivers an output between $-V_s$ to V_s. The output voltage must be above $V_s/2$ in absolute value for the update to be considered [3].

3.3 The Dot Product Engine

A memristor; the fourth circuit component proposed by Leon Chua in [36], is a resistor capable of saving its current resistance state based on the voltage across its two terminals. The

memristor correlates the charge q to the flux ϕ (Fig. 3.2). Figure 3.3 gives as an example a circuit to demonstrate the concept of a memristor, where the CCVS is a current controlled voltage source and the VCCS is a voltage controlled current source. A memristor has two modes of operation: programming and operation. During programming, it is subjected to a certain input voltage across its terminals, which drives it towards a corresponding resistance according to its characteristic curve. The memristor operates on this resistance as long as possible before changing it based on the next programming voltage potential across the memristor's terminals. Memristors are usually designed from semi conducting chemical materials; like titanium dioxide [7]. Different models exist for memristors based on the controlling window function and material used. The inner matter is divided into two portions: doped part and un-doped one. The mobility of the doped atoms (holes) from one region to another determines the resistance of the device. This mobility is highly influenced by the

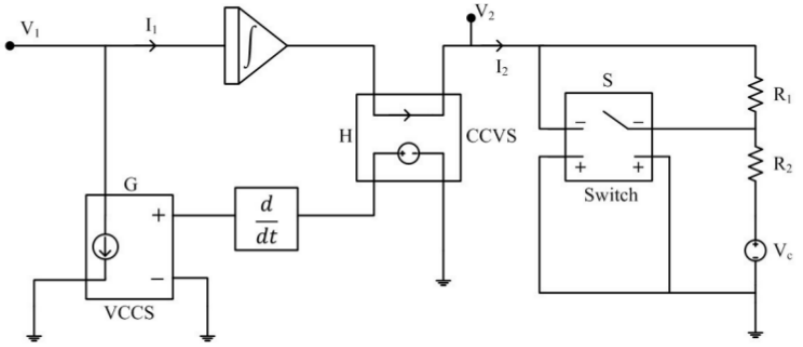


Figure 3.3 A circuit that resembles the memristor in action [7]

position of the boundary separating both regions. As the doped region increases, resistance increases and vice versa. In 2008, Hewlett Packard designed the first memristor after Chua's paper [4, 7]. Since then the company is trying to integrate memristor into different designs; one is the Dot Product Engine (DPE). The DPE is one of the design approaches used to implement a VMM using memristors. Similar to the RPU, input data is translated into voltages and then collected as currents at the columns level (Fig. 3.4). Each layer is composed of several interconnected memristors, with every device having its own resistance value. W is proportionally related to the conductance G defined by $G = \frac{1}{R}$ where R is the resistance. On each node of the crossbar in Fig.4, there is a memristor cell made of one memristor

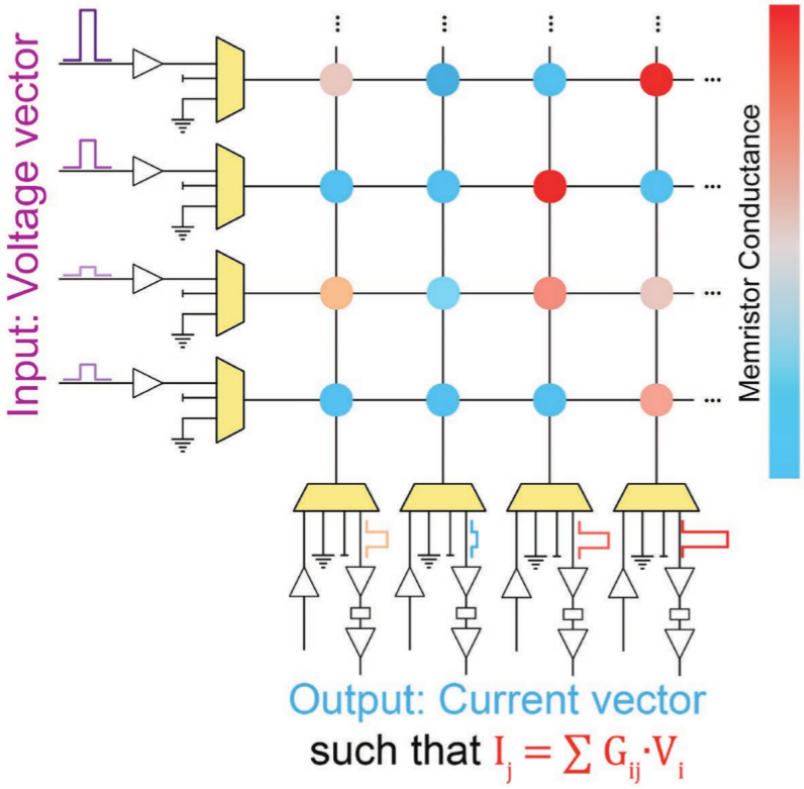


Figure 3.4 A DPE grid as suggested by [4]. Different colors shows different memristance values according to their conductances. Peripheral circuitry is responsible for controlling the input voltages and output currents

attached in series to a transistor; 1T1M cell [4]. All crossbars of the DPE are connected to a computer capable of tuning each memristor individually using the transistors and their gate voltages; V_G, which limits the current passing through the line. An update process of the weights is done once and follows the following algorithm. First the computer has a precomputed targeted set of conductances; G_{target}, that must be compared to a read matrix from the array; G_{read}, while setting all voltages to minimum. Then for each memristor in the array, the computer raises one out of two flags: SET or RESET. If $G_{read} < G_{target}$, it implies the device requires SET; RESET otherwise, until the error margin is within a certain range. During each cycle, the voltages are recorded, and increased with minimum step size (0.2V), until the desired conductance is read. When the voltage reaches its maximum before the conductance is set, V_G is increased for more current to pass through. The process is repeated every step until G_{target} is achieved within a maximum number of cycles allowed. It is to be noted that if a device passes its biased state with more than the permitted error; i.e. its

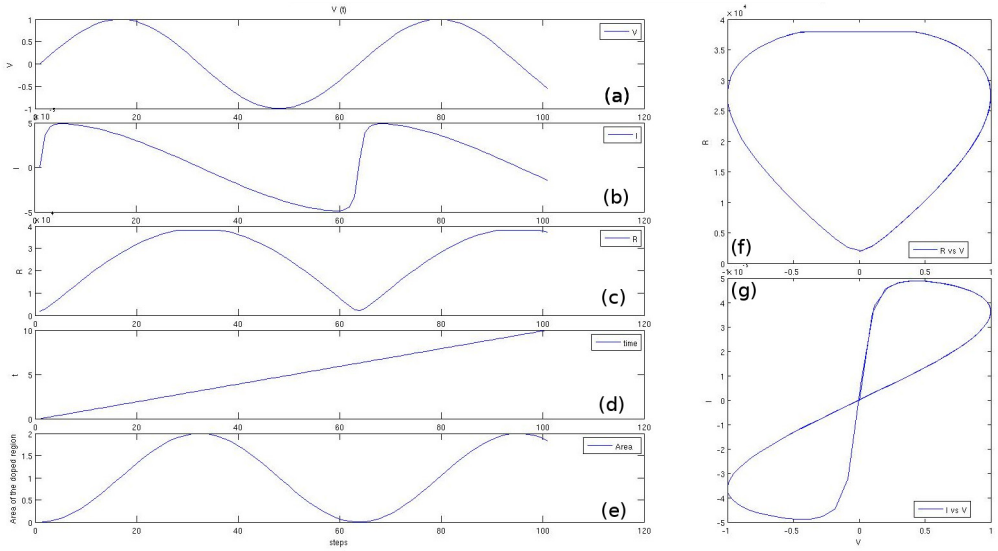


Figure 3.5 Simulated Memristor Characteristics using MATLAB Simulink (a) The input sinusoidal voltage in V (b) The current response of the device in mA (c) The change in resistance value according to the input voltage in $k\Omega$ (d) The time vector applied to the experiment in s (e) Variation of doped area with input voltage in nm (f) Memristor R-V characteristic curve (g) Memristor I-V hysteresis curve [8–10]

flag switches between SET and RESET, its input and gate voltages are set to their original values for re-programming.

3.4 Results

The DPE represents a promising approach to solve VMM, yet it needs enhancements at the level of device control to be integrated into computers. Memristors is a recent technology and has been integrated into many applications like reconfigurable computing: memory storage, analog implementations and digital designs. References [75, 76] are notable examples out of many available.

Equilibrium Propagation (EP) is a novel approach for solving NNs [18]. It is based on a novel back-propagation technique comprising three phases: forward and backward propagations in addition to weight update. EP does not depend on the energy function of the network. It rather applies to any network whilst having minimum knowledge about the energy function. The Hebian energy function; mentioned in [18], is just one example used to simplify the explanation of EP. EP can correct the neuron's output based on the system's states observed during training, testing and validation cycles of the algorithm. These states represent the

Table 3.1 Comparison of Analog Approaches versus Ideal Software [3–5]

Feature	**DPE**	**RPU**	**Ideal Software**
Matrix Size	128×64 hardware simulated	4096×4096	System related
Training for MNIST dataset	50 cycles overall	10 epochs for stability without the weight update	few hours in worst case
Complexity of Design for basic circuit	1T1M cell	1 AND Gate 1 Adder and weight storage circuitry	Software based
Accuracy with MNIST dataset	$89.9 - 92.4\%$	$88.3 - 90\%$	$+2.5\%$ over DPE
Computational Efficiency (in $\frac{TeraOps/s}{Watt}$)	115	$20 - 83$	7

system dynamics which must be maintained as long as EP applies. Our aim is to implement EP using a modified version of DPE. A memristor can be implemented using the circuit of Fig. 3.3 [7]. When this circuit is modified to a nano scale, then it can fit into the cross bar joints. Controlling this circuit to a certain memristance will help produce a good algorithm that solves first the VMM then EP. Figure 3.5 shows our initial tests in MATLAB Simulink for a single memristor. Initial VMM tests using this model revealed good timing values while increasing the size of the network from orders of milliseconds for small arrays to $50s$ when the array size reaches 60 M joints. These tests were performed on a core i7 Intel processor with 8GB of RAM. Fig. 3.5 shows the used-memristor characteristics during a programming phase. As the potential difference on the two terminals change (Fig. 3.5(a)), the device's resistance changes accordingly; (Fig. 3.5c), causing a deformation in the output current wave (Fig. 3.5(b)). Figure 3.5(e) shows the change in the doped area of the memristor which controls the resistance value of the device. Figures 3.5(f) and 3.5(g) present the R-V characteristics and I-V hysteresis loop respectively for the simulated ideal memristor in MATLAB Simulink based on the model in [8–10].

3.5 Discussion

The RPU crossbar proposed in [3, 5, 6] seems to be a good approach. The authors state that a VMM operation can be achieved in the order $O(1)$, while having the possibility to execute the three different cycles of a typical NN algorithm (forward/backward propagation and weight update) over three different time intervals. This represents a good starting point to implement NNs. Moreover, RPU provides 30,000X acceleration compared to other designs [5]. In addition, the RPU design uses simple concept simplifying the complex multiplication operations into linear and straightforward *and* operations, which simplifies the design considerations as well. On the other hand, RPU design has few drawbacks. First of all, everything presented is only design simulation with no real hardware implementation despite that board considerations and design parameters have been provided. Second, the modeled circuitry is not compact for a crossbar; (Fig. 3.1(a)), with multiple transistors for the *and* gate, the adder and the weight storage which implies bigger scaled-sizes for the array.

The DPE is another advance in the analog domain. The hardware is rather small and simple, thanks to the memristor functionality which helps saving space by storing its current value. Another advantage for DPE is providing board simulation tests illustrated in the support material of reference [4]. These boards can be tested for different algorithms. The DPE; on the other hand, must have a precomputed G_{target} which might require extensive pre-computing steps although it might be computed only once. Moreover, computing G_{target} is not performed per application; i.e. G_{target} is assumed to be the same for relatively close training datasets in order minimize pre-processing efforts. This may not be the most effective realization as a memristor will probably not be programmed to the proper value depending on the tolerance and process variations. Since DPE has precomputed weights, then it is not capable of executing the three phases of the algorithm, in which case intervention of a classical processor is essential for weights' update. Therefore both DPE and RPU may require controlling each unit separately, which is not efficient for large matrices and might add some undesired control overhead to the execution, consequently delaying the overall process.

Table 3.1 shows a direct comparison between the DPE and RPU approaches. RPU can accommodate larger matrices than DPE but the DPE can perform the VMM process faster.

However the RPU requires less training time and is more flexible with respect to the dataset, yet its design is more complex at the circuit level. As a main point of interest, analog crossbar arrays show big improvement in computational efficiency over the traditional digital solutions; however these numbers are not sufficient to beat the software approaches at the moment.

3.6 Conclusion

In this paper, we analyzed two different approaches to solve Vector Matrix Multiplication (VMM) highlighting their pros and cons. The first is the Dot-Product-Engine (DPE) based on memristors, while the second is the Resistive-Processing-Unit (RPU) approach based on a simplified version of the multiplication equation. We also concluded that both approaches are good to solve NN algorithms, but since memristors appear very promising, we started developing a modified version of DPE to implement Equilibrium-Propagation (EP); a novel algorithm for solving NNs.

Acknowledgment

The authors would like to thank IVADO for its financial support to the work.

CHAPTER 4 ARTICLE 2: MEMRISTOR EMULATORS FOR AN ADAPTIVE DPE ALGORITHM: COMPARATIVE STUDY

Hussein Assaf[1], Yvon Savaria[1] and Mohamad Sawan[1]
[1]Department of Electrical Engineering, Polytechnique Montreal,
Montreal, QC, Canada
Email: *hussein.assaf@polymtl.ca*

Through writing article 1 (presented in chapter 3), we became more confident that CMOS-based emulators for Resistive Switching Devices (RSDs) are needed due to the instability in RSD fabrication. To confirm such a need, we visited a memristor foundry in Canada where the people we met reassured our conclusion by explaining the challenges facing their production line. Hence, we wrote this article presenting a simplified yet improved version of the Dot Product Engine. The system, which we named the Adaptive Dot Product Engine, was based on spice models for the KNOWM memristor. Although the jury members (who evaluated this thesis) had comments on this article's content, we could not change any of the material presented since we are required to include it in its published version. This article is published at the 2019 IEEE International Conference on Artificial Intelligence Circuits and Systems (AICAS), pp. 13-17, doi: 10.1109/AICAS.2019.8771594.

Abstract

Vector Matrix Multiplication (VMM) is a complex operation requiring very large computational power to fulfill one iteration. Resistive computing; including memristors, is one solution to speed up VMM by optimizing the multiplication process into few steps despite the matrices' sizes. In this paper, we propose an Adaptive Dot Product Engine (ADPE) algorithm based on memristors for enhancing the process of resistive computing in VMM. The algorithm showed +5% error on preliminary results with one training step for one layered crossbar array circuit of memristors. However memristors require new fabrication technologies where the design and validation processes of systems using these devices remains

challenging. A comparison of various available circuits emulating a memristor suitable for ADPE is presented and models were compared based on chip size, circuit elements used and operating frequency.

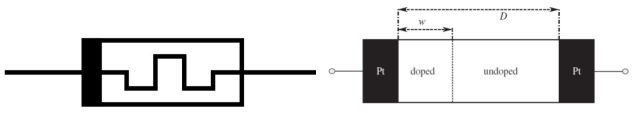
4.1 Introduction

The concept of memory-devices was first proposed in 1971; [36], as a link to bridge the gap between flux and charge. Such devices were proposed as the fourth circuit element. Memristor; Fig. 4.1, became the most famous in the series that includes memory-capacitor and memory-inductor.

Just like any other resistor, the current-voltage relationship in a memristor is linear with $V = M(r)I$ where $M(r)$ is the memristance of the device. $M(r)$ can change according to the potential difference across the terminals of a memristor. This change can be either linear or non-linear depending on the model deployed.

Memristors are designed to be in the scale of nanometers, and are made off two metalic plates with some semi-conductor substrate between them, like Titanium-dioxide. The latter is divided into two regions; one highly doped with electrons while the other is undoped. The width; w, of the doped region represents the window of a memristor which determines its resistance according to the mobility of holes from one region to another, Fig. 4.1(b) [9].

The non-linear behavior of holes cause $M(r)$ to change non linearly depending on the window function that is determined by the model [77]. This non-linearity in $M(r)$ change causes a hysteresis current-voltage characteristics; consequently, the measured area of the doped region is proportional to the device resistance [9].



(a) Symbol (b) Model

Figure 4.1 The memristor

Each memristor is capable of saving its current resistance state based on the voltage across its two terminals. Figure 4.2 shows the variation of an ideal device memristance with an input Piece Wise Linear (PWL) input voltage. When the device is at rest; i.e. zero input voltage, it starts varying from its last resistance value reached (e.g t=0.01ms and t=0.02ms in Fig. 4.2). This behavior continues until the resistance is set to its initial value (t=0.71 ms in Fig. 4.2) caused by sufficient negative input signal to reset it.

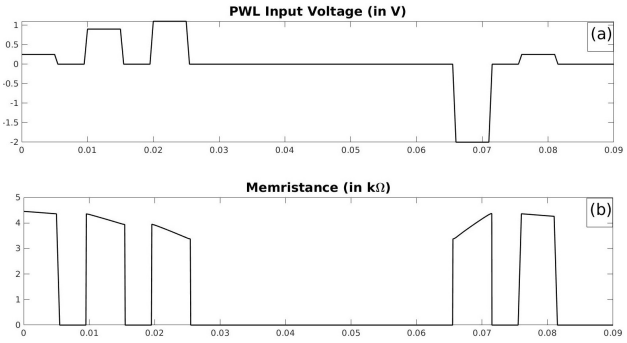


Figure 4.2 Response of SPICE simulation with PWL input voltage shows how the internal state of the memristor is preserved while voltage changes; with each new arriving signal the memristance starts varying from its previous value

The memristor correlates the charge q to the flux ϕ [36, 78]. This change in the memristance is one advantage of such device, where it enables the storage of non volatile data across the circuit. Some algorithms make use of these changes to store weights during run time when simulating neural networks. Another advantage is the size of these tiny components. If scaled to nanometers, memristors could operate at high frequencies and relatively close to current CMOS technology clock rates. While operating at such frequencies, those tiny circuit elements produce less heat than other components like CMOS and even can boot-up faster [79].

Some disadvantages of memristors, howevever, are delaying its commercialization. Still, a real memristor may not be as promising as its corresponding computer simulation is. Programming these hardware components requires a processor intervention like a computer, a micro-processor or a Field Programmable Graphical Array (FPGA) [79].

The remaining parts of this paper are organized as follows: Section 4.2 describes an adaptive

algorithm to modulate a memristor in crossbar arrays, section 4.3 will describe and compare some memristor emulators available in the literature, section 4.4 will present preliminary simulation results and section 4.5 will conclude the study.

4.2 The Adaptive Dot Product Engine

Nowadays, the concept of resistive computing is an active trend for analog computation as a solution for Vector Matrix Multiplication (VMM). With the increase in data size and in complexity of the Artificial Intelligence (AI) algorithms, VMM can grow in size up to the gigabytes limit requiring; therefore, huge computational power. Even with the best computer stations available, VMM operations can last for days before results appear. One option to solve the problem is using crossbars of resistive devices like Dot product Engine (DPE).

Table 4.1 Different window functions used for memristor

Name	**Function**	**Notes**
Joglekar [80]	$f(w) = 1 - \left(2\frac{W}{D} - 1\right)^{2p}$	w: doped region length D: Channel length p: Control parameter
Biolek [81]	$f(w) = 1 - \left(\frac{W}{D} - sgn(-i)\right)^{2p}$	w: doped region length D: Channel length p: Control parameter i: is the current $sgn(-i) = 1\ if\ i \geq 0$ $sgn(-i) = 0\ if\ i < 0$
Prodromakis [82]	$f(w) = j(1 - [(w - 0.5)^2 + 0.75]^p)$	w: doped region length D: Channel length
Benderli [83]	$f(w) = \frac{w(D-W)}{D^2}$	w: doped region length D: Channel length
Strukov [9]	$f(w) = \frac{w(1-w)}{D^2}$	w: doped region length D: Channel length
TEAM [78, 84]	$f_{on,off} = exp\left[-exp\left(\frac{\lvert x - x_{on,off}\rvert}{w_c}\right)\right]$	$x = \frac{w}{D}$ w: doped region length on: memristor is active off: Memristor not active

DPE proposed by Hu et al stands for a crossbar similar to the one in Fig. 4.3(a) [4]. It has two main parameters: input signal vector and memristor conductance. The only difference in architecture between Fig. 4.3(a) and DPE is that DPE uses the concept of 1M1T (i.e. 1

memristor and 1 CMOS transistor) for each node in the crossbar. The output of each column is a current defined by eq (4.1).

$$I_{col} = \sum_{row=1}^{m} G_{row,col}.V_{row} \tag{4.1}$$

where I_{col} is the current per column, $G_{row,col}$ is the respective conductance at the intersection of the corresponding row with the column and V_{row} is the voltage per row.

Not only does the Adaptive DPE (ADPE) differs from DPE in architecture, but also it does in concept. ADPE starts by estimating the influence of 1 unit change at the input on the output voltage per column. Each layer might have different recordings and behavior depending whether the layer is a peripheral layer; i.e. has direct communication with either inputs or output, or middle layer sandwiched between two others crossbar layers.

After recording the influence, a first round of input signals is applied; to record the corresponding output signal, followed by comparison of the recorded output to the targeted one. It depends on the designer's assumptions to figure out how to compare the real output to the desired one, assuming that the cross-row effects between neighboring memristors are negligible which is not a healthy assumption for real circuits. The measured difference between real and targeted outputs is converted; according to the influence relationship formulated before, into voltage change at the input terminals.

Next, conductances have to be modified so that the output current matches the targeted one. It is necessary to note that the deployed memristors are of unknown conductance states yet they are assumed to have similar characteristics; e.g. same switching resistances (R_{on} *and* R_{off}). Assume that the measured effect of 1mV at input voltage is 2 μV change per vector current and there is +100 μV error. It follows that the input voltage must be modified by +50 mV on all input signals to match the desired +100μV change. One good thing about this cross bar is that negative changes can easily be posted by simply applying negative change across the input signals. If the error was -100 μV, then the input signals must be altered by -50 mV. The algorithm keeps looping and measuring the error for a given input vector until convincing results with the minimum error are obtained. Once completed

Table 4.2 Comparison of different memristor models

Name	I-V relation	Notes
Linear ion drift [85]	$v(t) = \left(R_{on}\frac{w(t)}{D} + R_{off}\left(1 - \frac{w(t)}{D}\right)\right) i(t)$	$v(t)$: voltage $w(t)$: Window function $i(t)$: Current R_{on}: Min. resistance R_{off}: Max. resistance
Noninear ion drift [86, 87]	$i(t) = w(t)^n \beta sinh(\alpha v(t)) + \chi\left[exp\left(\gamma v(t)\right) - 1\right]$	$v(t)$: voltage $w(t)$: Window function $i(t)$: Current $\alpha, \gamma, \chi, \beta$: fitting Parameters n: Determines effect of state on current
Simmons tunneling barrier [88]	$v(t) = \left[R_{on} + \frac{R_{off} - R_{on}}{x_{off} - x_{on}}(x - x_{on})\right] i(t)$	$v(t)$: voltage $w(t)$: Window function $i(t)$: Current R_{on}: Min. resistance R_{off}: Max. resistance $x = \frac{w}{D}$
TEAM [78, 84]	$v(t) = R_{on} e^{\frac{\lambda}{x_{off} - x_{on}}(x - x_{on})} i(t)$	$v(t)$: voltage $w(t)$: Window function $i(t)$: Current R_{on}: Min. resistance R_{off}: Max. resistance $x = \frac{w}{D}$

for the first vector, it loops over all input vectors and so on.

Surprisingly, this algorithm has one main advantage which is its convergence speed due to the linear current-voltage relationship in the resistor-like devices, thus the assumption is that whatever the error is it could be corrected with one single step only. On the other hand, this crossbar is not a stand alone system and needs a micro-controller to be placed alongside for proper error correction. Algorithm 1 summarizes the complete algorithm presented here.

Algorithm 1 `ADPE Algorithm`

```
1: Inputs:
2:   - Voltages
3:   - Conductances
4: Outputs:
5:   - Current
6: Assumptions:
7:   - Memristors are similar
8:   - Memristor conductances unkown
9:   - Circuit is ideal and no coupling effect between neighboring memristors
10: Start:
11:   - Measure the influence of 1 unit input change on output
12:   - Apply input vector
13:   - Measure error at output
14:   - Correct input vector in 12 according to 11
15:   - Reapply corrected input
16:   - Go to 13 until error is minimum
17:   - Go to 12 until all vectors processed
18: End
```

4.3 Memristor Emulators

Although discovered 10 years ago, memristor fabrication is yet at its early stages and no satisfying single nanometer device is fabricated to fulfill the high speed demands in recent technology [89]. Different models have been proposed to achieve the best memristor device that matches Chua's desired fourth element.

Every memristor substrate material has its limitations suggesting that the memristor devices are not stable enough to gain manufacturers' trust. That is why emulating a memristor circuit is an active research trend nowadays. Tables 4.1-4.2 summarize the different existing memristor behavioral models and their corresponding window functions. These functions are very important to understand the functional behavior of a memristor so that any emulating circuit will probably implement one of these models and window functions.

This analysis of circuit emulators is biased towards finding a convenient memristor circuit emulator that can fit smoothly in a crossbar array structure supporting the ADPE algorithm. ADPE, DPE and similar approaches are supposed to beat currently available Central-Processing-Unit (CPU) and Graphical-Processing-Unit (GPU) simulations of vector matrix

Table 4.3 Selected memristor circuit emulators for the ADPE algorithm since 2014 and their characteristics

Ref.	**Year**	**Size**	**Components**	**Max. Frequency**	**Notes**
[90]	2014	Depending on CMOS technology	$\approx$ 50 CMOS Transistors	$10 - 100\ KHz$	-
[91]	2015	Macromodel	Resistors, Capacitors, Current Conveyors	$160\ KHz$	-
[92]	2015	Macromodel	Resistors, Capacitors, Current Conveyors	$5\ KHz$	Flux controlled
[93]	2015	Depending on CMOS technology	CMOS Transistors	$\approx 2\ KHz$	Includes OPAMPs
[94]	2017	Depending on CMOS technology, in the paper size in $(\mu m)^2$	7 CMOS transistors	$1\ MHz$	Very small bias curent, No ADC
[95]	2017	Depending on CMOS technology	16 CMOS transistors	$30\ Hz$	1 OPAMP

multiplication, as well as FPGA implementations [3]. That is why emulators are classified based on their synthesis size, and the maximum frequency they can support. Some basic macro-models in the literature can explain the basic operation of memristors using simple components like resistors, switches, current controlled voltage source and voltage controlled current source. Authors in [85] describe a very useful example to understand the theory of memory-devices. However these models will not be much beneficial when operating at nanometers scales.

Macromodels shown in [91, 92] will not be helpful either since they operate at low frequencies and they cannot fit into the tiny crossbar structures.

A model like the one itroduced in [95] is a useful choice, however it contains operational amplifier that consumes significant power budget and as a consequence, generates too much heat if embedded in a large arrays. Moreover the model is tested for low frequencies only; of the order of 30 Hz, and is not a competitive frequency which analog computation techniques hope for.

The disadvantages of using a model like the one in [90] is obvious for memristive crossbar array. Authors propose a structure requiring around 50 CMOS transistors to simulate one memristive element. In addition, it operates at low frequencies $10-100\ KHz$. But the design showed robustness and good results for a memristor circuit emulator.

An interesting circuit emulator is found in [94]. The circuit can have compact implementations as it requires only 7 transistors and it can operate at 1 MHz. This compact circuit is a good basis to develop a competitive design. The 1 MHz frequency; however, is still far away from the operating frequencies of digital simulators that can operate at GHz. The chip size is yet another disadvantage, in [94] the transistors are designed in μm which implies that a single element will size in $(\mu m)^2$. This might not be bad if the transistor technology is switched successfully to nm which in turn will increase device frequency response. Even if the transistor technology could not be changed, starting simulation with micro-meters scaled memristor remains promising.

Table 4.3 summarizes the comparison between the different aforementioned circuit emulators. Values reported in this table show that the suggested circuit in [94] is promising as a memristor emulator for crossbar array structures.

4.4 Preliminary Simulation Results

Figure 4.3 shows a simple ADPE crossbar array that is simulated using LTSPICE. This version of ADPE is only tested for one layered circuit of memristive crossbar array.

The algorithm starts by estimating the influence of 1 unit change at the input on the output voltage per column. Since we assume ideal simulation, then cross-row effects between neighboring memristors are assumed to be negligible. Thus each column of memristors operate independently to generate the output current measured at the peripheral circuit. The measured effect of 1 mV at input voltage induces 20 μV change per output voltage vector from four memristors.

After applying the first input voltage vector, the output voltages are summed up to compare it with the targeted voltage; Fig. 4.3(a), and the error difference is converted into a desired

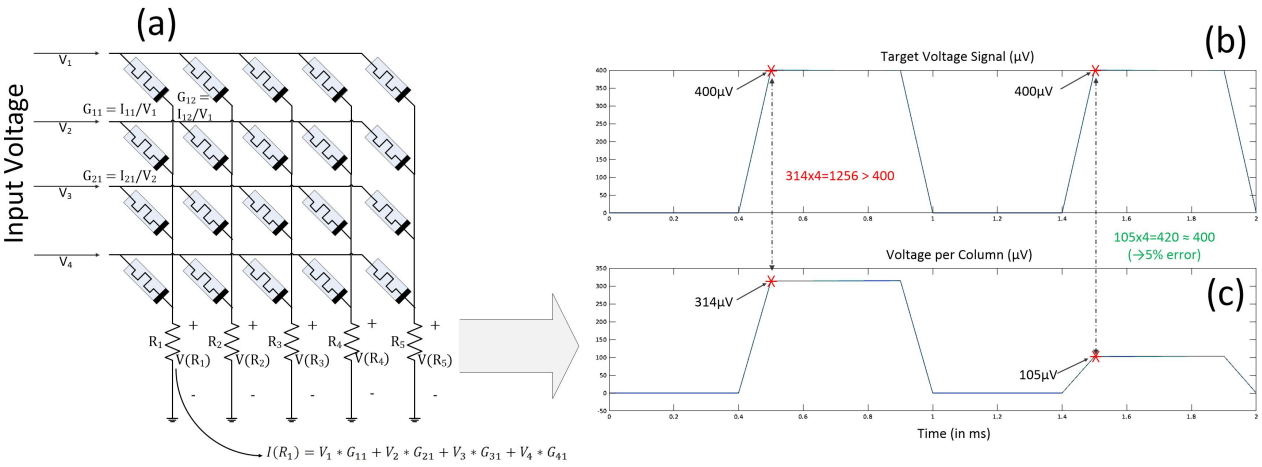


Figure 4.3 An adapted version of the Dot Product Engine (DPE) to solve the problem of vector matrix multiplication (a) Sample Circuit: Voltages v1-v5 are the input voltages. The output voltage is measured at the resistors R1-R5 (b) The targeted output voltage to achieve by adding the voltages across R1-R5 (c) The response per column of the circuit

change at the input voltage. In this case, negative conductances are needed to lower the total sum of column voltages. As aforementioned, it is enough to apply negative voltage at the input terminals. Figures 4.3(b-c), show $+856\mu$V $((314\ \mu V \times 4\ \mu V) - 400\ \mu V)$ which leads to $-42\ mV$ $(856\ \mu V \div 20\ \mu V)$ change across the input voltages with the best choice of the voltage source(s) to be modified. Thus through the next pulse, plots show only 5% $((105\ \mu V \times 4) - 400\ \mu V = 20\ \mu V;\ \frac{20}{400} \times 100 = 5\%)$ more than the targeted output voltage, Fig. 4.3.

4.5 Conclusion

We presented the main trends of a novel approach, an Adaptive Dot Product Engine (ADPE), for deploying memristors in vector matrix multiplication using a crossbar array structure. Preliminary results show that an ADPE can achieve 5% error in a single training step only for one layer of memristive crossbar array. This work also compares different memristor circuit emulators based on sizing and frequency response to select the best design suitable for resistive computing. The best means for emulating memristors utilizes only 7 CMOS transistors and one current source. It is capable of operating at a relatively high frequency

(1 MHz) yet the circuit dimensions depends on the designer's choice for a proper CMOS technology to use.

Acknowledgment

The authors would like to thank IVADO for its financial support to the project.

CHAPTER 5 ARTICLE 3: IMPLEMENTING FLOATING GATE TRANSISTORS WITH A THIN GATE DIELECTRIC 65 nm CMOS TECHNOLOGY

Hussein Assaf[1], Yvon Savaria[1], Mohamed Ali[1,2], Morteza Nabavi[1] and Mohamad Sawan[1,3]

The efforts we made to characterize charge tunneling in the CMOS 65 *nm* technology with thin gate dielectric are summarized in this article. The characterization process was necessary to develop a charge-trap CAD-model we needed to design a memristive cell with long data retention time. This article was submitted in May 2022 for a second round of reviews to IEEE Transactions on Electron Devices.

Abstract

Recent advances in analog-based circuits are calling for more reliance on Floating Gate Transistors (FGTs) as means for nonvolatile memories and resistive computing engines. However, developing a reliable simulation model matching a real FGT implemented with sub 70 nm technologies is difficult, as it demands proper characterization of the real device. This article establishes a basis for modeling FGTs in a standard 65 nm CMOS process with thin gate dielectrics. It focuses on FGTs' main features such as retention time, magnitude of tunneled current and effective voltage range. By analyzing the extracted measured characteristics, a new unexpected behavior of thin-dielectric FGTs is discovered and verified with an effective voltage range between 0 V and 550 mV whilst the tunneled current reaches a maximum of 8 nA. Moreover, the dielectric layer has proved to be very sensitive to the applied voltage exhibiting characteristic changes that alter its behavior. Nonetheless, a fabricated FGT has trapped charges corresponding to a 358 mV gate potential for over a year. This article also presents an experimentally verified analytic model for the targeted FGT that can be easily implemented using CAD tools.

[1]H. Assaf, Y. Savaria, M. Ali, M. Nabavi, and M. Sawan are with the department of Electrical Engineering, Polytechnique Montreal, Montreal, QC, Canada. *e-mail:hussein.assaf@polymtl.ca.*
[2]M. Ali is also with the department of Microelectronics, Electronics Research Institute, Cairo, Egypt
[3]M. Sawan is also with the CenBrain Lab, School of Engineering, Westlake University, Hangzhou, Zhejiang, China

5.1 Introduction

Floating Gate Transistors (FGTs) have been demonstrated in 1971 [96]. They were utilized in various applications that need non-volatile analog memories as in Electrically Erasable Programmable Read Only Memories (EEPROMs) [97]. They have also become an integral part in analog computing engines that depend on crossbar arrays, replacing the tiny resistive devices in these structures [74]. Moreover, FGTs were used in programmable analog solvers like Field Programmable Analog Arrays (FPAAs) which add flexibility to analog-based circuit solutions [73, 98].

Floating gate transistors can trap charges for long periods of time, 10 years or more in some cases. We refer to this characteristic as the retention time. The underlying CMOS process has a direct effect on this FGT's characteristic. Not only are FGTs characterized by their retention times, but also by other features that contribute to giving each FGT a unique identity. Some of these features are: (a) the transistor dimensions, length and width, which affect the tunneled current and induce more parasitic capacitance in a cell; (b) the dielectric thickness which has a direct effect on charge tunneling; (c) the charging capacity and (d) the discharging capacity, the last two factors define the time required to fully charge or discharge any capacitor attached to an FGT; (e) the junction breakdown voltage, where every *p*-*n* junction has a certain voltage tolerance beyond which the cell is damaged; (f) the maximum number of permitted write-operations through the dielectric including every charging and discharging process; (g) the effective voltage range, which determines the safe voltage limits for charge tunneling without breaking the *p*-*n* junction, and finally (h) the transistor *gate*-to-*bulk* and *gate*-to-*terminal* resistances.

Characterizing an FGT is very critical to produce a simulation model that matches the actual behavior of a real transistor. Researchers have always struggled to simulate any FGT-based design before proceeding to chip tape-out, a process that is usually costly in cutting-edge technologies. Some researchers used mathematical equations to estimate the gate leakage current in *sub* 70 *nm* MOSFETs [11]. Even with the use of advanced simulation models provided by recent computer aided systems, it is still difficult to simulate an FGT-based design.

Multiple works relate to the characterization of FGTs [99–108]. Some research led to the fabrication of custom FGTs [99–104], while other validated the developed theoretical models using Computer Aided Design (CAD) tools [68, 108]. On the other hand, some researchers characterized FGTs in standard 0.5 μm and 0.8 μm CMOS processes [105, 106]. Meanwhile other research started by a verified CAD model which was then utilized to fabricate an FGT in a standard CMOS 180 nm process [74, 109].

The reported effective voltage range for charge tunneling in 20 nm and 7 nm dielectric MOSFETs is between 1.2 V and 5 V for transistors fabricated under 0.25 μm and 0.18 μm CMOS processes [68]. The thinner the dielectric layer is, the trickier it becomes to reliably and controllably tunnel charges through it, because of the high sensitivity of the tunneling mechanism to the applied voltage. As a result, many recent applications that deploy FGTs are still not resorting to *sub* 70 nm transistors with *sub* 5 nm dielectric thicknesses [110, 111].

5.2 Methodology

The purpose of this chapter is to analyze the behaviour of an FGT cell fabricated in a 65 nm process. Two different cell topologies are proposed to study and analyze the properties of this FGT: (a) An Individual Transistor Cell (ITC) composed of 6 transistors is used to characterize the effective voltage range and the tunneled-current magnitude (b) A Capacitive Cell (CC), made of one ITC attached to a capacitor and other circuit components, is used to analyze the retention time and both charging and discharging currents. Table 5.1 summarizes the properties of the transistors used in these cells.

Table 5.1 Characteristics of the transistors used in the fabricated circuits

MOSFET	**Type**	**W** (m)	**L** (m)	**Vth** (V)	**Lmin** (m)	**Wmin** (m)	**Tox** (m)
NMOS	1 V	200 n	60 n	260 m	60 n	120 n	$\geq$ 1.8 n
PMOS	1 V	200 n	60 n	-198 m	60 n	120 n	$\geq$ 2.0 n
NMOS	2.5 V	400 n	500 n	540 m	400 n	500 n	$\geq$ 5.0 n
PMOS	2.5 V	400 n	400 n	-547 m	400 n	400 n	$\geq$ 5.0 n
NMOS	2.5 V	500 n	1.2 μ	-18 m	1.2 μ	500 n	$\geq$ 5.0 n
NMOS	2.5 V	1 μ	1.2 μ	-51 m	1.2 μ	500 n	$\geq$ 5.0 n
NMOS	2.5 V	2 μ	1.2 μ	-65 m	1.2 μ	500 n	$\geq$ 5.0 n

5.2.1 Individual Transistor Cells

Based on Table 5.1, the circuits of Fig. 5.1(a) and 5.1(b) are designed to stress 1 V type transistors ($M1$ and $M3$) with variable widths. PMOS transistors $M2$ and $M4$ of 2.5 V type have a higher voltage tolerance than 1 V MOSFETs. Each of these two transistors is used as a control transistor to activate and stress a desired target cell when a shared pulse-train bus is applied.

Figure 5.1(a) presents the schematic of the basic cell in which *nmos* $M1$ is the device under test having a minimum channel length of 60 nm and three possible width values: 2 μm, 4 μm and 8 μm. Since its body is grounded and tunneling will occur on channel edges, the 14-finger layout of $M1$ maximizes the contact area between the transistor's *source* or *drain* terminals and its *gate*. Devices $T1$ and $T2$ are transmission gates made of two 2.5 V MOSFETs placed as protections between thin dielectric sensitive transistor gates and bonding pads.

An external variable resistor is attached to node E during every measurement of the voltage $V_E(t)$. The tunneled current through $M1$ ($i_{M1}(t)$) can be calculated by dividing the aforementioned voltage ($V_E(t)$) by its corresponding resistance on node E according to the relation:

$$i_{M1}(t) = \frac{V_E(t)}{R_E(t)} \tag{5.1}$$

When a pulse train is applied on node $A1$ with $EN1$ voltage held low, transistor $M2$ will pass the applied train to node C, hence stressing $M1$ and forcing charges to tunnel from its *source/gate* and *drain/gate* junctions to node $D1$.

Some theoretical models are often used in MOSFET circuit simulators when dealing with cutting-edge technologies like the Berkeley Predictive Technology Models (*BPTMs*). The curves plotted in Fig. 5.2 correspond to a 65 nm *BPTM* and are adapted from [11]. The plots correspond to 1.3 nm and 1.7 nm dielectric thicknesses, which are close to MOSFETs' dielectric thicknesses in our targeted technology. Based on Fig. 5.2, when $V_{gate} = 1\ V$, the leakage current is considered high until V_{source} reaches 0.5 V that represents a turning point after which the current decreases on a logarithmic scale to reach values in the order of $10^{-21} A$ at $V_{source} = 800\ mV$. This implies that the voltage difference between the *source* or *drain*

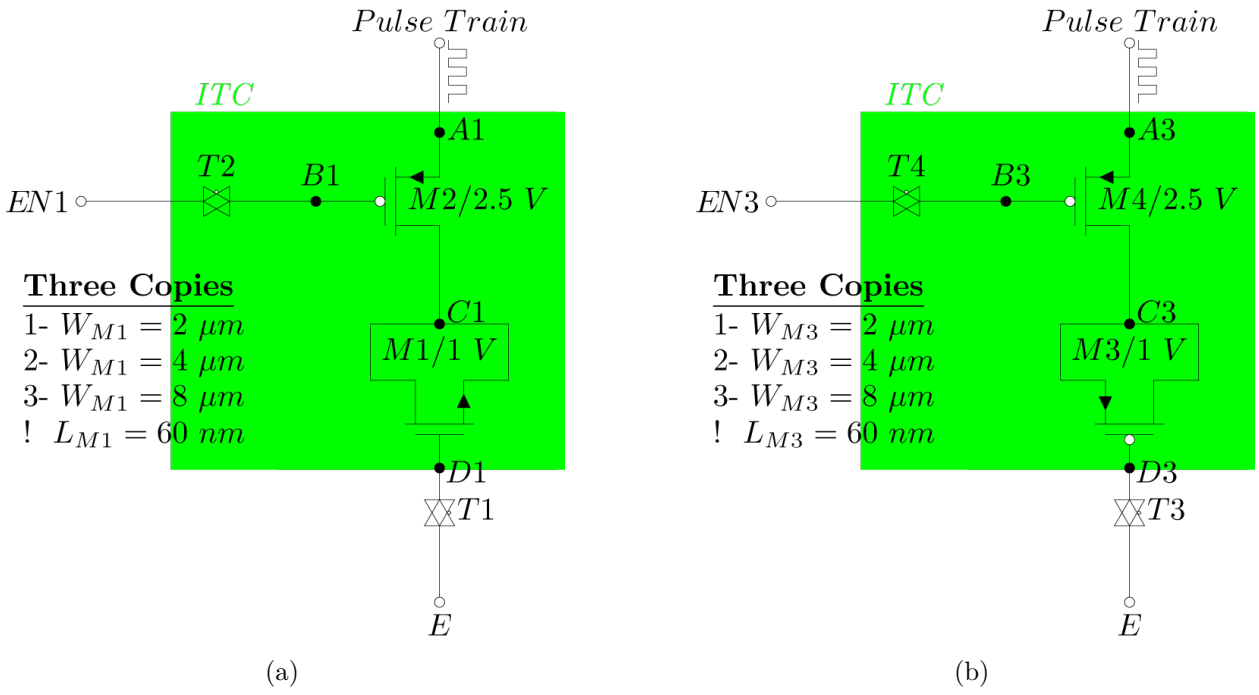


Figure 5.1 The schematics of: (a) The Individual Transistor Cell (ITC) used to characterize the floating transistor *nmos* $M1$ with 3 different width values (b) ITC with a *p-type* MOSFET, $M3$

nodes with the *gate* should be lower than 400 mV for the leakage to remain in the steep slope regime. That is for a transistor with characteristics similar to those plotted in Fig. 5.2, the region of operation where charges could be trapped on a floating gate transistor over a long period of time is that when $V_{gate} - V_{source(drain)} \leq 1\ V - 600\ mV = 400\ mV$.

It is expected that the gate leakage current decreases as the dielectric thickness increases. For example under *BPTM*, the 1.3 nm dielectric-thick MOSFET has a tunneling current much greater than a 1.7 nm MOSFET. Hence, it is safe to assume a leakage current to be even smaller for a 65 nm process in which the dielectric is slightly thicker ($\geq 1.8\ nm$).

To investigate the variation in tunneled current for 1 V transistors with respect to channel width, the cell presented in Fig. 5.1(a) was laid out and fabricated with multiple widths as explained before: 2 $\mu m, 4\ \mu m$ and 8 μm. Furthermore, to analyze the differences in behavior between *nmos* and *pmos* transistors, the same cell configurations were repeated for a cell comprising a *p*-type transistor $M3$, as shown in Fig 5.1(b).

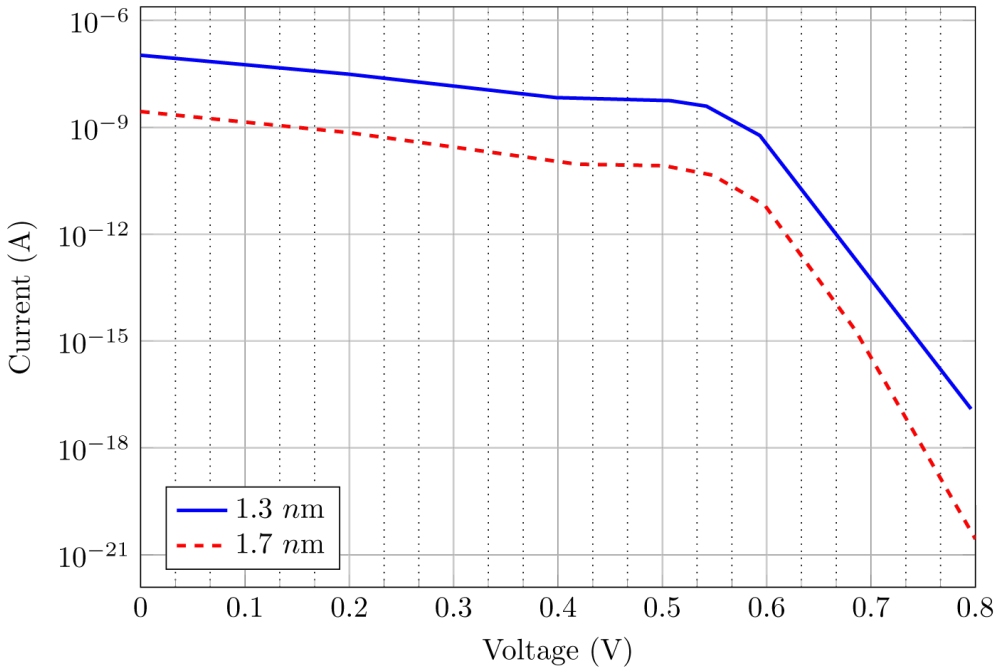


Figure 5.2 The predicted gate leakage current with respect to the source voltage for various dielectric thicknesses according to [11]

5.2.2 Capacitive Cells

The circuit shown in Fig. 5.3 is designed for characterizing the cell's retention time. This circuit reflects the effect of capacitor sizing on charging and discharging delays. When a pulse train is applied and transistor $M8$ is conducting by applying 0 V on node $EN7$, transistor $M7$ will be forced to tunnel charges according to pulses amplitude and duration.

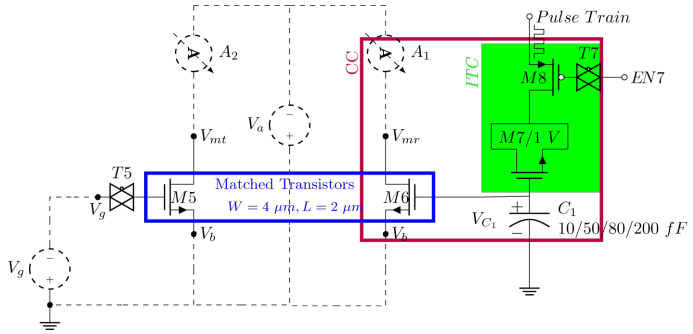


Figure 5.3 The schematic of a Capacitive Cell (CC) which is composed of one ITC connected to capacitor $C1$ and to the gate of $M2$ to indirectly measure $C1$'s voltage through V_g

In Fig. 5.3, capacitor $C1$ can have one of four different values: 10 fF, 50 fF, 80 fF and 200 fF. Each of these values corresponds to different charging and discharging times. Yet, the change in C1 value is not supposed to affect the aforementioned FGT voltage limit (400 mV), which defines the maximum charge that can be trapped on the node.

Transistors $M5$ and $M6$ are 2.5 V *nmos* transistors matched in layout implementation based on the common centroid method [112]. The design decision of using 2.5 V transistors was taken for safety at a time when we had no knowledge of the tunneling characteristics of the 1 V transistors in the target technology. Devices A_1 and A_2 are two ammeters placed off-chip to measure the currents traversing $M6$ and $M5$, respectively. The voltage source V_a supplies a DC voltage of 1 V whereas V_g is a variable DC voltage source. When the voltage across capacitor $C1$ increases, the current measured by A_1 increases with $V_a = 1\ V$. Sweeping V_g can translate the hidden voltage across C_1 when $I_{A_1} = I_{A_2}$; thus if transistors were perfectly matched $I_{A_1} = I_{A_2} \Longleftrightarrow V_g = V_{C1}$.

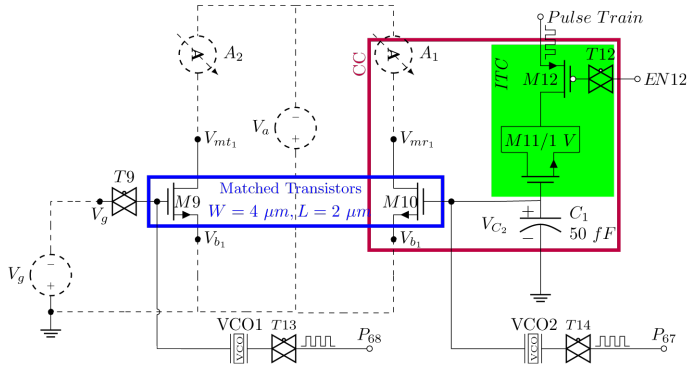


Figure 5.4 The schematic of the CC cell connected to two VCOs for indirect measurement $C1$'s voltage through V_g performed by comparing the VCOs' output frequencies

The circuit of Fig. 5.4 is similar to that of Fig. 5.3, but it also includes two matched copies of a Voltage Controlled Oscillator (VCO) whose internal structure is shown in Fig. 5.5. The voltage applied on *VCO Input* terminal in Fig. 5.5 tunes the VCO's output frequency. Hence, any two voltages are indirectly comparable by comparing the output frequencies of two matched copies of the VCO to which the two input voltages are connected. For example, in Fig. 5.4, the voltages V_{C_2} and V_g are assumed to be equal when the frequencies on nodes P_{67} and P_{68} are equal.

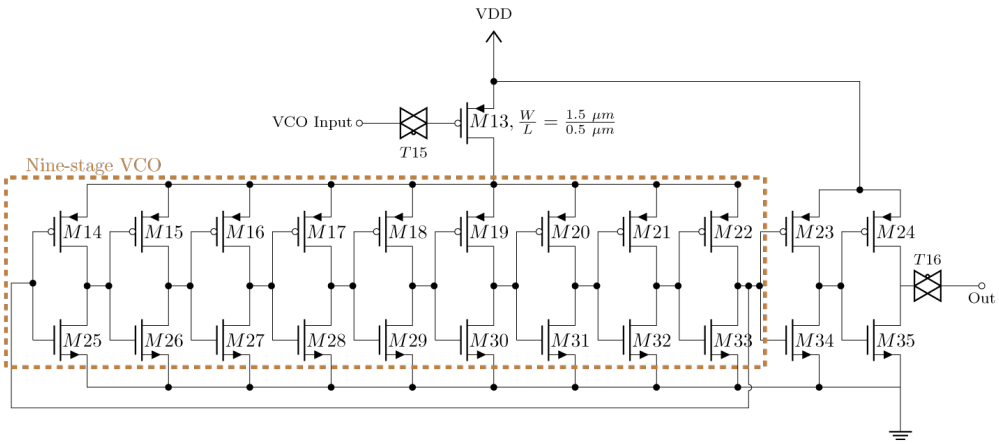


Figure 5.5 The schematic of the implemented nine-stage Voltage Controlled Oscillator (VCO) used for voltage to frequency conversion

Adding the two VCOs to our experimental setup allows providing another evidence of the voltage trapped on node V_{C_2} in Fig. 5.4. This type of indirect measurement is mandatory for observing trapped charges on a floating node. In addition, converting voltages to frequencies provides signals rather insensitive to parasitics on the readout paths.

5.3 Results

In this section, the measurement results of the fabricated test chip whose micro-graph is shown in Fig. 4 are reported in detail. These results were collected using: (a) A Keithley 2450 SMU source meter, (b) An Agilent 33220A waveform generator, (c) A Tektronix MDO4104-6 Oscilloscope, (d) A Keithley 4200 SCS parameter analyzer, (e) A Keysight N9010A spectrum analyzer, and (f) The MATLAB software running on a Lenovo notebook (Intel i7 fourth generation chipset and 8 gigabytes of RAM).

All the reported results in this section were gathered using MATLAB by direct communication with the equipment using the General Purpose Interface Bus (GPIB) protocol. The Keithley 2450 was the bottleneck in collecting data with around 80 *ms* per *read* command. This forced collecting the results in batches followed by estimating the correct timing in MATLAB.

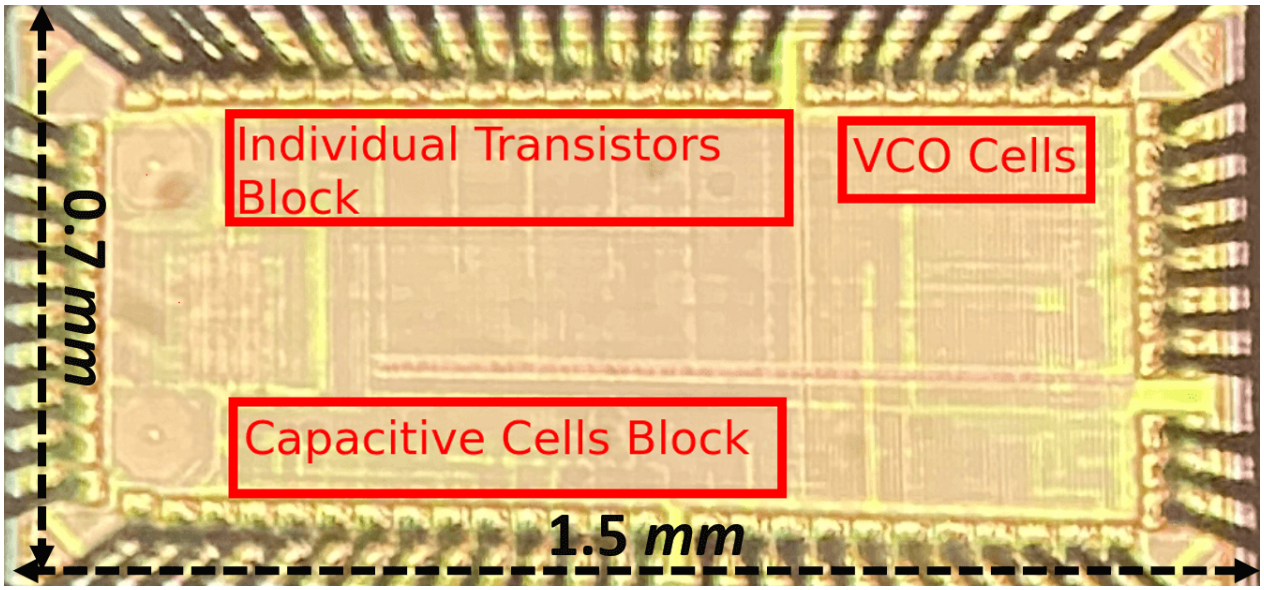


Figure 5.6 An image showing a healthy sample of the fabricated chip under the microscope with its dimensions

5.3.1 Individual Transistor Cells

5.3.1.1 Transistor Response

In order to properly characterize the voltage tolerance of the Capacitive Cell (CC) presented in Fig. 5.3, the threshold voltage of the 2.5 V MOSFETs, like $M5$ and $M6$, should be measured. Therefore, the gate voltage of $M5$ was swept while applying a 1 V on V_a, and the measured versus simulated *drain*-currents are plotted in Fig. 5.7. The curves in Fig. 5.7 show a threshold voltage of 350 mV for the fabricated devices versus a 550 mV threshold voltage for the same devices under a computer simulation. Since transistors $M5$ and $M6$ are matched in layout, the two transistors are assumed to have the same threshold voltage which is considered to be the minimum voltage that can cause a change in the reading of ammeter A_1, should V_{C_1} change as a result of the current tunneled through $M7$.

In addition, our chip was fabricated in a process where the maximum estimated voltage that any 1 V transistor can hold is 1.2 V [113]. For this chip, the purpose is to induce tunneling currents through the gates of the 1 V transistors, not the 2.5 V ones. Nominal supply voltages for both transistor models are 1 V and 2.5 V, respectively. Our initial assumption

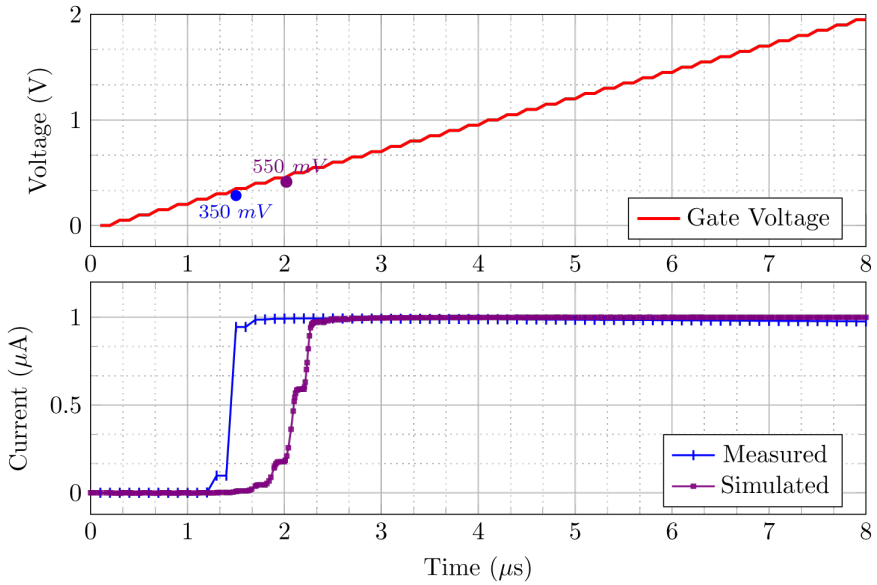


Figure 5.7 The simulated and measured drain current plots of the 2.5 V MOSFET observed when sweeping the gate voltage from 0 V to 1 V

while designing the test chip was that tunneling for 1 V devices would start at a voltage higher than 1 V, somewhere in the range of [1 V — 2 V]. Thus, to stress a 1 V MOSFET, it was expected that another transistor that can hold steadily the anticipated voltage range like the 2.5 V MOSFETs was needed. Computer simulations of the cells presented in Figs. 5.1(a) and 5.1(b) demonstrated their ability to reflect the DC components of every input signal. However, existing models do not accurately reflect the charge tunneling phenomenon.

In light of a lack of openly reported information on the tunneling voltage for devices with *sub* 5 nm gate dielectric, the circuits in Figs 5.1(a) and 5.1(b) were designed to flexibly stress only $M1$ or $M3$. Moreover, ESD protection cells were also removed for more flexibility.

5.3.1.2 Tunneling Voltage for $1V$ model MOSFET

Figure 5.8 shows the measured tunneled currents in the proposed ITCs (see Fig. 5.1) when applying a 1 MHz pulse train. The averaged leakage current of the ITC in Fig. 5.1(a) under no stress is plotted in Fig. 5.8(a) for $W = 8\ \mu m$ and $VDD = 0.8\ V$. This response indicates the sensitivity of the devices under test and shows that the measured current is a low-magnitude sinusoidal signal as long as the pulses' amplitudes are below 400 mV. Therefore, this low amplitude signal appears as a background interference component that adds to any

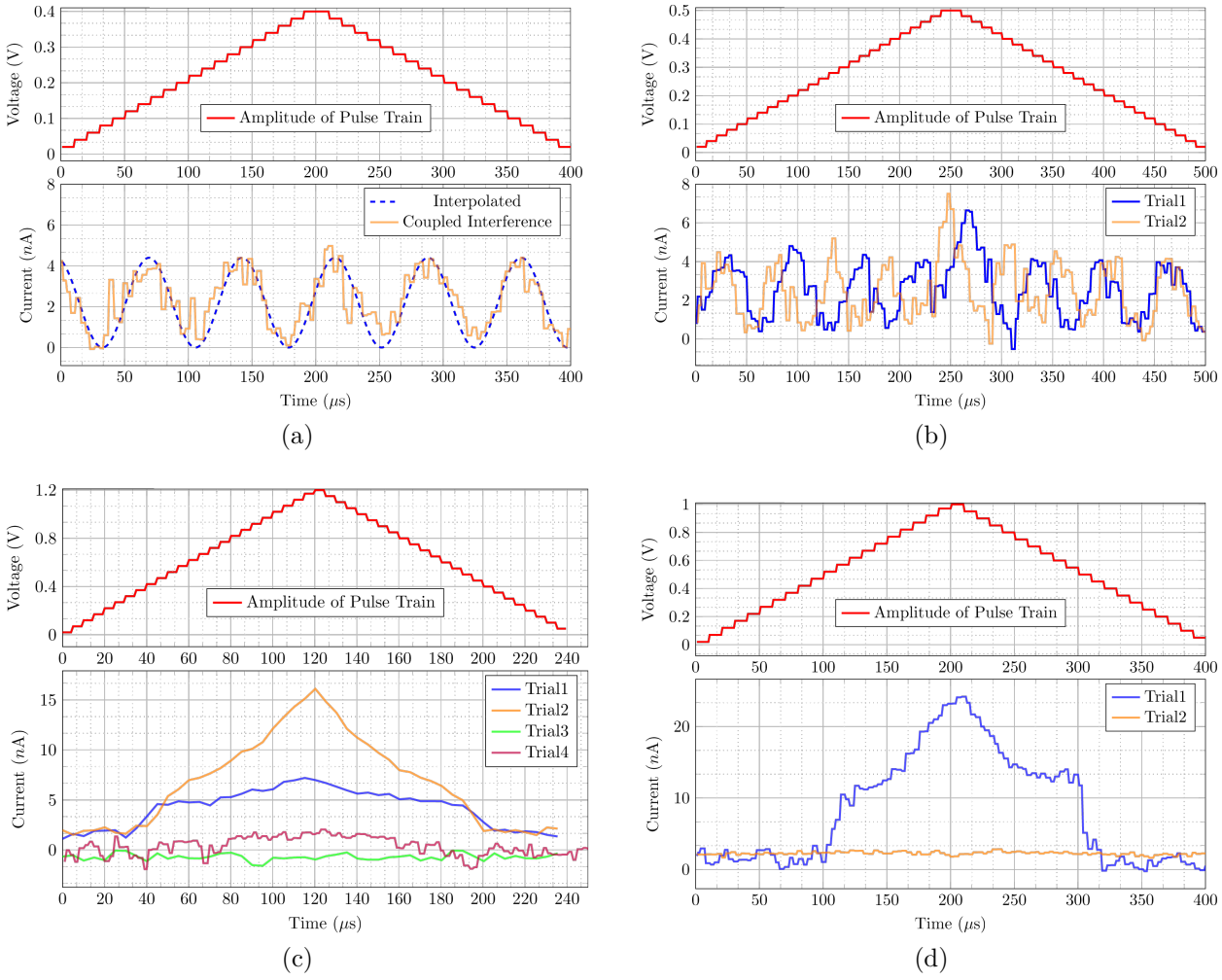


Figure 5.8 Measured tunneled currents in the proposed ITCs when applying a 1 MHz pulse train: (a) $W = 8\ \mu m$ and VDD $= 0.8\ V$, (b) $W = 8\ \mu m$ and VDD $= 0.8\ V$, (c) $W = 2\ \mu m$, VDD $= 0.5\ V$ for Trials 1 and 4, and VDD $= 0.8\ V$ for Trials 2 and 3, and (d) $W = 4\ \mu m$ and VDD $= 0.8\ V$

signal that we attempt measuring with our experimental setup. The collected measurement data was interpolated as a sine wave with a period of 73 μs, an amplitude of 2.2 mV, a vertical offset of 2.2 mV and a horizontal shift of 1.7 μs.

On the other hand, gradually increasing the pulse train amplitude on node $A1$ in Fig. 5.1(a) from 400 mV to 500 mV increases the current on node E up to 6.5 nA, as shown in Fig. 5.8(b), Trial 1. This experiment was repeated under the same setup, and similar results were obtained (see Fig. 5.8(b), Trial 2). As long as we respected the 500 mV limit, we were able to reproduce these results even with similar cells in different copies of our chip.

5.3.1.3 Substrate Break-Down Voltage

The on-chip pulse train bus is shared among all cells with an enable terminal specific to each cell. To measure the breakdown voltage of the devices under test, a pulse train of variable magnitude was applied to a circuit randomly selected from those in Fig. 5.1. While enabling the target cell by holding its corresponding EN voltage low and disabling other cells by holding their respective EN voltages high, some probes were connected to different chip pads with variable distances to the transistor under stress in order to detect any changes in the drawn currents. Applying any substrate voltage in the range between 0.8 V and 2.5 V did not have any effect on the chip when either the applied pulse train amplitude was below 800 mV, or each of the gate-currents detected by the measuring probes was below 50 nA. When the gate-current of any MOSFET exceeded the 50 nA limit, changes in other on-chip currents were detected, with values varying according to pulse train amplitudes. Chips that were subjected to this test could be easily distinguished from others under the microscope by substrate burnouts or silicon bubbles. Figure 5.9 shows an example of one damaged test chip before and after the stress test was applied.

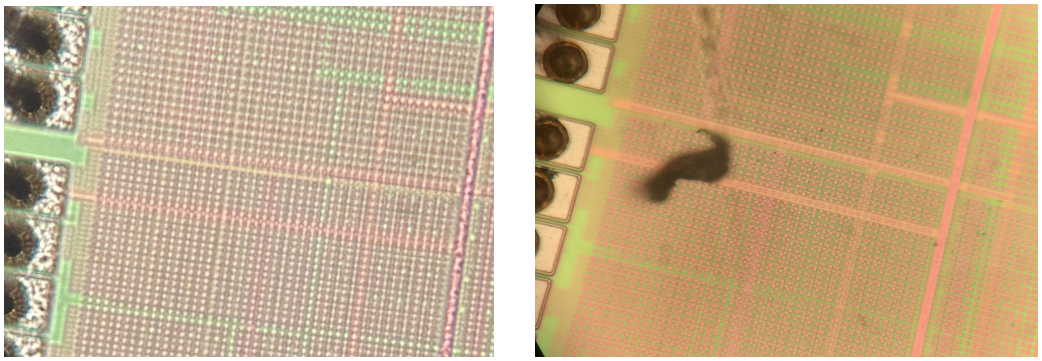
5.3.1.4 Oxide BreakDown Voltage

We experimented the cell of Fig. 5.1(a) with the following configuration for $M1$: $W = 2\ \mu m$ and $L = 60\ nm$. The gate dielectric thicknesses in the used technology are reported in Table 5.1. These thicknesses vary based on the corner in which the device falls after fabrication.

The plot of Trial1 in Fig. 5.8(c) shows the results of our first experimental trial in which we applied a maximum pulse train amplitude of 1.2 V and a *VDD* voltage of 500 mV to which the bulks of *p-type* MOSFETs are connected. In this test, the pulse period was 1 μs and the pulse width was 200 ns. We collected the current observed on terminal E of Fig. 5.1(a) by attaching an external 1 $M\Omega$ resistor to this node, and dividing the measured voltage by its corresponding resistance. The observed current started to increase at a pulse amplitude of 450 mV and reached its maximum (8 nA) when the amplitude became 1.2 V. This behavior was symmetrical when sweeping the pulse amplitude downwards from 1.2 V to 0 V.

In another experimental trial performed on the same circuit of Fig. 5.1(a), when we raised

VDD to 800 mV, a higher tunneling current was observed with a maximum of 15 nA at pulses' amplitude of 1.2 V. Yet, before the tunneled current reached 15 nA, where the pulse amplitude exceeded 650 mV, the device started a resistive response equivalent to a 40 $M\Omega$ resistor (see the curve of Trial 2 in Fig. 5.8(c)).



(a) (b)

Figure 5.9 An example of one the damaged test chips before and after the stress test: (a) before (b) after

All our setups followed standard conditioning for MOSFETs under the used process with a substrate voltage (*VDD*) close to 1 V and a maximum pulse amplitude less than or equals to 1.2 V. Nonetheless, repeating the last experimental trial, under similar conditions, revealed different measurement results from those previously obtained in Trials 1 and 2. The FGT transistor did not respond to the applied pulse train, even when increasing *VDD* from 500 mV to 800 mV. Figure 5.8(c) presents the new behaviors in the curves of Trials 3 and 4, where a maximum tunneling current of 2 nA is observed. The obtained current is 13 nA less than that measured in Trial 1. This current is even lower than the normal recorded output induced by pulse amplitudes lower than 400 mV, see Fig. 5.8(a). Interpreting these results leads to the conclusion that the dielectric layer was damaged. More precisely, it is believed that this happened when the tunneling current exceeded 8 nA.

To validate this assumption, another cell was placed under test. This cell was also a variant of Fig. 5.1(a) with $M1$ dimensions: $W = 4\ \mu m$ and $L = 60\ nm$.Under conditions similar to the previous experimental trials, Fig. 5.8(d) shows the output response of the first trial on this cell performed with $VDD = 800\ mV$. Therefore, by comparing the plots from Figs.

5.8(c) and 5.8(d), we can observe that:

- There exist a voltage threshold around 450 mV that is independent of MOSFET's dimensions. This threshold appears in multiple tests on different cells with different transistor dimensions ($W \in \{2\ \mu m, 4\ \mu m, 8\ \mu m\}$). It can be considered as the maximum voltage that can be trapped by a gate.

- Another threshold appears around 650 mV after which the output changed to a resistor-like behaviour. It starts when the value of the tunneled current is around 13 nA. The resistive behavior is an indicator of a change in the physical characteristics of the gate dielectric, and seems to be independent of transistor dimensions.

- The output response is symmetric on the up and down sweeps of the pulse train amplitude and is not reproducible once the tunneled current exceeds 13 nA.

- The measured *gate* leakage currents in the experimented circuits were in the range between 2 nA and 3 nA when the amplitudes were below 450 mV. Such low range results from normal Electro-Magnetic Interference (EMI) among on-chip components.

Figure 5.8(d) presents the results of a second experimental trial performed on the same transistor $M1$ in the previous experimental trial, where $W_{M1} = 4\ \mu m$. The measured current limited by 2.5 nA, see the curve of Trial 2 in Fig. 5.8(d), assures the change in gate dielectric properties of transistor $M1$.

5.3.1.5 Tunneling Independence on MOSFET Width

Another important characteristic of device's dielectric was revealed by comparing different measurements of all tests. The currents tunneled through FGTs' dielectric layers do not vary with the change in MOSFET dimensions. Figure 5.10 shows the variation in tunneled current versus the pulse sweep amplitude for different ITC configurations. As the amplitude increases, the tunneled current followed an exponential upwards trajectory independent of MOSFET's type and dimensions.

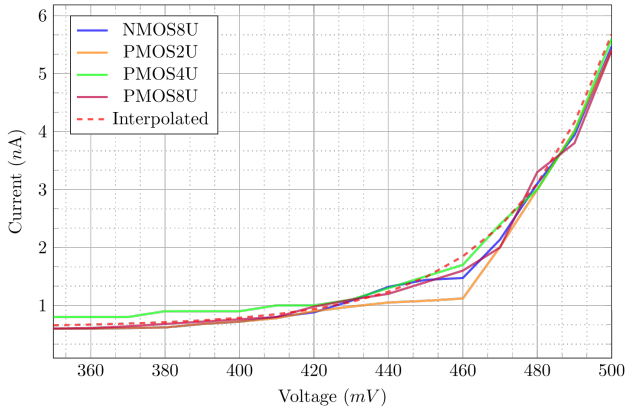
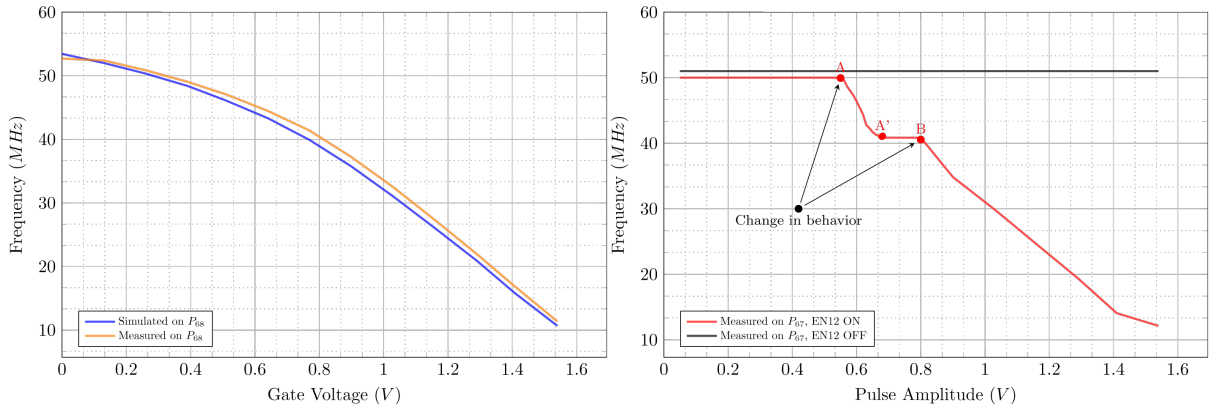


Figure 5.10 Variation of the tunneled currents interpolated from real-time measurements on all FGT variants vs the amplitude of the pulse train

5.3.2 Capacitive Cells

Additional FG features, like retention time, could be explored by experimenting the circuits of Figs 5.3 and 5.4.

5.3.2.1 VCO-based Verification



(a) (b)

Figure 5.11 Curves of measured and simulated VCO frequency output signals for CC in Fig. 5.4 with respect to voltage sweeps for (a) Vg , and (b) for amplitudes of the applied pulse-train

To verify the proper functionality of the VCOs in Fig. 5.4, a voltage sweep was applied on the gate of $M9$ (defined by V_g). The corresponding output frequency was measured on node

P_{68}. The measurement results are presented in Fig. 5.11(a) along with computer simulation results for the same cell. The two curves are very close to each other, descending from 52 MHz at $V_g = 0\ V$ to reach 12 MHz at $V_g = 1.6\ V$ which is $M13$'s threshold voltage.

Figure 5.11(b) shows two curves representing the measurement results collected for the floating-node VCO in the Capacitive Cell (CC) of Fig. 5.4. When the circuit was disabled by holding a high voltage on node $EN12$, no change in VCO2's output frequency on node P_{67} was noticed when sweeping the pulse-train amplitude. However after the circuit was enabled by forcing 0 V on node $EN12$, the output frequency dropped from 50 MHz at point A on the curve (see Fig. 5.11(b)) to 40.85 MHz at point A'. This frequency variation occurred for pulses' amplitudes increasing gradually from 550 mV at point A to 680 mV at point A'. The frequency remained stable at 40.85 MHz between points A' and B. At point B, when the amplitude reached 800 mV, another significant change in behavior was observed, and the VCO's output frequency started following that of node P_{68}.

Therefore, we can recognize different regions of operation in the curve of Fig. 5.11(b) which are compatible with the findings previously reported in this article. In each of these regions, both FGT's current and frequency curves are affected around the same voltages (550 mV and 800 mV), indicating changes in tunneling path(s). Around 550 mV (at point A in Fig. 5.11(b)), the change in VCO2's output frequency measured at node P_{67} is related to the observed increase, around the same pulses' amplitude, in transistor $M1$'s tunneling current either in the curve of Trail 1 in Fig. 5.8(d) or that of Trial 2 in Fig. 5.8(c). In addition, for the voltage range between 650 mV and 800 mV, the slight change (almost negligible) in VCO2's output frequency between points A' and B in Fig. 5.11(b) can be, once again, related to changes in slopes of the tunneled current curves of transistor $M1$ in the plots of Trails 2 and 1 in Fig. 5.8(c) and Fig. 5.8(d), respectively. Around 800 mV and beyond (after point B in Fig. 5.11(b)), VCO2's frequency started to decrease from 40 MHz to 12 MHz. When compared to the measured frequency curve on P_{68} in Fig. 5.11(a), both VCOs' frequencies are very close to each other. Moreover, the curves of $M1$'s tunneled currents in Trials 2 and 1, respectively in Fig. 5.8(c) and Fig. 5.8(d), show important changes in curves' slopes around this voltage (800 mV). We believe that around 800 mV one of the parallel leakage paths

between $M12$'s *drain* and C_2 broke making it possible for VCO2's frequency to decrease with a steep slope.

5.3.2.2 Retention Time

The threshold voltage for transistors $M5$ and $M6$ of Fig. 5.3 is between 350 mV and 565 mV depending on the post fabrication process corner. Measuring the retention time in these circuits was difficult with this high voltage threshold compared to the maximum possible voltage on C_1 that is around 450 mV. Nonetheless, one of the fabricated chips had a low threshold voltage for transistor $M6$, probably due to process variations. This was the only cell in which we were able to measure a trapped voltage on a floating-node. This voltage is detected due to the current traversing transistor $M6$ when no other on-chip cell or transistor was activated. The corresponding observed retention time for this device is 12 months at the time of writing this article. Figure 5.12 shows that transistor $M6$ current is 271.25 nA as recorded by ammeter $A1$ of Fig. 5.3 with a 1 $M\Omega$ in-series resistor. We were able to calculate the internal voltage of the floating-node from the MOSFET *drain*-current equation in *linear region* [114]:

$$I_D = \mu.C_{OX}.\frac{W}{L}.(V_{GS} - V_{TH}).V_{DS} \tag{5.2}$$

where I_D is the *drain* current, μ is the transistor mobility, C_{OX} is the oxide capacitance, W & L are the MOSFET width and length, respectively, V_{GS} is the potential difference between *gate* and *source* terminals of the MOSFET, V_{TH} is the MOSFET threshold voltage and V_{DS} is the potential difference between *drain* and *source* terminals. Table 5.2 summarizes the extracted parameters needed by eq. 5.2 which allows calculating that:

$$V_G - V_{TH} \simeq 8\ mV \tag{5.3}$$

In other words, V_G is very close to V_{TH} and can be estimated to be around 358 mV based on the discussion in section 5.3.A.1.

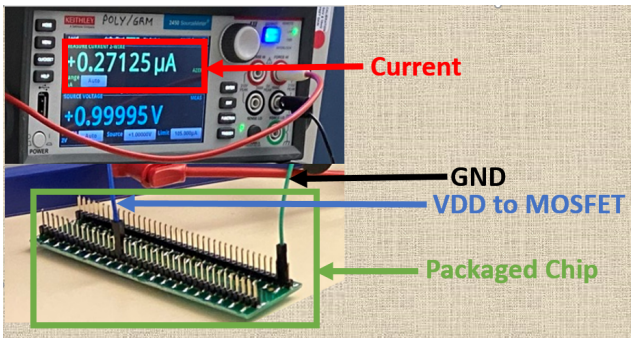


Figure 5.12 Picture of the test setup showing the drain current measured by ammeter $A1$ from Fig. 5.3

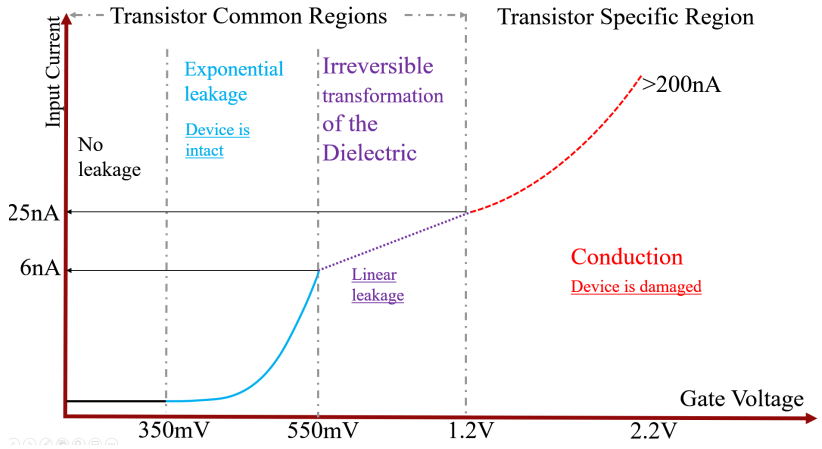


Figure 5.13 A general model representing the variation of the tunneled current vs the amplitude of the pulse train interpolated from physical measurements on floating gate transistors

5.4 Model

The plots of Fig. 5.10 show the effects of sweeping the pulse amplitude on different MOSFETs in the design with variable channel widths. Measured data proved that the current tunneled through MOSFET's dielectric is independent on channel width. All curves are exponential and started from below 1 nA (between 0.5 nA and 0.8 nA) at an amplitude of 350 mV and grew to reach their maximum (5 nA) at an amplitude of 500 mV. Using MATLAB interpolation tools, the polynomial equation of the trend line was approximated to a fourth-degree polynomial of the form

$$i(t) = a_4 v^4(t) - a_3 v^3(t) + a_2 v^2(t) - a_1 v(t) + a_0 \tag{5.4}$$

Where the polynomial coefficients are positive real numbers and are process related, $v(t)$ is the amplitude of the pulse train and $i(t)$ is the tunneled current. Using reverse Taylor expansion of the exponential equation:

$$e^{v(t)} = \sum_{n=0}^{\infty} \frac{v^n(t)}{n!} = 1 + v(t) + \frac{v^2(t)}{2!} + \cdots + + \frac{v^n(t)}{n!} \tag{5.5}$$

the curves can be modeled by following equation:

$$i(t) = I_0 e^{-Bv(t)} + 0.01Bv(t) + C \tag{5.6}$$

where I_0 is the initial current at time, $t = 0$, and $\{B, C\}$ are process related.

The plot of Fig. 5.13 summarizes the complete model inspired by the performed tests on 1 V MOSFETs in the chips. When the pulse amplitude is lower than 350 mV, the measured current is dependent on the exact circuit physical structure, and in this case, it is less than 1 nA. When the bias across the dielectric exceeds 350 mV, the current enters an exponential leakage region following eq. 5.6 until the curve enters a linear region when the pulse amplitude is in the vicinity of 550 mV for which a tunneled current of the order of 6 nA is observed. When entering this region, the device loses its ability to trap charges as the dielectric starts to develop a resistor like behavior with an equivalent resistance around 40 $M\Omega$. After crossing the 1.2 V threshold, which is the estimated maximum tolerated voltage for a 1 V MOSFET in a 65 nm process, the dielectric insulator starts to degrade. Therefore, at some point, when the insulator is completely damaged, the MOSFET's gate metal becomes connected to the substrate, thus creating a short circuit. In this region, the tunneled current is MOSFET specific, i.e. it changes according to the MOSFET's dimensions.

Combining these four regions summarizes the dielectric behavior in a 65nm process technology with 2 nm dielectric thickness. The first three regions are common among all transistors, independent of their dimensions, whereas the fourth region varies according to the dimensions

Table 5.2 The extracted parameters to apply in eq. (5.2) for estimating the gate voltage of MOSFET $M2$ in Fig. 5.4

Parameter	**Value**	**Reference**
I_D	$271.25nA$	Measured
μ	$398.7 \times 10^{-3} \frac{cm^2}{V.S}$	[114]
C_{ox}	$12 \frac{fF}{\mu m^2}$	[114]
W	16μ	Design
L	8μ	Design
V_{DS}	$0.73V$	Calculated

of the device under test. The device remains in this region until reaching a pulse amplitude high enough to cause substrate damage, hence the chip becomes unusable and unreliable.

5.5 Behavioural Hypotheses

Analyzing the gate insulator behavior based on the tunneled current in FGTs led to new behavioral hypotheses which might apply to all *sub* 70 *nm* processes with thin insulator films. The hypotheses formulated in this section, in relation to *sub* 70 *nm* processes with thin gate insulation, have not been addressed in detail in previous literature.

Hypothesis 1 (H1): *The physical characteristics of the gate dielectric change according to the applied voltage, or; in others words, they vary with the electric field traversing the dielectric.*

Corollary 1.1 (C1.1): *All transistors in sub 70 nm processes with thin gate dielectric lose their ability to trap charges when operating under nominal voltage way before they stop working as transistors.*

The observations supporting Hypothesis (H1) were detailed in section 5.4 and in the explanations of Fig. 5.13. More specifically, when the potential difference across the dielectric exceeds 600 mV, its physical characteristic transforms the material from a fully insulating to a resistive behavior. This leads to the direct corollary; C1.1, explaining the complete loss of charge trapping ability in FGTs caused by the high leakage current in the resistive gate to drain-channel-source material.

Hypothesis 2 (H2): *Charge Tunneling through thin gate dielectric insulators appears to take the shortest path from one terminal to another just like lightning sparks in the atmosphere. As this occurs at the atomic scale, it can explain the independence of tunneling current magnitude on MOSFET dimensions observed in numerous experiments.*

Hypothesis H2 was formulated as an attempt to explain the observed independence between the tunneled current and MOSFETs' dimensions in Fig. 5.13. When there exist a long contact area between both high and low voltage terminals, electrons will migrate from the highly negative side to the positive one following the shortest possible path.

5.6 Conclusion

In this article, Floating Gate transistors (FGTs) with thin gate dielectric insulator films were characterized using a 1 mm^2 chip fabricated under a standard 65 nm process. Despite normal challenges that accompany the process of characterizing FGTs, we were able to establish an FGT behavioral model that considers both the magnitude of an applied-pulse-train and the observed resulting tunneled-current.

Four regions of operation were discovered for a voltage range between 0 V and 1.2 V based on data extrapolated from chips' measurement results: (a) the no leakage region, (b) the exponential leakage region, (c) the linear leakage region, and (d) the dielectric break-down region. The first three regions were common among all tested transistors and the observed electrical behavior was found to be independent on their dimensions.

This study also presented an evidence of a successful charge trap using a fabricated device with over one year of retention time. Moreover, based on the obtained results, we proposed a couple of hypotheses which requires further analyses and studies before being generalized for other *sub* 70 nm processes.

CHAPTER 6 THE PROPOSED MEMRISTIVE CELL

6.1 High-Level Architecture

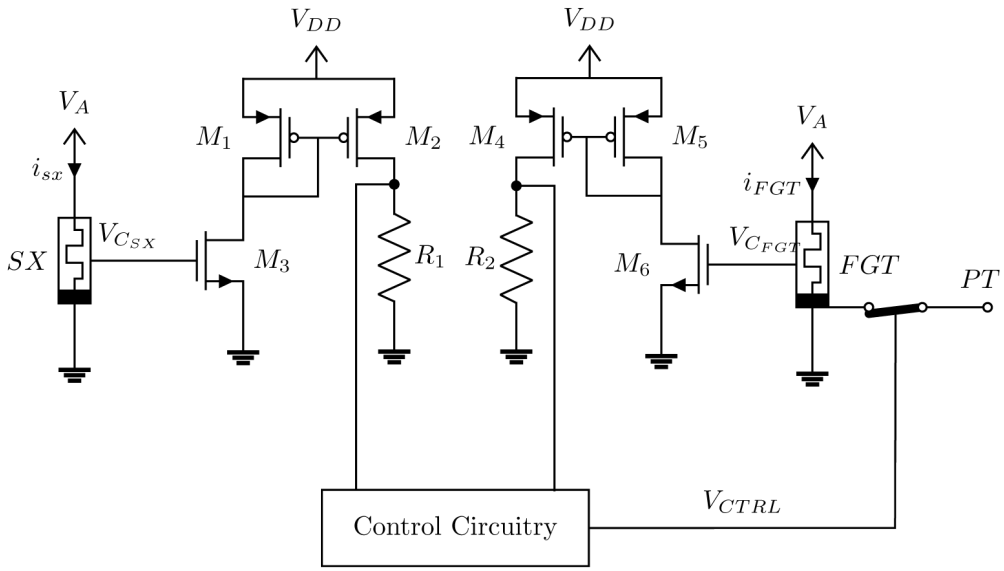


Figure 6.1 The general architecture of the proposed memristive cell

The circuit of Fig. 6.1 shows the general architecture of the proposed memristive cell, in which the SX component is a memristor emulator implemented in 65 nm CMOS process and adapted from [12] and the FGT represents a floating-gate cell that is discussed in previous chapters. The purpose of such memristive cell is copying $V_{C_{SX}}$ to $V_{C_{FGT}}$ since FGT has longer retention time (around 10 years) as compared to SX (around 1 μs). The voltage across the capacitor in SX, $V_{C_{SX}}$, controls the current traversing transistor M_3. This current is mirrored to R_1 and passed to the control circuitry.

On the other side of Fig. 6.1, and similar to $V_{C_{SX}}$, $V_{C_{FGT}}$ is mirrored to the control circuitry which, in turn, controls the flow of the charging/discharging Pulse Trains (PTs) through V_{CTRL}.

In this topology, the use of current mirrors is required in order to avoid non-gate leakage currents to the floating node in FGT. In addition, transistors M_3 and M_6 have zero-leakage currents on their gates. As the ultimate goal of the proposed cell is to match i_{FGT} with i_{sx},

transistors M_1, M_2, M_4 and M_5 must have the same W/L ratio, and so must M_3 and M_6. It follows that R_1 and R_2 should have equal resistance values as well.

In the following sections, we will discuss the proposed memristive cell thoroughly.

6.2 The Memristor Emulator

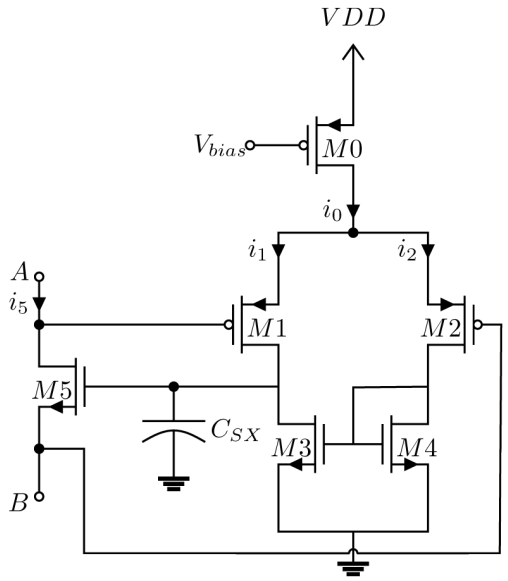


Figure 6.2 The schematic of the used memristor emulator which is adapted from [12] into 65 nm CMOS process

In order to implement the cell of Fig. 6.1, the memristor emulator implemented in [12] was adapted into 65 nm CMOS process. A detailed justification for choosing this emulator from the list of available emulators in literature is conducted in section 4.3.

In the proposed memristor, transistors M_1 to M_4 form a simple differential amplifier with M_0 controlling the branches' currents, i_1 and i_2, according to V_{bias}. The voltage across C_{SX} controls the conductance of M_5 which, in turn, affects M_5's drain current, i_5. As such, the memristive behavior of the circuit in Fig. 6.2 is directly dependent on the relations between V_{AB} and both: i_5 and $V_{C_{SX}}$. The voltage $V_{C_{SX}}$ can be referred to as the memristor state whereas the current, i_5, is the memristor current.

Figure 6.3 shows the computer simulation results of the circuit in Fig. 6.2. The memristor emulator response to a 1 MHz sinusoidal input on terminal A is a pinched hysteresis loop

for the current, i_5, and a cyclical hysteresis loop for the state, $V_{C_{SX}}$. The pinched loops are presented by Figs. 6.3(a) and 6.3(c) when the amplitude of the applied sinusoidal signal was 200 mV and 400 mV, respectively. Similarly, Figs. 6.3(b) and 6.3(d) correspond to the cyclical loops for the same amplitudes. It is important to note that the maximum voltage across C_{SX} is 550 mV when the amplitude of the applied sinusoidal signal is 400 mV. This voltage across C_{SX} does not exceed the FGT voltage threshold, as described in previous chapters. In other words, by copying $V_{C_{SX}}$ to $V_{C_{FGT}}$, FGT will operate in its exponential region, and it will not step into the linear leakage region where the device's behavior becomes incapable of trapping charges.

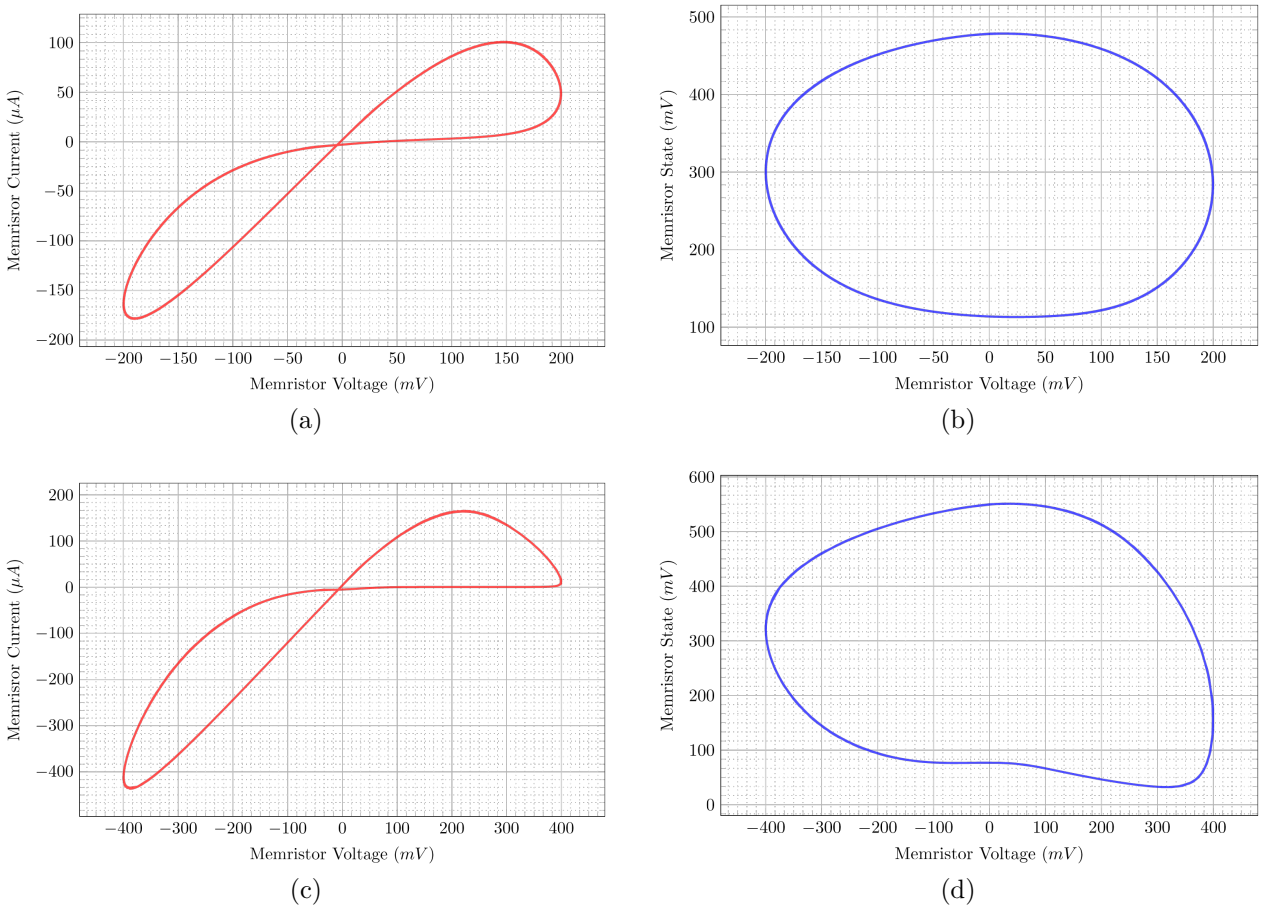


Figure 6.3 Simulation results of the memristor emulator in Fig. 6.2 showing both the pinched and the cyclical hysteresis loops when 0 V was applied on terminal B and a 1 MHz sinusoidal signal on A with: (a and b) 200 mV amplitude, (c and d) 400 mV amplitude

6.3 FGT Model

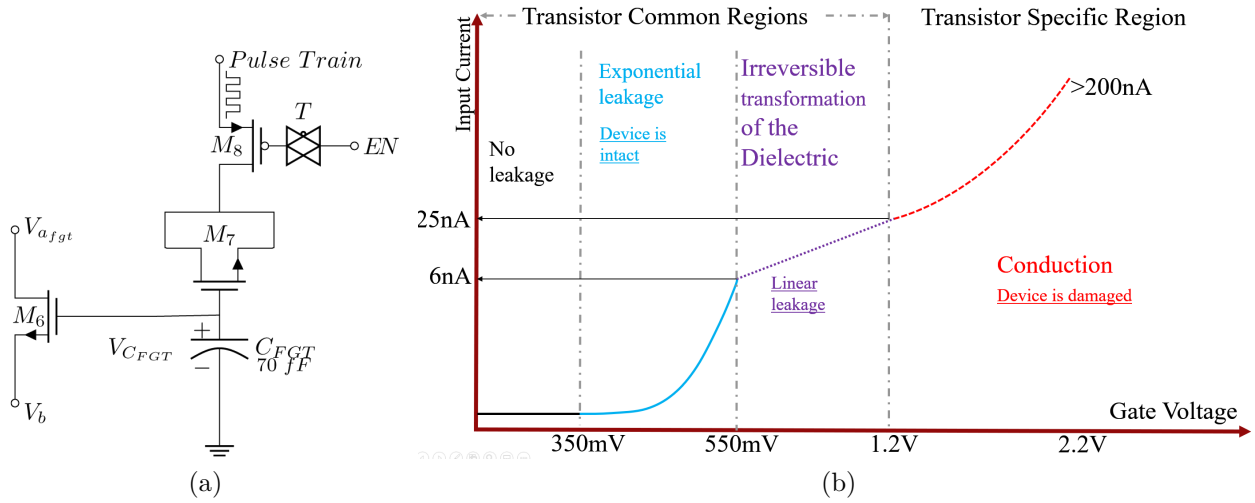


Figure 6.4 The basic FGT cell in our chips: (a) schematic (b) behavioral model

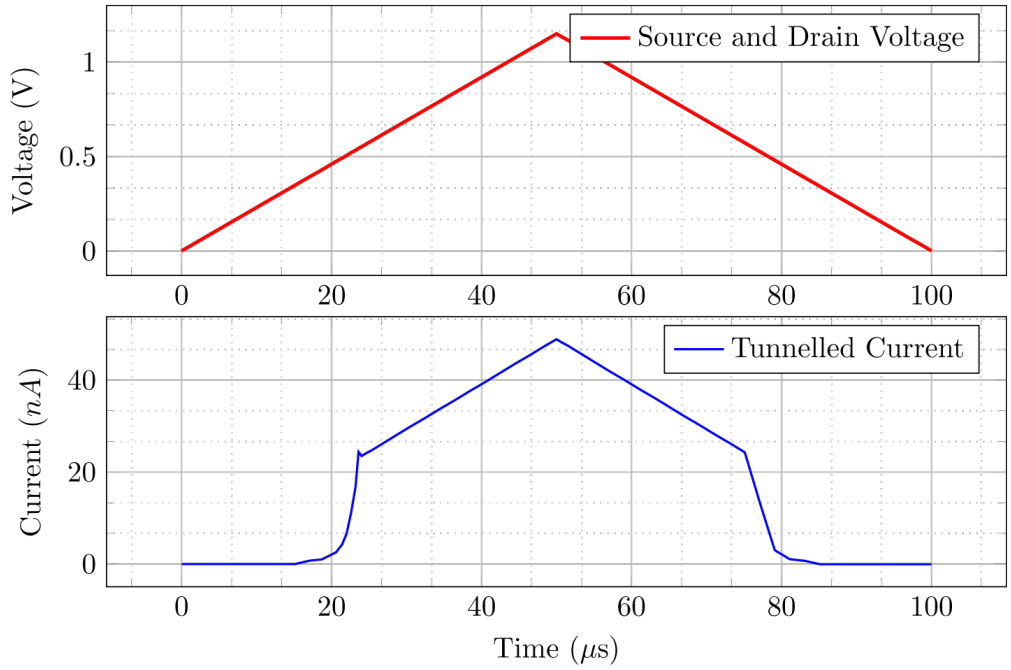


Figure 6.5 Computer simulation using VerilogAMS for the model in Fig. 6.4(b)

The circuit shown in Fig. 6.4(a) is the basic FGT cell that we chose to implement based on our characterization discussion in previous chapters. Computer models for simulating this circuit are rare, therefore we used VerilogAMS to simulate the device's model presented in Fig. 6.4(b).

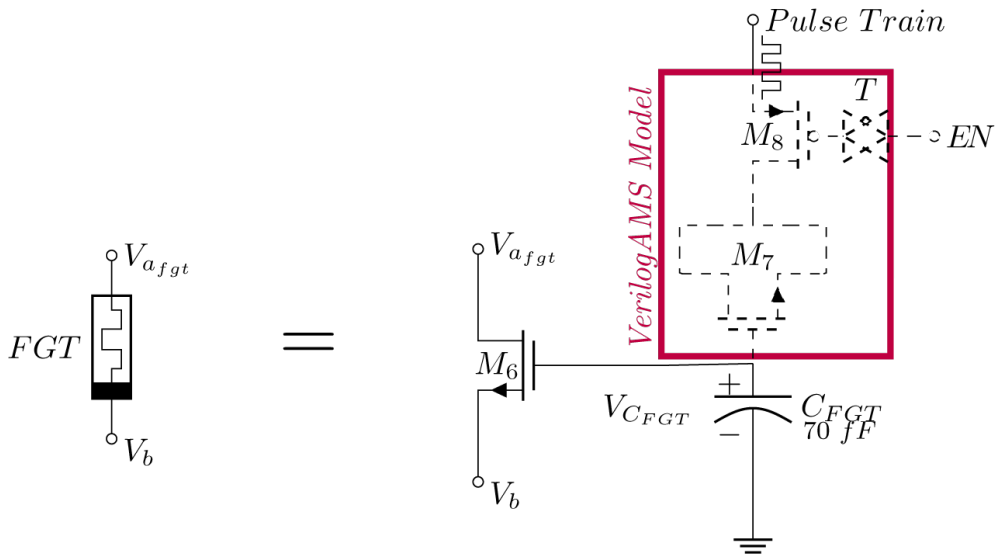


Figure 6.6 Schematic of the FGT cell used in computer simulation where the dashed components are replaced from the original circuit with VerilogAMS code

In order to implement the model presented in Fig. 6.4(b) using VerilogAMS, two assumptions should be taken into consideration:

- The dielectric behavior is symmetrical with respect to the the voltage polarity. That is when the gate voltage is higher than that of the *source* and *drain* connection, $V_{C_{FGT}}$ starts discharging according to the model in Fig. 6.4(b).
- The voltage difference across the transistor should not exceed 550 mV for both charging and discharging processes.

The VerilogAMS simulation for the FGT model, based on the aforementioned assumptions, is presented in Fig. 6.5. This model matches that predicted from device's characterization in previous chapters, and it will used in the rest of this chapter.

The circuit shown in Fig. 6.6 is used for simulating the FGT model presented in Fig. 6.4(b). The dashed components in the figure are replaced by VerilogAMS code for simulation purposes.

6.4 The Control Circuitry

6.4.1 The Pulse Train Generator Circuit

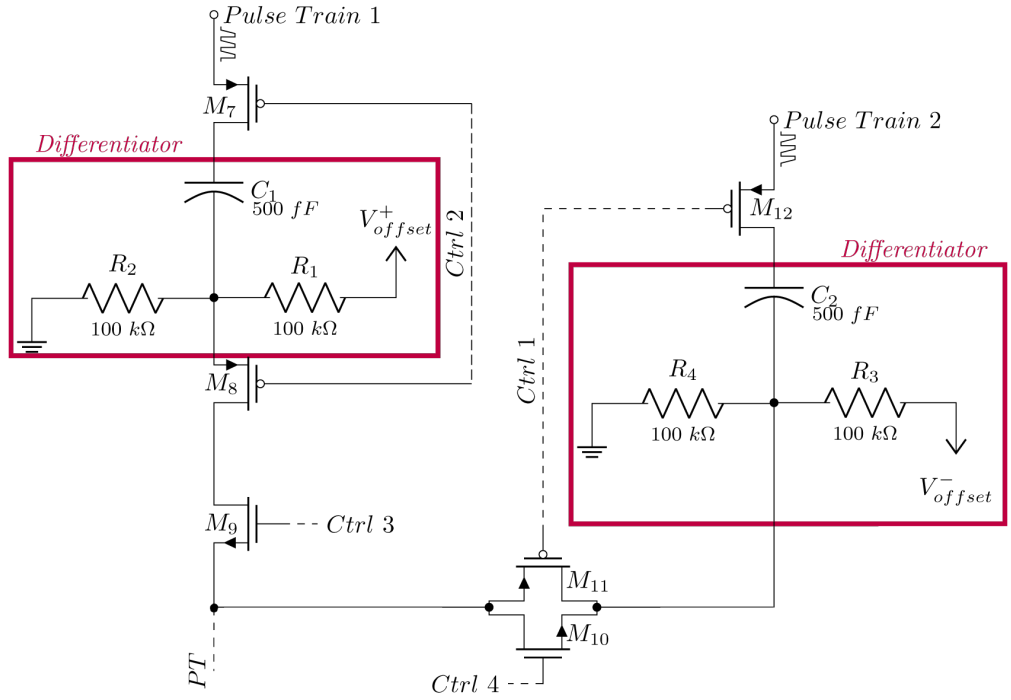


Figure 6.7 The PT generation circuit where the signals $Ctrl\ \{1-4\}$ are generated by some logic circuitry that will be explained later in this chapter

The circuit of Fig. 6.7 generates the PT signal in Fig. 6.1. The circuit contains two pulse trains, $Pulse\ Trian$ 1 and $Pulse\ Trian$ 2, which differ in their rising and falling times. $Pulse\ Trian$ 1 has short rising time and long falling time whereas $Pulse\ Trian$ 2 has long rising time and short falling time.

Both pulse trains are fed to two differentiator circuits, as labeled on Fig. 6.7. When the rising time is high, PT takes the shape of very short positive sparks. On the other hand, when the falling time is high, PT takes the shape of very short negative sparks. Generating such positive and negative sparks using the differentiator circuits is beneficial for three reasons: (a) it gives the decision circuitry more time to generate the proper control signals for transistors $M_{7-9,10-12}$, (b) it generates precise negative sparks for setups where no machinery can generate negative pulse trains, and (c) it helps in generating nano-second sparks since the laboratory equipment capable of generating similar pulses can be expensive.

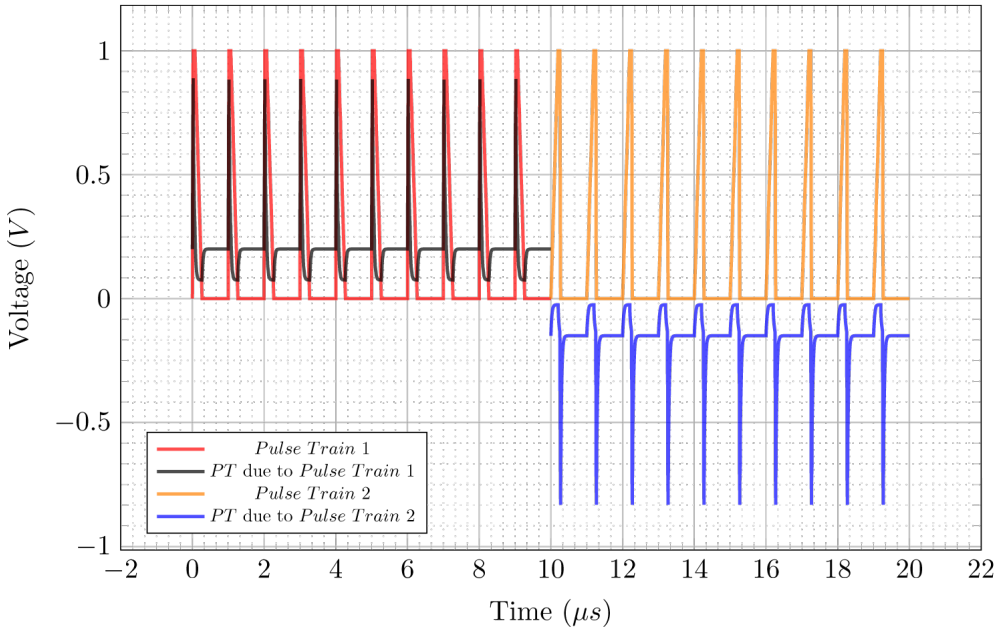


Figure 6.8 The PT output of the circuit presented in Fig. 6.7

Figure 6.8 shows the simulation results of the circuit presented in Fig. 6.7. When $Pulse\ Trian\ 1$ is applied with short rising time (20 ns) and long falling time (200 ns) for every pulse, the corresponding differentiator generates a positive pulse train PT with an amplitude proportional to the applied one and an $offset\ voltage = \frac{V^{+}_{offset}}{2} = \frac{0.4}{2} = 0.2\ V$. On the contrary, when the pulse train characteristics are reversed by reversing the rising and falling times for $Pulse\ Train\ 2$ (200 ns and 20 ns, respectively) with $offset\ voltage = \frac{V^{-}_{offset}}{2} = \frac{-0.3}{2} = -0.15\ V$, the corresponding signal generated on PT is a negative pulse train with very short pulse widths. Deciding the values of the simulation parameters (as rising time, falling time, V^{+}_{offset} and V^{-}_{offset}) are design and model dependent.

6.4.2 The Comparator

Figure 6.9 presents the comparator used in the memrisive cell. This comparator has a little hysteresis in it due to M_6 and M_7. The hysteresis range can be calculated given transistor dimensions and their respective threshold voltages.

In order to calculate the upper and lower hysteresis limits (V^{+}_{TRP} and V^{-}_{TRP}), the comparator should be analyzed at its switching, i.e. the points at which the output switches polarity. At

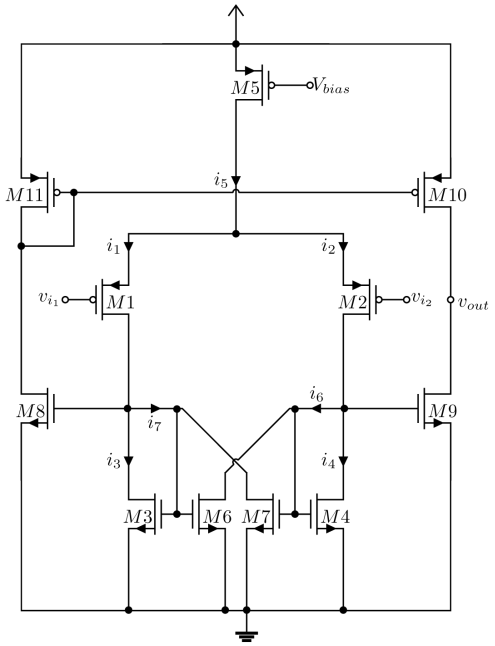


Figure 6.9 The comparator with hysteresis used in our chips

V_{TRP}^{+}, we assume that:

$$i_3 = i_1 \tag{6.1}$$

and

$$i_6 = i_2 \tag{6.2}$$

since M_4 is off. The current i_6 can also be calculated through the following equation:

$$i_6 = \frac{\beta_6}{\beta_3} i_3 \tag{6.3}$$

Consequently by assuming transistor M_7 to be OFF, the current i_5 can be written as:

$$i_5 = i_2 + i_1 = i_6 + i_3 = \frac{\beta_6}{\beta_3} i_3 + i_3 \tag{6.4}$$

Therefore, by rearranging the parameters, current i_3 becomes:

$$i_3 = \frac{i_5}{\frac{\beta_6}{\beta_3} + 1} \tag{6.5}$$

For the comparator of Fig. 6.9, the design parameters are: $W_6 = W_7 = 360\ nm$, $L_6 = L_7 = 60\ nm$, $\frac{\beta_6}{\beta_3} = 5$, $i_5 = 4.17\ \mu A$, $\mu_p = 61\ \frac{cm^2}{V.S}$ and $V_{tp_1} = V_{tp_2} = -270\ mV$. It follows that $i_1 = i_3 = 0.695\ \mu A$ and $i_2 = i_5 - i_1 = 3.475\ \mu A$. Assuming that MOSFET M_2 operates in saturation region, the voltage ($|V_{gs_2}|$) equals:

$$|V_{gs_2}| = \sqrt{\frac{2i_2}{\beta_2}} + |V_{tp_2}| = 453\ mV \tag{6.6}$$

Similarly, the voltage V_{gs_1} can be calculated to get:

$$|V_{gs_1}| = 352\ mV \rightarrow V_{TRP}^{+} = |V_{gs_2}| - |V_{gs_1}| = 101.4\ mV \tag{6.7}$$

And by following the same procedure, V_{TRP}^{-} can be estimated to $-101.4\ mV$ [115]. Figure 6.10 shows the simulation results of the used comparator in Cadence Virtuoso where $V_{TRP}^{+} = -V_{TRP}^{-} = 101.1\ mV$. The equivalent transient response is presented in Fig. 6.11. While setting v_{1_1} to $500\ mV$, sweeping v_{i_2} upwards from $0\ V$ to $1\ V$ and downwards from $1\ V$ to $0\ V$ stimulated a change in v_{out} with $V_{TRP}^{+} = 14\ mV$ and $V_{TRP}^{-} = -13\ mV$. The difference between the transient and DC responses of the circuit is normal and can be tolerated. This difference is due to the behavioral changes in devices' responses between both types of simulations.

Table 6.1 Truth table for generating the control signals in Fig. 6.7

$V_{C_{sx}} > V_{C_{fgt}}$	$V_{C_{sx}} < V_{C_{fgt}}$	Comparator 1	Comparator 2	Status
$True$	$False$	0	1	Charge
$False$	$True$	1	0	Discharge
Z	Z	0	0	Do Nothing
Z	Z	1	1	Do Nothing

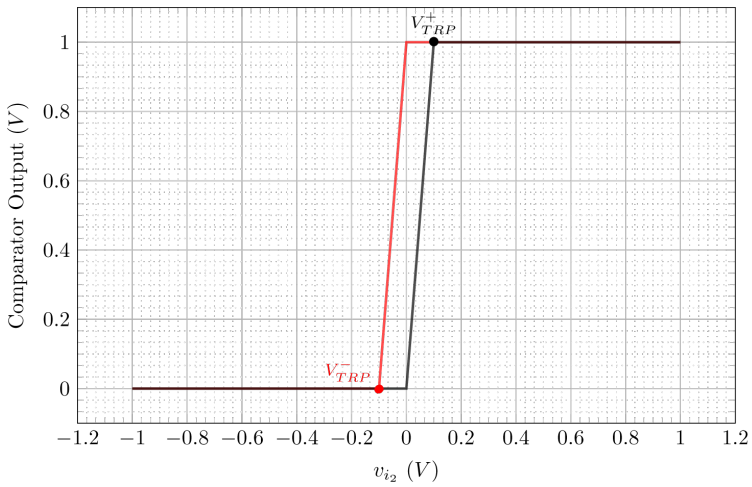


Figure 6.10 DC Response of the comparator with hysteresis used in our chips

6.4.3 Generating the Control Signals

In order to generate the control signals for the circuit shown in Fig. 6.7, the corresponding truth table is presented in Table6.1. Since the optimal goal is to level up $V_{C_{sx}}$ and $V_{C_{fgt}}$ in Fig. 6.1 for both discharging and charging phases, two copies of the comparator in Fig. 6.9 are required. The input voltages to both comparators need to be swapped; that is $v_{i_{1,comp\ 1}} = v_{i_{2,comp\ 2}}$ and $v_{i_{2,comp\ 1}} = v_{i_{1,comp\ 2}}$. This helps in generating the two different signals required to produce $Ctrl\ 1$, $Ctrl\ 2$, $Ctrl\ 3$, $Ctrl\ 4$ and an *enable* signal for the FGT model (called EN_{fgt}).

The voltages $Ctrl\ 1$, $Ctrl\ 2$ and EN_{FGT} are supposed to control p-type MOSFETs, whereas $Ctrl\ 3$ and $Ctrl\ 4$ control n-type MOSFETs. In this sense and based on Table6.1, the required signals can be written as :

$$EN_{FGT} = \overline{Comparator\ 1 + Comparator\ 2} \tag{6.8}$$

$$Ctrl\ 1 = \overline{\{Comparator\ 1 \cdot Comparator\ 2\}} \cdot Comparator\ 2 \tag{6.9}$$

$$Ctrl\ 2 = \overline{\{Comparator\ 1 + Comparator\ 2\}} + Comparator\ 2 \tag{6.10}$$

$$Ctrl\ 3 = \overline{\overline{Comparator\ 1} + Comparator\ 2} \tag{6.11}$$

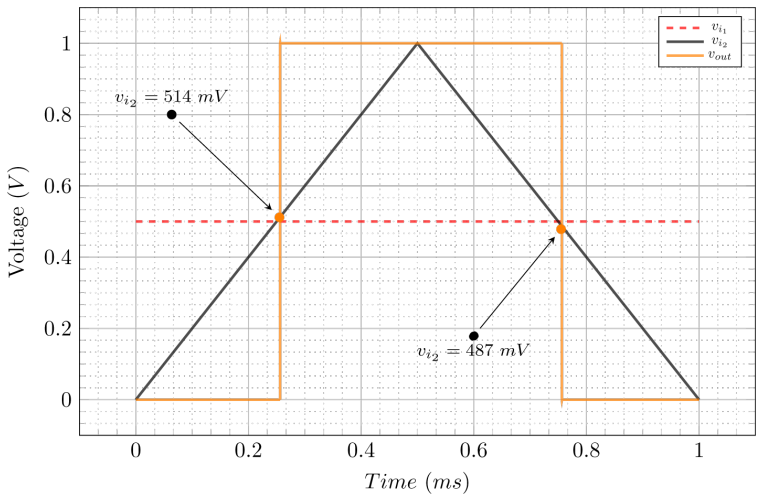


Figure 6.11 Transient Response of the comparator with hysteresis used in our chips

$$Ctrl\ 4 = \overline{\overline{Comparator\ 2} + Comparator\ 1} \tag{6.12}$$

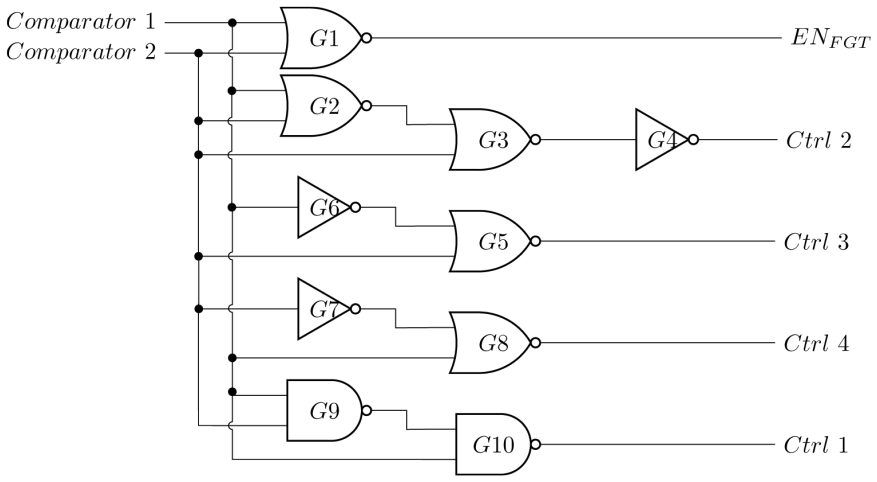


Figure 6.12 Decision circuitry that is responsible for generating the control signals of the pulse generator circuitry presented in Fig. 6.7

Figure 6.12 shows the decision circuitry that is responsible for generating the control signals in Fig. 6.7 ($Ctrl$ 1, $Ctrl$ 2, $Ctrl$ 3 and $Ctrl$ 4) in addition to the EN_{FGT} signal for the FGT device. The circuit is designed based on analyzing Table6.1 and based on equations 6.8 - 6.12.

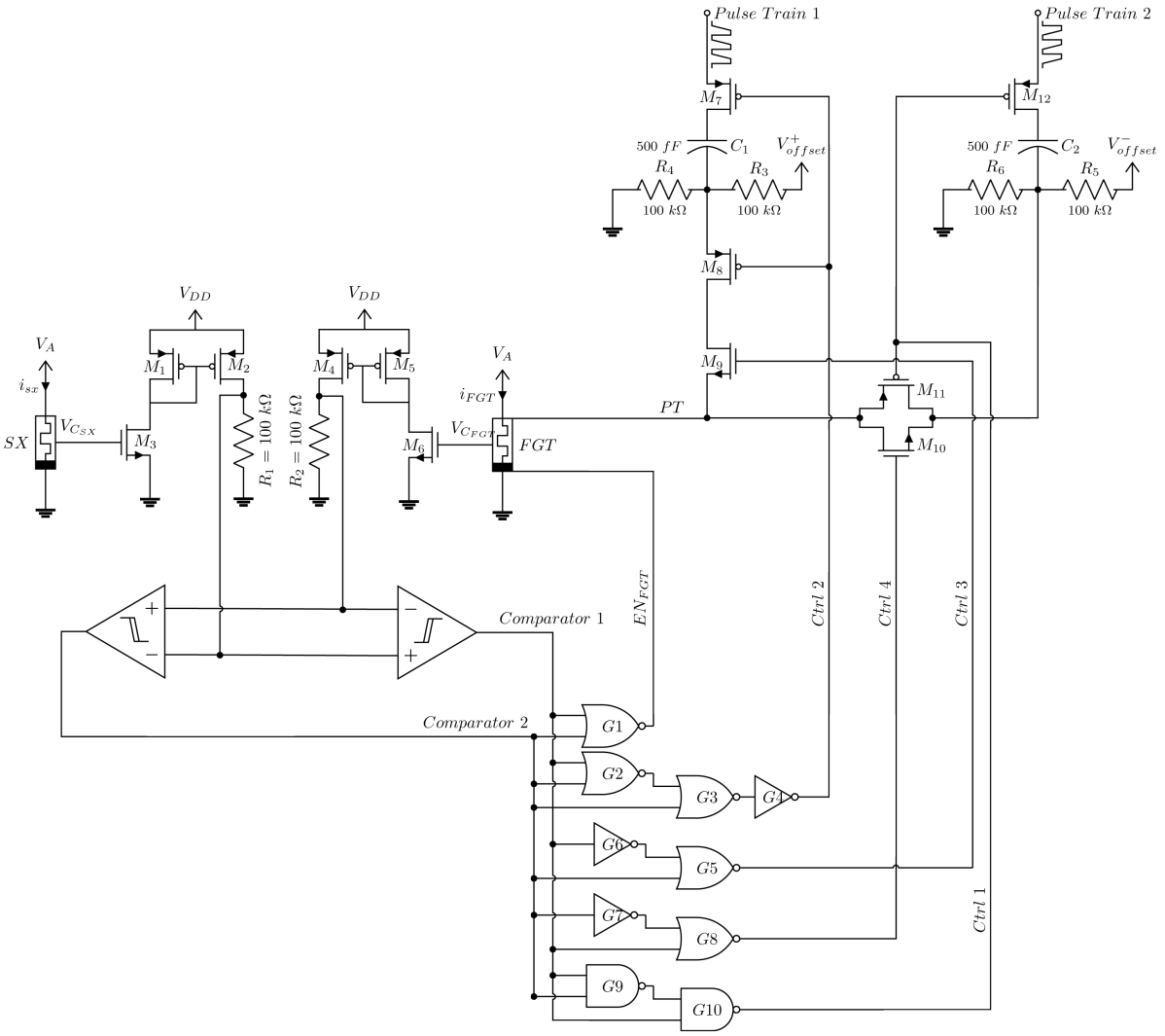


Figure 6.13 The detailed architecture of the proposed memristive cell which includes all sub-circuits explained in this chapter where each comparator is a copy of that in Fig. 6.9

6.5 Detailed Architecture of the Memristive Cell

This section presents the memristive cell overall architecture; detailed in Fig. 6.13, after integrating all sub-circuits discussed in this chapter. As aforementioned, the ultimate goal is to balance i_{FGT} with i_{SX}, $V_{C_{FGT}}$ with $V_{C_{SX}}$ or V_{R_2} with V_{R_1}. In the figure, each of the comparators with hysteresis is a copy of the circuit in Fig. 6.9.

When either comparator detects a change in either V_{R_1} or in V_{R_2}, a change in the output signals, *Comparator* 1 and *Comparator* 2, is induced and detected by the logic circuitry.

The latter takes a decision based on the detected change(s), and accordingly, the signals EN_{FGT}, $Ctrl\ 1$, $Ctrl\ 2$, $Ctrl\ 3$ and $Ctrl\ 4$ control the pulse generator to one out of three modes:

(i) Off mode: All branches of the pulse generator circuit are shut down and PT signal has a 0 V dc. In addition, EN_{FGT} is held high to 1 V. This happens when $V_{R_2} \approx V_{R_1} \implies V_{R_2} = V_{R_1} \pm \varepsilon$, where $\varepsilon \leq \frac{Hysteresis\ Voltage\ Range}{2}$.

(ii) $Charging$ mode: Only the $Pulse\ Train\ 1$ branch is activated (that is $Ctrl\ 1 = 1\ V$, $Ctrl\ 2 = 0\ V$, $Ctrl\ 3 = 1\ V$ and $Ctrl\ 4 = 0\ V$) and EN_{FGT} is held low to 0 V. As a result, PT takes the shape of positive pulses with some positive dc offset voltage ($\frac{V^{+}_{offset}}{2}$), see positive PT pulses in Fig. 6.8. This phase takes place when $V_{R_2} < V_{R_1}$.

(iii) $Discharging$ mode: Only the $Pulse\ Train\ 2$ branch is activated (that is $Ctrl\ 1 = 0\ V$, $Ctrl\ 2 = 1\ V$, $Ctrl\ 3 = 0\ V$ and $Ctrl\ 4 = 1\ V$) and EN_{FGT} is held low to 0 V. As a result, PT takes the shape of negative pulses with some negative dc offset voltage ($\frac{V^{-}_{offset}}{2}$), see negative PT pulses in Fig. 6.8. This phase takes place when $V_{R_2} > V_{R_1}$.

The behavior of this memristive cell is completely autonomous, and for simulation purposes, the FGT circuit is replaced by its VerilogAMS code, that is discussed earlier in this chapter. Figure 6.14 shows the transient response of this memristive cell and all its corresponding control signals. The simulation consists of three phases, two of which are charging and one is discharging.

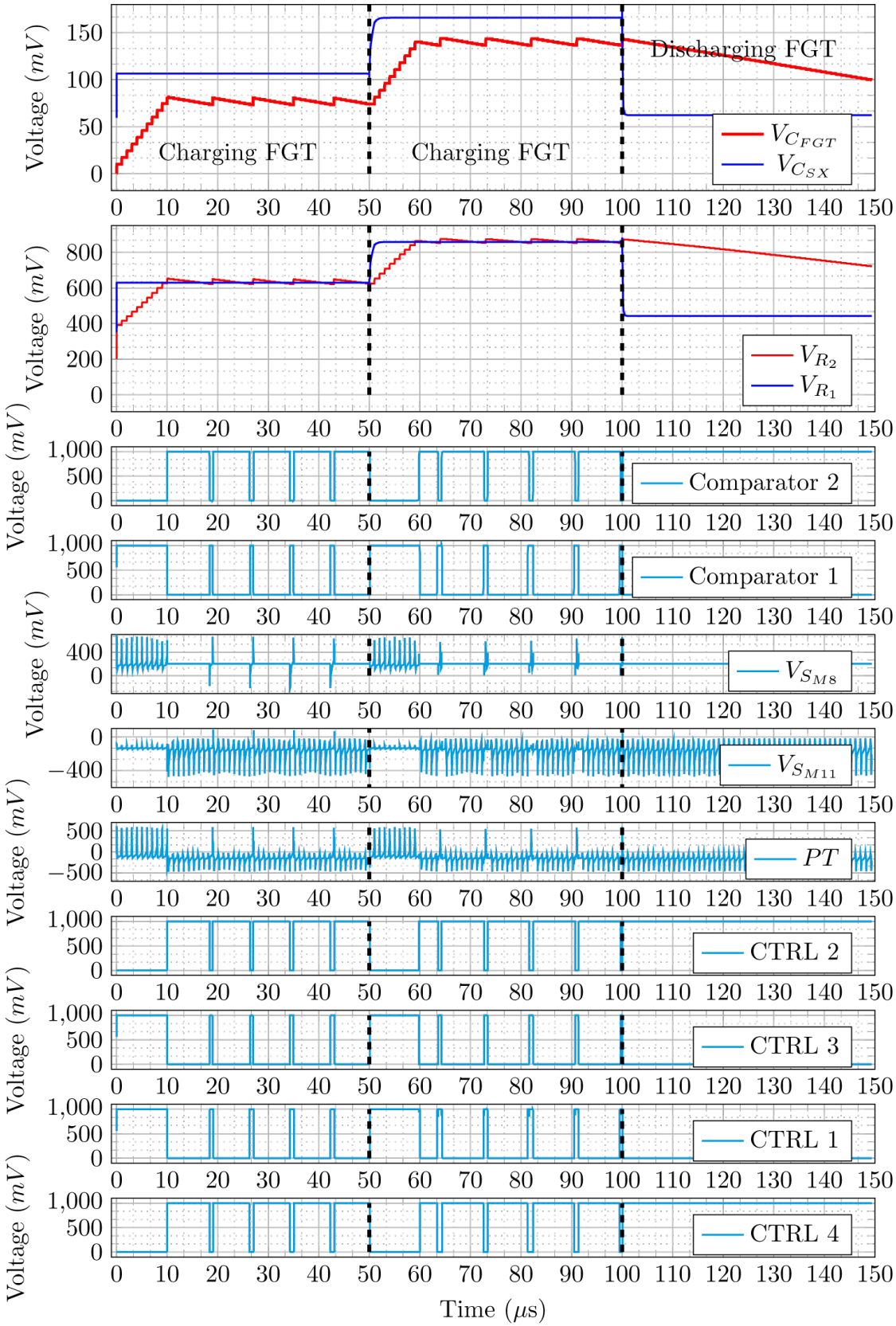


Figure 6.14 Transient response of the proposed memristive cell presented in Fig. 6.13

In this setup, $V_{C_{SX}}$ is considered as the reference voltage for $V_{C_{FGT}}$. However, we can only control $V_{C_{SX}}$ through V_A according to the plots in Fig. 6.3. Changing V_A (which is not shown in Fig. 6.14) causes the changes in $V_{C_{SX}}$, and based on $V_{C_{FGT}}$'s value, the comparators decide the operating mode of the pulse generator whether charging, discharging or off. Based on Fig. 6.14, it is clear that when $V_{C_{SX}}$ is larger than $V_{C_{FGT}}$, the decision circuitry orders a charge signal to the pulse generator, allowing PT to be positive. After around 12 μs, $V_{C_{FGT}}$ reaches $V_{C_{SX}} - \varepsilon$, a time instant at where FGT must retain its value with the help of other circuit components. Hence, we can notice the short charging signals between 10 μs and 50 μs. At 50 μs, when V_A changes, $V_{C_{SX}}$ changes accordingly, and consequently, the circuit re-enters the charging phase until $time \approx 62$ μs. Later, at $time = 100$ μs, $V_{C_{SX}}$ changes so that $V_{C_{SX}} < V_{C_{FGT}}$. This change forces the circuit to enter a discharging phase. It should be mentioned that, for design consideration, we configured the circuit such that the discharging time is longer than the charging time.

6.6 Summary

In this chapter, we presented an autonomous memristive cell that we proposed as an RSD substitute. Our proposed cell deploys the FGT behavioral model (presented in chapter 5) for long data retention time, and integrates this behavioral model with a memristor emulator (adapted from literature) for a memristor-like behavior of the cell. Such integration required additional circuitry to control the FGT's charging and discharging processes through charge tunneling.

Therefore, this chapter discussed all the components of our memristive cell with their respective simulation results using VerilogAMS. The simulation results were promising, and proved that such memristive cell can be deployed in different RSD applications. The cell has a 14 mV precision window between the two input voltages of the comparator, and a maximum in-cell storage voltage of 550 mV. In terms, this provides a voltage window range of 450 mV for the internal state of our proposed cell.

CHAPTER 7 INTEGRATING THE PROPOSED MEMRISTIVE CELL IN ANALOG COMPUTING FOR ARTIFICIAL INTELLIGENCE

In this chapter, we propose different useful architectures for the memristive cell shown in Fig. 6.13. Two main applications can be considered: (a) Vector Matrix Multiplication and (b) Non-Volatile Memory Arrays.

7.1 Vector Matrix Multiplication

Any application that belongs to the domain of analog artificial intelligence requires Vector Matrix Multiplication (VMM) as an essential component. As aforementioned in chapter 2, the main feature that characterizes VMM arrays is parallelism, which optimizes the computation time for the following weight equation in Deep Neural Networks (DNNs):

$$\overrightarrow{Y} = \sum W \cdot \overrightarrow{X} \tag{7.1}$$

where $\overrightarrow{Y}$ and $\overrightarrow{X}$ are the output and input vectors, respectively, and W is the weight matrix. Using VMM, equation (7.1) can be computed by substituting $\overrightarrow{X}$ with input voltages, W with the conductances in a VMM array and $\overrightarrow{Y}$ with the corresponding output currents. As a result, equation (7.1) can be written as:

$$\overrightarrow{I(t)} = \sum G(t) \cdot \overrightarrow{V(t)} \tag{7.2}$$

where $\overrightarrow{I(t)}$ is the output current vector at a time instant t, $\overrightarrow{V(t)}$ is the corresponding input voltage vector and $G(t)$ is the conductances of the VMM array used at the time instant t.

In a VMM array, every crossbar intersection represents a resistive device that should be able to hold its weight long enough until the operation is executed. For the cell presented in Fig. 6.13, this weight is $V_{C_{SX}}$. After this voltage is set during the programming (learning) phase, it must be copied to $V_{C_{FGT}}$. The voltage $V_{C_{FGT}}$ is used to run DNNs during prediction phase, not the learning phase.

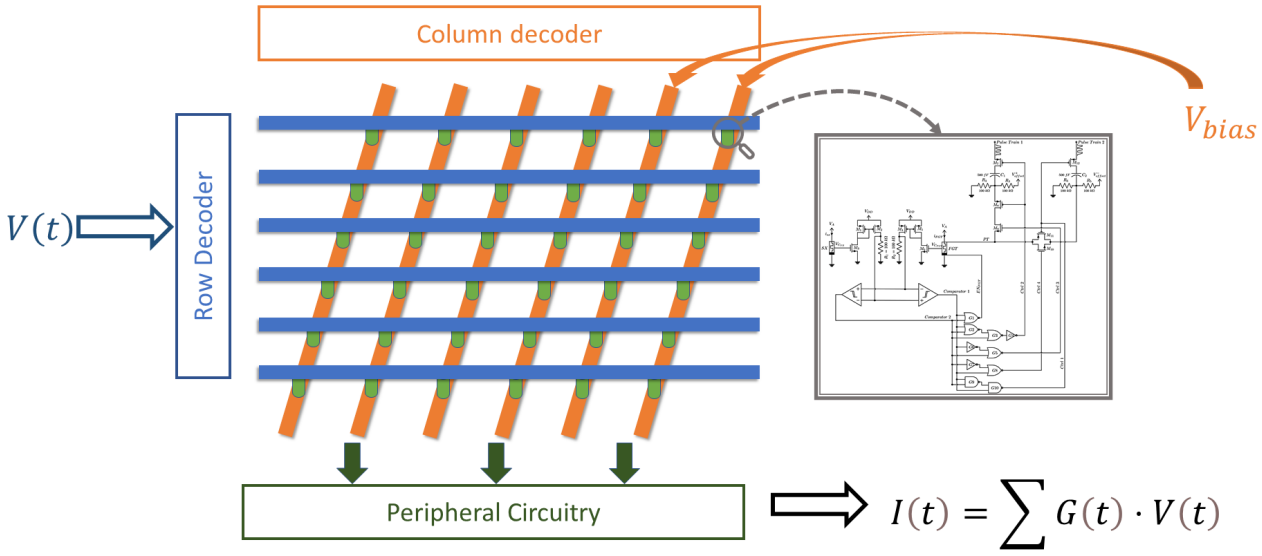


Figure 7.1 Schematic of one crossbar layer where at every crossbar intersection is the memristive cell of Fig. 6.13

7.1.1 VMM using Complete Memristive Cells

Figure 7.1 presents the schematic of one VMM layer configured to implement Deep Neural Network (DNN) algorithms using Dot Product Engine (DPE) that is presented in [4]. Every column is connected to a separate bias voltage V_{bias} which, in turn, is connected to the low voltage terminals of both, the memristor emulator circuit and its FGT counterpart in the same cell in Fig. 6.13, i.e. nodes B and V_b in Figs. 6.2 and 6.6, respectively. The difference among these bias voltages guarantees that each memristive cell is programmed to a different voltage level when the vector $V(t)$ is applied. Otherwise, the crossbar will include some redundant cells since all memristive cells in the same row will possess similar conductances after being subjected to the same terminal voltages. The row and column decoders are used to enable the targeted cell according to the $1T1M$ (1-Transistor-1-Memristor) structure presented in the DPE algorithm. For DPE, it is a requirement that the all weights ($G(t)$) should be pre-computed prior to programming the VMM.

7.1.2 VMM using Shared Memristor Emulator

In order to increase the cell density per one VMM layer on the expense of adding latency due to sequential processing, the memristive cell of Fig. 6.13 can be reconfigured to share one memristor emulator per a row of FGT cells. The drawing presented in Fig. 7.2 illustrates this configuration and the shared memristor emulator is a copy of the circuit shown in Fig. 6.2. In addition to this shared emulator, every row of FGT cells shares the same control circuitry that is presented in Fig. 6.1 and explained in details in Fig. 6.13. The mentioned control circuitry is integrated into the row decoder. The crossbar array configuration presented in Fig. 7.2 can also be efficient for data storage and memory applications.

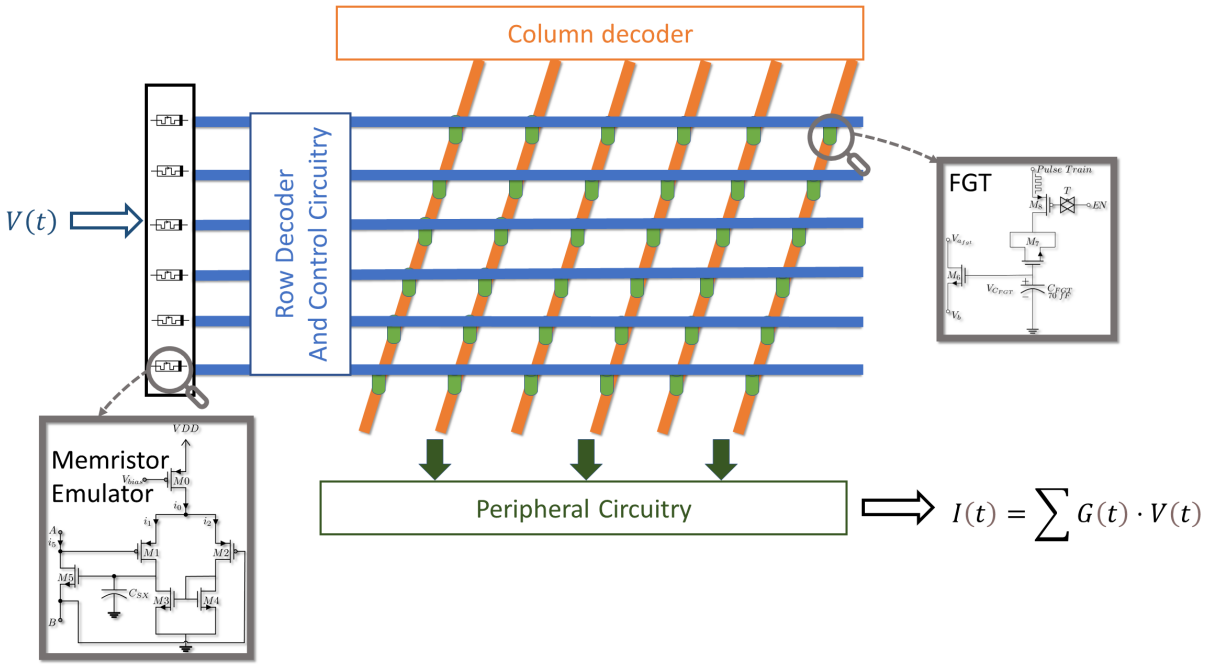


Figure 7.2 Schematic of the VMM layer shown in Fig. 7.1 optimized for density

7.2 Configuration

7.2.1 Weight Update

The process of weight update is significant for DNN learning phase since the main purpose of this phase is to keep adjusting connection weights among different neural network nodes until the targeted algorithm converges to a solution. In the memristive cell presented in Fig. 6.1, the FGT device will be the bottleneck of the weight update procedure; more specifically $V_{C_{FGT}}$.

As learned from chapter 5, the voltage range for $V_{C_{FGT}}$ is 430 mV before the FGT looses its ability to trap charges, (see Fig. 5.13). This voltage range is the difference between transistor M_6 threshold voltage, which is 120 mV, and 550 mV, the voltage around which the exponential leakage region for this particular FGT ends (see Fig. 5.13). The 430 mV voltage range can be quantized into 14 levels of accuracy with around 30 mV per level. The number of quantization levels is determined based on the comparator hysteresis range, which is around 27 mV. The 14-bit-accuracy weight might not be beneficial for analog implementation of DNN algorithm, but it is very useful for non-volatile memory applications.

Analog implementation of DNN algorithms includes two types of weight computation: (a) Precomputed weights and (b) learned weights. For algorithms like DPE, where weights are precomputed, the conductance of each cell can be approximated to a quantization level based on the final estimation of the currents per column according to the equation:

$$\sum_{j=1}^{\Im} I_j(t) = \sum_{j=1}^{\Im} \sum_{i=1}^{\Psi} M_{ij}^{-1}(r,t) \cdot V_i(t) \tag{7.3}$$

where $\Im$ and Ψ are the numbers of columns and rows, respectively, and $M_{ij}^{-1}(r,t)$ is the conductance (weight) at node i and row j. In such process, the goal is to program $M_{ij}^{-1}(r,t)$ to the targeted weight. On the other hand, for weight learning, the goal is to program the conductances per column until the targeted $I_j(t)$ is reached for a simple algorithm, where $I_j(t) = \sum_{i=1}^{\Psi} M_{ij}^{-1}(r,t) \cdot V_i(t)$.

7.2.2 Programming Procedure for Precomputed Weights

When the weights are precomputed, each node on the crossbar structure is programmed to a certain conductance according to the following steps:

(i) Prepare input voltage vectors, $V_i(t)$.

(ii) Compute output currents, $I_j(t)$, according to the desired application.

(iii) Calculate $M_{ij}^{-1}(r,t)$ per node. The weight of each node should be determined based on $V_i(t)$, $I_j(t)$ and column bias voltages $V_{bias,j}(t)$.

(iv) Apply the bias voltages per column.

(v) Apply input voltage vector, and using the row and column decoders, iterate over the crossbar array nodes.

(vi) For every column, iterate over its cells and, in steps of 20 mV, increase or decrease the input voltage ($V_i(t)$) until the targeted cell conductance is reached. In other words, keep looping until $\hat{I}_{i,j}(t) = I_{i,j}(t) + \epsilon$, where $\hat{I}_{i,j}(t)$ is the measured current due to the cell at row i and column j, $I_{i,j}(t)$ is the desired precomputed current and ϵ is the tolerated error per cell.

(vii) Activate the copy phase to translate the programmed conductances in the memristor emulator to the FGT counterpart based on the circuit architecture.

7.2.3 Programming Procedure for Weight Learning

The process of weight learning differs from that of weight precomputing. In weight learning, only input and output vectors, $V(t)$ and $I(t)$, are known. Yet, the two vectors must not be externally adjusted by the user. The column bias-voltages should be the only controllable circuit elements by the user. In short, the steps to program a crossbar layer during a weight learning process are the following:

(i) Prepare input voltage vectors, $V_i(t)$.

(ii) Create an initial mapping function to map the output currents, $I_j(t)$, to the desired output vector. A simple example of such function is 1 nA for 1 unit of measurement.

(iii) Identify ϵ value which is the error tolerance for the current per crossbar column, $I_j(t)$.

(iv) Apply the bias voltages to all columns of the crossbar array.

(v) Apply input voltage vector.

(vi) Collect $\hat{I}_j(t)$ for all columns.

(vii) If $\hat{I}_j(t)$ is below the circuit measurable current threshold, adjust $V_{bias,j}(t)$ so that every $\hat{I}_j(t)$ is above the mentioned threshold and can be detected by other circuit components.

(viii) Compare $\hat{I}_j(t)$ to $I_j(t)$, where $\hat{I}_j(t)$ and $I_j(t)$ are respectively the measured and the targeted currents. If $\hat{I}_{i,j}(t) > I_{i,j}(t) + \epsilon$, adjust $V_{bias,j}(t)$ in steps of 20 mV, either upwards or downwards until $\hat{I}_{i,j}(t) \leq I_{i,j}(t) + \epsilon$ holds for every j.

(ix) In cases where it becomes impossible to bring the value of $\hat{I}_{i,j}(t)$ near $I_{i,j}(t) + \epsilon$, adjust the mapping function in step (ii) and go back to step (iii).

7.2.4 Design Consideration for $V_{bias}(t)$

The column bias-voltages, $V_{bias}(t)$, are critical for VMM training, especially in cases where matrices are treated like black boxes, and the controllable external factors are few. In order to learn VMM weights, the voltage range for $V_{bias}(t)$ should be determined so that it is possible to store any learned weight inside the proposed memristive cell without damaging the gate dielectric.

As learned from chapter 5, the maximum voltage that a memristive cell can store is around 550 mV. This voltage is a safe limit we chose to assure the integrity of the gate-dielectric. However, this voltage limit can be relaxed by a 100 mV in order to increase $V_{bias}(t)$ voltage range and provide more flexibility on the crossbar structure.

To study the effect of changing $V_{bias}(t)$ on the memristive cell, two main characteristics should be respected: the cell's memristive behavior and its capability to retain charges for

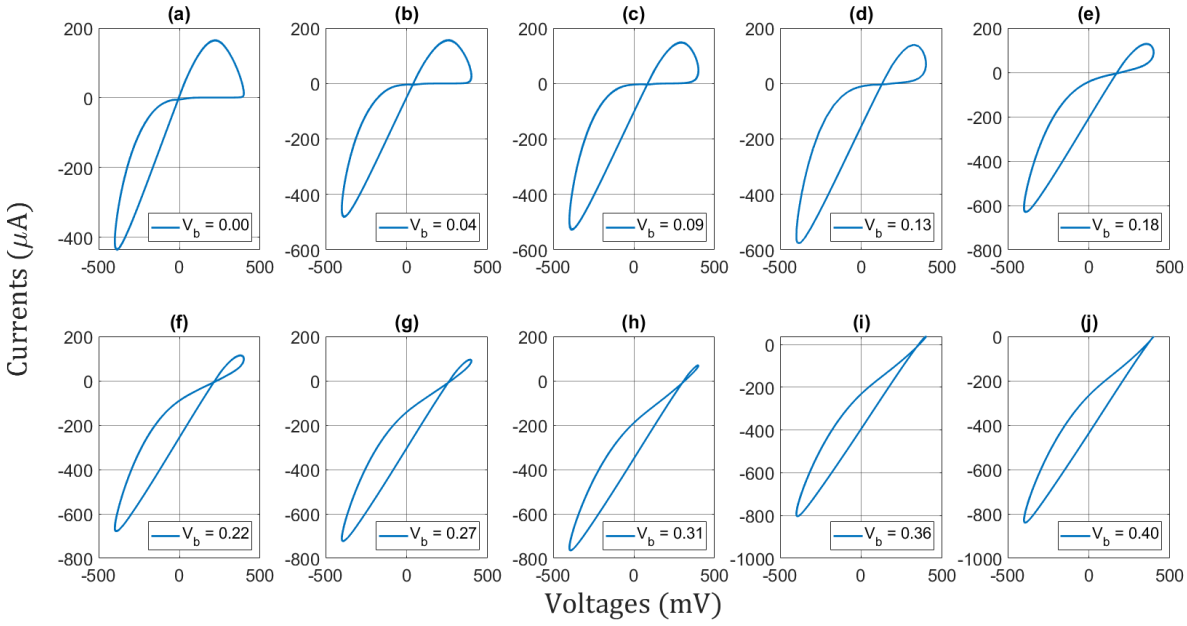


Figure 7.3 Transient current responses of the proposed memristive cell presented in Fig. 6.13 for different column bias-voltages swept between 0 V and 400 mV

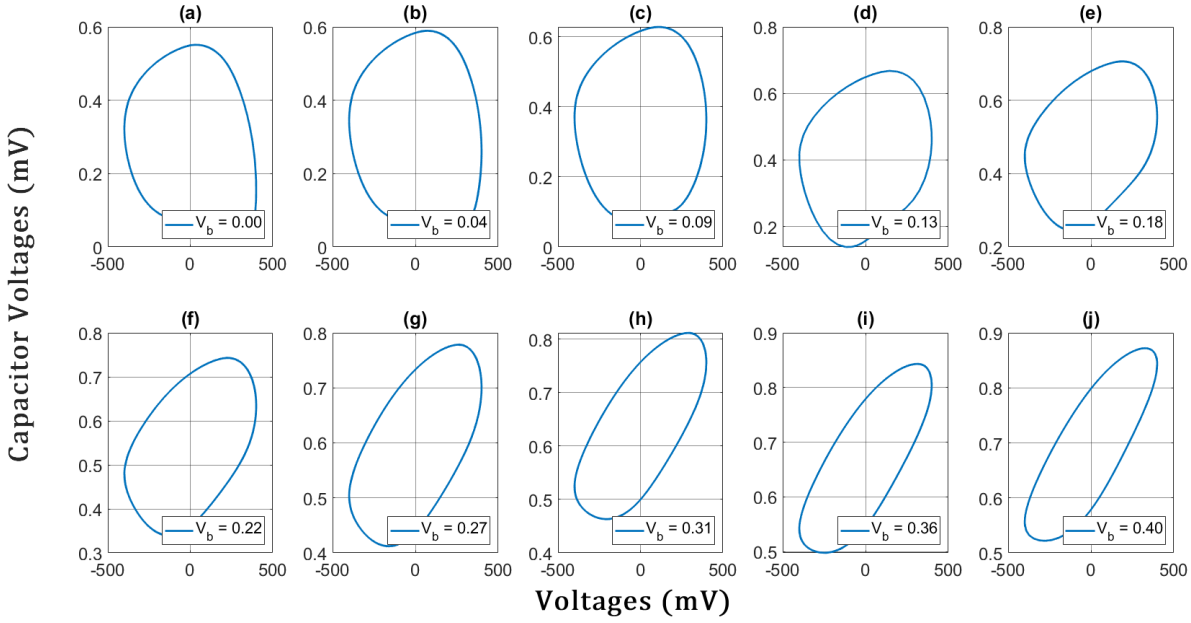


Figure 7.4 Transient state responses of the proposed memristive cell presented in Fig. 6.13 for different column bias-voltages swept between 0 V and 400 mV

long periods of time. As for the latter, such characteristic holds as long as $V_{C_{FGT}}$ is less than 650 mV beyond which FGT's gate dielectric starts to deteriorate.

For the other characteristic, Figs. 7.3 and 7.4 shows how a sweep on $V_{bias}(t)$ does not affect the memristive behavior of the cell. Figure 7.3 presents different current responses for the memristive cell when V_{bias} is swept between 0 V and 400 mV. The cell retains a pinched hysteresis response as long as V_{bias} is less than 310 mV. On the other hand, Fig. 7.4 shows cyclic state responses of the cell for all V_{bias} values. However, when V_{bias} exceeds 130 mV, the maximum state voltage surpasses 650 mV, a value which renders the FGT vulnerable.

In short, the best configuration for the proposed memristive cell is for a V_{bias} less than or equal to 130 mV, and all the crossbar columns must be subjected to a $V_{bias} \leq 130\ mV$.

7.3 A Proof of Concept

This section presents preliminary simulation results for the circuit shown in Fig. 7.1. A 4x4 array is simulated in which every crossbar intersection is a copy of the circuit shown in Fig. 6.13. For simplicity, all the cells are activated at once instead of having row and column decoders. In addition, every cell is autonomous in the sense that its $V_{C_{FGT}}$ is automatically following $V_{C_{SX}}$.

Figure 7.5 summarizes all input vectors, $V_i(t)$, to the array for a period of 1.5 ms. Each $V_i(t)$ randomly changes its value every 50 ms over an input range between 20 mV and 250 mV. On the other hand, the column bias-voltages, $V_{bias,j}(t)$, are programmed to acquire a DC voltage over the whole simulation period with $V_{bias} = [0\ V, 20\ mV, 60\ mV, 100\ mV]$.

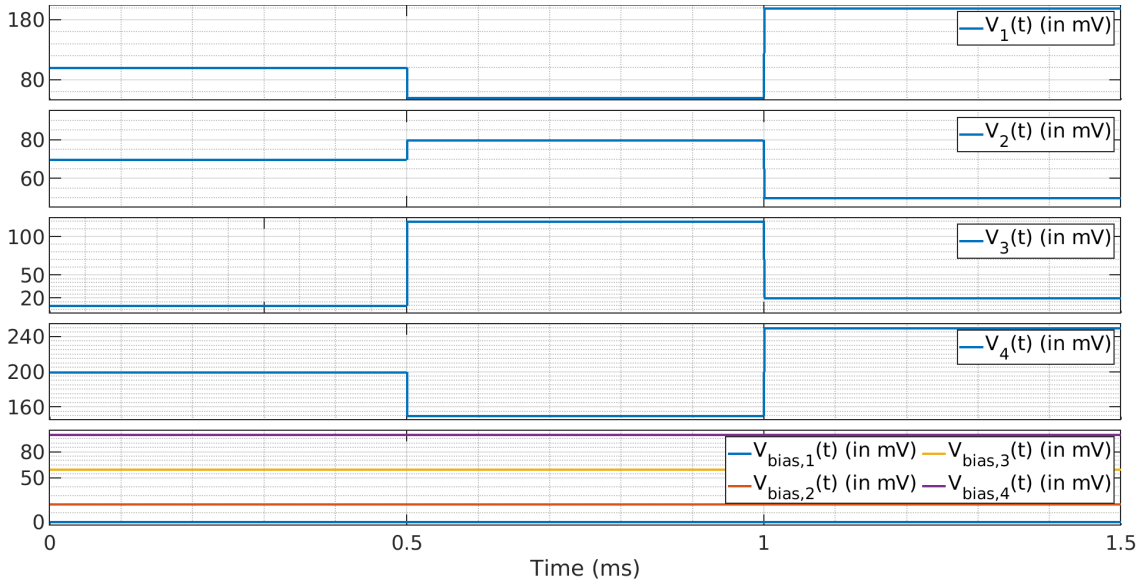


Figure 7.5 Transient input voltages, $V_i(t)$ and $V_{bias,j}(t)$, to the 4x4 crossbar array

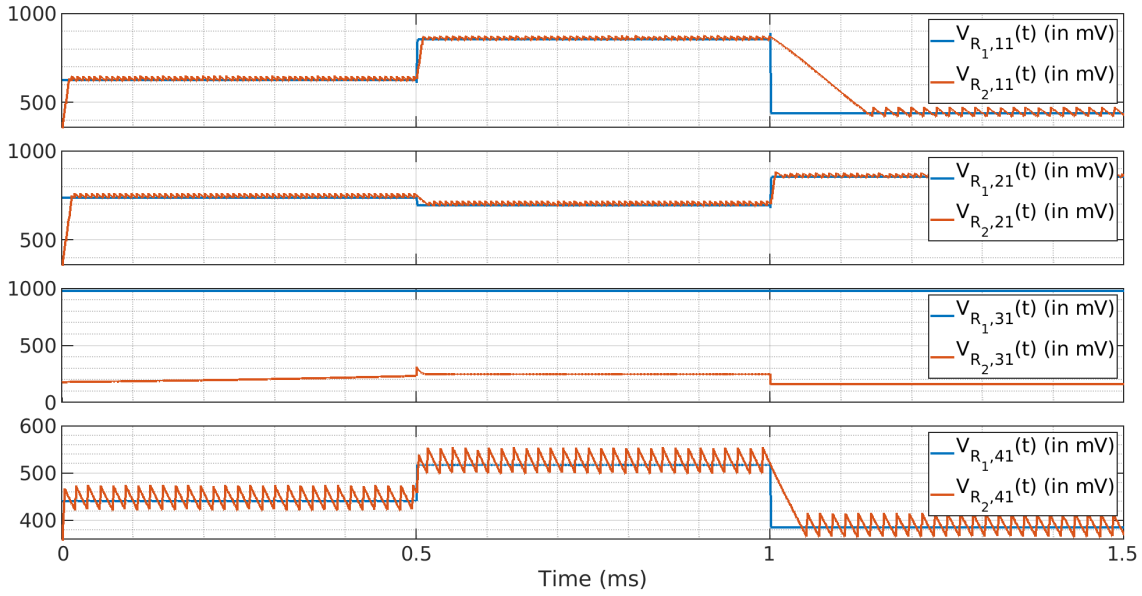


Figure 7.6 Transient input voltages, V_{R_1} and V_{R_2} (see Fig. 6.13), to the comparators of the memristive cells located in the first column of the 4x4 array for a $V_{bias,1}$ of 0 V and different values of $V_i(t)$ as shown in Fig. 7.5

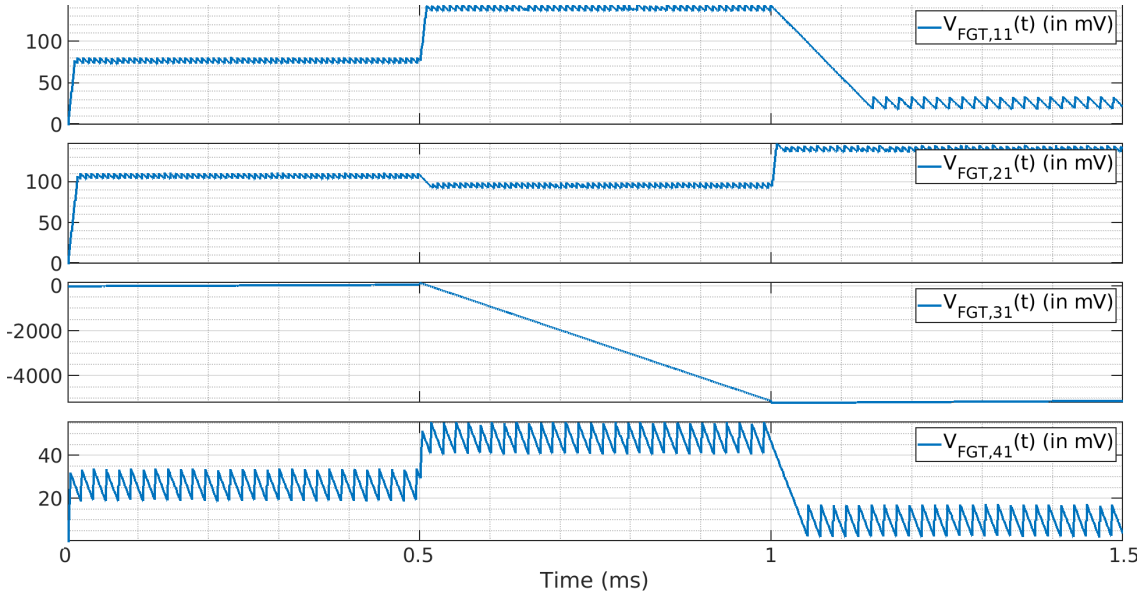


Figure 7.7 Transient FGT capacitor voltage responses of the memristive cells located in the first column of the 4x4 array for a $V_{bias,1}$ of 0 V and different $V_i(t)$ as shown in Fig. 7.5

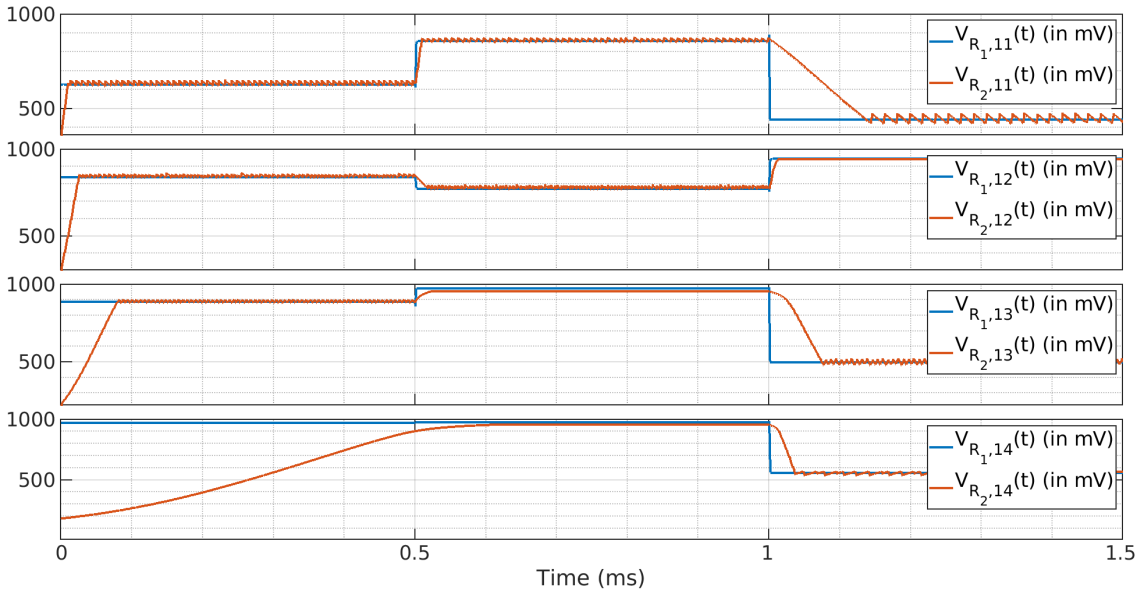


Figure 7.8 Transient input voltages, V_{R_1} and V_{R_2} (see Fig. 6.13), to the comparators of the memristive cells located in the first row of the 4x4 array for the same $V_1(t)$ and different values of $V_{bias,j}$ as shown in Fig. 7.5

For every $V_i(t)$, the values of $V_{R_1,ij}$ change, where V_{R_1} is shown in Fig. 6.13 and the indices i and j are cell's row and column, respectively. The changes in $V_{R_1,ij}$ are presented in Fig. 7.6. Since V_{R_2} follows V_{R_1} as explained in chapter 6, the changes in $V_{R_2,ij}$ are also recorded on the same figure. Similarly, the changes in $V_{C_{FGT},ij}$ are recorded in Fig. 7.7 although $V_{R_2,ij}$ is an indirect indicator for $V_{C_{FGT},ij}$. Figures 7.6 and 7.7 summarize the effect of changing $V_i(t)$ on the memristive cells. On the other hand, to study the effect of $V_{bias}(t)$ on the same cells, Fig. 7.8 shows the simulation results for $V_{R_1,ij}$ and $V_{R_2,ij}$ when $V_{bias}(t)$ changes.

A detailed analysis of Figs. 7.6-7.8 leads to the following remarks:

(i) The memristive cell is sensitive to the slightest change in its input voltages. For example, a 10 mV change on $V_2(t)$ at time $t = 50$ ms provoked changes by proportional amounts in the corresponding voltages: $V_{C_{FGT},21}$, $V_{R_1,21}$ and $V_{R_2,21}$.

(ii) The memristive-cell internal capacitor-voltage, $V_{C_{SX},ij}$ is cyclic. In other words, an increase or decrease in $V_i(t)$ does not necessarily lead to the same effect on $V_{C_{SX},ij}$. It rather depends on both: $V_{C_{SX},ij}$'s starting voltage before $V_i(t)$ changes and on the amount of that change.

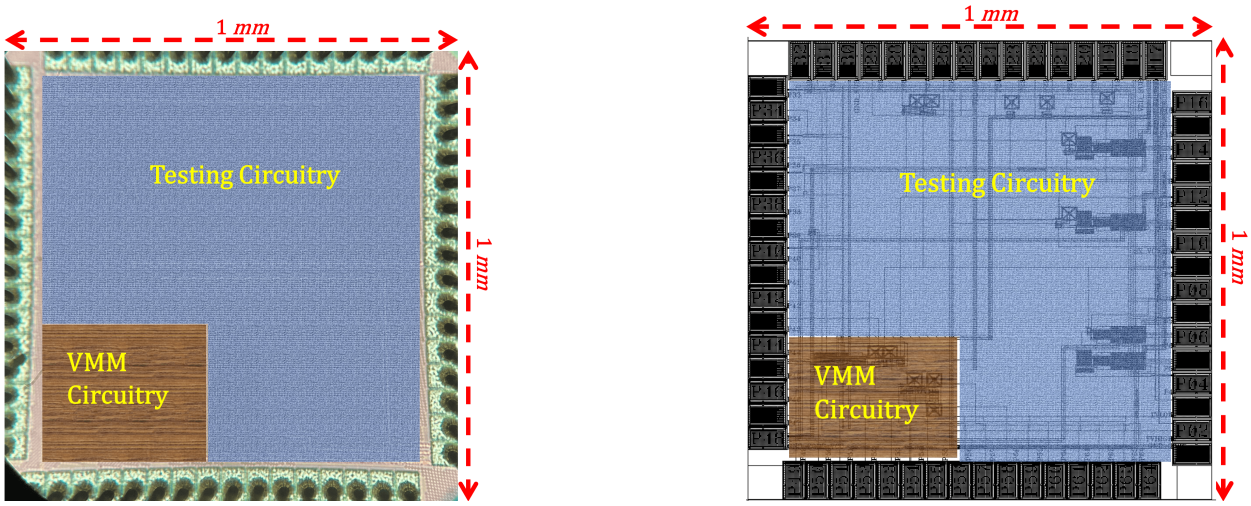
(iii) In the proposed VerilogAms model, charging $V_{C_{FGT},ij}$ is faster than discharging. For example, an 80 mV change in $V_{C_{FGT},ij}$ requires around 1 ms in charging time, and around 12 ms in discharging time.

(iv) There still exist some instability in $V_{C_{FGT},ij}$. This the cause of $V_{R_2,ij}$ turbulence around $V_{R_1,ij}$ in all cells.

(v) The values of $V_{C_{FGT},ij}$ in all cells are less that 550 mV, which is good for the integrity of FGTs gate-dielectric.

(vi) The changes in $V_{bias,j}$ affect $V_{C_{SX},ij}$ and, consequently, $V_{R_1,ij}$ which is driving the whole circuit. The change in $V_{bias,j}$ also affects the charging and discharging times of $V_{R_2,ij}$.

7.4 A Design Verification Chip



(a) (b)

Figure 7.9 An image of the second fabricated chip in 65 *nm*: (a) A microscopic photograph after fabrication (b) CAD layout

Figure 7.9 presents a chip fabricated based on lessons learned from experimenting on the chip presented in Fig. 5.6. We fabricated two 2×1 VMM crossbar rows in order to verify the simulation results previously mentioned in this current chapter. This chip was fabricated with the intention of verifying the unanticipated measurements captured from the previously fabricated chip. Hence, it was designed comprising of many testing circuits each corresponding to a sub-circuit in the crossbar. In short, we designed the chip of Fig. 7.9 to be able to characterize the behavior of the following components: (a) isolated individual transistors, (b) transistors matched based on the common centroid method, (c) the embedded voltage comparator, (d) the embedded voltage control oscillator, (e) the memristor emulator, (f) one memristive cell, (g) a 2×1 memristive crossbar, and (h) a 2×2 memristive crossbar.

Unfortunately, we encountered two major problems that prevented us from validating the new designed chip. Due to the COVID-19 pandemic, the ordered chips took a longer delivery time than usual, and most of the delivered dies are defective.

7.5 Summary

In this chapter we presented two different possible configurations of the memristive cell (proposed in chapter 6) for VMM, and we used one of these two configurations to simulate a 4×4 crossbar array. The simulated array proved that the potential difference across the terminals of each cell led to a different resistive state for every cell in the array while maintaining its memristor-like behavior. This was evident from the transient plots of cells within the same row or the same column, where each memristive cell showed a different resistive state. The obtained results were promising despite that neither the crossbar architecture nor the memristive cell were optimal. The results also proved that our memristive cell represents a possible RSD substitute that is fit for many recent RSD applications.

CHAPTER 8 GENERAL DISCUSSION

This dissertation contributes to improving the means of implementing AI algorithms using analog integrated circuits. We believed that FGTs are necessary for these circuits since they can provide long data retention, and we relied on existing literature in our preliminary circuit design. During our time waiting for the fabricated chips to arrive, we tried to minimize delay consequences on the project by ordering memristors from KNOWM and an FPAA from a well-known foundry. The delays we faced to get hold of some of these products (particularly the FPAA) were unexpected as all ordered devices arrived after our fabricated chips. We hoped to compare the performance of our Adaptive DPE system on the three devices: our fabricated memristive cells, KNOWM memristors, and the FPAA. However, due to the previously discussed reasons, we went through tedious and lengthy experiments on our fabricated chips for FGT characterization. Yet, our conducted experiments were fruitful and led to the discovery of new characteristics for charge tunneling in FGTs with thin-gate dielectric. Thus, the uncovering of FGT characteristics in a standard 65 nm CMOS process with 2 nm thin gate-dielectric is novel to the literature, and it presents our first contribution.

For our next contribution, we focused on modeling our newly discovered FGT characteristics in VerilogAMS and integrating the resulting model into one memristive cell with a CMOS memristor emulator. The resulting memristive cell represents a novelty for its characterizing properties. Firstly, it is a hardware-proved model based on extracted measurement results from our chips. Secondly, it is fabricated using a robust and reliable CMOS process. Finally, it possesses a robust memristive behavior with a long data retention time.

The final contribution of this dissertation is represented by integrating the obtained memristive circuit into a 4×4 crossbar array capable of implementing VMM. This is the first step towards implementing a fully integrated memristive inference engine which we believe it to be an interesting subject for another Ph.D research project. In the rest of this chapter, we will discuss some challenges we faced during this research.

A few words on our decision to adopt 65 nm CMOS process Our decision to characterize FGTs in a standard 65 nm CMOS process might be challenged since CMOS technologies have progressed a lot ever since 65 nm technology was launched. In sub 10 nm processes, for example, dielectric thicknesses are less than or equal to 1 nm, whereas the 65 nm process has a dielectric thickness of around 2 nm. When we decided on 65 nm as a target technology for our research, we tried to predict the behavior of its dielectric based on existing literature, and we were confident that this 65 nm; that is and will remain an industry workhorse for years to come, would suit our end-application.

Our initial prediction for the tunneling voltage thresholds, based on knowledge we had and on literature available at the time, was quite far from our laboratory measurements, despite our strong belief that gate tunneling could successfully be used with to 65 nm transistors. Our characterization efforts for 65 nm yielded critical observations that changed our understanding about how gate tunneling works in advanced CMOS technologies. Although we believe that our conclusions theoretically apply to all CMOS processes with thin gate dielectric, the voltage thresholds separating different behavioral regions of the tunneled current will vary based on dielectric thicknesses, materials, and nanostructure. For example, we expected that charge tunneling would start around 1.5 V, and that the dielectric breakdown voltage would occur around 5 V. However, the numbers, extracted from laboratory measurements, were 350 mV and 1.2 V for the tunneling voltage threshold and the breakdown voltage threshold, respectively. As these numbers relate to transistors rated at 1 V for long term operation, this was a quite unpleasant surprise to us. On the other hand, with our observations concerning 65 nm, we believe that after our work is published, the perception about gate tunneling and the feasibility of exploiting the floating gate structure will change. We believe that is very risky to speculate about the exact numbers for onset of tunneling and dielectric breakage with any more advanced technology without in-laboratory measurements, especially with the lack of reported (or CAD-tools embedded) experimentally validated models for floating gate devices.

In addition, it is important to not forget the challenges that academic researchers face to secure access to state-of-the art processes notably those offering finfets. At the time of

completing this thesis, there is no access to any technology below 12 nm for academia in our country, and for us, such access can only be obtained as part of industry university collaborations typically aimed at short term returns. Performing curiosity-driven academic research with sub 10 nm technology may become possible in 3 years for us at Polytechnique Montreal, but it was absolutely not possible when our technology selection was made for the research reported in this dissertation.

Finally, it is to be stressed that it is the researcher's duty to validate the use of a technology before proceeding to circuit design. The verification can be either through experimenting on fabricated chips or through foundry provided models built in CAD tools, if they exist.

A few words on the memristive cell The proposed memristive cell in chapter 6 is based on our characterization efforts for floating gate devices using a standard 65 nm process. Yet, in to order integrate the derived model into CAD tools without loss of generality, similar efforts are needed on FGTs with variable dielectric thicknesses and from different processes.

Moreover, during circuit simulations, we discovered some shortcomings in existing CAD models. This forced us to revisit those models and re-assess them. For example, it was difficult to simulate an FGT charge trap with the existence of redundant gate resistors (inside the model) which always drain circuit capacitors to 0 V. Due to these errors, we were forced into unexpected delays in simulation while investigating modeling errors. One of the measures we took was to dig into conventional VerilogAMS transistor models, and use them to replicate the I-V characteristic of MOSFETs from our fabricated chip. The leakage in the developed model was lower than that of the default internal model.

From another perspective, the proposed memristive cell is a proof of concept that a cell with such characteristics can be fabricated using a standard 65 nm process. For us, the main goal was to design, fabricate, model and simulate a memristor with a long retention time using a reliable CMOS process. Thus, we did not optimize the circuit for scalability, power consumption or precision.

Finally, a few words on vector matrix multiplication The 4x4 array, that was presented in chapter 8 as a proof of concept for VMM, is quite small for complex AI algorithms,

such as Back-Propagation [18]. The 4x4 array is sufficient to solve simple functions as the XOR function, but not to demonstrate a complex simulation such as the one reported in [116] using our proposed cell. For the 4x4 VMM array, a simple simulation task for a period of 1.5 ms required 5 hours on a computer with an $i7$ Intel processor. This is mainly caused by the sequential simulation of the proposed VMM array. It is thus necessary to gain access to CAD tools capable of deploying parallelism in their simulation, just similar to CAD tools available in industry.

The proposed memristive cell, if used in its current configuration, limits the crossbar array dimensions. As the number of rows depends on NN architecture, the number of columns is limited by column bias-voltages. The bias-voltage window is between 0 V and 120 mV. We recommend a 20 mV step in bias-voltage of neighboring columns. Thus, the maximum recommended number of columns with circuits that we proposed is 6. Somehow, this is quite smaller than our preliminary estimate for the crossbar array dimensions as a means to integrate a DPE comprising millions of memristive crosspoints in a practical device.

CHAPTER 9 CONCLUSIONS AND RECOMMENDATIONS

9.1 Summary of the Work Done

Many recent articles express concerns for transistor scalability and the end of Moore's law, since recent progresses of technology are demanding denser structures, especially in digital domains. By this dissertation, we contribute to exploring a means to switch computing systems from digital architectures to more analog and continuous ones, particularly for artificial intelligence and machine learning.

This led us to propose a memristive cell that gets over two main challenges facing resistive switching devices: data retention and I-V characteristic curve, while being fabricated using a robust, reliable and advanced CMOS process with thin gate-dielectrics. On the way towards solving these challenges, we dove deep into the technology to discover transistors characteristics, hoping to be able to use its MOSFETs as floating gate transistors and, later, use them in re-configurable charge traps where data retention time is 10 or more years. We therefore discovered an unreported behavior for charge tunneling in floating gate transistors in a 65 nm CMOS process with thin gate-dielectric. The discovered behavior can be described as delicate due to transistor fragility, sensitivity and vulnerability. In brief, charge tunneling kicked-in at around 350 mV and continues until reaching 1.2 V, defining four behavioral regions of operation for the gate-dielectric starting by a no-leakage region before 350 mV followed by an exponential leakage region while below 550 mV then a linear and irreversible resistive leakage region while below 1.2 V at which the device became prone to damage, and enters the damage region. On the other hand, to maintain the cyclical pinched I-V hysteresis for the cell, we leveraged a memristor emulator, adapted from literature, to match all operating conditions of the experimented floating gate transistors.

Afterwards, we brought together using control circuitry the memristor emulator and the floating gate transistor circuit. To simulate the resulting circuit in a well-known CAD software, we modeled the floating gate charge trap in VerilogAMS. The resulting architecture was our memristive cell that was proved to have a pinched I-V hysteresis curve and long data retention.

Finally, we placed the resulting memristive cell in a 4×4 crossbar structure suitable to implement vector-matrix multiplication, an indispensable process for analog computing. All memristive cells in the 4×4 array showed autonomy as well as variability in cell-conductance programming.

9.2 Summary of Contributions

The main contributions of this dissertation are:

(i) Characterization of floating gate transistors in a standard 65 nm CMOS process with thin gate-dielectric.

(ii) Implementing a hardware-proved memristive cell that possesses the characteristics of conventional memristors, and that has long data retention time.

(iii) Integrating the designed memristive cell into a crossbar array as an initial step towards implementing a complete memristive inference engine.

9.2.1 Publications

In the process of completing this dissertation, two articles have been published, one article has been submitted, and another is still under development. These publications are presented below:

(i) Assaf, H., Savaria, Y., & Sawan, M. (2018, December). Vector Matrix Multiplication Using Crossbar Arrays: A Comparative Analysis. In 2018 25th IEEE International Conference on Electronics, Circuits and Systems (ICECS) (pp. 609-612). IEEE. (See chapter 3).

(ii) Assaf, H., Savaria, Y., & Sawan, M. (2019, March). Memristor emulators for an adaptive dpe algorithm: Comparative study. In 2019 IEEE International Conference on Artificial Intelligence Circuits and Systems (AICAS) (pp. 13-17). IEEE. (See chapter 4).

(iii) (Under Revision) Assaf, H., Savaria, Y., Ali, M., Nabavi, M., & Sawan, M. (2022, May). Implementing floating gate transistors with a thin gate dielectric 65 nm CMOS technology. Submitted to IEEE Transactions on Electron Devices. (See chapter 5).

(iv) (Under Development) A memristive cell with high endurance and long retention time for vector matrix multiplication: From hardware to simulation.

9.3 Limitations

Existing CAD tools were the main limitation in this project, and will continue to be for this type of work in the near future. The lack a proved and validated simulation model for floating gate transistors delayed us a significant amount of time (around two years). We were unable to predict the behavior of our circuits prior to fabrication, and the measurement result were an unpleasant surprise to us.

Another limitation in this project is the size of the crossbar array. Despite that the number of rows in a crossbar array deploying our memristive cell is only bounded by the neural network architecture, the number of columns is limited by the voltage range for column bias-voltage which, in turn, is limited by the studied floating gate transistor sensitivity. In other words. it is limited by the upper margin of the exponential leakage region in thin gate-dielectric floating gate devices, which is 650 mV.

9.4 Future Research

On the short term, we plan to develop an error model for the complete crossbar array starting by the memristive cell. This model ought to account for errors from all sources such as: model approximation, measurement errors, circuit sensitivity and, above all, sneak path currents. Moreover, we plan to optimize the memristive cell in terms of size, accuracy, response time and power consumption.

As for the long term, we plan to design a feedback path from output currents to input voltages. This step is missing to close the loop for a simple training task, such as the XOR function. By completing this step, the system will be ready for a large-scale memristive inference engine that can be fabricated on chip.

REFERENCES

[1] Yibo Li, Zhongrui Wang, Rivu Midya, Qiangfei Xia, and J Joshua Yang. Review of memristor devices in neuromorphic computing: materials sciences and device challenges. *Journal of Physics D: Applied Physics*, 51(50):503002, 2018.

[2] Emanuel Carlos, Rita Branquinho, Rodrigo Martins, Asal Kiazadeh, and Elvira Fortunato. Recent progress in solution-based metal oxide resistive switching devices. *Advanced Materials*, 33(7):2004328, 2021.

[3] Tayfun Gokmen and Yurii A Vlasov. Accelerated neural network training using a pipelined resistive processing unit architecture, January 4 2018. US Patent App. 15/196,350.

[4] Miao Hu, Catherine E Graves, Can Li, Yunning Li, Ning Ge, Eric Montgomery, Noraica Davila, Hao Jiang, R Stanley Williams, J Joshua Yang, et al. Memristor-based analog computation and neural network classification with a dot product engine. *Advanced Materials*, 2018.

[5] Tayfun Gokmen, Murat Onen, and Wilfried Haensch. Training deep convolutional neural networks with resistive cross-point devices. *Frontiers in neuroscience*, 11:538, 2017.

[6] Seyoung Kim, Tayfun Gokmen, Hyung-Min Lee, and Wilfried E Haensch. Analog cmos-based resistive processing unit for deep neural network training. *arXiv preprint arXiv:1706.06620*, 2017.

[7] Vahid Keshmiri. A study of the memristor models and applications, 2014.

[8] Ahmed G Radwan, M Affan Zidan, and KN Salama. On the mathematical modeling of memristors. In *Microelectronics (ICM), 2010 International Conference on*, pages 284–287. IEEE, 2010.

[9] Dmitri B Strukov, Gregory S Snider, Duncan R Stewart, and R Stanley Williams. The missing memristor found. *nature*, 453(7191):80, 2008.

[10] Mohamed E Fouda and Ahmed G Radwan. Fractional-order memristor response under dc and periodic signals. *Circuits, Systems, and Signal Processing*, 34(3):961–970, 2015.

[11] Charan Thondapu, Ramalingarn Sridhar, et al. A gate leakage reduction strategy for sub-70 nm memory circuits. In *Proceedings of the 2004 IEEE Dallas/CAS Workshop Implementation of High Performance Circuits*, pages 145–148. IEEE, 2004.

[12] Vishal Saxena. A compact cmos memristor emulator circuit and its applications. In *2018 IEEE 61st International Midwest Symposium on Circuits and Systems (MWSCAS)*, pages 190–193. IEEE, 2018.

[13] James Moor. *The Turing test: the elusive standard of artificial intelligence*, volume 30. Springer Science & Business Media, 2003.

[14] Alan M Turing. Computing machinery and intelligence. In *Parsing the turing test*, pages 23–65. Springer, 2009.

[15] Seung Hwan Lee, Xiaojian Zhu, and Wei D Lu. Nanoscale resistive switching devices for memory and computing applications. *Nano Research*, 13(5):1228–1243, 2020.

[16] Robert R Schaller. Moore's law: past, present and future. *IEEE spectrum*, 34(6):52–59, 1997.

[17] Chris A Mack. Fifty years of moore's law. *IEEE Transactions on semiconductor manufacturing*, 24(2):202–207, 2011.

[18] Benjamin Scellier and Yoshua Bengio. Equilibrium propagation: Bridging the gap between energy-based models and backpropagation. *Frontiers in computational neuroscience*, 11:24, 2017.

[19] Fei-Yue Wang, Jun Jason Zhang, Xinhu Zheng, Xiao Wang, Yong Yuan, Xiaoxiao Dai, Jie Zhang, and Liuqing Yang. Where does alphago go: From church-turing thesis to

alphago thesis and beyond. *IEEE/CAA Journal of Automatica Sinica*, 3(2):113–120, 2016.

[20] Tom De Smedt and Walter Daelemans. Pattern for python. *The Journal of Machine Learning Research*, 13(1):2063–2067, 2012.

[21] Michael J Crawley. *The R book*. John Wiley & Sons, 2012.

[22] Giuseppe Ciaburro. *MATLAB for machine learning*. Packt Publishing Ltd, 2017.

[23] Andrew Ling and Jason Anderson. The role of fpgas in deep learning. In *Proceedings of the 2017 ACM/SIGDA international symposium on field-programmable gate arrays*, pages 3–3, 2017.

[24] T Hirtzlin, Marc Bocquet, M Ernoult, J-O Klein, E Nowak, E Vianello, J-M Portal, and D Querlioz. Hybrid analog-digital learning with differential rram synapses. In *2019 IEEE International Electron Devices Meeting (IEDM)*, pages 22–6. IEEE, 2019.

[25] Zongjie Shen, Chun Zhao, Yanfei Qi, Wangying Xu, Yina Liu, Ivona Z Mitrovic, Li Yang, and Cezhou Zhao. Advances of rram devices: Resistive switching mechanisms, materials and bionic synaptic application. *Nanomaterials*, 10(8):1437, 2020.

[26] Tayfun Gokmen and Wilfried Haensch. Algorithm for training neural networks on resistive device arrays. *Frontiers in Neuroscience*, 14:103, 2020.

[27] Yuetong Fang, Shuaibo Zhai, Liang Chu, and Jiasong Zhong. Advances in halide perovskite memristor from lead-based to lead-free materials. *ACS Applied Materials & Interfaces*, 13(15):17141–17157, 2021.

[28] Bonan Yan, Yiran Chen, and Hai Li. Challenges of memristor based neuromorphic computing system. *Science China Information Sciences*, 61(6):1–3, 2018.

[29] Amirali Amirsoleimani, Fabien Alibart, Victor Yon, Jianxiong Xu, M Reza Pazhouhandeh, Serge Ecoffey, Yann Beilliard, Roman Genov, and Dominique Drouin. In-memory

vector-matrix multiplication in monolithic complementary metal–oxide–semiconductor-memristor integrated circuits: Design choices, challenges, and perspectives. *Advanced Intelligent Systems*, 2(11):2000115, 2020.

[30] Said Hamdioui, Shahar Kvatinsky, Gert Cauwenberghs, Lei Xie, Nimrod Wald, Siddharth Joshi, Hesham Mostafa Elsayed, Henk Corporaal, and Koen Bertels. Memristor for computing: Myth or reality? In *Design, Automation & Test in Europe Conference & Exhibition (DATE), 2017*, pages 722–731. IEEE, 2017.

[31] Stefan Lai. Non-volatile memory technologies: The quest for ever lower cost. In *2008 IEEE International Electron Devices Meeting*, pages 1–6. IEEE, 2008.

[32] Vivienne Sze, Yu-Hsin Chen, Tien-Ju Yang, and Joel S Emer. Efficient processing of deep neural networks. *Synthesis Lectures on Computer Architecture*, 15(2):1–341, 2020.

[33] Tuo Shi, Rui Wang, Zuheng Wu, Yize Sun, Junjie An, and Qi Liu. A review of resistive switching devices: performance improvement, characterization, and applications. *Small Structures*, 2(4):2000109, 2021.

[34] Mario Lanza, H-S Philip Wong, Eric Pop, Daniele Ielmini, Dimitri Strukov, Brian C Regan, Luca Larcher, Marco A Villena, J Joshua Yang, Ludovic Goux, et al. Recommended methods to study resistive switching devices. *Advanced Electronic Materials*, 5(1):1800143, 2019.

[35] R Stanley Williams. Aftermath of finding the memristor. In *Chaos, CNN, Memristors and Beyond: A Festschrift for Leon Chua With DVD-ROM, composed by Eleonora Bilotta*, pages 490–493. World Scientific, 2013.

[36] Leon Chua. Memristor-the missing circuit element. *IEEE Transactions on circuit theory*, 18(5):507–519, 1971.

[37] Leon O Chua and Sung Mo Kang. Memristive devices and systems. *Proceedings of the IEEE*, 64(2):209–223, 1976.

[38] Daniele Ielmini and H-S Philip Wong. In-memory computing with resistive switching devices. *Nature electronics*, 1(6):333–343, 2018.

[39] Meiran Zhao, Bin Gao, Jianshi Tang, He Qian, and Huaqiang Wu. Reliability of analog resistive switching memory for neuromorphic computing. *Applied Physics Reviews*, 7 (1):011301, 2020.

[40] Sung Hyun Jo, Kuk-Hwan Kim, and Wei Lu. High-density crossbar arrays based on a si memristive system. *Nano letters*, 9(2):870–874, 2009.

[41] Cong Wang, Shi-Jun Liang, Chen-Yu Wang, Zai-Zheng Yang, Yingmeng Ge, Chen Pan, Xi Shen, Wei Wei, Yichen Zhao, Zaichen Zhang, et al. Scalable massively parallel computing using continuous-time data representation in nanoscale crossbar array. *Nature nanotechnology*, 16(10):1079–1085, 2021.

[42] Qiangfei Xia and J Joshua Yang. Memristive crossbar arrays for brain-inspired computing. *Nature materials*, 18(4):309–323, 2019.

[43] Javier Del Valle, Juan Gabriel Ramírez, Marcelo J Rozenberg, and Ivan K Schuller. Challenges in materials and devices for resistive-switching-based neuromorphic computing. *Journal of Applied Physics*, 124(21):211101, 2018.

[44] Ilia Valov. Redox-based resistive switching memories (rerams): Electrochemical systems at the atomic scale. *ChemElectroChem*, 1(1):26–36, 2014.

[45] C Sun, SM Lu, F Jin, WQ Mo, JL Song, and KF Dong. The resistive switching characteristics of tin/hfo2/ag rram devices with bidirectional current compliance. *Journal of Electronic Materials*, 48(5):2992–2999, 2019.

[46] H-S Philip Wong, Simone Raoux, SangBum Kim, Jiale Liang, John P Reifenberg, Bipin Rajendran, Mehdi Asheghi, and Kenneth E Goodson. Phase change memory. *Proceedings of the IEEE*, 98(12):2201–2227, 2010.

[47] J Joshua Yang, Dmitri B Strukov, and Duncan R Stewart. Memristive devices for computing. *Nature nanotechnology*, 8(1):13–24, 2013.

[48] Fabien Alibart, Elham Zamanidoost, and Dmitri B Strukov. Pattern classification by memristive crossbar circuits using ex situ and in situ training. *Nature communications*, 4(1):1–7, 2013.

[49] Can Li, Miao Hu, Yunning Li, Hao Jiang, Ning Ge, Eric Montgomery, Jiaming Zhang, Wenhao Song, Noraica Dávila, Catherine E Graves, et al. Analogue signal and image processing with large memristor crossbars. *Nature electronics*, 1(1):52–59, 2018.

[50] Mirko Prezioso, Farnood Merrikh-Bayat, BD Hoskins, Gina C Adam, Konstantin K Likharev, and Dmitri B Strukov. Training and operation of an integrated neuromorphic network based on metal-oxide memristors. *Nature*, 521(7550):61–64, 2015.

[51] Lei Deng, Guoqi Li, Ning Deng, Dong Wang, Ziyang Zhang, Wei He, Huanglong Li, Jing Pei, and Luping Shi. Complex learning in bio-plausible memristive networks. *Scientific reports*, 5(1):1–10, 2015.

[52] Lei Deng, Dong Wang, Ziyang Zhang, Pei Tang, Guoqi Li, and Jing Pei. Energy consumption analysis for various memristive networks under different learning strategies. *Physics Letters A*, 380(7-8):903–909, 2016.

[53] Geoffrey W Burr, Robert M Shelby, Severin Sidler, Carmelo Di Nolfo, Junwoo Jang, Irem Boybat, Rohit S Shenoy, Pritish Narayanan, Kumar Virwani, Emanuele U Giacometti, et al. Experimental demonstration and tolerancing of a large-scale neural network (165 000 synapses) using phase-change memory as the synaptic weight element. *IEEE Transactions on Electron Devices*, 62(11):3498–3507, 2015.

[54] Zhongrui Wang, Saumil Joshi, Sergey E Savel'ev, Hao Jiang, Rivu Midya, Peng Lin, Miao Hu, Ning Ge, John Paul Strachan, Zhiyong Li, et al. Memristors with diffusive dynamics as synaptic emulators for neuromorphic computing. *Nature materials*, 16(1): 101–108, 2017.

[55] Thomas Breuer, Lutz Nielen, Bernd Roesgen, Rainer Waser, Vikas Rana, and Eike Linn. Realization of minimum and maximum gate function in ta2o5-based memristive devices. *Scientific reports*, 6(1):1–9, 2016.

[56] Selina La Barbera, Adrien F Vincent, Dominique Vuillaume, Damien Querlioz, and Fabien Alibart. Interplay of multiple synaptic plasticity features in filamentary memristive devices for neuromorphic computing. *Scientific reports*, 6(1):1–11, 2016.

[57] Alexander Serb, Johannes Bill, Ali Khiat, Radu Berdan, Robert Legenstein, and Themis Prodromakis. Unsupervised learning in probabilistic neural networks with multi-state metal-oxide memristive synapses. *Nature communications*, 7(1):1–9, 2016.

[58] Isha Gupta, Alexantrou Serb, Ali Khiat, Ralf Zeitler, Stefano Vassanelli, and Themistoklis Prodromakis. Real-time encoding and compression of neuronal spikes by metal-oxide memristors. *Nature communications*, 7(1):1–9, 2016.

[59] Patrick M Sheridan, Fuxi Cai, Chao Du, Wen Ma, Zhengya Zhang, and Wei D Lu. Sparse coding with memristor networks. *Nature nanotechnology*, 12(8):784, 2017.

[60] Peng Yao, Huaqiang Wu, Bin Gao, Sukru Burc Eryilmaz, Xueyao Huang, Wenqiang Zhang, Qingtian Zhang, Ning Deng, Luping Shi, H-S Philip Wong, et al. Face classification using electronic synapses. *Nature communications*, 8(1):1–8, 2017.

[61] Max M Shulaker, Gage Hills, Rebecca S Park, Roger T Howe, Krishna Saraswat, H-S Philip Wong, and Subhasish Mitra. Three-dimensional integration of nanotechnologies for computing and data storage on a single chip. *Nature*, 547(7661):74–78, 2017.

[62] Bhaswar Chakrabarti, Miguel Angel Lastras-Montaño, Gina Adam, Mirko Prezioso, Brian Hoskins, M Payvand, A Madhavan, A Ghofrani, L Theogarajan, K-T Cheng, et al. A multiply-add engine with monolithically integrated 3d memristor crossbar/cmos hybrid circuit. *Scientific reports*, 7(1):1–10, 2017.

[63] Xinman Chen, Guangheng Wu, and Dinghua Bao. Resistive switching behavior of pt/mg 0.2 zn 0.8 o/pt devices for nonvolatile memory applications. *Applied Physics Letters*, 93(9):093501, 2008.

[64] Nadine Gergel-Hackett, Behrang Hamadani, Barbara Dunlap, John Suehle, Curt Richter, Christina Hacker, and David Gundlach. A flexible solution-processed memristor. *IEEE Electron Device Letters*, 30(7):706–708, 2009.

[65] Sungho Kim, Hanul Moon, Dipti Gupta, Seunghyup Yoo, and Yang-Kyu Choi. Resistive switching characteristics of sol–gel zinc oxide films for flexible memory applications. *IEEE Transactions on Electron Devices*, 56(4):696–699, 2009.

[66] Chiyui Ahn, Zizhen Jiang, Chi-Shuen Lee, Hong-Yu Chen, Jiale Liang, Luckshitha S Liyanage, and H-S Philip Wong. 1d selection device using carbon nanotube fets for high-density cross-point memory arrays. *IEEE Transactions on Electron Devices*, 62 (7):2197–2204, 2015.

[67] HY Lee, YS Chen, PS Chen, PY Gu, YY Hsu, SM Wang, WH Liu, CH Tsai, SS Sheu, PC Chiang, et al. Evidence and solution of over-reset problem for hfo x based resistive memory with sub-ns switching speed and high endurance. In *2010 International Electron Devices Meeting*, pages 19–7. IEEE, 2010.

[68] Paolo Pavan, Luca Larcher, and Andrea Marmiroli. *Floating gate devices: operation and compact modeling*. Springer Science & Business Media, 2007.

[69] Souvik Mahapatra, S Shukuri, and Jeff Bude. Chisel flash eeprom. i. performance and scaling. *IEEE Transactions on Electron Devices*, 49(7):1296–1301, 2002.

[70] Cory K Perkins, Melanie A Jenkins, Tsung-Han Chiang, Ryan H Mansergh, Vasily Gouliouk, Nizan Kenane, John F Wager, John F Conley Jr, and Douglas A Keszler. Demonstration of fowler–nordheim tunneling in simple solution-processed thin films. *ACS applied materials & interfaces*, 10(42):36082–36087, 2018.

[71] Paolo Pavan, Roberto Bez, Piero Olivo, and Enrico Zanoni. Flash memory cells-an overview. *Proceedings of the IEEE*, 85(8):1248–1271, 1997.

[72] Roberto Selow, Heitor S Lopes, and Carlos R Erig Lima. A comparison of fpga and fpaa technologies for a signal processing application. In *2009 International Conference on Field Programmable Logic and Applications*, pages 230–235. IEEE, 2009.

[73] Suma George, Sihwan Kim, Sahil Shah, Jennifer Hasler, Michelle Collins, Farhan Adil, Richard Wunderlich, Stephen Nease, and Shubha Ramakrishnan. A programmable and

configurable mixed-mode fpaa soc. *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, 24(6):2253–2261, 2016.

[74] Loai Danial, Evgeny Pikhay, Eric Herbelin, Nicolas Wainstein, Vasu Gupta, Nimrod Wald, Yakov Roizin, Ramez Daniel, and Shahar Kvatinsky. Two-terminal floating-gate transistors with a low-power memristive operation mode for analogue neuromorphic computing. *Nature Electronics*, 2(12):596–605, 2019.

[75] Wei Lu and Mohammed A Zidan. Field-programmable crossbar array for reconfigurable computing, April 5 2018. US Patent App. 15/723,668.

[76] Mohammed Zidan, YeonJoo Jeong, Jong Hong Shin, Chao Du, Zhengya Zhang, and Wei Lu. Field-programmable crossbar array (fpca) for reconfigurable computing. *IEEE Transactions on Multi-Scale Computing Systems*, 2017.

[77] Varsha Venkatesh and R Swarnalatha. Review and simulation of memristors using matlab and pspice. In *Electrical, Computer and Communication Technologies (ICECCT), 2017 Second International Conference on*, pages 1–6. IEEE, 2017.

[78] Shahar Kvatinsky, Eby G Friedman, Avinoam Kolodny, and Uri C Weiser. Team: Threshold adaptive memristor model. *IEEE Transactions on Circuits and Systems I: Regular Papers*, 60(1):211–221, 2013.

[79] Chun Jason Xue, Guangyu Sun, Youtao Zhang, J Joshua Yang, Yiran Chen, and Hai Li. Emerging non-volatile memories: opportunities and challenges. In *Hardware/Software Codesign and System Synthesis (CODES+ ISSS), 2011 Proceedings of the 9th International Conference on*, pages 325–334. IEEE, 2011.

[80] Yogesh N Joglekar and Stephen J Wolf. The elusive memristor: properties of basic electrical circuits. *European Journal of Physics*, 30(4):661, 2009.

[81] Zdeněk Biolek, Dalibor Biolek, and Viera Biolkova. Spice model of memristor with nonlinear dopant drift. *Radioengineering*, 18(2), 2009.

[82] Themistoklis Prodromakis, Boon Pin Peh, Christos Papavassiliou, and Christofer Toumazou. A versatile memristor model with nonlinear dopant kinetics. *IEEE transactions on electron devices*, 58(9):3099–3105, 2011.

[83] TA Wey and S Benderli. Amplitude modulator circuit featuring tio 2 memristor with linear dopant drift. *Electronics letters*, 45(22):1103–1104, 2009.

[84] Shahar Kvatinsky, Keren Talisveyberg, Dmitry Fliter, Eby G Friedman, Avinoam Kolodny, and Uri C Weiser. Verilog-a for memristor models. *CCIT Technical Report*, 801, 2011.

[85] Omid Kavehei, Azhar Iqbal, YS Kim, Kamran Eshraghian, SF Al-Sarawi, and Derek Abbott. The fourth element: characteristics, modelling and electromagnetic theory of the memristor. In *Proceedings of the Royal Society of London A: Mathematical, Physical and Engineering Sciences*, page rspa20090553. The Royal Society, 2010.

[86] Eero Lehtonen and Mika Laiho. Cnn using memristors for neighborhood connections. In *Cellular Nanoscale Networks and Their Applications (CNNA), 2010 12th International Workshop on*, pages 1–4. IEEE, 2010.

[87] J Joshua Yang, Matthew D Pickett, Xuema Li, Douglas AA Ohlberg, Duncan R Stewart, and R Stanley Williams. Memristive switching mechanism for metal/oxide/metal nanodevices. *Nature nanotechnology*, 3(7):429, 2008.

[88] John G Simmons. Generalized formula for the electric tunnel effect between similar electrodes separated by a thin insulating film. *Journal of applied physics*, 34(6):1793–1803, 1963.

[89] Faisal Budiman, Detiza Goldianto Octensi Hernowo, Reetu Raj Pandey, Hirofumi Tanaka, et al. Recent progress on fabrication of memristor and transistor-based neuromorphic devices for high signal processing speed with low power consumption. *Japanese Journal of Applied Physics*, 57(3S2):03EA06, 2018.

[90] Abdullah Yeşil, Yunus Babacan, and Fırat Kaçar. A new ddcc based memristor emulator circuit and its applications. *Microelectronics Journal*, 45(3):282–287, 2014.

[91] C Sánchez-López, MA Carrasco-Aguilar, and C Muñiz-Montero. A 16 hz–160 khz memristor emulator circuit. *AEU-International Journal of Electronics and Communications*, 69(9):1208–1219, 2015.

[92] Hasan Sözen and Uğur Çam. New memristor emulator circuit using otas and cciis. In *Electrical and Electronics Engineering (ELECO), 2015 9th International Conference on*, pages 10–14. IEEE, 2015.

[93] Muhammad Taher Abuelma'atti and Zainulabideen Jamal Khalifa. A continuous-level memristor emulator and its application in a multivibrator circuit. *AEU-International Journal of Electronics and Communications*, 69(4):771–775, 2015.

[94] Vishal Saxena. A compact cmos memristor emulator circuit and its applications. *arXiv preprint arXiv:1711.06819*, 2017.

[95] Yunus Babacan and Fırat Kaçar. Floating memristor emulator with subthreshold region. *Analog Integrated Circuits and Signal Processing*, 90(2):471–475, 2017.

[96] Tor Sverre Lande. Floating-gate circuits and systems. In *A versatile memristor model with nonlinear dopant kinetics*, pages 115–137. Springer Science & Business Media, 2004.

[97] Fujio Masuoka, Masamichi Asano, Hiroshi Iwahashi, T Komuro, and S Tanaka. A 256k flash eeprom using triple polysilicon technology. In *1985 IEEE International Solid-State Circuits Conference. Digest of Technical Papers*, volume 28, pages 168–169. IEEE, 1985.

[98] Alejandro Silva-Juárez, Esteban Tlelo-Cuautle, Luis Gerardo de la Fraga, and Rui Li. Fpaa-based implementation of fractional-order chaotic oscillators using first-order active filter blocks. *Journal of advanced research*, 2020.

[99] Samuel Tuan Wang. On the iv characteristics of floating-gate mos transistors. *IEEE Transactions on Electron Devices*, 26(9):1292–1294, 1979.

[100] ML Ostraat, JW De Blauwe, ML Green, LD Bell, ML Brongersma, J Casperson, RC Flagan, and HA Atwater. Synthesis and characterization of aerosol silicon nanocrystal nonvolatile floating-gate memory devices. *Applied Physics Letters*, 79(3):433–435, 2001.

[101] M Ziegler, M Oberländer, D Schroeder, WH Krautschneider, and H Kohlstedt. Memristive operation mode of floating gate transistors: A two-terminal memflash-cell. *Applied Physics Letters*, 101(26):263504, 2012.

[102] S Danzeca, J Cesari, M Brugger, L Dusseau, A Masi, A Pineda, and G Spiezia. Characterization and modeling of a floating gate dosimeter with gamma and protons at various energies. *IEEE Transactions on Nuclear Science*, 61(6):3451–3457, 2014.

[103] P Maier, F Hartmann, T Mauder, M Emmerling, C Schneider, M Kamp, S Höfling, and L Worschech. Memristive operation mode of a site-controlled quantum dot floating gate transistor. *Applied Physics Letters*, 106(20):203501, 2015.

[104] Takaho Yokoyama, Naoyuki Hirata, Hironori Tsunoyama, Yuichi Negishi, and Atsushi Nakajima. Characterization of floating-gate memory device with thiolate-protected gold and gold-palladium nanoclusters. *AIP Advances*, 8(6):065002, 2018.

[105] Chenling Huang, Nizar Lajnef, and Shantanu Chakrabartty. Calibration and characterization of self-powered floating-gate usage monitor with single electron per second operational limit. *IEEE Transactions on Circuits and Systems I: Regular Papers*, 57 (3):556–567, 2009.

[106] Maite Álvarez, Carlos Hernando, Joan Cesari, Alvaro Pineda, and Eugeni Garcia-Moreno. Total ionizing dose characterization of a prototype floating gate mosfet dosimeter for space applications. *IEEE Transactions on Nuclear Science*, 60(6):4281–4288, 2013.

[107] C Riggert, M Ziegler, D Schroeder, WH Krautschneider, and H Kohlstedt. Memflash device: floating gate transistors as memristive devices for neuromorphic computing. *Semiconductor Science and Technology*, 29(10):104011, 2014.

[108] Yongbeom Cho, Jae Yoon Lee, Eunseon Yu, Jae-Hee Han, Myung-Hyun Baek, Seongjae Cho, and Byung-Gook Park. Design and characterization of semi-floating-gate synaptic transistor. *Micromachines*, 10(1):32, 2019.

[109] Loai Danial, Nicolás Wainstein, Shraga Kraus, and Shahar Kvatinsky. Breaking through the speed-power-accuracy tradeoff in adcs using a memristive neuromorphic architecture. *IEEE Transactions on Emerging Topics in Computational Intelligence*, 2(5): 396–409, 2018.

[110] Yao Yao, Lei Liu, Zhi-Yuan Ye, Min-Zhi Lin, Zheng-Yuan Su, Jun Wu, and Peng-Fei Wang. Investigation of device physics and modeling of semi-floating gate image sensor cell. *Microelectronic Engineering*, 217:111111, 2019.

[111] Ruishan Xin, Mao Ye, Jia Wang, Kai Hu, and Yiqiang Zhao. Data deletion method for security improvement of flash memories. *IEICE Electronics Express*, pages 15–20180152, 2018.

[112] Di Long, Xianlong Hong, and Sheqin Dong. Optimal two-dimension common centroid layout generation for mos transistors unit-circuit. In *2005 IEEE International Symposium on Circuits and Systems*, pages 2999–3002. IEEE, 2005.

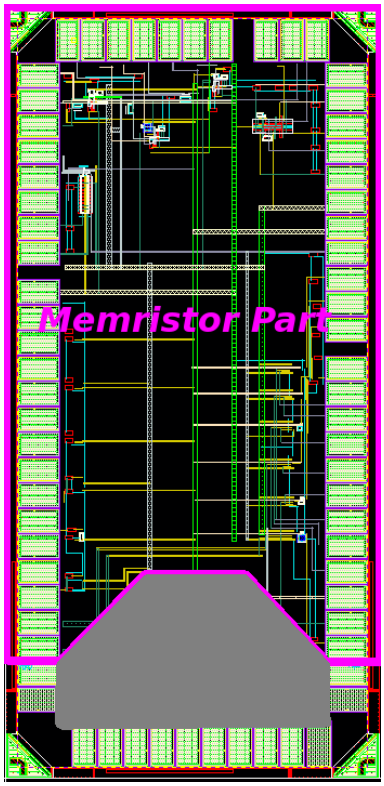
[113] Gianluca Traversi, L Gaioni, L Ratti, M Manghisoni, and V Re. Perspectives of 65 nm cmos technologies for high performance front-end electronics. In *21st International Workshop on Vertex Detectors*, volume 26, 2012.

[114] Morteza Nabavi. *Designing faster CMOS subthreshold circuits using transistor sizing and paralled transistor stacks*. PhD thesis, Carleton University, 2012.

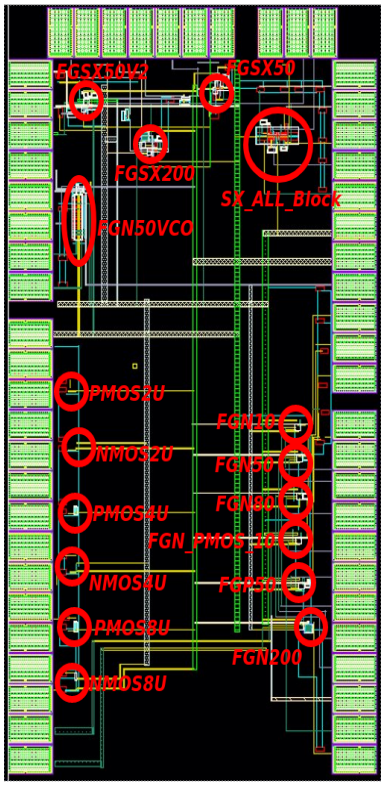
[115] Phillip E Allen and Douglas R Holberg. *CMOS analog circuit design*, page 480. Elsevier, 2011.

[116] Olga Krestinskaya, Khaled Nabil Salama, and Alex Pappachen James. Learning in memristive neural network architectures using analog backpropagation circuits. *IEEE Transactions on Circuits and Systems I: Regular Papers*, 66(2):719–732, 2018.

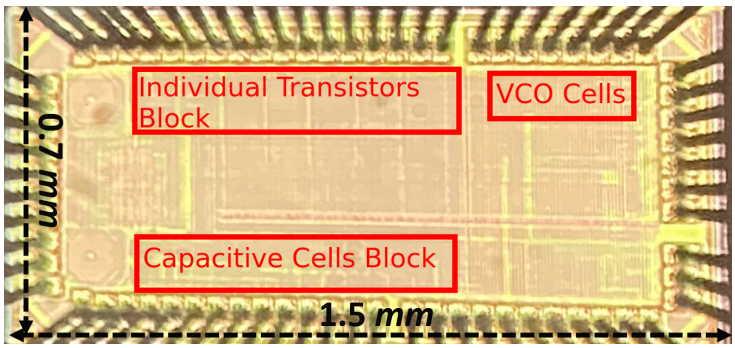
APPENDIX A CHIP 1 DESCRIPTION: SUPPORT MATERIAL FOR CHAPTER 5



(a) Complete Chip Layout



(b) Core Design Layout



(c) Fabricated Chip

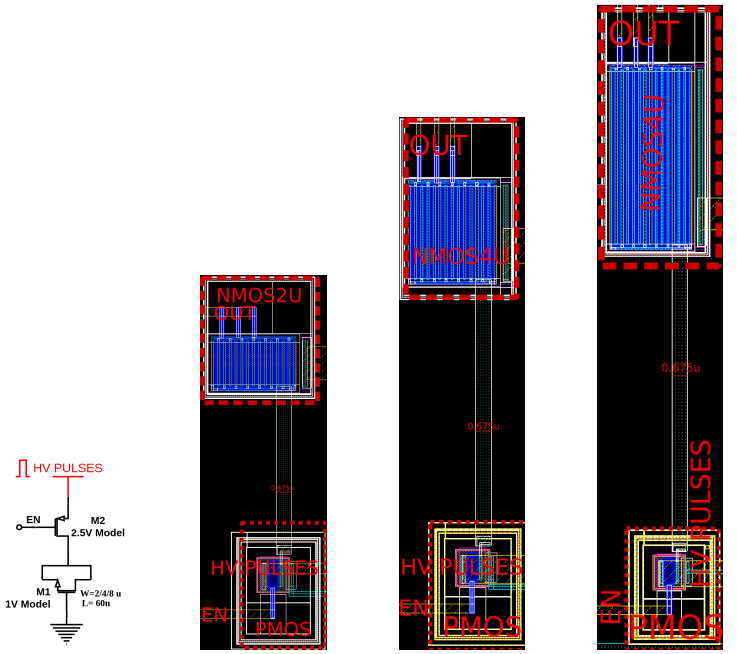
Figure A.1 Chip Overview

The main purpose of this chip is to characterize its different components as a preliminary step to design another chip for analog computing based on memristors. The desired memristor will contain a floating gate cell, a comparator and a memristor emulator.

Chip Summary

The full chip layout is shown in FigureA.1(a). It is composed of two parts with 68 PADs. Figure A.1(b) shows the memristor part of the chip, specifying the name and position of every cell.

Single FG transistors



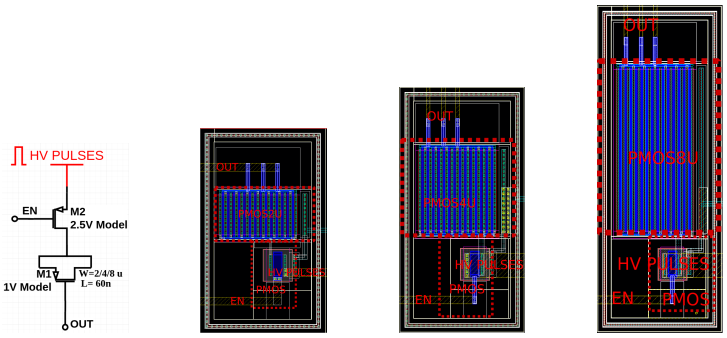
(a) Schematic

(b) 2um NMOS FG

(c) 4um NMOS FG

(d) 8um NMOS FG

Figure A.2 Single NMOS transistors

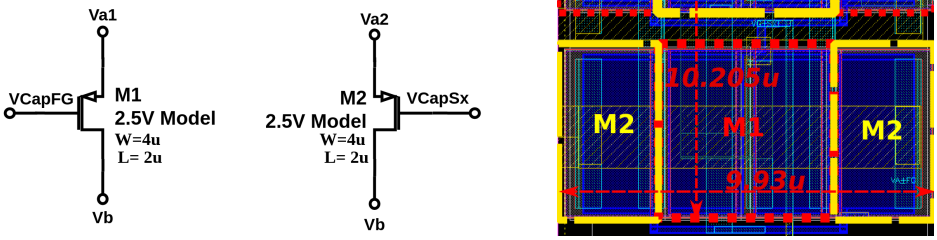


(a) Schematic (b) 2um PMOS FG (c) 4um PMOS FG (d) 8um PMOS FG

Figure A.3 Single PMOS transistors

Matched Transistors

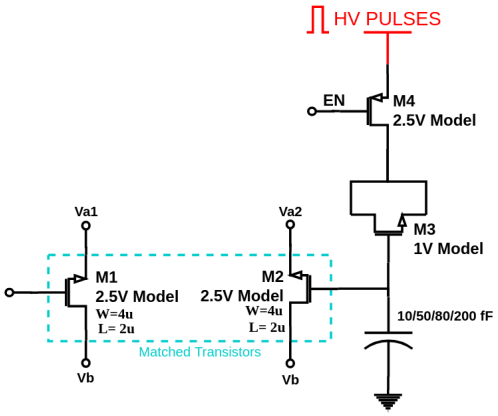
In this chip, we matched some transistors in layout design to make sure that the current traversing though the two transistors are identical



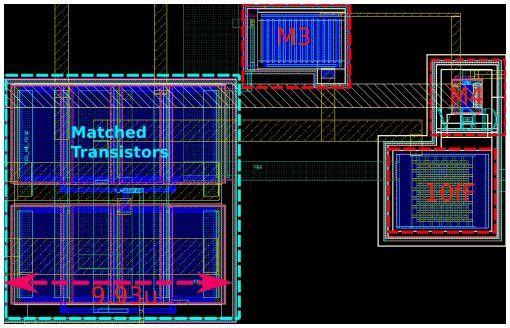
(a) Schematic (b) Layout

Figure A.4 Matched transistors using common centroid method

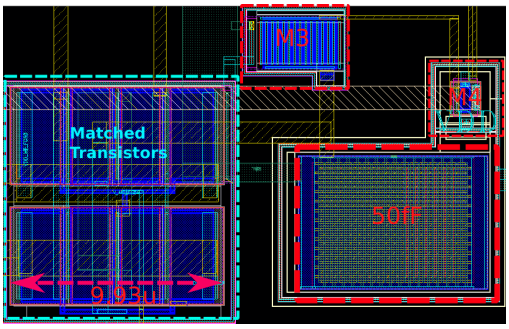
FG Cells with Capacitors and Matched Transistors



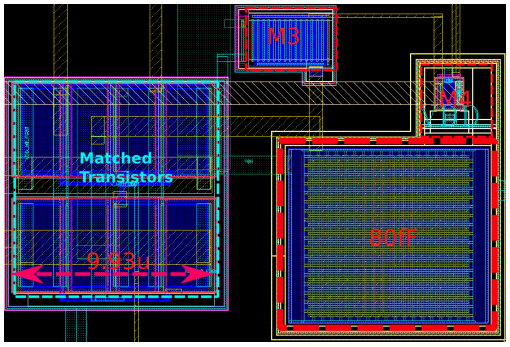
(a) Schematic



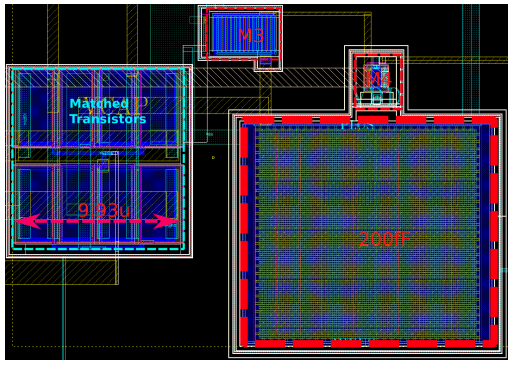
(b) FGN 10fF



(c) FGN 50fF

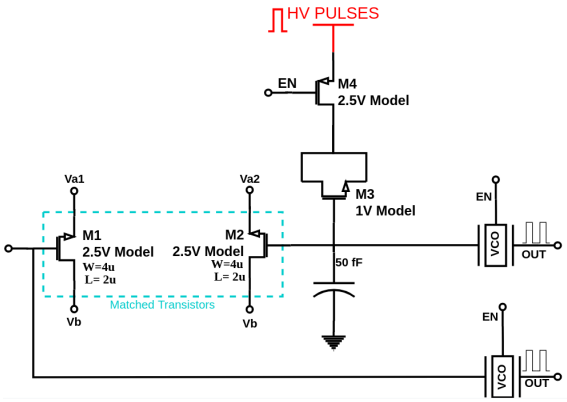


(d) FGN 80fF

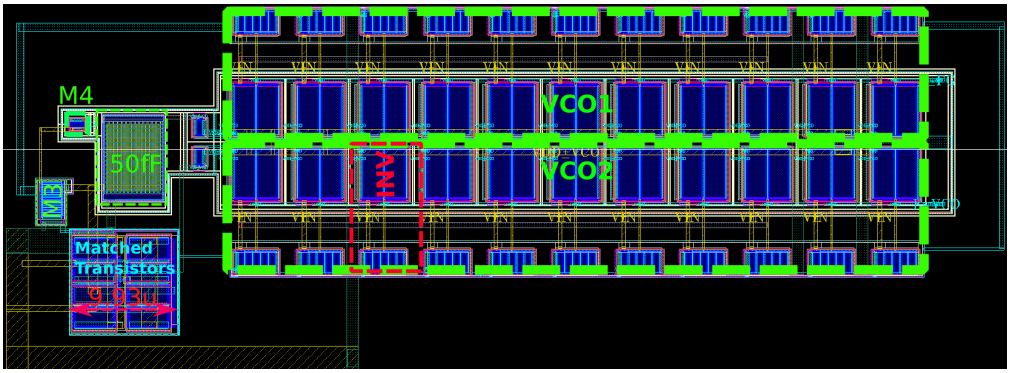


(e) FGN 200fF

Figure A.5 FG cells with an NMOS for the FG transistor and variable capacitance values

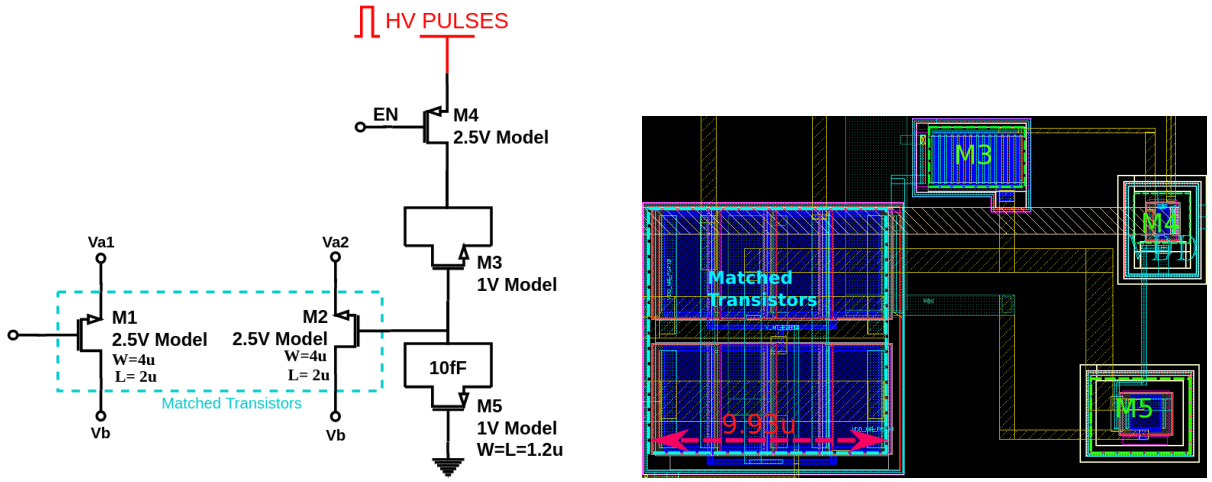


(a) Schematic



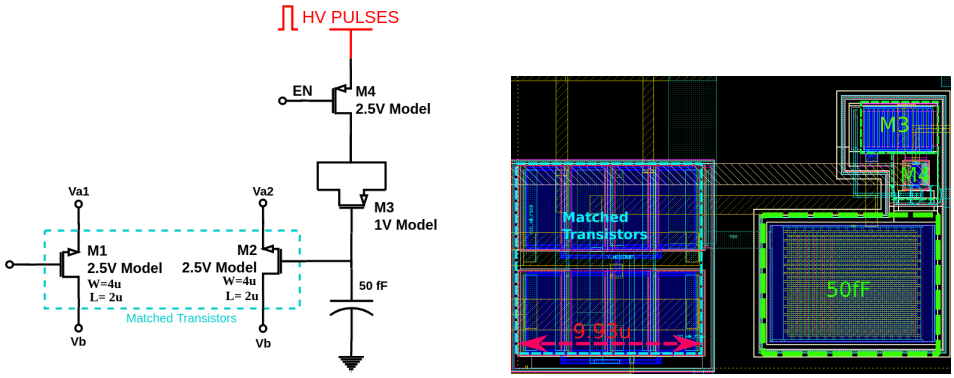
(b) FGN 50fF with VCO

Figure A.6 FG cell with an NMOS for the FG transistor and a capacitance value of 50fF attached to a VCO to match the cells based on frequency



(a) Schematic (b) FGN PMOS 10fF

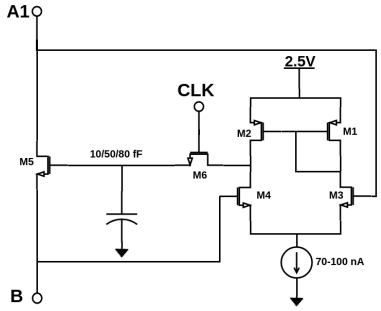
Figure A.7 FG cell with an NMOS for the FG transistor and a PMOS transistor to substitute a capacitor of capacitance value $10fF$



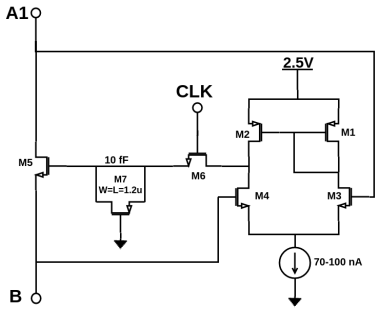
(a) Schematic (b) FGP 50 fF

Figure A.8 FG cell with a PMOS for the FG transistor and a capacitance value of 50 fF

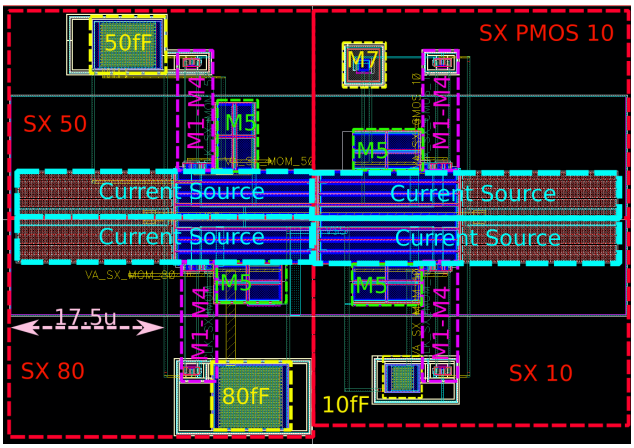
Memristor Emulator Cells



(a) Schematic of SX Cell with Capacitor



(b) Schematic of SX Cell with a PMOS as a Capacitor

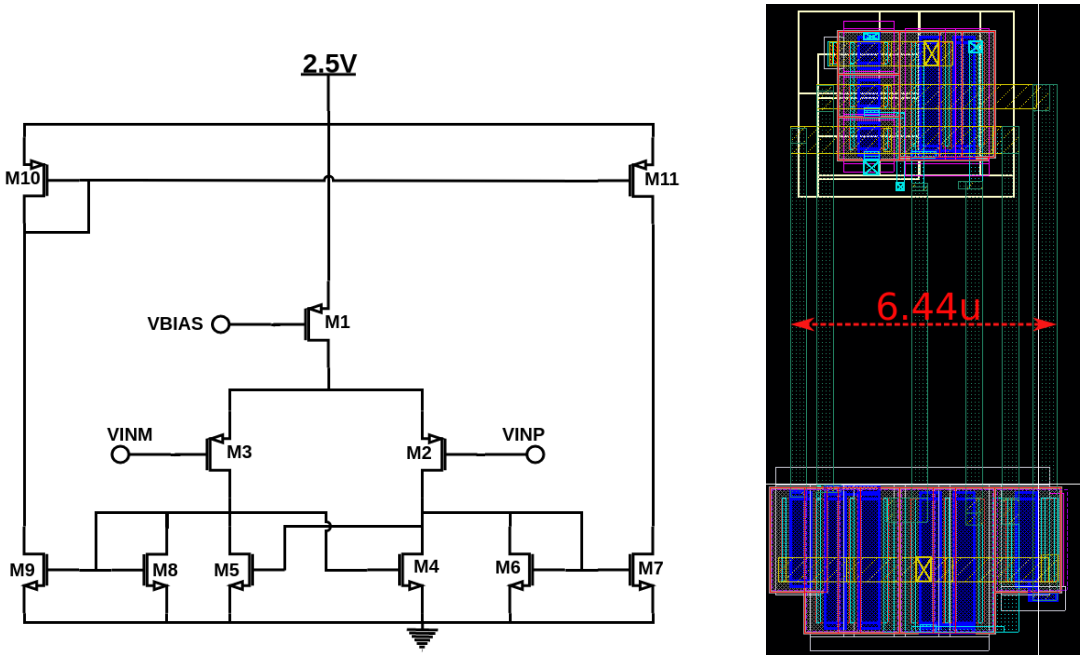


(c) Layouts

Figure A.9 Different variants of the SX cells

Memristive Cells

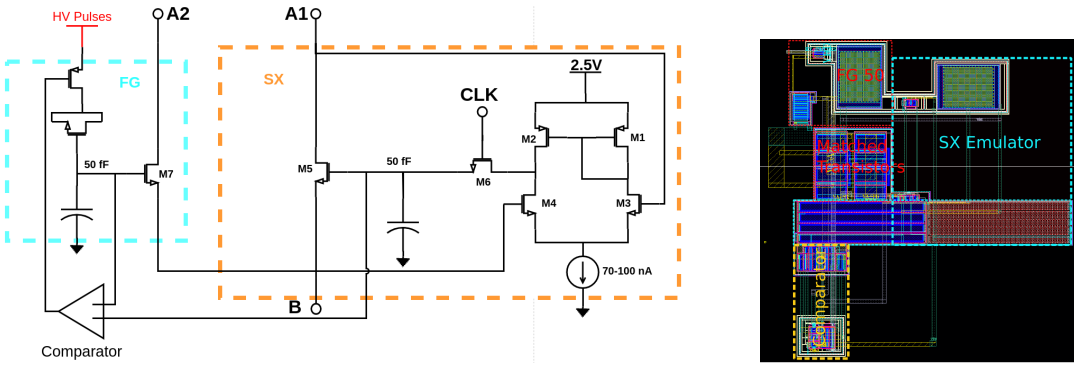
Comparator



(a) Schematic (b) Layout

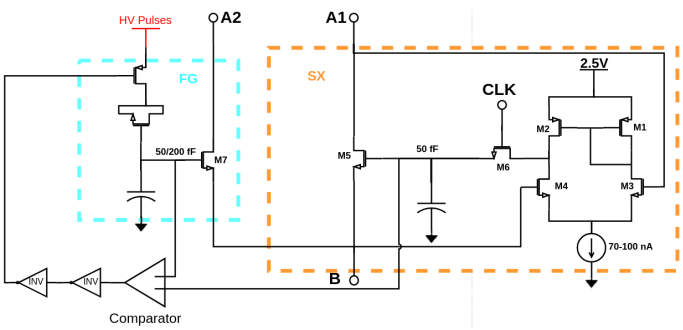
Figure A.10 The used comparator in the chip

Memristive Cells

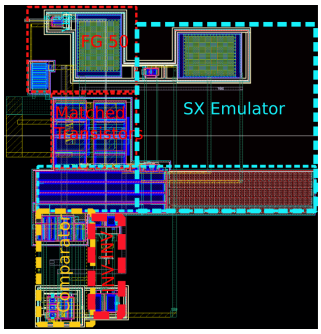


(a) Schematic (b) Layout

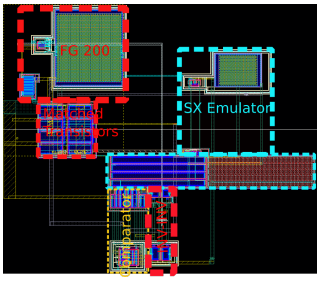
Figure A.11 The first memristive cell with a 50 fF capacitance



(a) Schematic



(b) FG50 - SX50 layout



(c) FG200 - SX50 layout

Figure A.12 The second memristive cell with a 50 fF capacitance for the SX emulator and a capacitance of 50 fF or 200 fF for the FG cell