University of Trento

XXIX Cycle

Memristor-Based Computing Architecture with Advanced Signal Processing Capabilities

Author:

Olufemi Akindele Olumodeji

A thesis submitted in fulfillment of the requirements for the degree of Doctor of Philosophy

in the

Doctoral School in Materials, Mechatronics and Systems Engineering

Department of Industrial Engineering

December 2017

Memristor-Based Computing Architecture with Advanced Signal Processing Capabilities
Olufemi Akindele OLUMODEJI

Supervisors

Dr Massimo GOTTARDI
Centre for Materials and Microsystems
Fondazione Bruno Kessler
Via Sommarive, 18, 38123 Trento TN
Italy

Prof. Gianfranco Dalla Betta
Department of Industrial Engineering
University of Trento
Via Sommarive, 9, 38123 Trento TN
Italy

Reviewers

Prof. Luca Benini
Department of Electrical, Electronic,
and Information Engineering
University of Bologna
Via Zamboni, 33 - 40126 Bologna
Italy

Dr. Victor Erokhin
National Research Council - IMEM
Parco Area delle Scienze, 37/A,
43124 Parma PR, Italy

Examiners

Prof. Paolo Bassi
Department of Electrical, Electronic,
and Information Engineering
University of Bologna
Via Zamboni, 33 - 40126 Bologna
Italy

Dr. Victor Erokhin
National Research Council - IMEM
Parco Area delle Scienze, 37/A,
43124 Parma PR, Italy

Dr. Lucio Pancheri
Department of Industrial Engineering
University of Trento
Via Sommarive, 9, 38123 Trento TN
Italy

University of Trento - Department of Industrial Engineering
Doctoral Thesis

Olufemi Akindele Olumodeji - 2017

Published in Trento (Italy) – by University of Trento
ISBN: _ _ _ _ _ _ _ _ _ _ _ _

Declaration of Authorship

I, Olufemi Akindele OLUMODEJI, declare that this thesis titled, "Memristor-Based Computing Architecture with Advanced Signal Processing Capabilities" and the work presented in it are my own. I confirm that:

- This work was done wholly or mainly while in candidature for a research degree at this University.
- Where any part of this thesis has previously been submitted for a degree or any other qualification at this University or any other institution, this has been clearly stated.
- Where I have consulted the published work of others, this is always clearly attributed.
- Where I have quoted from the work of others, the source is always given. With the exception of such quotations, this thesis is entirely my own work.
- I have acknowledged all main sources of help.
- Where the thesis is based on work done by myself jointly with others, I have made clear exactly what was done by others and what I have contributed myself.

Signed: Olufemi A. Olumodeji

Date: December 4, 2017

"A writer is a person for whom writing is more difficult than it is for other people."

Thomas Mann

UNIVERSITY OF TRENTO

Abstract

Doctoral School in Materials, Mechatronics and Systems Engineering
Department of Industrial Engineering

Doctor of Philosophy

Memristor-Based Computing Architecture with Advanced Signal Processing Capabilities

by Olufemi Akindele OLUMODEJI

Memristor-based computing architecture with advanced signal processing capabilities investigates analogue applications of memristors, particularly in image processing, combining them with conventional electronic circuitry.

The concept of memristor (short for memory resistor) was first theorised in 1971 by Prof. Chua while reasoning on the theoretical grounds of of the symmetry of equations governing the fundamental passive circuit theory.

This thesis can be split into two parts as follows: i) Memristor Device, Modelling, Characterisation and Programming and ii) Memristor Circuit Applications. In the first part, an overview of the theory of memristors which gives an introductory background is discussed alongside the device modelling to fit experimental data. Electrical characterisation was carried out on fabricated memristors to validate the fundamental fingerprint of these devices and finally, the different programming techniques, particularly the pulse-based technique which was widely used in this work, was exhaustively treated.

In terms of applications, starting from a novel memristor-based light to resistance encoder, a more complex architecture based on adaptive background subtraction for scene interpretation used for the analyses of motion and its association to a particular object in the scene is treated in this work. Throughout this thesis, the intention of an overview on the application of memristors in image processing algorithm is emphasised and the last chapter discusses a neural network architecture based on memristors. The intended neural network architecture was trained to perform colour classification targeting applications based on gesture detection.

In essence, *Memristor-based computing architecture with advanced signal processing capabilities* gives an insight into the advantages to come having an hybrid system of standard CMOS image processing techniques with memristive devices particularly in computation-intensive applications requiring high speed and massive parallel signal processing. Typically the realisation of such powerful and dense networks in an integrated

circuit with acceptable size which is not resource hungry by using the commonly encountered elements and conventional CMOS technology is becoming increasingly difficult to achieve. Computing architectures based on memristors present the advantages that could help overcome the limitations of an overall implementation with conventional electronic elements.

Acknowledgements

The research leading to this thesis has been carried out under the framework of the MaDEleNA project funded by the Autonomous Province of Trento in collaboration with the University of Trento and Fondazione Bruno Kessler (FBK). I sincerely acknowledge everyone who has made an impact towards the success of this thesis during the course of my research.

First and foremost I would like to thank my principal supervisor, Dr Massimo Gottardi of Fondazione Bruno Kessler. I sincerely appreciate his enthusiastic supervision and active participation in the research which has led to the success of this thesis. It has really been an honour working with him and the rest of the team at the Integrated Radiation and Image Sensors group. I am also indebted to my co-advisor from UNITN Prof. Gian Franco Dalla Betta for his invaluable support during the course of my research. My gratitude goes to Alessandro Paolo Bramanti (STMicroelectronics) for his collaboration in co-authoring several publications.

I wish to thank Dr Leandro Lorenzelli and Dr Cristian Collini of FBK BioMEMs Lab for their support in the provision of samples and characterisation of inorganic memristor with the resources of the BioMEMs lab. Likewise I would like to acknowledge the support Dr Victor Erokhin and Dr Silvia Battistoni for their support in providing me with samples of organic memristors and valuable comments in their characterisation.

I was honoured to be hosted at the Chair of Computer Science 3 (Computer Architecture), University of Erlangen-Nuremberg where I worked on memristor parallel processing architectures. I would like to thank Prof. Dr.-Ing. Dietmar Fey for the invitation and the excellent working conditions.

I owe countless number of people gratitude for the time, care and advice rendered to me during the course of my research. I would like to thank Dr Safa Dhaher, Federica Setti and family, Yady Tatiana Solano Correa, Claudia Owoola, Giulia Roncador, my PhD colleagues at FBK with whom I've shared the office space for the last three and the half years, for their friendship, encouragement and support. A special thanks goes to Dr Milena Belloni and family for their support and encouragement through out the course of my PhD.

Finally I would like to thank my family for the love, support and guidance I have received throughout all my life. I dedicate this thesis to my late parents whom I know would have been proud of my accomplishments despite the odds. . . .

Contents

List of Figures

List of Tables

List of Abbreviations

ADC	Analogue to Digital Converter
ANN	Artificial Neural Network
CAD	Computer Aided Design
CCII	Second Generation current Conveyor
CMOS	Complementary Metal Oxide Semiconductor
COMP	Comparator
CS	Chip Select
CNN	Convolutional Neural Network
DC	Direct Current
DDCC	Differential Difference Current Conveyor
DigPot	Digital Potentiometer
DR	Dynamic Rnage
DRMMSE	Doctoral School in Materials, Mechatronics and Systems Engineering
DSP	Digital Signal Processing
DUT	Device Under Test
FBK	Fondazione Bruno Kessler
FPN	Fixed Pattern Noise
HP	Hewlett Packard
IRIS	Integrated Radiation and Image Sensors
L2F	Light to Frequency
LCO	light-controlled oscillation
LID	Linear Ion Drift
LPP	Low Pass Filter
LUT	Look Up Table
MAND	Memorised AND Gate
MOS	Metal–Oxide–Semiconductor
NI	National Instrument
Op-amp	Operational Amplifier
OTA	Operational Transconductance Amplifier
PANI	Polyaniline
PCB	Printed Circuit Board
PC	Personal Computer

PEO	Polyethylene Oxide
PFM	Pulse Frequency Modulation
PMCS	Pulsed Microplasma Cluster Source
PMOS	P-type Metal–Oxide–Semiconductor
PWM	Pulse Width Modulation
RGB	Red Blue Green colour mode
SDI	Serial Digital Interface
SMU	Source Measuring Unit
SPI	Serial Peripheral Interface Bus
VCR	Voltage Controlled Resistor
VLSI	Very Large Scale Integration

In loving memory of my beloved parents Mr and Mrs Olumodeji. . .

Chapter 1

Introduction

This thesis reports the research path I followed during the course of the 29th cycle of the PhD program of the Doctoral School of Materials, Mechatronics and Systems Engineering (DRMMSE) at the University of Trento between January 2014 and December 2016.

As we will see in this thesis, Memristor-based computing architecture investigated analogue applications of memristors combining them with conventional electronic circuitry. The non-volatility and the relatively nanoscale sizes of memristors have been taken advantage of to proffer solutions to some analogue applications that requires non-volatile storage, therefore enabling improvements to architectures which are difficult to realise and are area and power hungry with conventional CMOS circuitry.

Starting from memristor device fabrication, characterisation and modelling, this work analysed the various techniques of programming memristors all in the aim of using them in programmable analogue circuits. Particularly, the direction this work follows is in the applications of memristor in image processing with a target application in motion and gesture recognition. As we will see in this work, memristor-based computing will be advantageous in large-scale, highly parallel mixed-mode processing architectures.

My research was carried out in the Integrated Radiation and Image Sensors (IRIS) Unit of the Fondazione Bruno Kessler (FBK) under the supervision of Dr. Massimo Gottardi. The research I present in this thesis was carried out under the framework of the MaDEleNA ("*Developing and Studying novel intelligent nanoMaterials and Devices towards Adaptive Electronics and Neuroscience Applications*") project financed by the Provincia Autonoma di Trento (I), Call Grandi Progetti 2012.

1.1 Motivation

The limitations imposed by power consumption, the end of Dennard scaling theory (describing consistent improvements in transistor density, cost performance and power) and high variability in nanoscale technology are very significant problems with respect to Moore's Law. Even if few concepts in our time have had as much influence on the economy in the last 60 years, it is now very clear that we are approaching the end of the exponential growth with the computer performance doubling every two years. By current estimates the 5 nm technology is projected to be reached by semiconductor companies in the 2020 time frame. This roadmap has been based on the continuing extension of CMOS technology for at least 25 years, but this roadmap does not guarantee any more that silicon-based CMOS will extend beyond. On the other hand, a new electronic component, the memristor, has been physically realised in 2008 at Hewlett-Packard Labs using a thin film of titanium dioxide. This new type of component, for which several realisations are now available, combines the functions of memory and logic.

Significant interest has been placed on developing systems based on memristors since the initial fabrication by HP Labs in 2008. The memristor is a nanoscale device with dynamic resistance that is able to retain the last programmed resistance value after power is removed from the device. This property shows that the memristor can be used as a non-volatile memory component, and has potential to enhance many types of systems, such as high-density memory, and neuromorphic computing architectures.

For nearly 180 years, it has been accepted that there are three fundamental passive circuit elements, the resistor (1827), the capacitor (1745), and the inductor (1831). In 1971, Prof. Leon Chua theorised that mathematically there should be a fourth fundamental circuit element based on the symmetry of the equations that govern passive circuit theory [1, 2, 3, 4, 5]. Dr Chua called this device the memristor (short for memory-resistor), it was not until 2008 that results of the device in physical form were published [6, 7, 8]. Details of the memristor theory based on the the symmetry of the passive circuit elements is further discussed in Chapter 2.

The physical memristor is a nanoscale device that has unique properties that can be used to greatly improve existing electronic systems and computing architectures. The memristor can be thought of as a time varying resistor where the resistance changes due to the summation of current that has passed through the device [9, 10]. When the current flowing through the device is zero, the summation of current becomes constant, and thus the resistance remains unchanged. This shows that the memristor can be used as a non-volatile memory component.

Furthermore, the dynamics of a memristor closely resemble those of a synapse in brain tissue. Just as the values of synaptic weights change with the application of neural spikes, the resistance value of a memristor can be changed with the application of a voltage pulse. This could provide significant advancements in the field of neuromorphic computing as electronic systems using memristors could be fabricated with a device density similar to that of the human brain.

Since the memristor's physical discovery, memristors have been fabricated using a variety of different materials and device structures of which the TiO2-based memristors remains popular. Different device structures are still being developed to determine which memristor device would be the best option for commercial use. This is based on many factors such as size, switching speed, power consumption, switching longevity, and possible integration with CMOS technology.

Several SPICE device models of the memristor exist already which could be used to simulate the basic behaviours of memristor. As a result of the complexity and unpredictability of the device behaviour as a result of different fabrication processes, device structure among other factors, it is therefore impossible to have a standalone model which can exist as a generalised memristor model. In order to better simulate different circuital configuration with memristors matching the ones fabricated by FBK and the University of Parma, in was inevitable to have a model closely fitting experimental measurements. The memristor device model developed during the course of this research served as the first steps in validating electronic systems and computing architectures based on memristors.

1.2 The MaDEleNA Research Project

The MaDEleNA research project within which this thesis evolves around stands for Developing and Studying novel intelligent nanoMaterials and Devices towards Adaptive Electronics and Neuroscience Applications. The objective of the project is the development of neuro-bio-inspired electronic systems based on elements mimicking the function of natural nervous systems and brain, thus memristive systems, by combining materials research and novel hardware design.

MaDEleNA officially started in September 2013 and it is scheduled to end in August 2017. It is composed of a consortium of five institutions of which the University of Trento and FBK are participants. My thesis evolved around several work-packages in the projects with core contribution in the electrical modelling of the organic/inorganic memristors and the design of novel electronic circuits for memristors. The MaDEleNA project has produced over 100 peer reviewed publications, 10 of which originated totally or in part, by the research activity reported in this thesis.

Some major tasks in the work-packages this work is based on could be described below:

- Electrical model of the memristor using SPICE and CADENCE: will report the description of the electrical model of both the memristor and memristive components supported by simulation results. The models will be included in the libraries of electrical components for design tasks. The developed model will be compared regarding the possibility of the describing properties of real experimentally fabricated devices. The possibility to adopt this model in more complex circuital configurations for validating the design of circuit network in order to implement more complex functions.

- Memory arrays and prototype processor: memristive devices will be integrated with traditional electronics to implement computing blocks and will be capable of adaptation (learning). Both properties mimic the basic working principles of natural nervous systems, which are known to outperform traditional computers as to flexibility and robustness in complex computing tasks. A memristive processor will be demonstrated, meaning by processor an adaptive network, which can be trained to associate pairs of input and output stimuli in a previously unknown and modifiable way.

The outcomes of this work-package have had highly innovative technological, analytical and microelectronic contents.

1.3 Thesis Outline

This thesis encompasses the areas of memristor device, modelling, fabrication, characterisation and several novel memristor circuit applications. As my PhD followed the evolution of the MaDEleNA project, this thesis is a reflection of the paths through different research fields around which the project has been executed.

In the following, I review the rest of the Chapters in this thesis, with reference to relevant related publications.

- **Chapter 2: Memristors and Memristive Systems**
  This introductory section of this chapter touches on the theoretical definition, the concept of the memristor and its general properties with a review of the fundamental properties of the memristor starting from the basic mathematical equations governing memristive systems. Furthermore the device fabrication steps, experimental characterisation for both organic and inorganic memristive devices as well as device modelling are treated.

  Related publications:

  - O. A. Olumodeji and M. Gottardi. "Behavioural modelling of memristive devices targeted to sensor interfaces". In: *AISEM.*. IEEE. 2015, pp. 1–4

- **Chapter 3: Memristor Emulator**
  This chapter discusses an overview of memristor emulators. A novel way of emulating memristive devices is presented. This particular technique which is of

educational value is based on a digital potentiometer and a microcontroller in the form of an Arduino. The developed emulator played an important role in this thesis as we will see in subsequent chapters where the emulator was used to validate the working principle of the memristor.
Part of the work presented in this Chapter was given as an oral talk in the IEEE PRIME conference 2016, held in Lisbon, Portugal and won the IEEE gold leave award for its originality.

Related publications:

- Olufemi A Olumodeji and Massimo Gottardi. "Emulating the physical properties of HP memristor using an arduino and a digital potentiometer". In: *Ph. D. Research in Microelectronics and Electronics (PRIME), 2016 12th Conference on*. IEEE. 2016, pp. 1–4

- Olufemi Akindele Olumodeji and Massimo Gottardi. "Arduino-controlled HP memristor emulator for memristor circuit applications". In: *Integration, the VLSI Journal* 58 (2017), pp. 438 –445. ISSN: 0167-9260. DOI: `http://dx.doi.org/10.1016/j.vlsi.2017.03.004`. URL: `http://www.sciencedirect.com/science/article/pii/S0167926017301499`

- **Chapter 4: Memristor Resistance Modulation**
  Chapter 4 discuses memristor high resolution state tuning which is an aspect important in order for memristors to truly take their rightful place in analogue electronics. Several resistance modulation techniques and their setbacks are presented, particularly the pulse-based programming technique which is arguably the most popular. A DC programming technique which led to the development of the switched memristor circuit is also presented. Experimental characterisation on memristors developed by FBK was also carried as a proof of concept for both the pulse-based and DC programming techniques. This was done by first investigating the current-voltage Lissajous behaviour of the memristors. Finally, two memristor circuit based on self terminating programming methods are also presented.
  Part of the work in this Chapter was presented in the IEEE PRIME conference 2015, held in Glasgow, Scotland and in the IEEE ISCAS conference held in Baltimore-Maryland, US.

Related publications:

- O. A. Olumodeji and M. Gottardi. “A pulse-based memristor programming circuit”. In: *2017 IEEE International Symposium on Circuits and Systems (ISCAS)*. 2017, pp. 1–4. DOI: `10.1109/ISCAS.2017.8050793`

- O. A. Olumodeji and M. Gottardi. “A Pulse-based Memristor programming Circuit”. In: *Circuits and Systems (ISCAS), 2017 IEEE International Symposium on*. IEEE. 2017, pp. 232–235. DOI: `10.1109/PRIME.2015.7251377`

- O. A. Olumodeji and M. Gottardi. “Memristor-based comparator with programmable hysteresis”. In: *Microelectronics and Electronics (PRIME), 2015 11th Conference on Ph. D. Research in*. IEEE. 2015, pp. 232–235. DOI: `10.1109/PRIME.2015.7251377`

- **Chapter 5: Memristor-Based Light-to-Resistance Encoder**
The implementation of a novel memristor-based light to resistance converter based on the principle of multiple-reset pulse frequency modulator. The proposed pixel architecture embedding a single memristor as an analogue counter is presented.

Related publications:

- Olufemi A Olumodeji, Alessandro Paolo Bramanti, and Massimo Gottardi. “A memristor-based pixel implementing light-to-resistance conversion”. In: *Optical Engineering* 55.2 (2016), pp. 020501–020501. DOI: `10.1117/1.OE.55.2.020501`

- **Chapter 6: Memristor-Based Pixel Architecture for Dynamic Background Subtraction**
Based on the light to resistance encoding treated in the previous Chapter, a pixel architecture relying on memristive devices to perform pixel-level adaptive background is presented in this Chapter subtraction. Core of the processing is the pixel, containing a light-to-frequency converter and two additional memristors used to store the dynamic boundaries, outside which the behaviour of the photo-generated signal being converted to a resistance value is recognised to be anomalous.
Part of the work in this Chapter was presented in the IEEE SENSORS conference 2015, held in Busan, South Korea.

Related publications:

- Olufemi Akindele Olumodeji, Alessandro Paolo Bramanti, and Massimo Gottardi. “A Memristive Pixel Architecture for Real-Time Tracking”. In: *IEEE Sensors Journal* 16.22 (2016), pp. 7911–7918. DOI: `10.1109/JSEN.2016.2606599`

- O. A. Olumodeji, A. P. Bramanti, and M. Gottardi. “Memristor-based pixel for event-detection vision sensor”. In: *SENSORS, 2015 IEEE*. 2015, pp. 1–4. DOI: `10.1109/ICSENS.2015.7370688`

- **Chapter 7: Memristor-Based Neural Network for Image Processing** Design and simulation of an RGB colour sensor neural network based on experimentally acquired datasets. The neural network was designed and validated on Matlab. Hardware design of the board carried out within the framework of MaDEleNA for use by other partners is also described in the annex related to this chapter.

  Related publications:

  - Olufemi A Olumodeji et al. “Estimating illuminant chromaticity with a low-power color pixel”. In: *AISEM Annual Conference, 2015 XVIII*. IEEE. 2015, pp. 1–4. DOI: `10.1109/AISEM.2015.7066815`

Chapter 2

Memristors and Memristive Systems

2.1 Introduction

The concept of memristor (short for memory resistor) was first theorised in 1971 by Prof. Chua. Before then, it was assumed that there were three basic fundamental passive circuit element, the capacitor (1745), the resistor (1828) and the inductor (1831). While reasoning on theoretical grounds of the symmetry of equations governing the fundamental passive circuit theory, Chua predicted that apart from resistors, capacitors and inductors, there should be a fourth fundamental passive circuit element, the memristor which is defined by the constitutive relationship between electric charge q and magnetic flux Φ [21, 22].

The theory of the fourth fundamental circuit element however remained a puzzle for decades as there was no physical implementation to back it up. As some scholars have argued, this was due in part to the existing technology and also the memristor being a nano-scale device even though for decades, memristive behaviours have been unwittingly observed [23, 24, 25, 26, 27]. It was therefore difficult to for the memristors to be identified until advancements were made in nanotechnology The memristor can be considered as a time varying resistor whose resistance depends on the history of current that has passed through it [1],[28]. Its resistance changes as a result of the summation of current that has passed through the device. When the current flowing through the device is zero, the summation of current becomes constant, and thus the resistance remains unchanged. This gives the memristor the potential of being adopted as a non-volatile memory component [29].

In this Chapter, the memristor upon which this work is based, is introduced. Experimental characterisation for both organic and inorganic memristive devices as well as device modelling is treated. This chapter is organised as follows. In Section 2.2, the fundamental properties of the memristor are defined starting from the postulated theoretical notion on Chua. Section 2.3 describes the first solid state memristor fabricated by HP labs in 2008. A general overview of the polyaniline-based memristor together with some

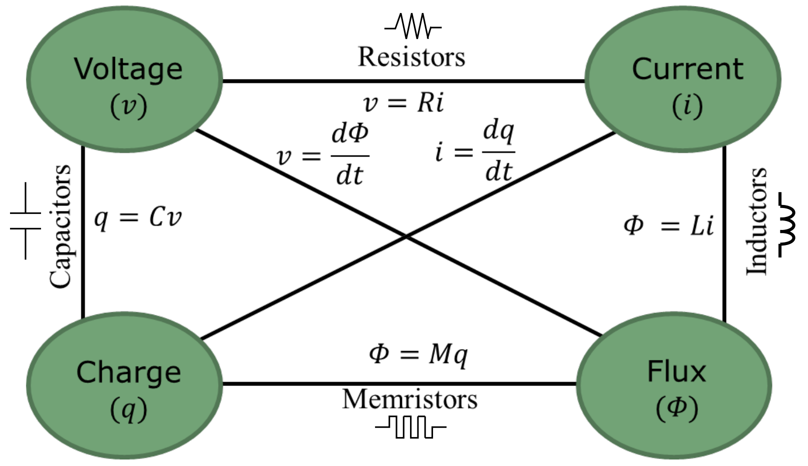


FIGURE 2.1: Symmetry diagram showing the six distinct possible realisations based on the four fundamental two-terminal circuit elements: resistor, capacitor, inductor and memristor. The relationships are drawn from the four fundamental circuit variables, voltage (v), current (i), charge (q) and flux (Φ)

experimental measurements is presented in Section 2.4. Results of FBK Memristor device fabrication and characterisation are presented in Section 2.5. Finally, the memristor device modelling and equations are given in Section 2.6 before concluding in Section 2.7.

2.2 Fundamental Properties of Memristors

Memristors have unnoticeably been existing for centuries. In the 70's, Leon Chua, a circuit theorist published his seminal paper on the memristor's working concept. There, he theorised that apart from the three already existing passive circuit element (resistor, capacitor and inductor), there should be a fourth element based on the symmetry of relationships between the equations governing the fundamental passive circuit variables [30, 31]. Let us recall that three basic circuit elements – resistor, capacitor, and inductor – are defined by a relationship between two of the four axiomatic circuit variables: the voltage, $v(V)$ – work done required to bring charge from ∞ to an electric field; the current, $i(A)$ – flow of electric charge; the charge, $q(C)$ – energy per electron; and the flux $\Phi(W)$ – rate of flow through an area. Where current is the derivative of charge $i{=}\frac{dq}{dt}$ and voltage is the derivative of magnetic flux, $v{=}\frac{d\Phi}{dt}$.

As shown in the symmetry diagram in Fig. 2.1, the resistor is identified by the correlation between voltage and current $v = Ri$, the capacitor is defined as the correlation between voltage and charge $q = Cv$, and inductor, the linkage between current and magnetic flux $\Phi = Li$. The relationship charge and flux is what Chua was puzzled about which is defined as memristance $M = \frac{d\Phi}{dq}$.

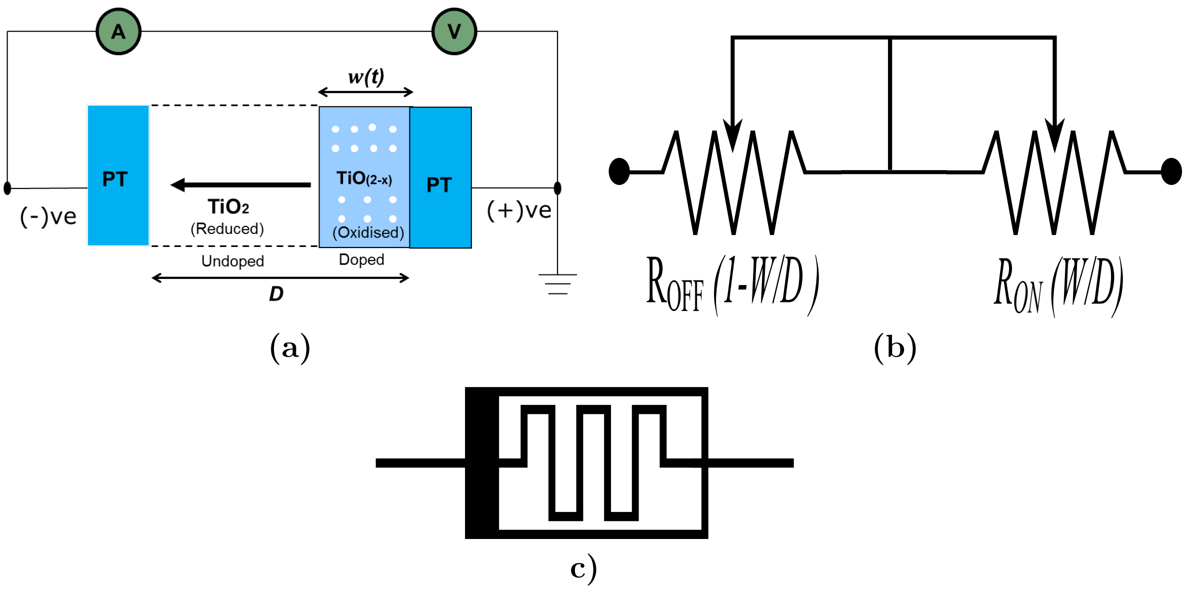


FIGURE 2.2: Diagram of the memristor with simplified equivalent circuit. (a) Cross-section of the memristor showing the Structure of the TiO_2 memristor consisting of a high conductive (doped) and a low conductive (undoped) layer sandwiched in between two platinum electrodes. The boundary between the two parts is dynamic and is moved back and forth as a function of the passing charge carriers. The parameter w(t) is state variable that describes the position of this boundary, (b) Equivalent resistor circuit, (c) Symbolic representation a memristor in an electric circuit

In 1976, Chua and kang introduced a generalised class of non-linear devices known memristive systems [28] defined by the relationships in Equations 2.1 & 2.2.

$$v = R(w,t)i, \tag{2.1}$$

$$\frac{dw(t)}{dt} = f(w,t), \tag{2.2}$$

Equations 2.3 & 2.4 is a system of two equations which describe memristors as a generalised concept of memristive systems. This generalised memristive systems represent a non-linear state-dependent version of Ohm's law where the internal state variable is described by w, v & i represent the voltage and current across the device and R is a resistance, better known as memristance (short for memory resistance) with the physical units of Ohm (Ω).

$$I = g(w,V).V, \tag{2.3}$$

$$\frac{dw(t)}{dt} = f(w,V) \tag{2.4}$$

Equations 2.3 & 2.4, w is the physical variable indicating the internal memristor state, the function $g(.)$ is the memristor's conductance. The state variable w, is in theory, bounded between zero and D such that $0 < w < D$. D being the thickness of the transient material oxide thin-film sandwiched between two metal electrodes. I represent the current flowing through the device and V represents the voltage across the device.

2.3 The HP Memristor

In 2008, HP announced the realisation of the first physical model based on two regions of TiO_2 [6]. An undoped, highly resistive region and a doped region with highly conductive oxygen vacancies TiO_{2-x} both sandwiched between two metal electrodes as shown in Fig. 2.2a. The doping process involves removing the negative charged oxygen atom from its substitutional site in TiO2, creating a positively charged oxygen vacancy [32]. The forming of these oxygen vacancies occurs at the time of crystallisation. The boundary between the doped and undoped layers shifts with the application of an external bias across the device. This shift which is a function os the applied voltage or current causes a change in resistance. The resistance change between the two electrodes due to the drift of charged dopant [33].

2.4 Polyaniline-Based Memristor: PANI and PEO

Properties of the memristor have been observed in organic materials even before the claim by HP to have discovered the missing fourth passive circuit element. Erokhin *et*

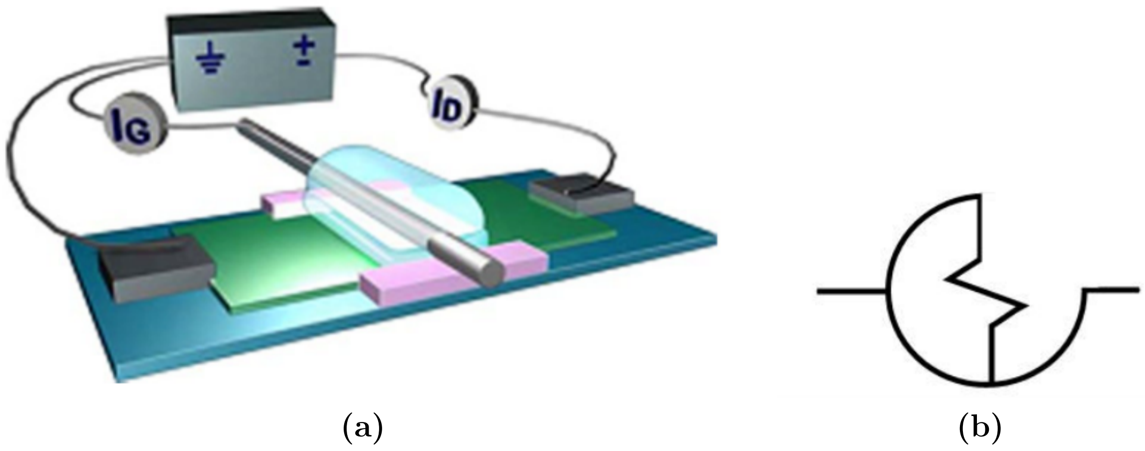


FIGURE 2.3: Structure of the polymeric memristor element and its symbol for electric circuits. **(a)**, The active layer was formed from the conducting polymer (PANI) attached two metal electrodes. A stripe of solid electrolyte (PEO) was deposited in the central part of the PANI layer in order to provide a suitable medium for redox reactions. The area of PANI layer under the electrolyte is the active zone. The reference potential was provided by a silver wire inserted into the electrolyte. The wire was connected to the one of two metal electrodes as shown in the schematics **(b)**, Symbolic representation of the organic memristor in an electric circuit. Diagram of the organic memristor structure adapted from [35].

al.'s organic memristor is a two terminal polymeric electronic device developed at the university of Parma [34]. The organic memristor is made up of a thin film of polyaniline (PANI) and a lithium doped solid polyelectrolyte (polyethylene oxide (PEO)). Figure 2.3 is the schematic of the organic memristor showing its measurement interconnection. The active layer is formed by PANI in its emeraldine salt (Chlorine-doped) form which is a conductive polymer. A stripe of PEO (a water solution of polyethylene oxide doped $LiClO_4$ used as a solid electrolyte) is deposited on the centre of the PANI layers in a crossed configuration. As seen in the diagram, the PANI is connected to device terminals (chrome electrodes) and a silver wire is connected to the PEO which works as a reference electrode. The active zone where all redox reactions and the conductivity variations occur is the area of contact between the PANI layer and the PEO. The two chrome electrodes are referred to as source (S) and drain (D) and the reference electrode is called the gate (G). As seen in the circuit diagram, when connected, the device would have two working electrodes only S and D. The reference electrode is connected to one of the terminals S or D (typically the one kept at ground potential). Thus there are two currents flows (electronic and ionic) which can be easily measured.

The working principle of the device is based on the variation of the conductivity of a the conducting polymer multilayer (polyaniline, PANI) in its oxidised and reduced states[36]. This variation is due to the ionic flux flowing through PANI multilayer at the

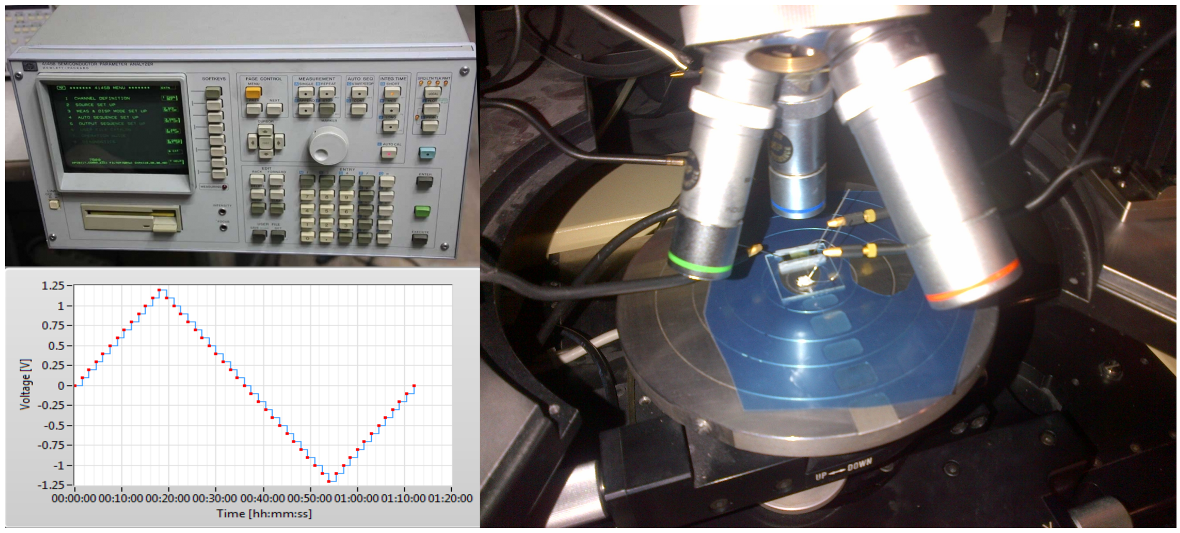


FIGURE 2.4: Organic Memristor Measurement Setup. The HP4145B Semiconductor Parameter Analyser, Programme voltage sweep and the device under test (DUT) are clearly seen in the figure.

junction with a with a film of a solid electrolyte (Li-doped Polyethylene oxide, PEO) [37].

2.4.1 Organic Memristor Characterisation

This section touches on the experimental characterisation carried out on organic memristor devices fabricated at the university of Parma.

In order to have a good characterisation of the device, during the application of the voltage cycles, two currents are measured — the ionic current flowing through the reference electrode otherwise referred to as the gate current I_G and the total current flowing through the drain electrode referred to as drain current I_D. In a typical memristor characterisation setup at the University of Parma, the application of the voltage cycles and the measurements of the total current are performed with a Keithley 236 Source Measure unit while the gate current is registered with a Keithley 6514 system electrometer [38].

The ionic and and electronic charges which characterises the device flow in perpendicular directions. The actual potential of the active layer with respect to a reference point, in this case gorund potential, determines the ionic flow and provides the PANI layer with incoming-outgoing Li+ ions, therefore varying its conductivity [39].

The time integral of the ionic current (transferred charge) actually determines the resistance of the active zone. It is therefore imperative that the value of the ionic current is less than the current flowing through the PANI layer by at least one order of magnitude. This means that the actual measured current of the element in the conducting state,

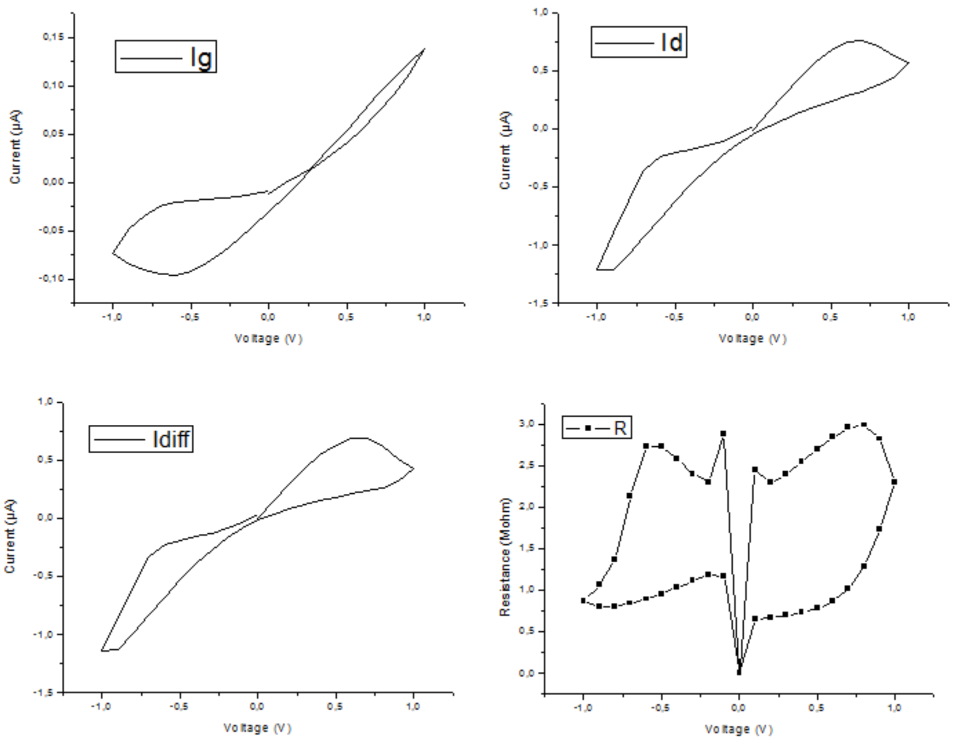


FIGURE 2.5: Organic Memristor Characterisation result showing the cyclic current-voltage characteristics.

i.e. the sum of the electronic and ionic currents, is chiefly determined by the first contribution.

The electrical conductivity of PANI is not only highly sensitive to the doping ratio of the polymer but also to its redox state. This give the film the characteristics of being easily switched to a conducting or an insulating state by simply applying an adequate electric potential at its terminals. The application of sufficient electric potential causes a displacement of lithium ions between the polyelectrolytes and the PANI film changing reversibly the conductivity of the film by up to four orders of magnitude, although the exact values, as for the actual threshold values at which these changes begin to happen, vary from device to device.

Regarding the results obtained from the experimental characterisation of the organic memristor with our setup as shown in Figure 2.4, the different measurement approach and setup is presented and results discussed. The interconnections of the organic memristor is the same as the one given in Figure 2.3. I_g is the ionic current that gives us information about the redox activity of the conductive polymer. In the electrical characterization this current is named "Gate". I_d is the total current that flows through the PANI layer, from the source to the drain electrode. It's a composition of ionic current and electronic current. So, in order to obtain only the electronic current I_{diff}, we subtract I_g from I_d.

The electrical measurements were performed using the HP4145B Semiconductor Parameter Analyser. The HP4145B is a Source Monitor Unit (SMU)-based architecture. It consists of 4 SMUs which can alternately act as a voltage source/current monitor or current source/voltage monitor. A four terminal device can be characterized without changing device connection but by simply changing the SMU's current/voltage operating mode. In our setup we only need 3 SMUs for the 3-terminal organic memristor measurements ss seen in Figure 2.4

The instrument was connected to a personal computer and controlled with LabVIEW codes. Most of the electrical measurements were performed as follows: the source and gate electrodes were at ground potential; the drain potential was swept between -1.2V and +1.2V, with steps of 0.1V, time step of 60 s and sampling time of 60 s to presumably allow the device to settle [40, 41, 42, 43, 44, 45, 46, 47]. Each sweep was started from 0 V, increasing it up to +1.2V, then decreasing it until -1.2V and finally increasing it back to 0 V as shown in the voltage sweep of Figure 2.4. An example of characterisation measurement carried out on the device is given in Figure 2.5 showing the cyclic current-voltage characteristics. I_d the current at the drain electrode, while I_g is the current at the gate. I_{diff} is the deduced electronic current and R is the cyclic resistance variation.

2.5 FBK Memristor

This Section reports the research activity developed during the first year of the MaDEleNA project. From the technological side, new approaches involving material science and micro-fabrication processes will provide the basic building blocks for realising more complex architectures based on memristive components. The different technologies and the prototypes of memristive test structures implemented are reported and particularly:

- Outline of the adopted fabrication process of the first test prototypes.
- Developed test prototypes and preliminary experimental setup.

2.5.1 Device Fabrication

A simple and low cost fabrication process was adopted for the development of test structures for exploiting the various memristive properties. Therefore, starting materials for substrates having higher insulation performances with respect the silicon wafer such as the 6" Electric Fused Quartz wafers, (double side polished and $650\mu m$ of thickness) was one of techniques exploited. As for the fabrication of the metal electrodes, the first implemented structures consist of a sandwich made of a bottom metal layer, a metal oxide and a top metal layer. The first fabrication phase was the realisation of the bottom electrodes by the following deposition process on the two wafers of quartz:

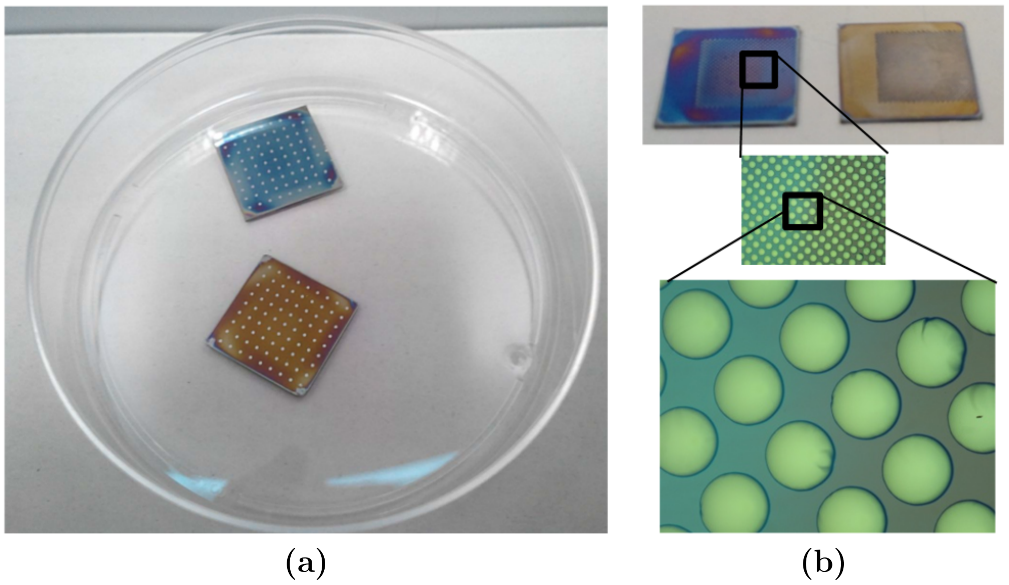


Figure 2.6: FBK memristor fabrication deposition techniques. **(a)** Aluminium dishes (∼1mm of diameter) deposited on a one-layer and on a two-layer TiO_2 by Sol-gel. The area of the dices is 2x2cm. **(b)**, Platinum dish-shaped patterns (∼0.35 mm of diameter) on a two layer TiO2 Sol-gel dices.

- 5nm of Titanium and 50nm of Platinum evaporated by electron beam technique and patterned using a hard mask
- a layer of 60 nm of Titanium deposited by sputtering

A summary of the first phase of the fabrication steps of the technological process is outlined below:

- The 2 electric fused quartz wafers (DSP, 650650μm thickness) are the starting materials
- Initial cleaning with:
  - Sulfuric acid
  - Hydrofluoric acid
  - Ammonium hydroxide
  - Chloric acid

  (D.I. water rinsing and Nitrogen drying is carried out between each cycle.)

Wafer One

- An oxygen plasma is performed
- A layer of Ti/Pt (5/50nm) was evaporated by electron beam technique

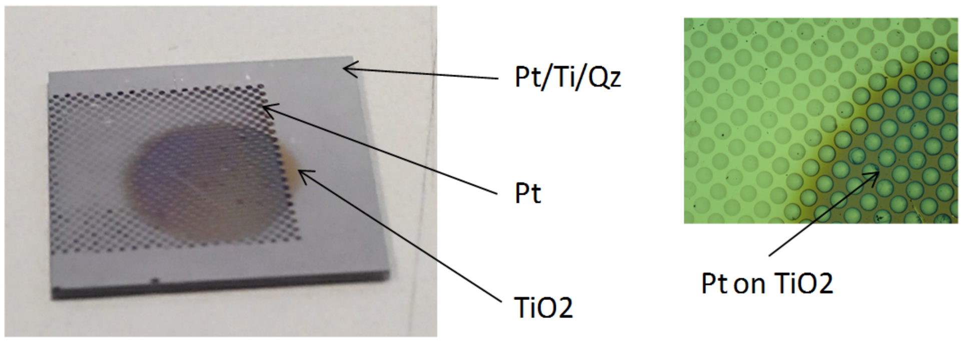


FIGURE 2.7: Quartz/Titanium/Platinum dices (1.5x1.5cm) with TiO_2 deposited by PMCS and with evaporated Platinum dish-patterns as top electrode (∼0.35 mm of diameter).

Wafer Two

– A layer of Pt (60nm) was deposited by sputtering

The second phase involves the use of Spin-coating to deposit the metal oxide on the bottom electrode in a clean room environment. Spin-coating of Sol-Gel synthesis of TiO_2 has tha advantage of producing thin films with a good level of reproducibility in terms of thickness and homogeneity.

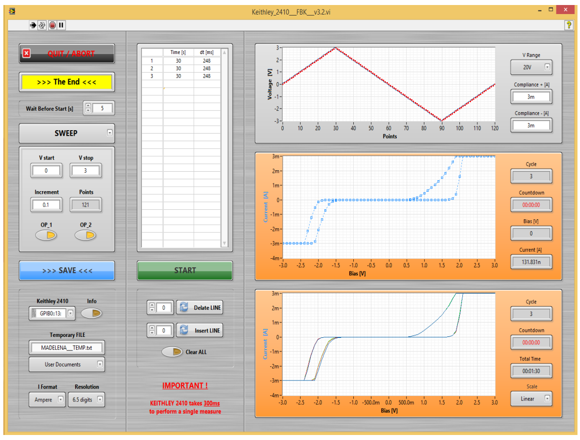
The third fabrication phase is the deposition of the top electrode. In order to achieve a good working device, proper interface between Pt/Ti metal layer and TiO_2 metal oxide must be guaranteed for the memristor reproducibility. In this deposition process, Al metal layer which did not require any adhesion primer to oxides, as well as having an advantage of reducing the dimensions (area) of the top electrode thus reducing delamination, was used as as top electrode. The deposition of this layer of metal (Al) was performed by electron beam technique and patterned using a shadow mask technique (Figure 2.6a). In the second option shown in Figure 2.6b, the deposition of the top layer of metal (Pt) was performed by standard electron beam technique and patterned using a shadow mask with smaller dish-shaped patterns to avoid cracks due to intrinsic stress of Platinum deposited by this technique.

Similar approach has also been adopted for samples where the metal oxides have been deposited by Pulsed Microplasma Cluster Source (PMCS) technique (Figure 2.7);

2.5.2 Experimental Setup and Device Characterisation

The **measurement setup** used for the characterisation of the fabricated memristive devices is given in Figure 2.8a. The measurement setup is composed of a Faraday cage, stereoscopic microscope, micro-manipulators with Tungsten probes and Platinum wire, a 2410 High Voltage Source Meter (Keithley) connected to a PC through a GPIB card and

(a)



Wait Before Start [s]: 5
V start: 0
V stop: 3
Increment: 0.1
Points: 121
OP_1
OP_2
Keithley 2410: GPIB0::13:
Info
Temporary FILE: MADELENA__TEMP.txt
User Documents
I Format: Ampere
Resolution: 6.5 digits

	Time [s]	dt [ms]
1	30	248
2	30	248
3	30	248

IMPORTANT !
KEITHLEY 2410 takes 300ms to perform a single measure

V Range: 20V
Compliance + [A]: 3m
Compliance - [A]: 3m
Cycle: 3
Countdown: 00:00:00
Bias [V]: 0
Current [A]: 131.831n
Cycle: 3
Countdown: 00:00:00
Total Time: 00:01:30
Scale: Linear

(b)

FIGURE 2.8: Inorganic memristor measurement and data acquisition setup. **(a)**, Measurement setup composed by a Faraday cage, stereoscopic microscope, micro-manipulators with tungsten tips or Platinum wire, 2410 Keithley High Voltage Source Meter. **(b)**, LabVIEW data acquisition control panel of program for inorganic memristive device testing

controlled by a software user interface (LabVIEW, National Instruments). A preliminary characterisation was carried out with the application of a controlled voltage-current waveform on several structures such as Pt/TiO_2 (Sol-gel)/Pt wire, Pt/TiO_2 (PMCS)/Pt wire, Pt/TiO_2 (Sol-gel)/Aluminium dishes to determine their resistive switching effect. These devices under test were realised on fused silica quartz and having a bottom electrode on Pt/Ti where Titanium has been used as adhesion. The TiO_2 layer has a thickness of around 50nm in each case and is diced in 2 dimension in order to allow fitting with several deposition technique (2x2 cm or 1.5x1.5 cm).

The measurement process is has been automated using LabVIEW reference tool to allow a large degree of freedom in managing time, sampling rate and applied voltage. The LabVIEW program written for the KEITHLEY 2410, a 1 channel SMU, allows the application of a voltage sweep between 2 electrodes or a constant value. In the LabVIEW data acquisition control panel shown in Figure 2.8b, different types of measurements can be carried out allowing us the flexibility to set the kind of waveform the current compliance and number of points we want. The table seen in the figure is used to plan a list of measurements that can change for the duration only. The system automatically calculates the interval of time "dt" between 2 consecutive points. Because of some limits of the instrument, is not possible generate a list of point with interval less than 300ms. The measurement setup is the what was used for the experimental results of the FBK memristors present in Chapter 4.

The **Experimental characterisation procedure** for all fabricated devices follows the below listed steps:

- Opening of the conduction channel by a process known as *electroforming* [48, 49, 50, 51, 52]. This is achieved by the applying a constant voltage (typically 4V) with a current compliance value of 1mA in order to avoid any possible failure due to the generation of extra currents through the device.

- After opening up the channel by electroforming, measurement is then taken to plot the typical Lissajous *i-v* characteristic. Measurements carried out were done with voltage sweeps ranging from -1.5V to +1.5V at different rates in order to evaluate the switching behaviour of the TiO_2 memristor. In some tests, other similar procedures have been adopted without electroforming but with more extended sweeps (typical range from -3.5V to +3.5V).

Figure 2.9 is an example of a plot showing hysteresis behaviour of the memristor for 5 consecutive sweeps. In this experiment, a series of voltage sweeps from -3.5V to 3.5V is applied to the closed channel device. The hysteresis behaviour and switching phenomenon for five consecutive sweeps are plotted on the graph conforming to symmetric PT/TiO2/Pt structures and a memristive behaviour [53, 54, 55, 56]. Electroforming steps was intentionally not performed here in order to focalise attention on the effect of the change in current hysteresis. The curves reported in the graphs are performed in the session on a single device in a single point of measure.

Even though it is not so obvious in the plot due to scaling. the first cycle shows the first switching at negative sweep, the second cycle shows the switching at positive and negative sweeps. Meanwhile, the third cycle shows non-switching at both positive and negative sweeps, this is assumed due to the short time of cycle. The fourth cycle shows switching at positive sweep but not at negative sweep. The fifth cycle show the non-switching behaviour; supporting the hypothesis of short time cycles.

2.6 Memristor Device Modelling

In this Section, the electrical model of a memristor which was developed and validated using experimental results on memristive devices fabricated within the MaDEleNA project. The model has been one of the basic building blocks with which I have been able to carry out my research activities on memristive system. The model has been adopted along the project to simulate basic memristive-base processing circuitries as well as more complex computational architectures, taking advantage from the built-in memory and resistance properties of this device. The behavioural model has been developed in CADENCE using Verilog-A and is currently available in the CAD library. The memristor was modelled from the equations governing the state-dependent Ohm's law derived from the fabricated solid state memristor by HP labs.

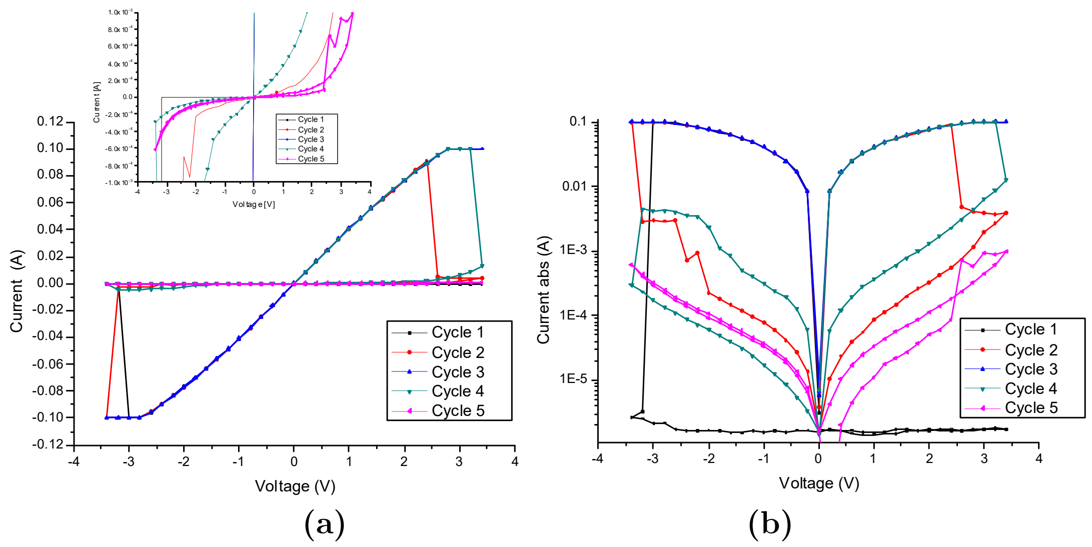


FIGURE 2.9: Experimental measurement of the FBK memristor i-v plot for five distinct cycles. **(a)**, All five cycles plotted in linear scale. A zoom of linear graph is plotted in the inset in order to highlight behaviour at low current values. **(b)**, All five cycles plotted in log scale.

2.6.1 Ion Drift Model

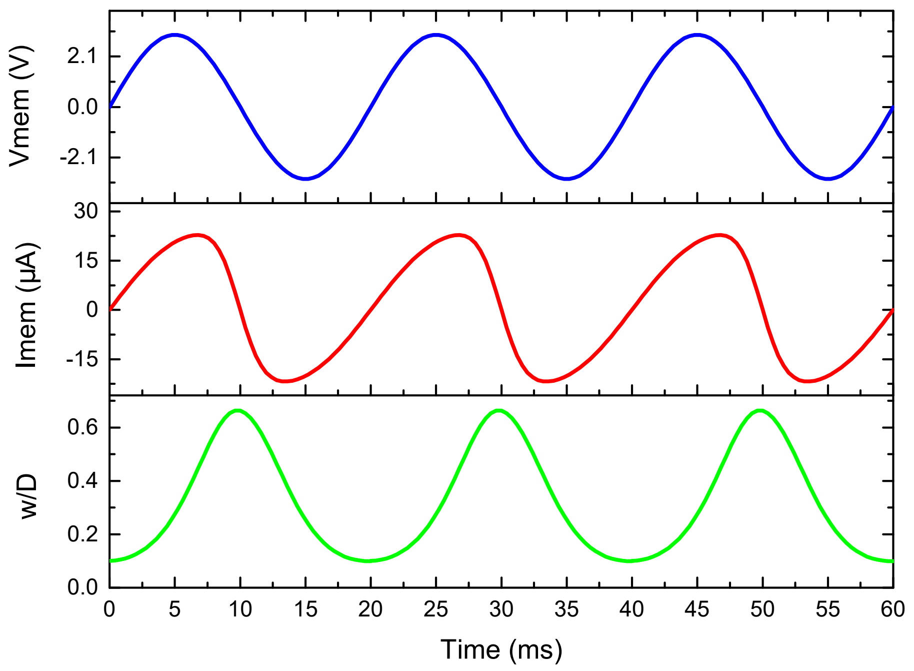
In the linear ion drift (LID) model, as described in the device structure given in Figure 2.2, the device is considered as having a physical width D containing two regions (doped and undoped), and modelled as two resistors in series (one representing the doped region and the second region representing the undoped region). The doped region forms the highly conductive region while the undoped region represents the low conductive oxide region. The variable length of the doped region, w(t), is the internal state variable such that when $w \to 0$, we have a low conductive channel and when $w \to D$, the channel is highly conductive [57].

The state-dependent current–voltage $(i - v)$ relationship is given in Equation 2.5 for a simplified case of ohmic conduction.

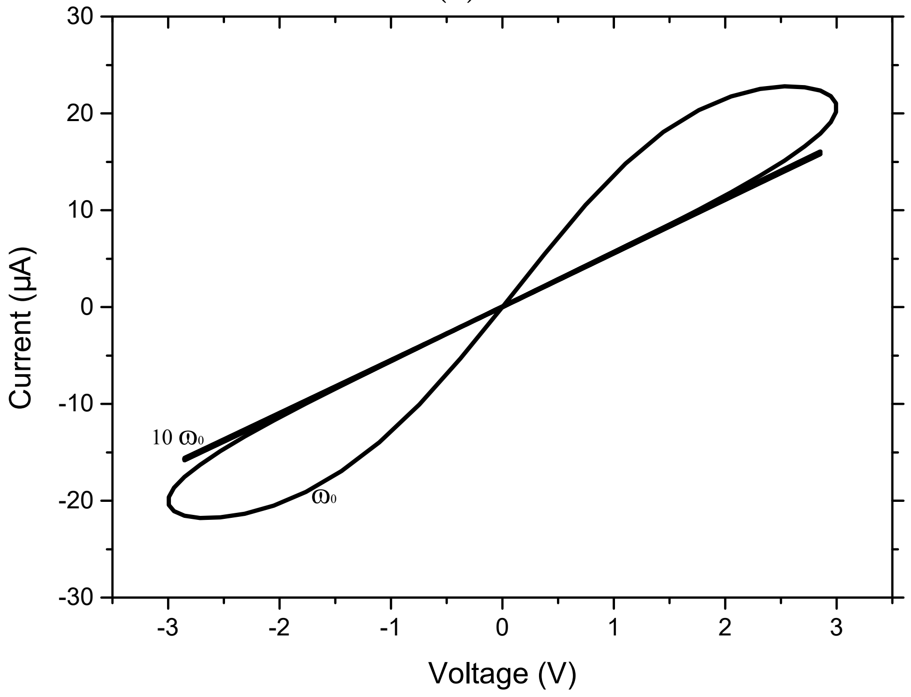
$$v(t) = \left(R_{ON}\frac{w(t)}{D} + R_{OFF}\left(1 - \frac{w(t)}{D}\right)\right)i(t), \tag{2.5}$$

$$\frac{dw(t)}{dt} = \mu_\nu \frac{R_{ON}}{D} i(t), \tag{2.6}$$

where R_{ON} is the highly conductive doped region of the semiconductor film with high concentration of dopant atoms, R_{OFF} is the highly resistive undoped region, D is the length of the device, $w(t)$, the state variable is the doping ratio and u_ν is the dopant



(a)



(b)

Figure 2.10: Simulation of the Behaviours for a voltage-driven memristive devices using our LID memristor model. (a) Applied voltage (blue) for a symmetrical input and the corresponding current (red) as a function of time. The corresponding state variable (green) is also plotted, (b) corresponding $i - v$ behaviour to the symmetrical input. The collapsed line correspond to a sweep of an order of magnitude higher. The applied voltage is $\pm v_0 \sin(\omega_0 t)$, where $\omega_0 = 2\pi f_0 = \frac{2\pi\mu_\nu}{D^2}$.

mobility. Integrating (2.6) gives the formula for $w(t)$

$$w(t) = w_0 + \mu_\nu \frac{R_{ON}}{D} \int_0^t i(t)dt = w_0 + \mu_\nu \frac{R_{ON}}{D} q(t), \tag{2.7}$$

The expression of the memristance is obtained by inserting (2.7) into (2.5).

$$M(t) = R_{OFF} \left\{ \left(1 + \frac{w_0}{D} \left(\frac{R_{ON}}{R_{OFF}} - 1 \right) \right) - \frac{u_\nu R_{ON}}{D^2} \left(1 - \frac{R_{ON}}{R_{OFF}} \right) q(t) \right\}, \tag{2.8}$$

For $R_{ON} \ll R_{OFF}$ the memristance can be expressed as

$$M(t) = R_{OFF} \left(1 - \frac{u_\nu R_{ON}}{D^2} q(t) \right). \tag{2.9}$$

The memristor behaves symmetrically when the system is within the bounded range of $M \in (R_{ON}\ , R_{OFF})$ and functions as a linear resistor when either one of its boundaries is reached. It holds its the resistance value as long as the input polarity is not reversed [58, 59].

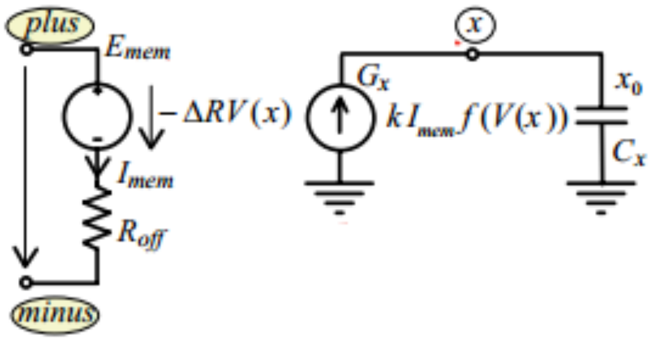
Fig. 2.10 is a plot of some fundamental curves relating to the characteristics of memristors. These interesting behaviours of memristors form the fundamental finger prints for identifying memristive devices. Fig. 2.10a is a plot of the applied voltage and the resulting current versus time t. Also plotted in Fig. 2.10b, is the corresponding $i - v$ characteristics. As observed in the plot, the hysteresis is pronounced for $\omega \leq \omega_0$ and shrunk when $\omega \gg \omega_0$. According to Chua, the fundamental finger prints for identifying memristive devices were drawn from these important characteristics of the memristor.

Modelling organic memristor starting from known inorganic devices

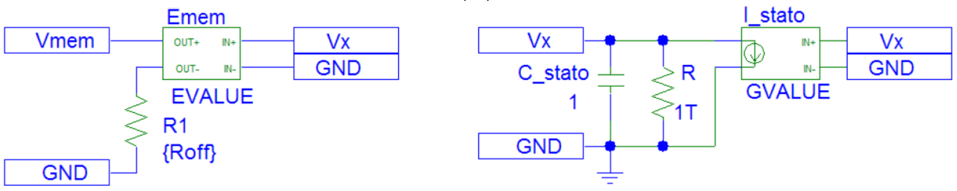
Due to the fact that the organic memristor shares the same common fingerprint of the inorganic memristor, as well as the generalized memristor theorized by Chua in 1971, it is possible to behaviourally model the organic memristor starting from existing modelling equation of the inorganic memristor. The organic memristor shares the common fingerprint of the generalized memristor because it passes the three experimental tests for memristors postulated by Chua [60], which are:

(i) The Lissajous figure in the voltage-current plane is a pinched hysteresis loop when driven by any bipolar periodic voltage v(t), or current i(t), and under any initial conditions;

(ii) The area of each lobe of the pinched hysteresis loop shrinks as the frequency of the forcing signal increases;



(a)



(b)

FIGURE 2.11: Organic Memristor modelling starting from inorganic devices. **(a)**, Structure of SPICE model for inorganic memristive devices. **(b)**, P-Spice model for the organic memristor.

(iii) As the frequency tends to infinity, the pinched hysteresis loop degenerates to a straight line through the origin, whose slope depends on the amplitude and shape of the forcing signal.

The implemented p-Spice electrical model of the organic memristor from previous work shown in Figure 2.11 is an example of the organic memristor modelled starting from inorganic devices.

The relationship between the memristor's voltage and current is modelled with a voltage-controlled voltage source and a current-controlled voltage source. This Spice model of the inorganic memristor was first introduced by [59]. The schematic in Figure 2.11 is an adapted electrical model of the organic memristor. The model presents some simulation limitations in stability, speed and can only mimic a specific implementation. As a result of this we decided to use more powerful tool, Verilog-A; which is more stable than Spice models, very fast and uses the fundamental charge equations rather than trying to mimic a specific implementation.

Why Verilog-A?

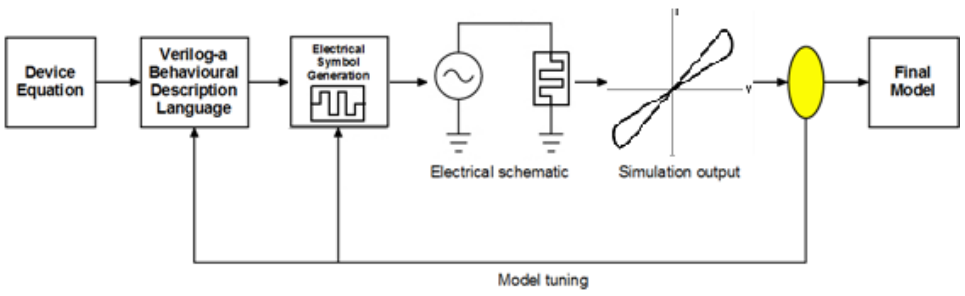
The Verilog-A language is a high-level language that uses modules to describe the structure and behaviour of analogue systems and their components. With the analogue statements of Verilog-A, it is possible to describe a wide range of conservative systems and signal-flow systems, such as electrical, mechanical, fluid dynamic, and thermodynamic systems [61]. Spice: physical model needs to be translated into an electrical one Verilog-A: model described through equations disregarding the physical nature. The advantages of using Verilog-A instead of Spice is that Verilog-A provides a simple, efficient, more concise and clear language of describing analogue behaviour in simulators, simulation run time much shorter and the implementation in CADENCE which is a standard CAD software for IC design give the model the advantage of being integrated in a more complex network with other electrical components. To specify the behaviour of individual modules, you define mathematical relationships among their input and output signals. After defining the structure and behaviour of a system, the simulator derives a descriptive set of equations from the netlist and modules. The simulator then solves the set of equations to obtain the system response.

The block diagram of Figure 2.12 describes the model of the organic memristor starting from the device equation to the Verilog-A and then the final model by tuning certain parameters. The set of equations governing the memristor functionality from the input terminal to the output terminal are described in Verilog-A i.e., the structure and the behaviour of its components.

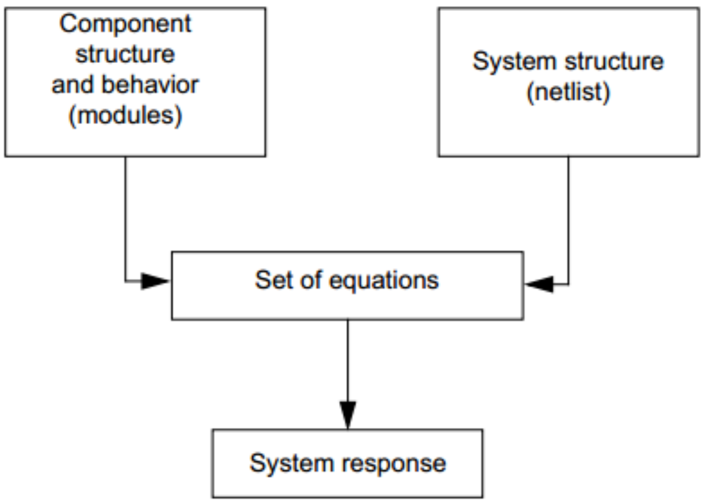
A circuit symbol of the memristor is then generated, which can be used in a more complex network in any schematic with conventional electronic components. On the first instance, during the device modelling, the model can be tuned to fit the organic memristor characteristics either by adjusting the equation governing the current-voltage relationship or by adjusting certain predefined editable parameters of the memristor. After the desired characteristics is obtained and the final model is frozen and the only parameters that can be edited are those defined in the symbol generation. Table 2.1 summarises the tunable model parameters.

Even though it has been argued that the pinched hysteresis loop is not a circuit model because both the shape and the area enclosed by the hysteresis lobes change with the input signal, and therefore cannot be used to predict the solution waveforms when the device is embedded as part of an electronic circuit [62, 63]. It is important to note that understanding how the memristor reacts when driven by a voltage can nevertheless help us understand some certain behaviours of the memristor as seen from simulation results of a model of the organic memristor fitted from experimental data, implementing logic with memory.

Figure 2.13a shows the model-to-hardware correlation fit for the organic memristive device, the black line is the measured data while the red line is the simulated data. After fitting the model with the experimental results, we validated it by simulating an AND gate which has been implemented in [35]. The schematics of the Memorised-AND (MAND) function is shown in the inset of Figure 2.13b. Even though the simulation



(a)

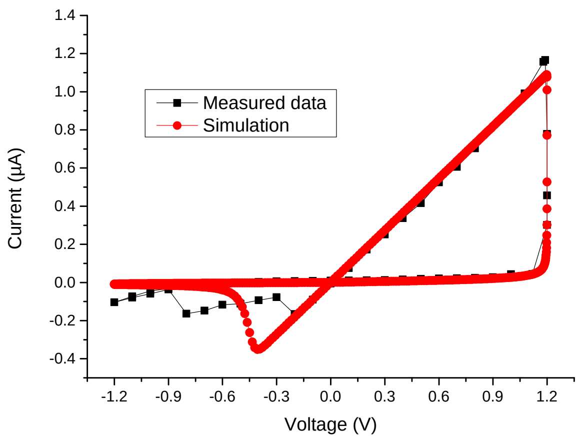

Component structure and behavior (modules)
System structure (netlist)
Set of equations
System response

(b)

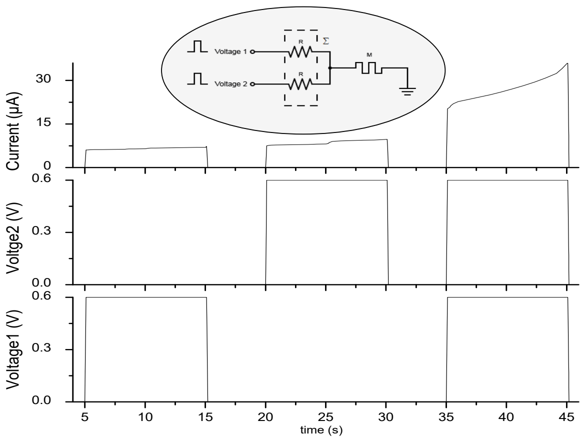
FIGURE 2.12: Organic Memristor modelling starting from inorganic devices. **(a)**, Organic memristor model implementation. **(b)**, Verilog-A operational flowchart.

Table 2.1: Model Parameter Definition

Parameters	
Name	**Description**
$init_{state}$	Initial state of the memristor (w/D) $\in$ [0:1]
R_{ON}	Memristor ON state resistance
R_{OFF}	Memristor OFF state resistance
V_{th1}	Positive bias threshold voltage
V_{th2}	Negative bias threshold voltage
u_v	Linear ion mobility
D	Film thickness (physical width of memristor)
t	Simulation time step
window_coef	Value of the coefficient in the window function
window_type	The type of window function required



(a)



(b)

FIGURE 2.13: Organic Memristor modelling starting from inorganic devices. **(a)**, Organic memristor hardware and model fitting with i-v curve. **(b)**, Temporal dependence of the output of the MAND function.

result does not accurately fit the experimental result due to certain characteristics of the device which must be considered such as the stability of the device, early ageing of the device. It is also worth noting that device characteristics varies from one device to another and measurements carried out on the same device varies as well. The model, nevertheless, sufficiently describes the behaviour of the organic memristor for the realisation of the logic AND gate.

The operation of the MAND circuit is to check the status of the output current for different logic configurations of two inputs. When both inputs are activated, a gradual increase of the output signal can be observed, while the output remains constant when only one input is ON. The value of the output signal depends on the duration of the simultaneously applied input. Thus, the implementation of a logic AND gate.

2.7 Conclusion

This chapter introduces a review of the fundamental properties of the memristor starting from the basic mathematical equations governing memristive systems. Memristor device theory, fabrication and characterisation of both organic and inorganic memristor within the framework of the MaDEleNA project were discussed. The properties of both organic and inorganic memristors were verified with in the characterisation experiments. In the experiments of the organic devices, it was a bit difficult to replicate the kind of waveform obtained by the researchers from the University of Parma principally because of the device to device variability and or ageing of the device. Nevertheless, the fundamental principles identifying the memristors, switching and non-volatile memory among others were observed. In the organic devices, several promising characteristics were observed. One promising properties of the organic memristor based on its ability to mimic biological systems, is that it could act as a basic building block for complex adaptive electrical systems. Such systems based on basic cognitive processes of biological systems, serve as test-benches for different algorithms and learning theories.

The last section of this Chapter was devoted memristor device modelling. A simple an accurate behavioural model was developed and placed in our CAD library which is being used for memristive systems simulation. The developed LID model is based on the mathematical equations describing the TiO_2 memristor. Thanks to the advantage of memristive devices share the same common fingerprint as the generalised memristor theorised by Chua in 1971, it is possible to behaviourally model a generalised memristive system (in our case the organic memristor) starting from existing modelling equation of the TiO_2 -based model. The memristor is modelled as a thin film of two variable resistors in series where the resistance of each is dependent on the doping ratio w(t).

Based on measurement and experimental results, the memristor model can effectively be tuned by acting on certain physical parameters contained in the model to fit experimental data. An example of model to hardware fitting was also presented in this Chapter (TiO_2 -based model was easily adapted with some additional parameters to fit organic memristors). The construction of an accurate and compact model for memristive devices is necessary for the design and modelling of complex adaptive systems. Our goal was to design a behavioural model which approximates the measurements of the memristor carried out in our labs, which in turn is used to simulate memristive systems. All the simulations carried out in this thesis were done with the model described in this Chapter except otherwise mentioned. The model has been implemented using Verilog-A, a high-level language that uses modules to describe the structure and behaviour of analogue systems and their components. A behavioural model was chosen in part due to the advantages of using Verilog-A over Spice, moreover the former gives us the ability to integrate our model to our CAD tool.

Chapter 3

Memristor Emulator

3.1 Introduction

The theory of the fourth fundamental circuit element however remained a puzzle for decades as there was no physical implementation to back it up. As some scholars have argued, this was due in part to the existing technology and also the memristor being a nano-scale device, even though for decades, memristive behaviours have been unwittingly observed [23, 24, 25, 26, 27]. It was therefore difficult to for the memristors to be identified until advancements were made in nanotechnology. The memristor as earlier defined, can be considered as a time varying resistor whose resistance depends on the history of current that has passed through it [1, 28]. Its resistance changes as a result of the summation of current that has passed through the device. When the current flowing through the device is zero, the summation of current becomes constant, and thus the resistance remains unchanged, making the memristor act as a non-volatile memory component [29].

After the realisation of a solid state memristor, there has been a lot of interest in exploiting this emerging circuit element especially in their applications in analogue circuitry, particularly exploiting their non-volatile memory. This is in revenge, due to their fine resolution programmability An examples of memristor application in programmable analogue ICs is in demonstrated in [64] where a memristor is designed for a pulse-programmable mid-band differential gain amplifier, the total output resistance is made programmable by use of a memristor. Another practical example of memristor application is in a programmable threshold comparator [16], here, a technique is used to control the charge integrated in the memristor thereby implementing a programmable hysteresis. Recently, chances of Memristor-based Vision Processors have also been investigated. In [17], a light-to-resistance encoder exploiting the properties of a memristor is presented whereas other memristor-based image processing algorithms are continually being exploited [65, 19, 18].

In order to investigate the memristor device properties in programmable analogue electronics, researchers have so far relied on SPICE simulation models of memristive devices for numerical simulation. Several SPICE macromodels based on the HP memristor equations have also been published to emulate the behaviours of memristors [59, 66, 67, 68, 69, 70, 71, 72, 73, 74]. Although SPICE macromodels are important for the simulation of these characteristical properties of memristors, they posses certain limitation in terms of the accuracy needed to simulate devices of such dynamical properties, which also have to be improved. Moreover in order to implement practical electronic circuits based on memristors, SPICE models cannot be used. It is therefore, pertinent to have emulators which could be use for real-world application which are practical.

Most of the emulators which have been published rely on discrete electrical components which are either complicated to build or are resource hungry. In this thesis, I propose an easy way of mimicking the properties of memristors with an Arduino-controlled digital potentiometer. This approach takes advantage of the Arduino's capability to communicate with other components through its SPI ports and most especially it can be easily programmed and reconfigured. Since the whole idea is to simplify things, data acquisition is done by a PC interfaced with the Arduino through the serial port. This way data acquired from the Arduino is plotted in real-time using some sort of serial monitor. The entire process is further explained in subsequent sections addressing the memristor emulator implementation. It is worth the mention that the Arduino is an open source development board with a microcontroller.

This chapter consists of five sections which are organised as follows. Section 3.2 gives an overview of the state of the art of memristor emulators and architectures. In Section 3.3, the proposed memristor emulator architecture based on this work alongside its hardware implementation are addressed in details. Experimental setup and results are presented in Section 3.3.2. And finally in Section 3.4, conclusions and some discussion related to the technical implementation of the emulation are presented.

3.2 An Overview of Memristor Emulators

Due to among others certain factors like cost and the difficulties involved fabricating nano-devices, reliable commercial memristors are not readily available in order to validate the principle of memristor in analogue signal processing. Several circuit models for numerical simulations are already being employed an in order to study memristors and it's applications, researchers rely solely on them. Therefore, there is a need to implement electronic circuits which are able to emulate the properties of the memristor. Several memristor emulators have already been proposed relying on voltage and current-controlled models. In [75], the first attempt of a CMOS-based memristor emulator was realised employing Differential Difference Current Conveyor (DDCC-based) circuit blocks. The proposed technique was implemented with four differential difference current conveyor (DDCC) as integrator, squarer, multiplier and summer. Hussein *et al.* later proposed a simpler version of a MOS-based emulator using only voltage controlled resistor (VCR) and a second generation current conveyor (CCII) which has the advantage of being less complex than its predecessor and has a higher functionality to work in a wider frequency range [76]. A flux-controlled memristor emulator circuit has also been proposed. The emulator circuit realised by operational transconductance amplifiers (OTAs) and CCIIs, consists of three operational transconductance amplifiers, four second generation current conveyors, six resistors and one capacitor elements. The authors claimed that the frequency dependent pinched hysteresis loop in the current versus voltage plane could up to 5 kHz [77]. Other examples of a memristor emulator based on a current-controlled mode have also been presented [78, 79].

Kim *et al.* also presented a memristor emulator made from off-the-shelf solid state electronic components which suffers the setback of being resource hungry due to its complexity. Moreover, the circuit can only be programmed in one direction (incremental or decremental) [80]. Pershin *et al.* presented a memristor emulator using digital potentiometer and a micro-controller [81]. This work is slightly similar to Pershin's implementation and can easily be used in a programmable analogue circuits to be interfaced with other electronics, a feature in which other previous implementation suffers from. It has some drawbacks due to resolution of digital signal and digital potentiometer. These issues will be further discussed in Section 7.

3.3 Memristor Emulator Architecture

The architecture implemented for the memristor emulator is based on a micro-controller and a digital potentiometer (here referred to as DigPot). The functional diagram of the proposed architecture is given in Figure. 3.1. The schematic of the memristor emulator consists of an Arduino due, and an MCP4251 digital potentiometer. The MCP4251 DigPot has an 8-bit dual potentiometer configuration. It is possible to exploit both the resistor networks ($P0$) and ($P1$) as seen in functional block diagram. Floating the terminals POA POB, configures the device as a rheostat *(variable resistor)* [12]. Each potentiometer of the MCP4251 has an 8–bit resolution resistor network which allows connection from 0 to full scale. Zero scale typically being the wiper (W) resistance value which forms part of the resistor network as seen in Figure. 3.2. Analogue pins $A0$, $A1$, $A2$ and $A3$ of the arduino are used to read input voltages are connected to the MCP4251 pins ($P0B$), ($P0W$), ($P1B$) and ($P1W$) respectively. Communication between the Arduino and the digital potentiometer is established through the SPI interface, whereas the Arduino communicates with the PC through the serial ports. Equations of the state

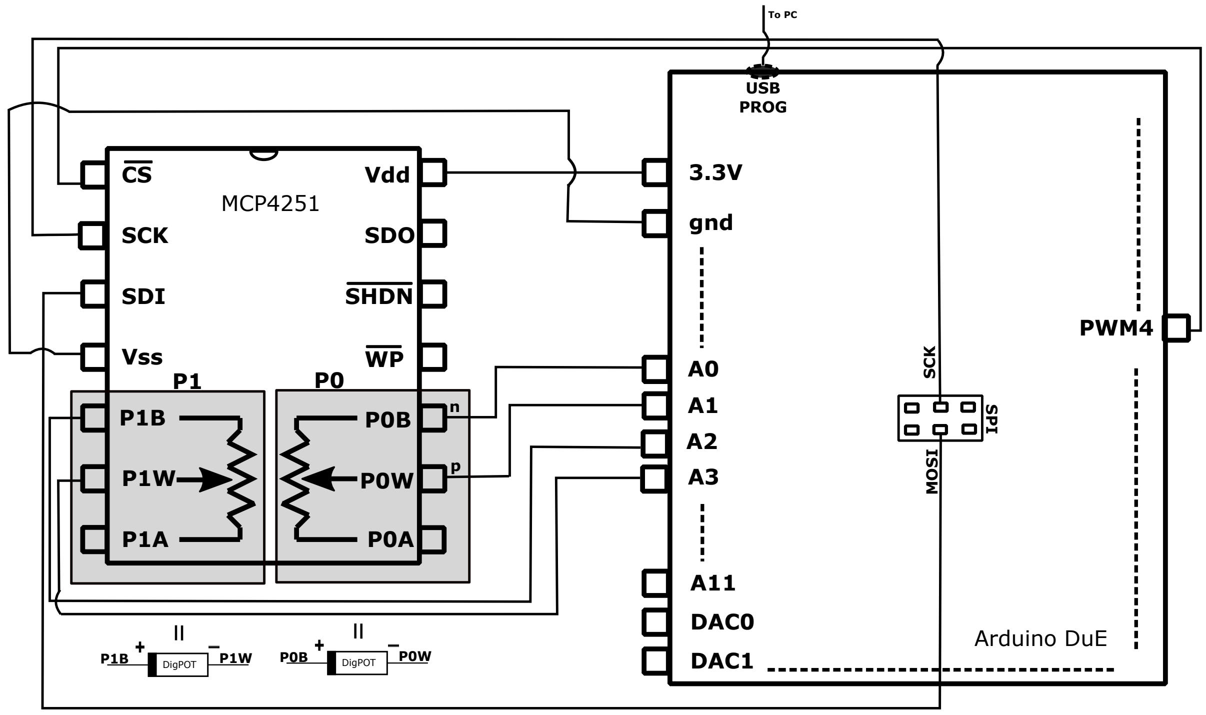


FIGURE 3.1: Pinout Schematic of the Arduino-controlled memristor emulator.

dependent current-voltage $(i - v)$ relationship of the memristor which were presented in the previous section have been implemented on the Arduino and executed with respect to the flow diagram given in Figure. 3.3b. The digital potentiometer has its default wiper position (W) in the midscale of the resistor network, therefore, we initialise to the our emulator initial position $(Rinit)$ during device power-up. The arduino analogue pins continually samples the voltages across the emulator which is used to calculate the potentiometer's wiper position $P0W$ which is in turn updated through the SPI port. Communication via the SPI port is attained by the following definition

3.3.1 Memristor Emulator Implementation

The memristor emulator has been implemented in such a way that the input resistance is calculated as a function of the applied current or voltage to satisfy the equations of the memristor given in Equations 2.5. The working principle of the emulator is simple and self explanatory. Figure. 3.3 is a clearer version of the functional block diagram of the emulator together with the operational flow starting from initialisation. The $P0W$ terminal and the Wiper terminal $P0B$ of the digital potentiometer serve as the external connections of the memristor emulator. The $P0A$ terminal is left floating, this way the DigPot acts as a rheostat. The analogue-to-digital converter (ADC) of the Arduino continuously samples the voltage drop across the emulator which in turn is used to

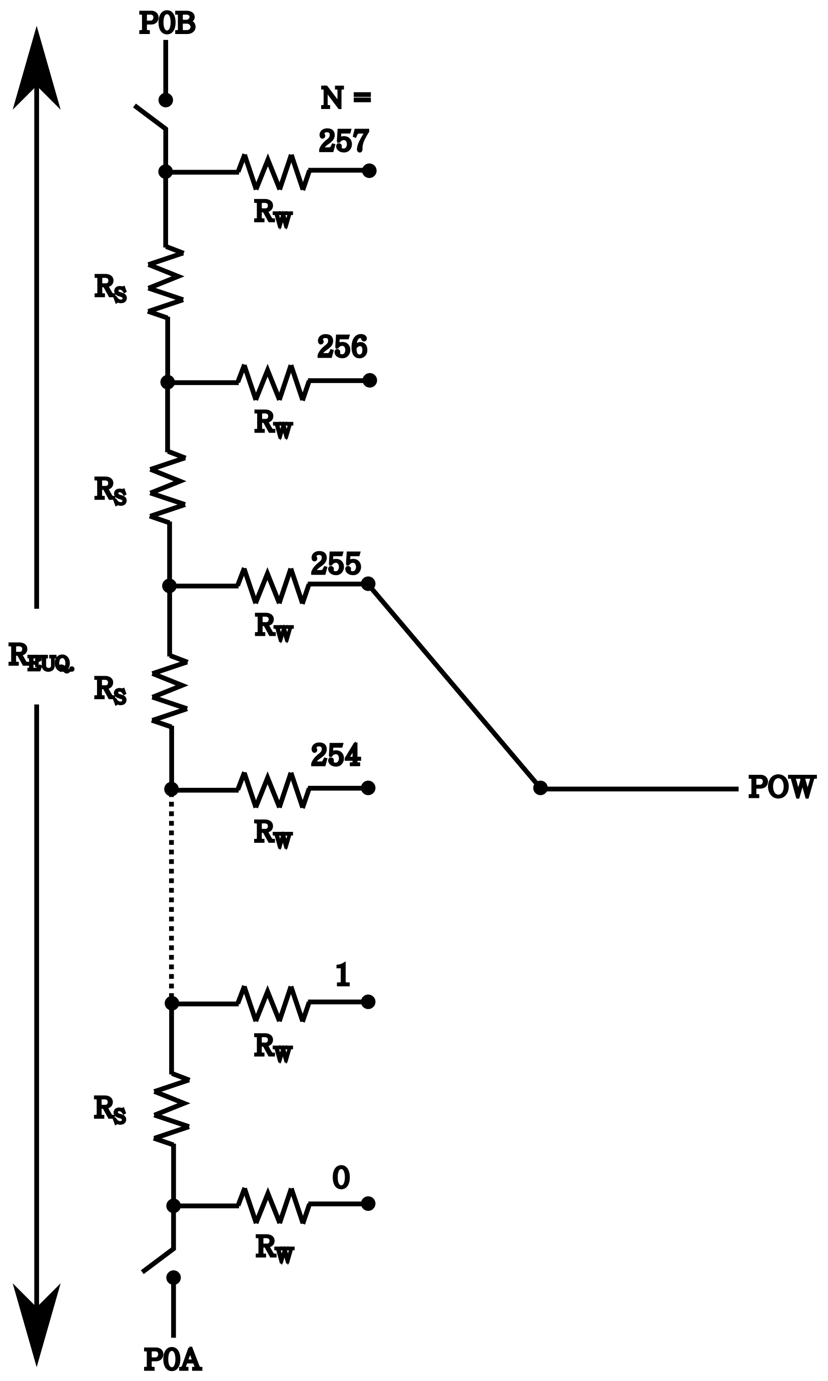


FIGURE 3.2: DigPot 8-bit resistor Network consisting of an analogue multiplexer allowing zero scale to full scale connections. The network which consists of–a resistor ladder having a series of equal value resistors $R_s = \frac{R_{EQU.}}{256}$ and a wiper terminal of equivalent resistor R_W. $R_{WB} = \frac{R_{EQU.}N}{256} + R_{EQU.}$ is the equivalent resistance for an 8-bit programming for $N = 0$ to 256

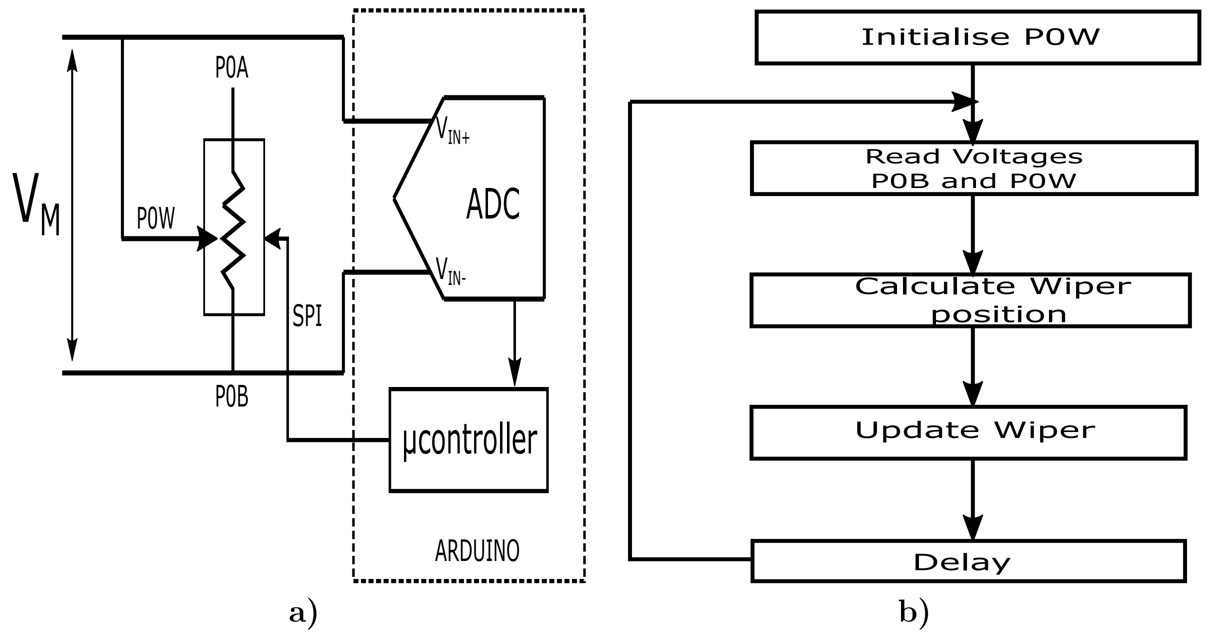


FIGURE 3.3: Memristor Emulator Circuit. (a) The arduino-controlled Emulator, (b) algorithm of the arduino-controlled Emulator

determine the emulator's equivalent resistance based on the mathematical equations governing the functionality of the memristor. Communication between the DigPot and the Arduino is attained through Serial Peripheral Interface (SPI) bus meanwhile the emulator can be easily connected to other electronic devices with its eternal connectors. Communication via the SPI port between the two blocks that make up the emulator is attained by the following definition predetermined from the datasheet.

CODE 3.1: Command Definition

```
#define INCREASE_W0  4   //INCREMENT WIPER W0 by one step
#define INCREASE_W1  20  //INCREMENT WIPER W1 by one step
#define DECREASE_W0  8   //DECREMENT WIPER W0 by one step
#define DECREASE_W1  24  //DECREMENT WIPER W1 by one step
#define WRITE_W0 0       //WRITE W0 command: Modify wiper position
#define WRITE_W1 16      //WRITE W01command: Modify wiper position
#define init_W0 230      //Initialise W0 command
#define init_W1 0        //Initialise W1 command
define SLAVESELECT 4     //SS Pin for the DigPot
```

Meanwhile the function given below receives the defined command as argument and transfers the operation to be carried out to the DigPot.

CODE 3.2: SPI Function

```
void digitalPotWrite(int command, int value){
//send in the command and wiper address via SPI:
SPI.transfer(SLAVESELECT, command, SPI_CONTINUE);
```

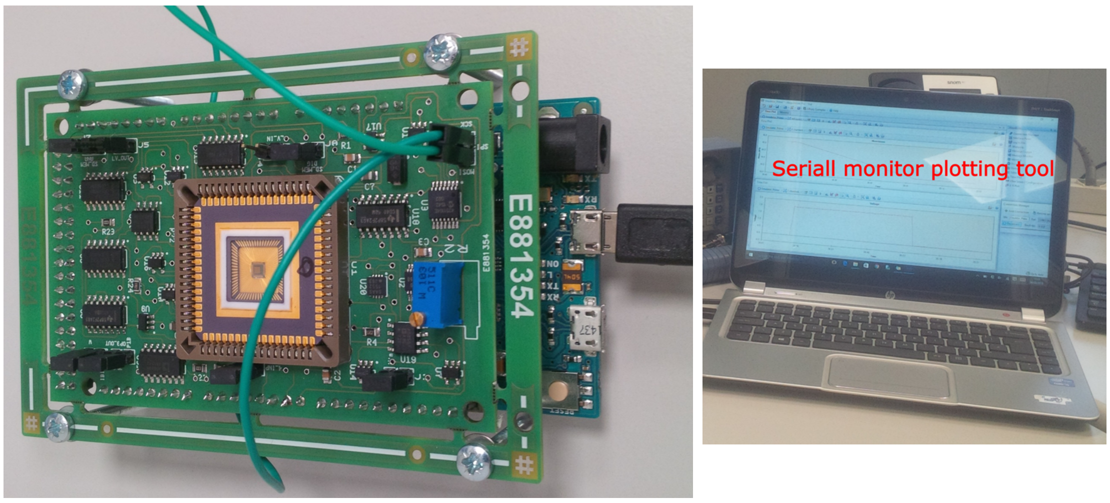


Figure 3.4: Memristor Emulator Experimental Setup

```
SPI.transfer(SLAVESELECT, value);}
```

A summary of the characteristics of the emulator and its comparison with the actual physical device is presented is Table 3.1.

3.3.2 Experimental Setup & Results

The setup used to validate the memristor emulator is given in Figure 3.4. The setup consists of Arduino stacked to a PCB implementation fo the digital potentiometer. On the PCB, there is a power supply system for the emulator an a programmable waveform generator (AD9833) to generate the sin waveform abd programming pulses. Both stacked board are connected to a computer used to visualise the graphs in real-time.

The entire validation setup is intended to be simple and easily replicable, therefore, in order to validate the emulator, we exploit the Arduino's serial communication capability through the USB ports. As shown in the schematic of the emulator in Figure3.1, the Arduino is connected to the computer (*PC*) and communication is established via USB ports. Data coming in through the serial port is tracked by Megunolink, a plotting tool software installed on the computer. It acts as some sort serial monitor which tracks and plots data in real-time [82]. The following two functions permits serial communication from the Arduino to the PC for real-time plot and Lissajous figure respectively.

Table 3.1: Characteristic of the Emulator and comparison with the physical device.

Parameter	**Physical device**	**Emulator**	**Description**
Resistance range	Determined by device structure	$75\Omega < R < 50k\Omega$	Digital Potentiometer Dynamic range
Discretisation of R	Continuous	257 steps	$8 - bit$ resolution
Response	Determined by device structure	Programmed function	Mathematical model
Applied voltage	Less than device structure breakdown voltage	0, $+3.3V$	Unipolar ADC
Supply voltage	–	$Vss = 0$, $Vdd = 3.3V$	Power supply

Code 3.3: Continuous Time Plot Function

```
void TimePlot(float data,
String seriesName,
String channelName){
Serial.print("{TIMEPLOT:");
Serial.print(channelName);
Serial.print("|data|");
Serial.print(seriesName);
Serial.print("|T|");
Serial.print(data, 3);
Serial.println("}");}
```

Code 3.4: Lissajous Function

```
void XYPlot(float xData,
float yData, String seriesName,
String channelName){
Serial.print("{XYPLOT:");
Serial.print(channelName);
Serial.print("|data|");
Serial.print(seriesName);
Serial.print("|");
Serial.print(xData);
Serial.print("|");
Serial.print(yData);
Serial.println("}");}
```

The Arduino has a unipolar analogue-to-digital converter (ADC) with an output swing limited to $0V - 3.3V$ (ground to the full-scale input voltage). As a result of this limitation, we're are restricted to working within this range. The input signals used in this experiment are provided with an offset of $1.65V$ (ADC's mid-swing) while the other terminal of the emulator is always tied to a reference voltage of $1.65V$ (See inset of Figure 3.4). This way, the entire dynamic range (DR) of the ADC is exploited.

The aim of the results presented in this section is to validate the fundamental fingerprints of the memristor identified by Chua [83], which was presented in Section 2.

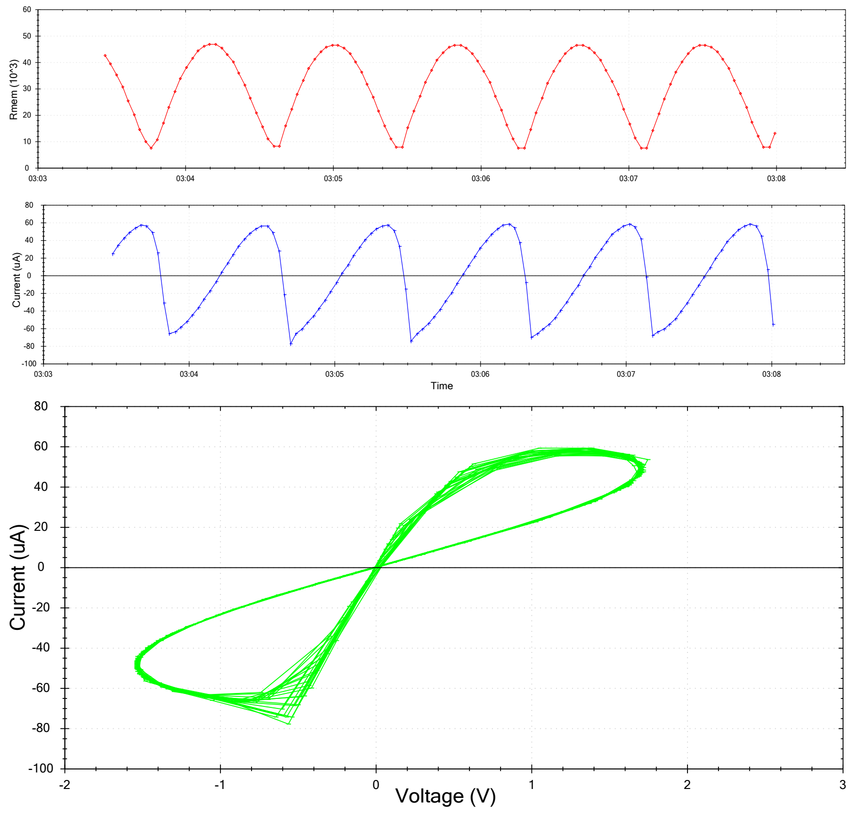


FIGURE 3.5: Memristor emulator behaviour for an input sinus waveform $\pm v_0 \sin(2\pi f t)$ with a DC offset of $1.65V$ where $v_0 = 1.65V$ and $f = 100Hz$. (a) Measured memristance and current, (b) corresponding $i - v$ plot. The negative terminal of the emulator ($P0W$) is tied to $Vref = 1.65V$ as shown in the inset of the setup in Figure 3.4

The graph in Figure 3.5 is the Lissajous behaviour for an input sine waveform $\pm v_0 \sin(2\pi f t)$ with a DC offset of $1.65V$ where $v_0 = 1.65V$ and $f = 100$Hz. From the $i - v$ plot we observe a pronounced pinched hysteresis loop which expectedly shrinks with an increase in frequency as observed in Figure 3.6a – Figure 3.6d for frequencies of 200Hz, 300Hz, 400Hz and 500Hz respectively. Thereby validating the fundamental fingerprint of a memristor. Another interesting property of memristors is their ability

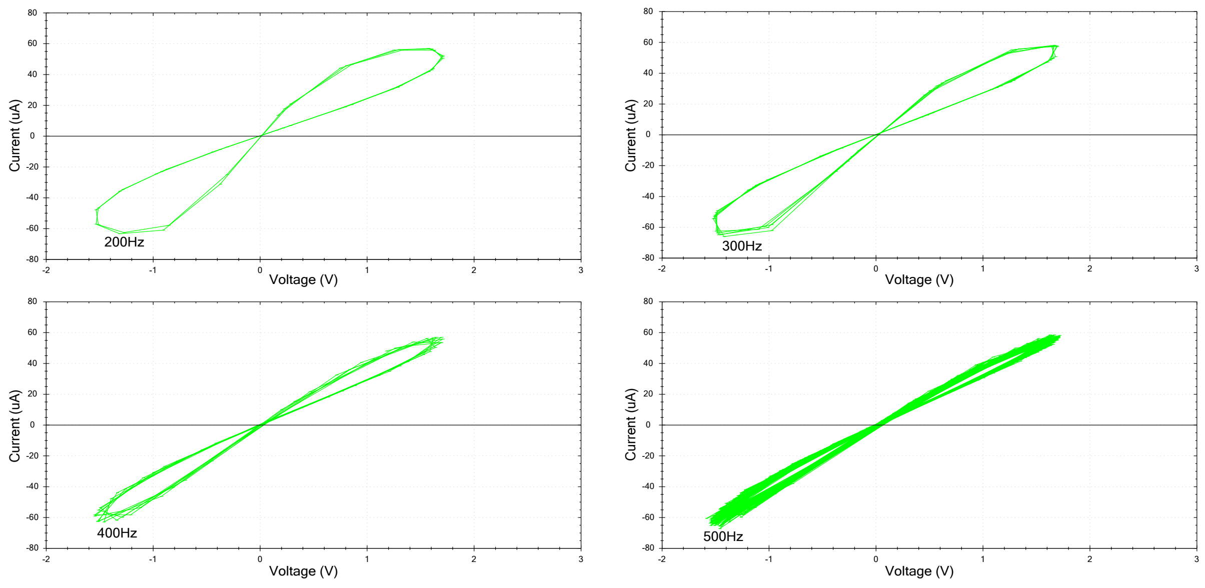


FIGURE 3.6: $i-v$ curve for different frequencies for the same input signal parameters of Figure 3.5

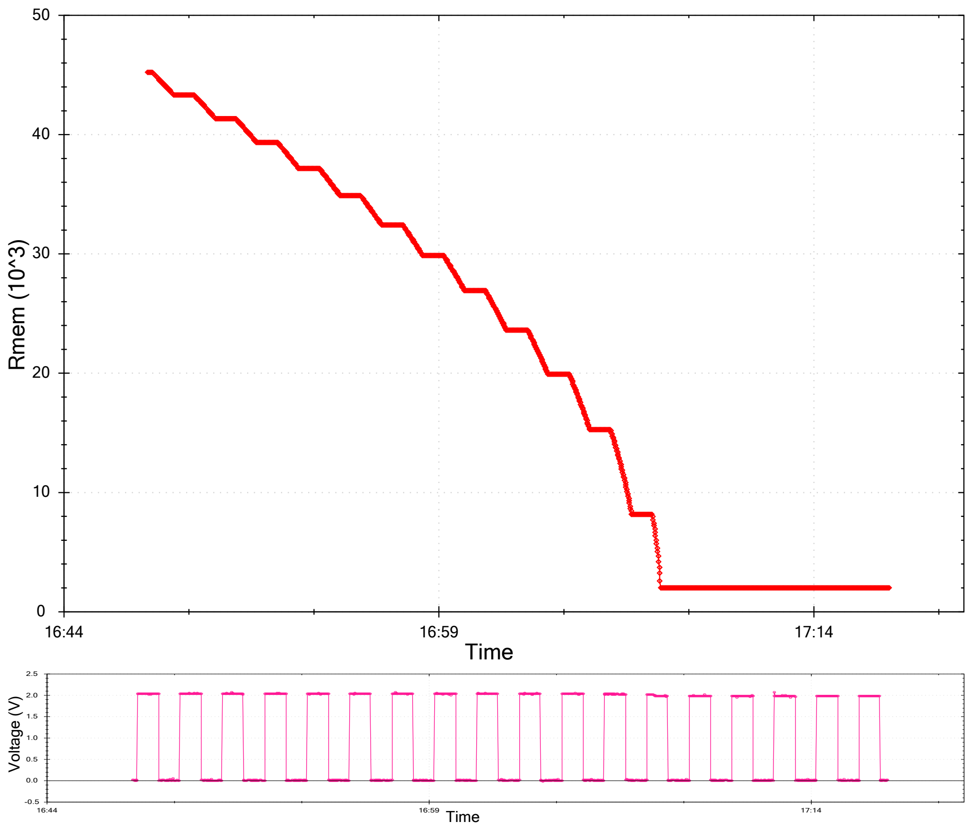


FIGURE 3.7: Behaviours of memristors for an input pulse signal. Memristance rate of change for a train of pulses

to be programmed with pulses ($voltage or current$) Given in Figure 3.7, is a plot of the memristance rate of change for an input train of pulses of a frequency of 100Hz with a 50% duty cycle and an amplitude of $2V$,. The memristor emulator was first set to a high resistance state (less conductive state) and then gradually reduced to a more conductive state R_{ON} with the application of pulses. During inter-pulse period, the memristance maintains it's state. A decrease in memristance which is proportional to the integration charges is observed. Hence, as more charges are integrated in the device with the application of pulses, step size increases.

Commenting on some features and certain limitation of the proposed memristor emulator, we'd like to discuss some issues with regards to the fitting of the memristor characteristics with regards to the digital potentiometer resolution. The fitting of the memristor characteristic is computed and approximated by the micro-controller. The Arduino Due's ADC has a resolution of 12 bits, 1MHz doesn't severely affect the fitting instead it is the DigPot's resolution of 8 bit. During analytical calculation of the mathematical equations of the memristor the Digpot's wiper position is rounded up to the nearest integer with some error margin. This coupled with the drift present in the devices characteristics over time considerably affects the i-v plot. Also, the frequency of operation in our experiment is actually limited by the experimental setup. Data acquisition was carried out through the serial port at 9600 bps. The Arduino due was chosen for its fast computing capability and the MCP4251 for its high speed read-write rate (10 MHz). A better acquisition tool like an oscilloscope would permit to run at a much higher frequency.

3.4 Discussion and Conclusion

In this chapter, the memristor emulator composed mainly of a digital potentiometer and an Arduino (micro-controller) is presented. The emulator made up of off-the-shelf electronic components is relatively easy to implement as demonstrated in this chapter. The HP memristor mathematical model was implemented on the micro-controller. Analogue pins of the Arduino were used to read the voltages at terminal of the digital potentiometer, this is thanks to it's in-built 12-bit analogue-to-digital converter on the Arduino. According to the implemented algorithm, the digital potentiometer is continuously updated through the SPI interface based on calculations of the micro-controller. The Arduino as well, communicates with a PC through the serial port in other to send data to be plotted on a serial monitor The fundamental properties of the memristor have been experimentally validated with memristor emulator. This simple and easy to make emulator could be useful to initiate students into the basic theory of memristor. The Arduino programming code is presented in the appendix.

Chapter 4

Memristor Resistance Modulation

4.1 Introduction

Rising from the interests in exploiting the properties of memristors due to their fine resolution programmability especially in their applications in programmable analogue circuitry, particularly exploiting their non-volatile resistance memory, accurately programming memristors is therefore essential [17, 18].

In programmable analogue IC design, precision variable resistors are important. Therefore, accurate and efficient methods for tuning the resistance of memristors to a desired value are necessary. Memristance programmability has to be precise and reproducible given an adequate modulation range. Nevertheless, the programming of these devices is not straight-forward and can sometimes be difficult or tricky while the device reproducibility with respect to the kind of programming technique plays an important role.

Moreover, an efficient reproduction of the state tuning of memristors in order to fully utilise them in analogue circuits, would require sufficient knowledge of the device characteristics and deal with its unpredictable switching variations. Depending on the application, an insignificant switching variation could be tolerated but it is important to note that the accuracy of a memristor-based programmable circuit would strongly depend on how accurately the memristor can be tuned.

There are several existing techniques of accurately tuning the memristor which are all based on either increasing or decreasing the memristor's conductivity to a predetermined value. The pulse-based programming technique is notwithstanding the most popular. This technique suffers from the setback of requiring external processing to accurately tune the memristor, this will be discussed in subsequent sections. An analogue programming approach which takes advantage of the dynamic modulation of the memristor under a constant DC bias exists as well [84]. Although accurate, the present approach has the drawback of not being able to program devices of high ON/OFF ratio.

The programming techniques which will be discussed in this chapter can be summarised into three main categories: (i) pulse-based programming technique which utilises pulses of distinct width and amplitude to read, write and reset a device to an intermediate state, (ii) DC programming technique and (iii) self-terminating programming method.

This chapter which consists of six sections is organised as follows. Section 4.2 gives an overview of the existing programming techniques & architectures. In Section 4.3, the pulse-based programming technique is discussed in details with some simulation and experimental results carried out on memristors fabricated at FBK. Section 4.4 touches on the DC programming technique under which we present a novel switched memristor circuit and some characterisation results under constant DC bias of the FBK memristor. In Section 4.6, we present and compare two memristor programming circuit: a digital and an analogue solution. Finally, we conclude in Section 4.7 discussing some important advantages and draw-backs of the proposed circuits.

4.2 State of the Art – An Overview of the Existing Programming Techniques

Before going on to discuss the several programming techniques within the scope of Chapter 4 with respect to the state of the art, I would like to discuss some existing memristor programming techniques and circuital configuration.

One the most popular and easiest techniques adopted today for memristor resistance modulation is the utilisation pulses of either voltage or current of distinct pulse width and amplitude for read, write and reset [85, 86, 87, 88]. In this technique, in order to read the memristor state, a pulse of sufficiently low voltage or short pulse width could be applied in order to keep the memristor's state unperturbed [89, 90]. Device reset is typically straight forward, this is achieved by applying a reverse bias voltage to device in order to reset it back to a less conductive state [91, 92, 93]. The pulse-based techniques suffers the draw-back of requiring massive external processing to accurately determine write voltage and pulse timings due to device–device non-linearities [94, 84]. It is also common to use current compliance in the laboratory instrument to program the memristor to a predetermined value. Other scientists as well have implemented a system of applying iterative write and read pulses to converge onto a desired resistance value [95, 96, 97, 98, 99, 100, 101, 102, 103].

Another method which has been published is the use of a conventional resistor as reference and force the memristor to latch onto the reference resistor. Kim *et al*. proposed a circuit comprising of an array of reference resistance [104]. They argue that their method enables the memristor to be used as multilevel memory by forcing the memristor to latch onto one of the predetermined fixed reference resistors.

In [105], Merrikh-Bayat *et al*. proposed a procedure for tuning the state of the memristor given a predetermined analogue input value. The schematic of the proposed circuit is shown in Figure 4.1a. According to the schematic, when aM is lower than aV_{in}, the output of the comparator is low, this causes the left current source to drive the memristor. This way the memristor is programmed from a high conductive state to a low conductive state. This implies an increase in Memristance. The reverse is the case when aM is higher than aV_{in}. In this case the output of the comparator is saturated to Vdd with the memristor programmed to a higher conductive state (decremental programming)

In this configuration the memristor is programmed to a corresponding V_{in} with a current source controlled by the output of the operational amplifier within which it is connected. The memristor is then further used in other circuital configuration to perform some basic arithmetic operations. The same author also proposed another memristor programming technique based on the automatic application of sequential levels of positive & negative voltages to the memristor to try and converge to a desired value with a corresponding increase in time [106]. A process referred to as sliding mode control [107]. Shin *et al.* proposed a pulse coded programming method where the memristors are programmed by input patterned waveform. The proposed circuit is given in Figure 4.1b and intended to be used in high frequency circuits such as amplifiers and filters with differential circuit topology [108]. As observed in the schematic diagram, their approach also utilises blocking capacitors to isolate DC mismatch and other mismatch effects.

Another approach to programming memristors was proposed by Pershin *et al.*. Their approach involves the use of MOS switches to drive the memristor with programming voltages relatively higher than the memristor threshold voltage. In their proposed circuit shown in Figure 4.1c, the positive terminal of the memristor is tied to ground while the negative terminal is connected to the switches that are used to program the memristor. The control signals V_{pn} & V_{pp} are used to program the memristor within it's two boundary limits with the programming voltage V_{pr}. The circuit is intended to form a part of an analogue circuitry with the negative terminal of the memristor connected to the analogue circuitry.

Berdan *et al.* an analogue memristor state tuning circuit having an AC voltage source as input signal [84]. Their circuit exploit the dynamic resistance modulation of memristors under constant DC bias.

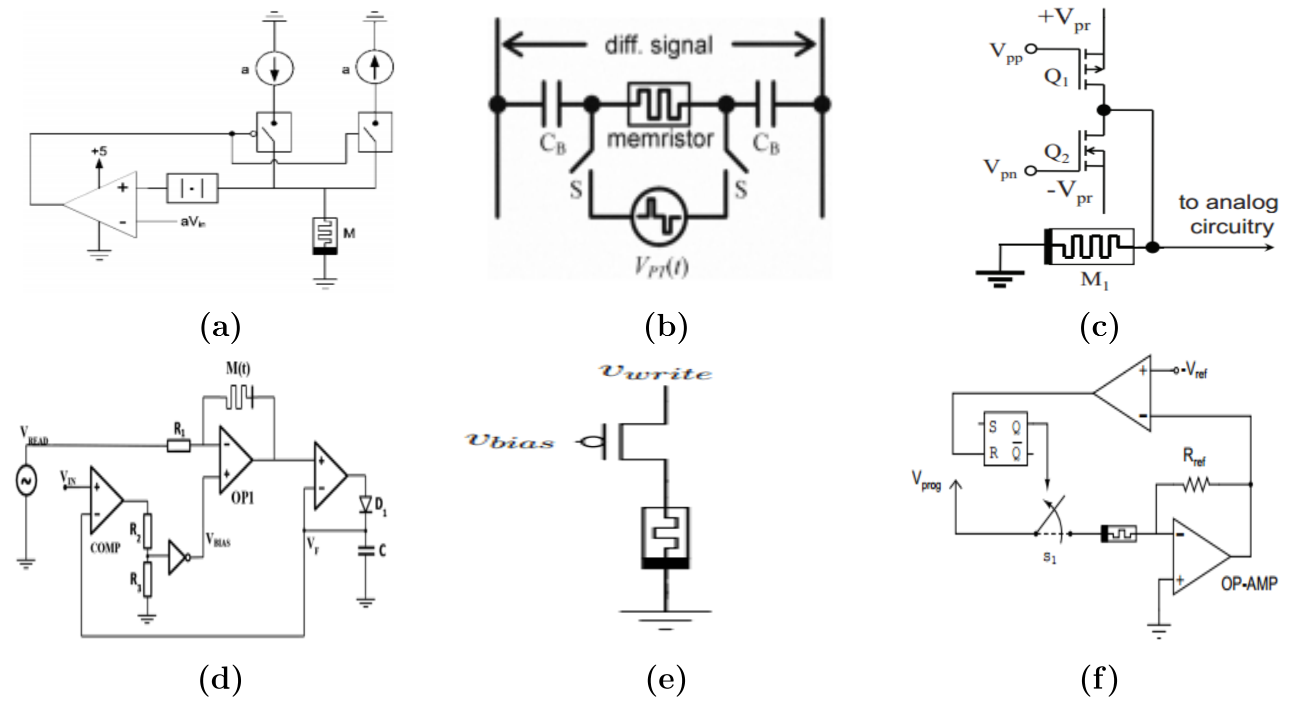


FIGURE 4.1: Examples of Existing Memristor Programming Circuit Techniques: **a)** Proposed circuit in [105] used for adjusting the memristance of the memristor with a predetermined input V_{in}. **b)** Proposed pulse coded programmable resistor in [108] having two blocking capacitors used to isolate DC mismatch. **c)** Memristor-based digital potentiometer proposed in [81] which consists of a memristor and a couple of FETs Q_1 & Q_2. **d)** Analogue programming circuit with AC voltage source proposed in [84]. See detail description of the working principle in Section 4.6. **e)** Self-terminating decremental programming circuit [109]. **f)** Continuous monitoring programming circuit [109].

The proposed circuit is given in Figure 4.1d, where the memristor is connected as feedback device for an inverting gain stage operational amplifier. The circuit employs feedback for converging the state of the memristor to a given level within a decade, given an analogue input voltage V_{IN}. The authors claim an accuracy of 8 bit precision was achieved with the HP memristor model. Details of a modified version of this circuital configuration is presented in Section 4.6.

Similar to the implementation of Pershin *et al.*, is another type of Self-terminating programming circuit proposed by Lehtonen [109]. The proposed circuit given in Figure 4.1e consists of a PMOS–transistor in series with the memristor. The PMOS acts as a current source, when biased with v_{bias} it turns on to program the memristor with a current i_{write}.
And finally, to conclude this section is another programming architecture proposed by Lehtonen which he referred to as memristor state tuning by continuous monitoring. The proposed circuit is given in Figure 4.1f. It employs an operational amplifier to set a virtual ground to one electrode of the memristor in order to convert the current through the memristor to the voltage across the resistor Rref. The output of the operational amplifier is then compared to a reference voltage which is fed into a decision block that controls a switch S_1 connected to the other terminal of the memristor in order to isolate the programming input signal V_{prog}.

The programming architectures which have been discussed in this section were chosen as a result of their similarities in programming technique to our architecture which will be presented later on in this Chapter and the next. By the end of this Chapter a table will be drawn up comparing the several programming techniques in terms of accuracy, programmability, power and resource requirements.

4.3 Memristor Pulse-Based Programming Technique

In subsequent Chapters to come, the pulse-based technique will form the basic programming method in terms of *read*, *write* and *reset* of the memristor. A large portion of this thesis has been built around this technique. Therefore, as a result of the importance of the pulse-based programming technique to this work, this Section is dedicated to presenting a detailed analysis of the entire memristor pulse programming process. Simulation results of current and voltage pulse programming techniques will be discussed with pulse timing analysis and the effect of balanced and unbalanced input signals. Experimental measurements carried out on memristors fabricated in FBK will also be presented to show the unique properties of charge accumulation and non-volatility.

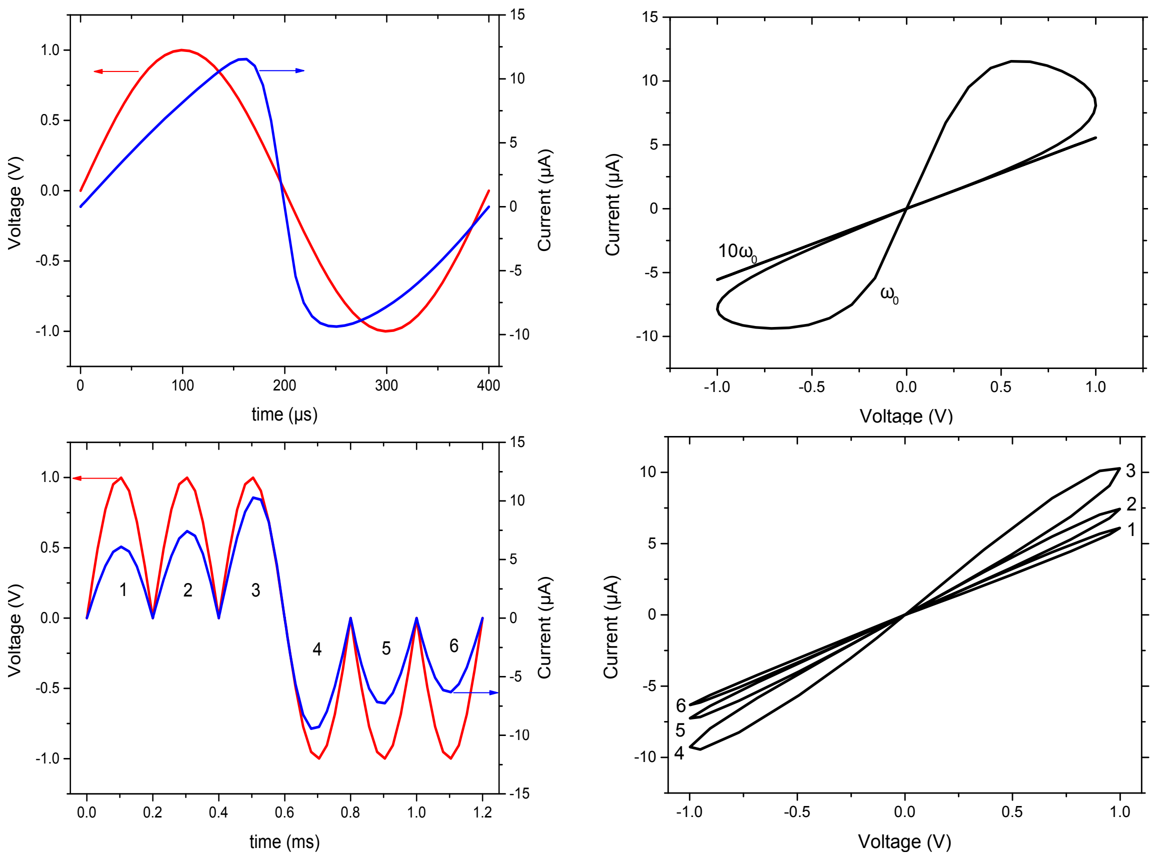


FIGURE 4.2: Behaviours for a voltage-driven memristive devices.
a, Applied voltage (red) for a symmetrical input and the corresponding current (blue) as a function of time. **b**, corresponding $i-v$ Behaviours to the symmetrical input. The collapsed line correspond to a sweep of an order of magnitude higher. **c**, Assymetrical input applied voltage (red) and resulting current (green) as a function of time. **d**, corresponding $i-v$ Behaviours to the asymmetrical input. The applied voltage in **a** is $\pm v_0 \sin(\omega_0 t)$ while the applied voltage in **c** is $\pm v_0 \sin^2(\omega_0 t)$, where $\omega_0 = 2\pi f_0 = \frac{2\pi\mu_\nu}{D^2}$.

4.3.1 Simulation of the Pulse-Based Memristor Programming

Let us recall the memristor fundamental characteristics described in Chapter 1 which we referred to as behaviours which form the fundamental finger prints for identifying memristive devices. Figure 4.2a is a plot of the applied voltage and the resulting current versus time t. Also plotted in Figure 4.2b, is the corresponding $i-v$ characteristics. As observed, the hysteresis is pronounced for $\omega \leq \omega_0$ and shrunk when $\omega \gg \omega_0$. Figure 4.2c is the plot for the behaviour of the memristor for an unbalanced input signal. In this case, we observe a gradual increase in $w(t)$ for the first three periods. This is as a result of the accumulation of the net charge over time. Applying three consecutive periods of the signal with reversed polarity reduces $w(t)$ back to the initial state. In summary, as shown in Figure 4.2b, any symmetrical alternating-current voltage bias results in double-loop $i-v$ hysteresis that collapses to a straight line for high frequencies. Moreover, for

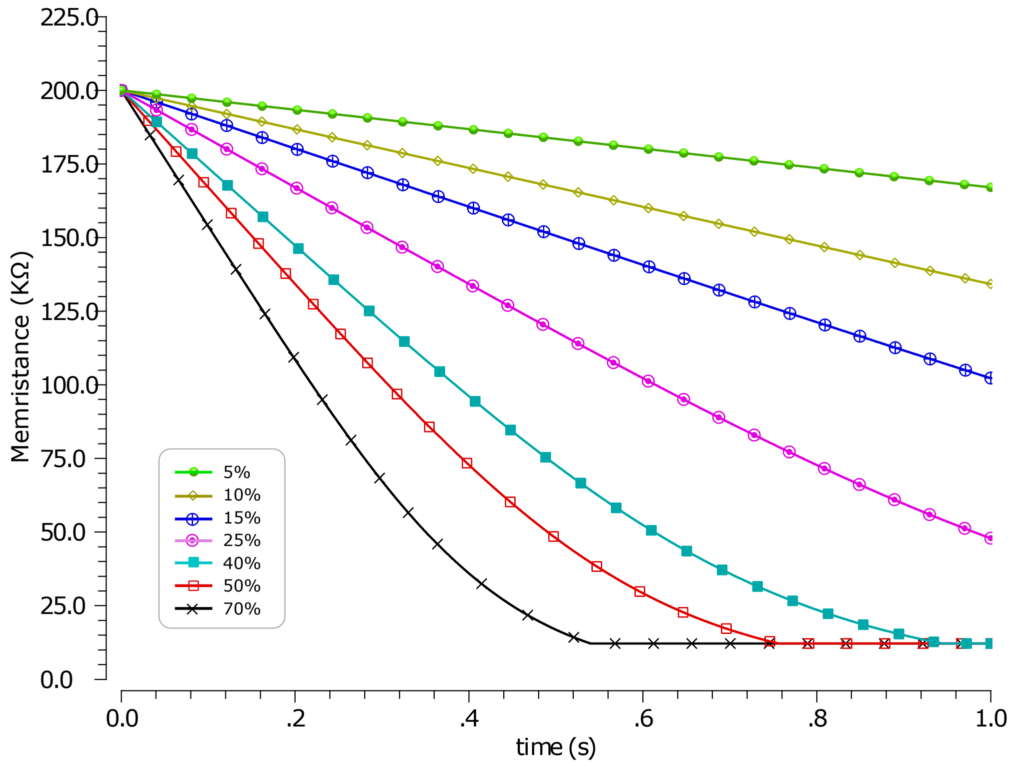


FIGURE 4.3: Memristance rate of change over time for different Duty cycles of a train of current pulses with amplitude $i_p = 160\mu A$ and programming frequency $\omega_p = 50\omega_0$.

any sort of asymmetry in the applied bias as seen in Figure 4.2d, we observed a multiple double-loop $i - v$ hysteresis which becomes more pronounced as the current increases. Therefore we can conclude that the behaviour memristors is symmetrical as long as the system is within the bounded range of $M \in (R_{ON}\ ,\ R_{OFF}\ )$. When either one of its boundaries is reached, the memristor is thought of as operating like a linear resistor holding its boundary resistance. This resistance value is held as long as the input polarity is not reversed [6, 58, 59].

Based on our assumption in Figure4.2c & Figure 4.2d, we can gradually increase the conduction of the memristor by simply applying charges of defined quantities Q. This can be achieved from pulses of distinct width and amplitude. To reverse the device's conduction, switching the polarity of the applied voltage or current is enough. Therefore a change in the memristors state can be achieved by patterning the frequency and amplitude dependent flux. Memristors typically feature an initial high-resistance state whose resistance is reduced by one polarity and increased by the opposite polarity. As observed in Figure 4.3, the application of an impulsive conduction lowers the resistance non-linearly, down to a low-resistance plateau [64]. The application of a voltage of reverse polarity can restore the initial high resistance state [110, 111]. In Figure 4.3 the

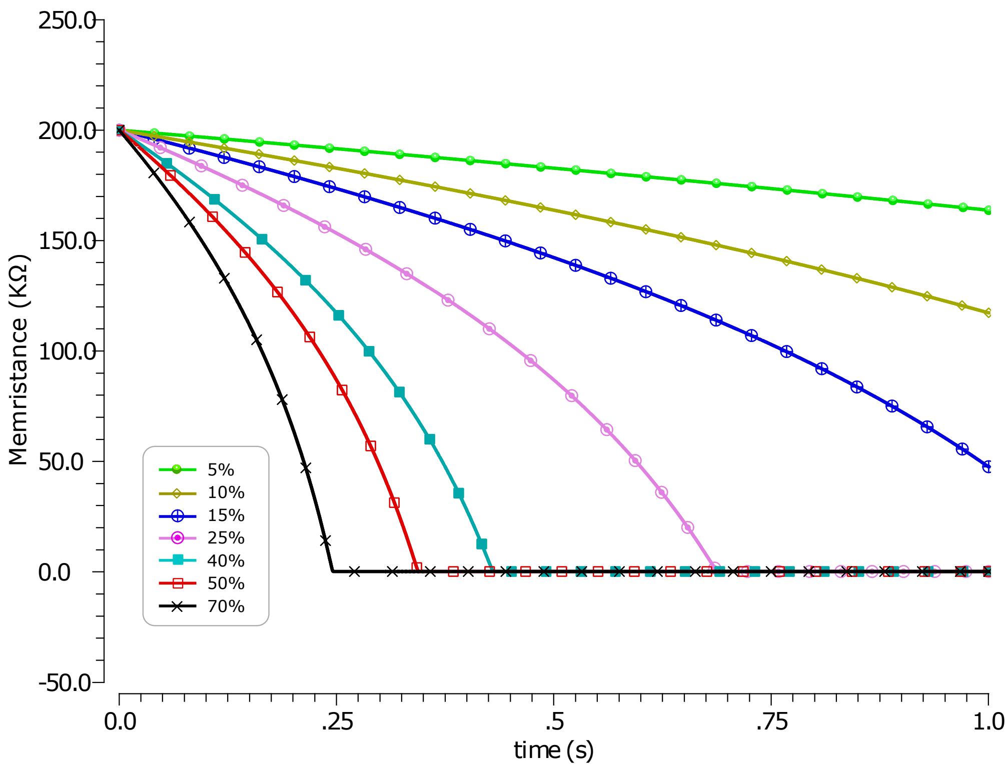


FIGURE 4.4: Memristance rate of change over time for different Duty cycles of a train of voltage pulses with amplitude $v_p = 500mV$ and programming frequency $\omega_p = 50\omega_0$.

memristor was first set to a high resistance state before it is programmed with a duty-controlled train of current pulses of frequency $\omega_p = 5\omega_0$ and current amplitude $i_p = 160\mu A$. The larger the duty, the more the charge integrated in the device and thus, the quicker the rush to the ON state.

Similar to Figure 4.3, in Figure 4.4 the memristor is also programmed from a high low conductive state down to a high conductive state but this time around with a duty-controlled train of voltage pulsed of frequency $\omega_p = 5\omega_0$ and voltage amplitude $v_p = 160\mu A$ as seen in the inset of the diagram. The same characteristics is replicated here and as observed in both plots, during inter–pulse period, the memristance maintains it's state. Even though it is obvious that the memristance programming behaviours are not proportional to the number of input pulses we observe a decrease in memristance proportionally to the integration of applied voltage. Hence, as more charges are integrated in the device with the application of pulses, step size increases (high rate for low memristance and low rate for high memristance).The memristor can nevertheless be programmed either by controlling the number of input pulses or by patterning the duty ratio [64].

Finally, the last experiment was to program the memristor with a bipolar train of

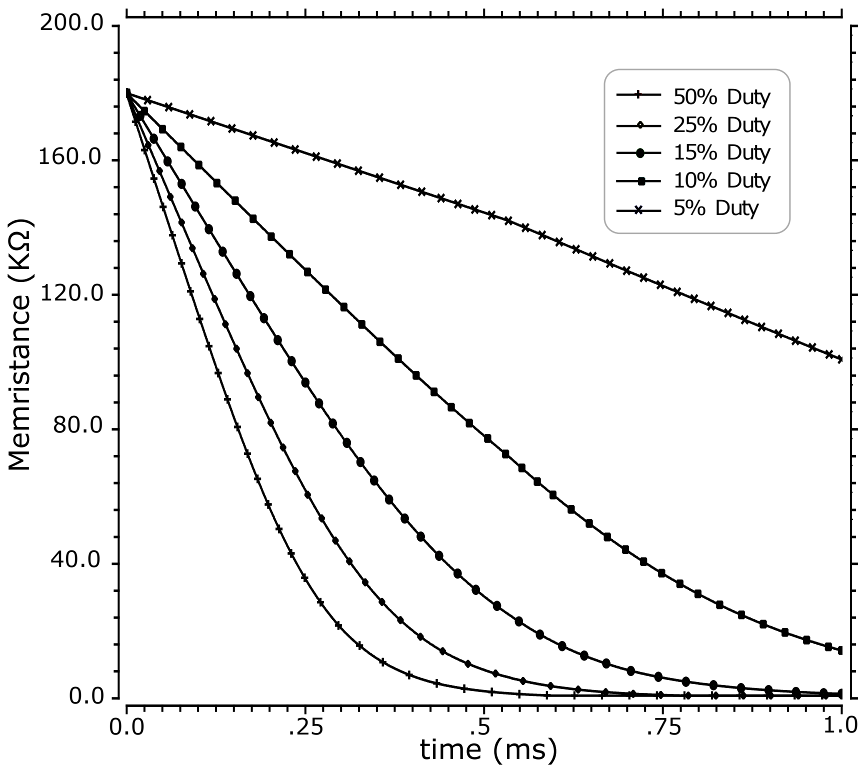


FIGURE 4.5: Memristance rate of change over time for different Duty cycles of a bipolar train of current pulses with amplitude $v_p = 500mV$ and programming frequency $\omega_p = 50\omega_0$.

pulses. The idea here is to return the memristor back to its initial state after programming with the application of pulses with reversed polarity (reset). Figure 4.5 shows the change in memristance with a sequence of applied current pulses. Notice that during the application of the positive pulses, the memristor gradually programmed in a decremental configuration. The attempt to return the memristor back to its original state means we have to apply equal number of pulses to the memristor, albeit with an opposite polarity. We note again the memorisation of state during inter–pulse period.

4.4 DC Programming

We have already seen that varying the flux across the memristor will considerably change the resistance state of the device. This means that passing a current through the memristor should either increase or decrease its memristance depending on the direction in which the current flows and also in what configuration *(polarity)* the device is mounted. The memristor can be considered as a time varying resistor where the resistance changes as a result of the summation of current that has passed through it. When the current

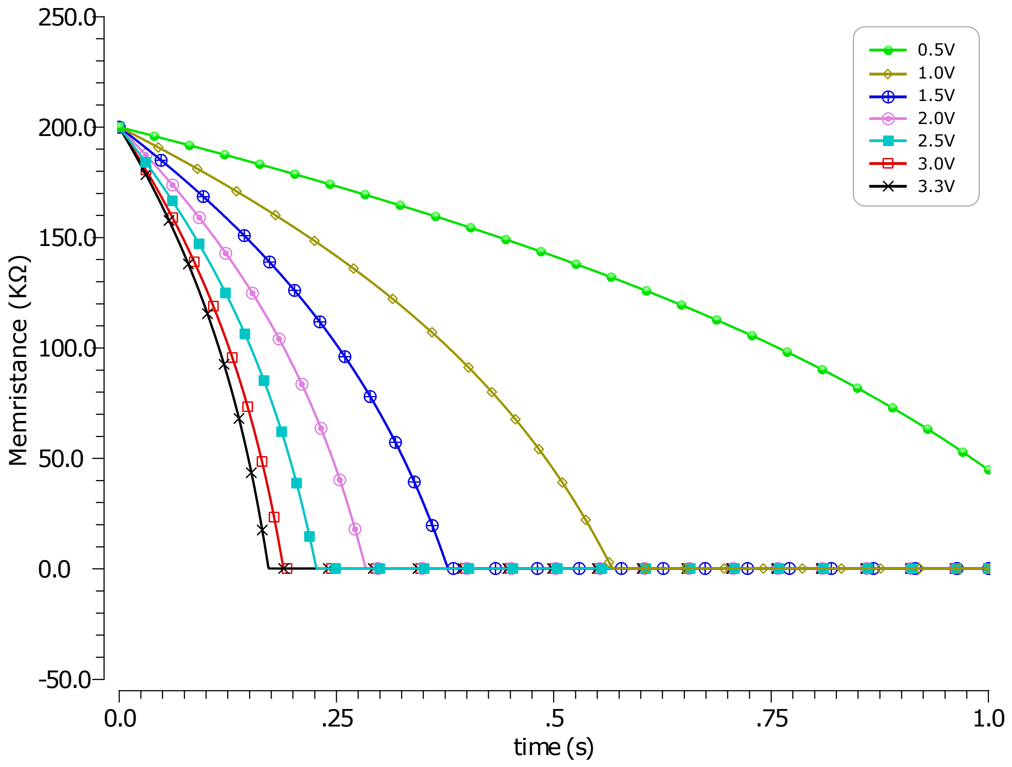


FIGURE 4.6: Memristance rate of change over time for different constant DC bias.

flowing through it is zero, the summation of current becomes constant, and thus the resistance remains unchanged.

Assuming the memristor is subjected to a constant bias voltage; the change in state of the device should depend on the polarity of the applied voltage and of the device itself. Therefore, we can also program the memristor by simply applying the a DC voltage albeit with very little precision. The issue of accuracy and programmability is not within the scope of this Section and would be discussed later on.

In this experiment, we first reset the device, we then apply a bias voltage over a period and observe the change resistance state for various bias voltages applied. Figure 4.6 is the simulation output of bias voltages ranging from *5V* to *3.3V*. As in the previous cases, the curves are self explanatory.

4.4.1 Switched Memristor Circuit

As we already know, under constant DC bias the memristor state changes as long as the applied voltage is greater than the memristor's threshold voltage. If the memristor is to be used in a programmable analogue circuitry, then we have to find a way of preserving its state under certain conditions. The question we asked ourselves is, what if we wanted the memristor to work as a conventional resistor all the time even in the presence of

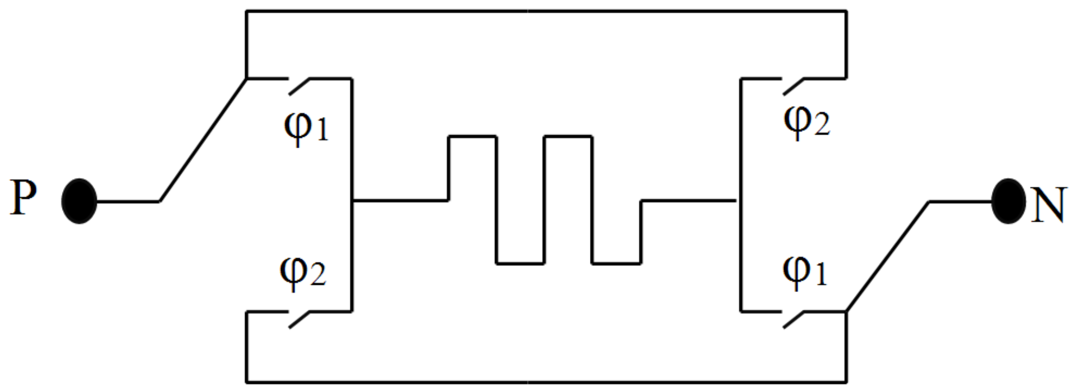


FIGURE 4.7: Schematic of the Switched Memristor sub-circuit.
The internal terminals of the memristors are inverted by a non-overlapping two-phase clock. Switches φ_1 and φ_2 are simply MOS switches. The sub-circuit can be adopted in a more complex circuit and will be suitable for IC applications.

certain conditions which should change its state. An example of such an implementation is the pulse coded programming circuit proposed by Shin *et al.* for high frequency circuits having two blocking capacitors to isolate DC components [64]. See Section 4.2 for more details about the architecture.

Unlike the configuration proposed by Shin *et al.*, the approach presented in this work is designed to work in all frequency domain. We simply employed a switched memristor circuit to achieve our goal. Figure 4.7 is the block diagram of the Switched Memristor circuit. The memristor is chopped in the circuit by constantly switching its polarity with CMOS switches controlled by a non-overlapping two phase clock with adjustable duty cycle for compensation. The practice of chopping [112] the memristor, is to make it function like a resistor all the time [16]. To change its state, it could be pulsed externally with some programming pulses. Depending on the application, in order to minimise drift effects (increase or decrease in memristance) during the switching of the two-phased clock, the chopping frequency could be chosen with respect to the memristor frequency sensitivity [113]. This implies a trade-off between the device time constant and the chopping frequency Therefore, prior information about the memristor operation model is needed.

Simulation of the Switched Memristor Circuit – In order to validate the functionality of the circuit, a simulation of the Switched Memristor Circuit and a normal memristor having the same physical characteristics was carried out. Both memristors were set at their midscale value (*100K*Ω) and then biased with a constant voltage of *3.3V* for *1.5s* similarly to the simulation carried out in Figure 4.6. The simulation result of the Switched Memristor Circuit is shown in Figure 4.8. As observed, the Switched Memristor is able to retain its state without any significantly change from its initially programmed state, whereas, the normal memristor circuit exhibited a significant change

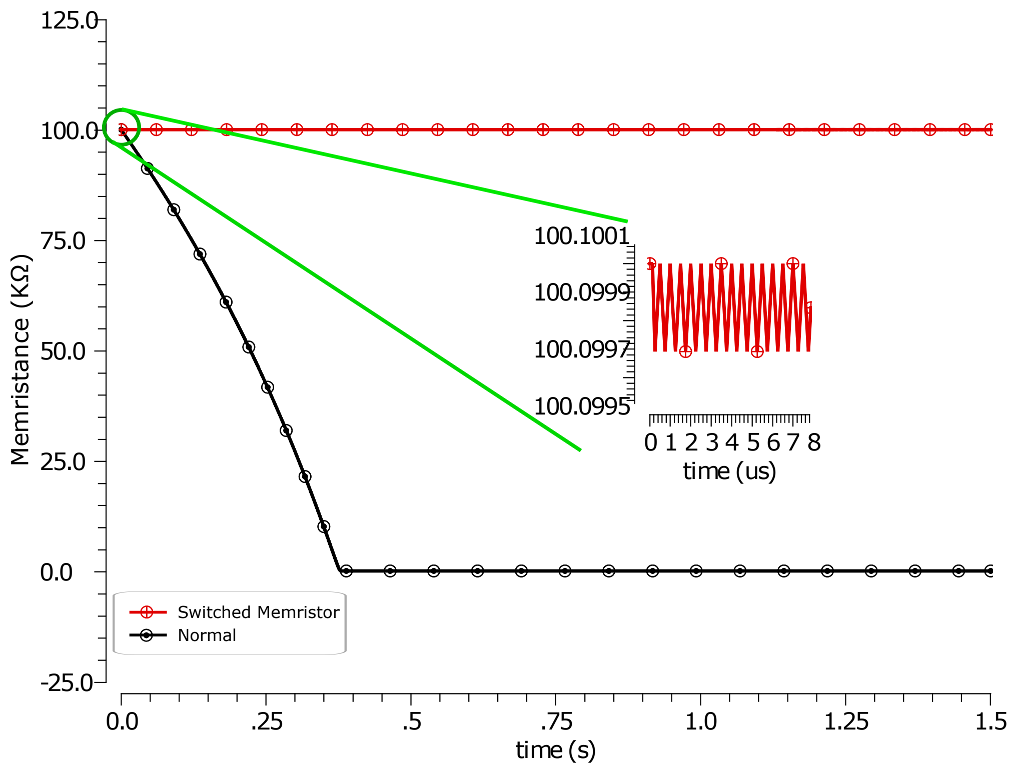


FIGURE 4.8: Behaviour of the Switched Memristor Circuit under a constant DC bias.

of resistance down to a higher conductive state. The inset is a blow-up of the impact of the chopping frequency. It can be observed that the positive and negative levels of the chopping clock over memristance is negligible. Nevertheless, it is worth to note that, over a long period of operation, due to drift effects, the device should be re-programmed in order to guarantee the performance of the entire circuit. Therefore, constant recalibration of the circuit could be necessary to curtail this problem.

Additional information about the two phase non-overlapping clock frequency, the Switched Memristor Circuit and its application in a Comparator with Programmable Hysteresis is provided in Appendix A.

4.5 FBK Memristor Experimental Characterisation

To buttress the point I have been trying to make in the previous Sections of this Chapter, some experimental results related to memristor resistance modulation techniques carried out on memristors fabricated in FBK is presented in this Section. Several measurements were carried out on various samples on an the array of fabricated solid state memristor. In this experiment the devices under test were first electroformed, the devices were then programmed with pulses and one experimental instance and on another, subjected

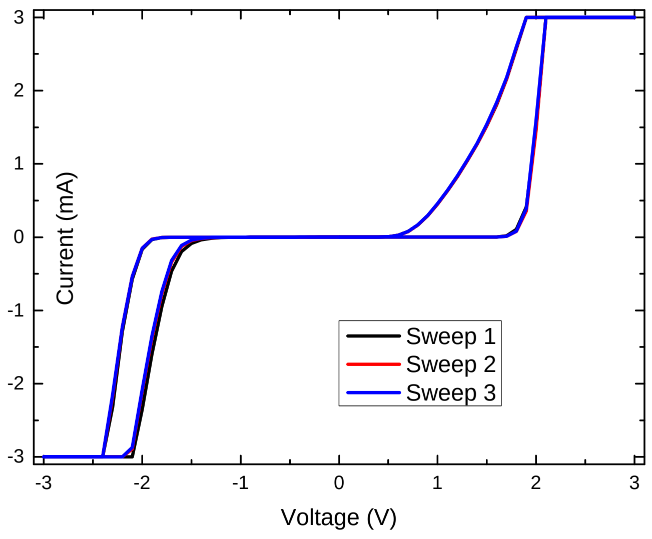


FIGURE 4.9: Electrical measurement of the current–voltage behaviour of the memristor device.
The i–v curve is measured by applying a bipolar triangular waveform of amplitude v_p $= 3V$ and frequency of 33 mHz with a 50% duty. The reason for the saturation of the current branch is due to a compliance of $\pm 3mA$ of the measurement instrument.

to constant DC bias. Both techniques based on charge accumulation to increase the conductive state of the devices were then compared. For all experiments carried out, the measurement setup is the same as the one presented in Section 2.5.

Pulse Characterisation – In this experiment we applied a train of pulses at a programming frequency of 500 mHz. After the electro-forming, the device operating channel is opened by a couple of voltage sweeps using a triangular waveform of amplitude $v_p = 3V$ running at a frequency of 33 mHz. The $i--V$ plot of the device considered for this experiment is given in Figure 4.9 before the application of pulses, we made sure the device was reset back to it's lowest conductive state.

In the pulse characterisation experiment, two distinct measurements were carried out and unlike the simulation of the pulse-based programming previously presented (Section 4.3.1), we measured the current for applied input train of voltage pulses with a duty ratio of 50%. What was varied here was the amplitude v_p. Figure 4.10 is a plot of the measured current versus time. We applied a total of 200 pulses for a period $400s$ with a pulse width (PW) of $1s$ per pulse. The time scale in Figure 4.10 has been normalised $(f = \frac{2}{T})$ to present the measurement in a more realistic way in order to show and compare the evolution of the measured current at different v_p because during the inter pulse period, no current is flowing through the device. Notice that for higher value of amplitude, more charges are integrated into the device with an increases amount of current. Both curves of $v_p = 1.6V$ and $v_p = 1.65V$ were performed on the same device. After the first measurement for an amplitude of $1.6V$, the device was then reset again

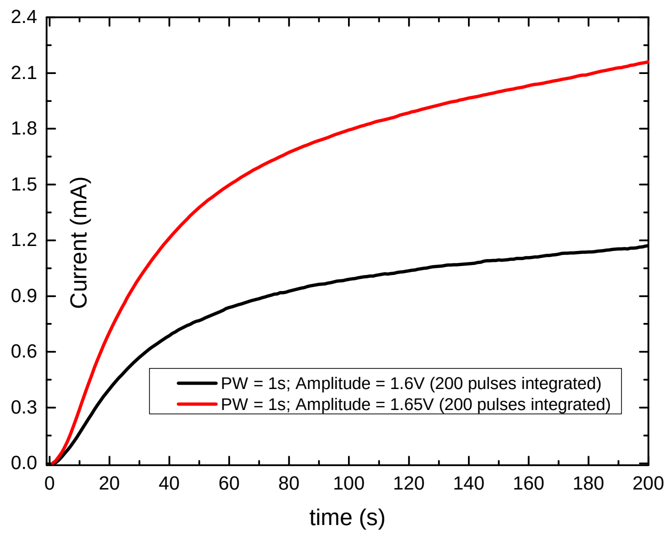


FIGURE 4.10: Current rate of change over time for a train of pulses of 500 mHz programming frequency, 50% duty cycle and two different amplitudes v_p of $1.6V$ (black) and $1.65V$ (red) respectively with the measured current evolving as expected

back to its low conductive state (R_{OFF}), typically by the application of negative voltage (opposite polarity to the programming voltage), before the next measurement of $1.65V$ amplitude was taken.

Constant DC bias – In this experiment we subject the DUT to a constant DC bias and observe the state tuning from a high resistance state (low conductive state) down to a low resistance plateau (high conductive state). The experimental procedure (reset, program, read, reset) is the same as in the previous experiment. Thanks to our setup, during this kind of decremental programming, with the set current compliance, when the device is in the low resistance state we avoid having excessive current which could damage the device.

Figure 4.11 is the measurement result for the application of a constant DC bias for a duration of *200s* for two different bias voltages of $1.65V$ and $1.65V$ applied consecutively. It is worth again to emphasise that for both experiments, the memristor was first reset to its OFF state. As shown by the curve, within a period of *200s*, more charges were injected for during the $1.65V$ bias (red curve) which is as expected.

Given the assumption that an equal amount of charge would have been injected to the device both in the pulse-based technique and the constant DC biasing, if we plot both experiments on the same curve we should see a similar characteristics in their curves. Figure 4.12a is the rate of change of the current over time for a comparison between results obtained from pulse-based programming of amplitude $v_p = 1.6V$ and the DC-bias technique of $1.6V$.

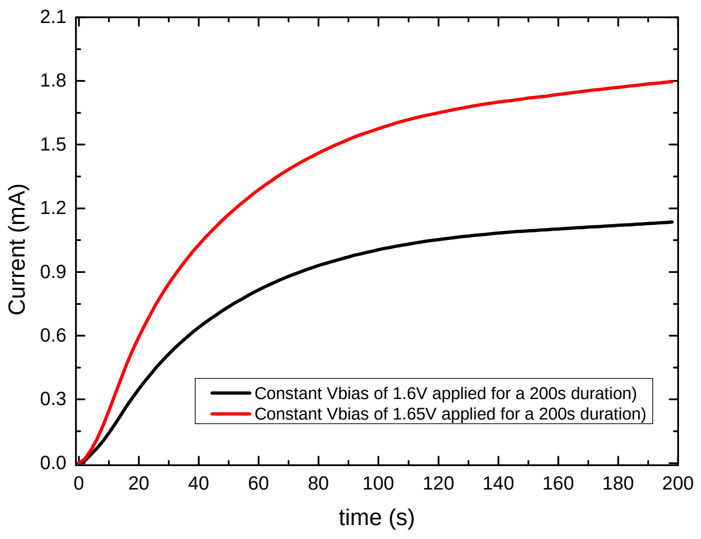


FIGURE 4.11: Current rate of change over time for a constant DC bias

Similarly, Figure 4.12b is a plot of the current rate of change over time for a comparison between results obtained from pulse-based programming of amplitude v_p = *1.65V* and the DC-bias technique of *1.65V*. As observed, the curves were reproduced for the different programming methods with the *1.6V* experiment giving a better reproducibility. Thus we can conclude that at low voltages, the programming is much more uniform. Same could be said at low frequencies.

4.6 Memristor Programming Circuitry

This Section is dedicated to the implementation of two different programming techniques: *(i)* an analogue approach and *(ii)* a digital solution. Both solutions exploit the dynamic state modulation of the memristor. In each architecture, I will proceed by first introducing the block diagram of the technique being introduced, then move on the functional description of each blocks by presenting an illustration if a programming cycle before showing some simulation results of each programming circuits. hardware implementation of the circuits and experimental results will be presented in the following Chapter.

4.6.1 Digital Programming Circuit

The functional block diagram of the programming circuit is given in Figure 4.13. The circuit is capable of programming a memristor in both a decremental and an incremental fashion. The circuit takes as input, a digital signal (I_N) and outputs another digital signal (Q). For a decremental programming configuration, switches $S1$ are turned on and the memristor is connected to the feedback branch of the operational amplifier (op-amp) to

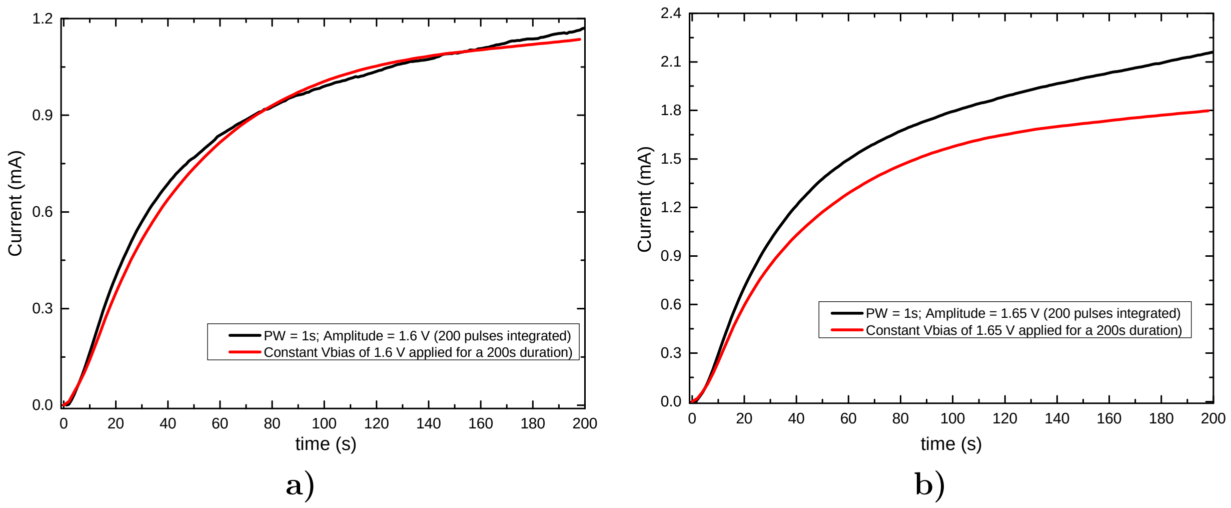


FIGURE 4.12: Current rate of change over time for a comparison between results obtained from pulse-based programming and the DC-bias with the assumption that an equal amount of charges would have been accumulated by the device

form a gain stage with its positive terminal connected to the op-amp's output as shown in Figure 4.14. Incremental programming is achieved with switches $S2$ turned on (this implies $S1$ going to a low state). With an incremental programming configuration, the memristor is connected to the input stage oriented with its positive terminal connected to the inverting terminal of the op-amp as depicted in Figure 4.15.

Circuit Operating Principle – Both programming configurations work in the same way with the memristor and resistors connected to the inverting voltage amplifier to form a gain stage. The output of the gain stage is continually sampled and compared to a reference voltage vth. What we imply by decremental programming configuration entails programming the memristor from a low conductive state (high resistance value R_{ON}) down to a high conductive state (low resistance value R_{OFF}). The reverse is the case for an incremental programming configuration.

Let us consider the schematic diagram given in Figure 4.13. I_N is the digital input signal driving the programming voltage levels V_H & V_L. V_H is used for incremental programming (I_N=*logic* 1), meanwhile V_L is used for decremental programming (I_N=*logic* 0). At the initial stage, the input is completely disconnected for the network, output of op-amp is connected to its virtual ground 0, the output is then compared to a reference voltage vth which could fall in this condition $vss<vth<vdd$. If $vth>0$ then $Q=0$ and the circuit is configured for a decremental programming of the memristor. If $vth<0$ then $Q=1$, we have incremental circuit configuration. This is achieved thanks to the control block. After the kind of programming configuration has been determined, the input is then connected to the rest of the circuit and the memristor is programmed. The control

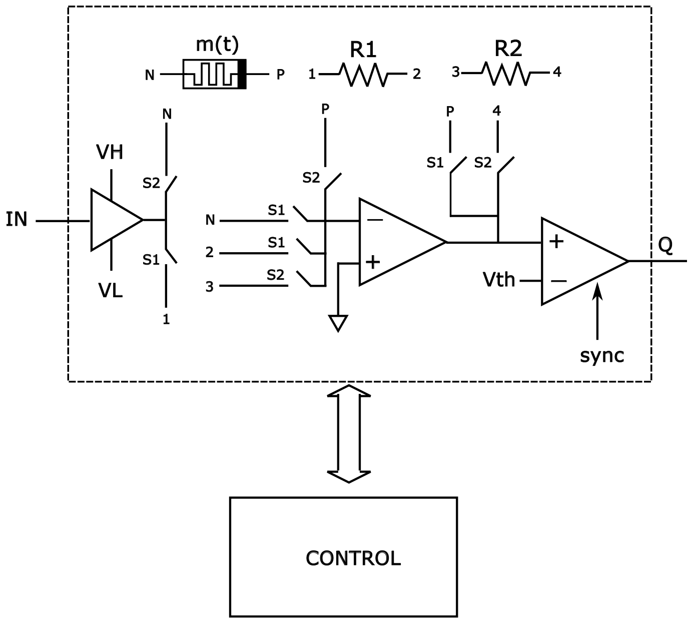


FIGURE 4.13: Functional Block Diagram of the Programming Circuit

block continuously samples the output of the comparator Q. Programming ends once Q toggles, triggering the control block to set the input programming signal to tristate and another programming cycle can commence.

The digital programming circuit can be divided into three blocks:

The input stage – the circuit basically has a binary input which triggers the CMOS drivers to supply the appropriate analogue voltages to the input state of the operational amplifier. The input goes to tristate once programming has been achieved, i.e., the toggling of the output Q.

The inverting amplifier stage – which consists of an op-amp mounted as an inverting

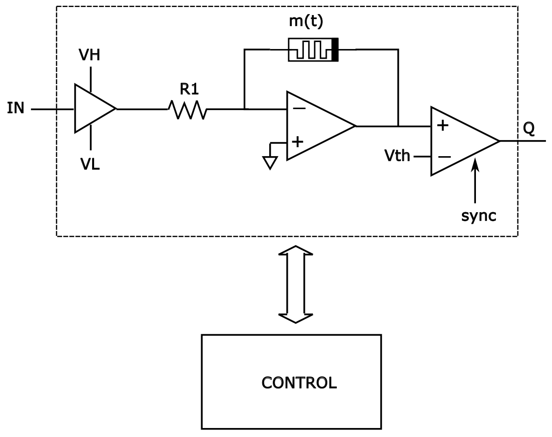


FIGURE 4.14: Decremental Programming Circuit Configuration.

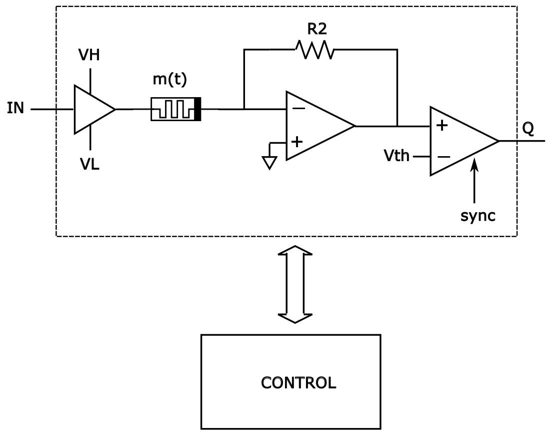


FIGURE 4.15: Incremental Programming Circuit Configuration.

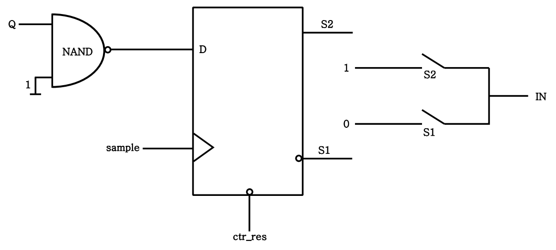


Figure 4.16: Function implementing the Decision Block.

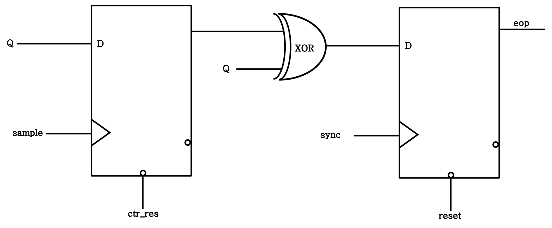


Figure 4.17: End of Programming Block.

voltage amplifier. Based on the programming configuration being implemented, the op-amp's gain should gradually increase or decrease during memristor state tuning. A clocked-comparator is used to periodically sample the output of the gain-stage. In the decremental programming (Figure 4.14), as the memristance decreases, the gain also decreases, therefore, the output of the op-amp is gradually reduced until it reaches vth. However, during an incremental programming, as the memristance increases, the gain is also increased, therefore, the output of the op-amp gradually increases until it reaches vth.

The control & decision block – The implemented control circuit consists of digital blocks realising two main functions to close the loop of the programming circuit. By default, the network is configured for a decremental programming.

- Sample and Decision: Before the memristor is programmed with respect to the given reference voltage vth, the control block first decides in what direction the memristor will be programmed by activating the appropriate switches ($S1$ or $S2$). This block is also charged with providing the programming circuit with the appropriate digital input signal (IN). By default, the network is configured for a decremental programming, i.e., during reset, signal $S1$ is high.

- End of state tuning: The other function realised by the control block is a logic function that signals the end of memristor state programming. This is achieved thanks to the circuit given in Figure 4.17. During the Sample and Decision phase, the state of Q is memorised, Q is then periodically checked every time the memristor is programmed. As soon as the Q toggles (high to low or vice versa), the signal *eop* goes high, signalling the end of programming, therefore, disconnecting input voltages. Another programming cycle can now commence.

Figure 4.18 is the timing diagram illustrating one programming cycle.

The functionality of the overall circuit has been validated through simulation on using Cadence Spectre. The simulation parameters are given in Table 4.1. The type of memristor used is based on the HP model [6].

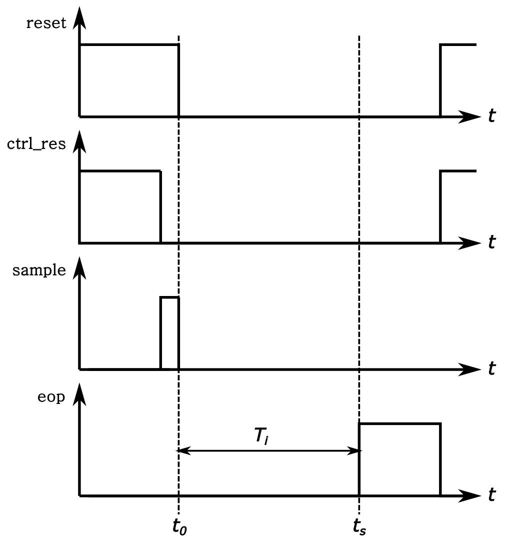


FIGURE 4.18: Timing Diagram for One Programming Cycle

TABLE 4.1: Digital Programming Circuit Simulation Parameters.

Parameter	**Value**
$R1$	$200K\Omega$
$R2$	$50K\Omega$
V_H	$1.65V$
V_L	$-1.65V$
Incremental R_{t0}	$30k\Omega$
Decremental R_{t0}	$200k\Omega$
Supply voltage	$0, +3.3V$

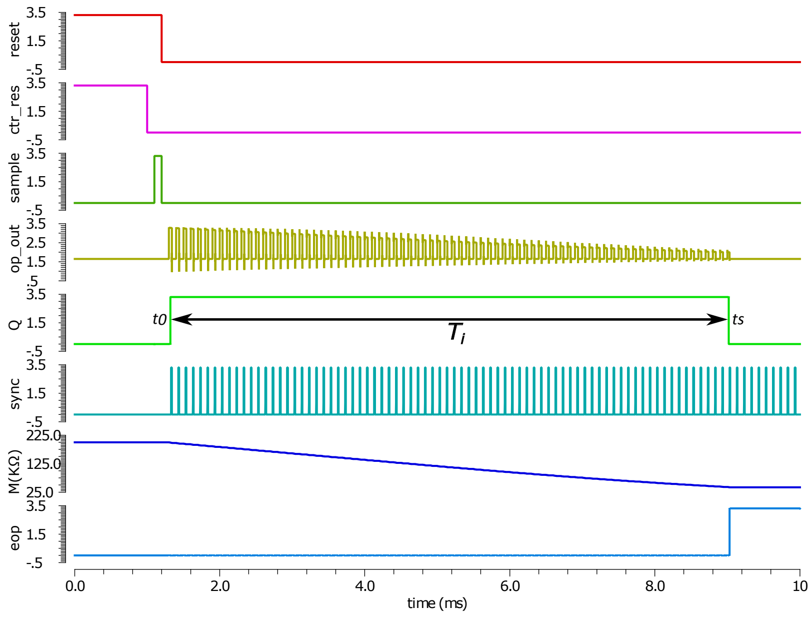


FIGURE 4.19: Simulation of the Decremental Programming Circuit

Decremental Programming – Figure 4.19 is a simulation of one programming cycle for a reference voltage $(vth = 2.0V)$. The memristor was first set to a high resistance state (@$t0$ $R_{init} = 200k\Omega$), after reset, programming starts. Input voltage V_L is used to program the memristor for a duration of $40\mu s$ every $100\mu s$. For every applied pulse, the output of the op-amp as well as the memristor decreases by a certain Δt. Once the op-amp's output latches unto vth, Q toggles and signal *eop* goes high signalling the end of the programming cycle. For the simulated value of $vth = 2.0V$, @ts, the programmed memristance $R_{final} = 42k\Omega$. Figure 4.21a is a plot of the programmed memristance versus the vth showing a good linear relationship between these two parameters.

Incremental Programming – Figure 4.20 is a simulation of one programming cycle for a reference voltage $(vth = 1.0V)$. The memristor was first set to a low resistance state (@$t0$ $R_{init} = 30k\Omega$), after reset, programming starts. Input voltage V_H is used to program the memristor for a duration of $40\mu s$ every $100\mu s$. For every applied pulse, the output of the op-amp as well as the memristor increases by a certain Δt. Once the op-amp's output latches unto vth, Q toggles and signal *eop* goes high signalling the end of the programming cycle. For the simulated value of $vth = 2.0V$, @ts, the programmed memristance R_{final} $= 127k\Omega$. Figure 4.21b is a plot of the programmed memristance versus the vth. As observed in the graph, in terms of programming accuracy, the incremental configuration suffers a little set back compared to its decremental counterpart. The memristance vs vth plot did not express good linearity, possible reason for this is discussed in the next section.

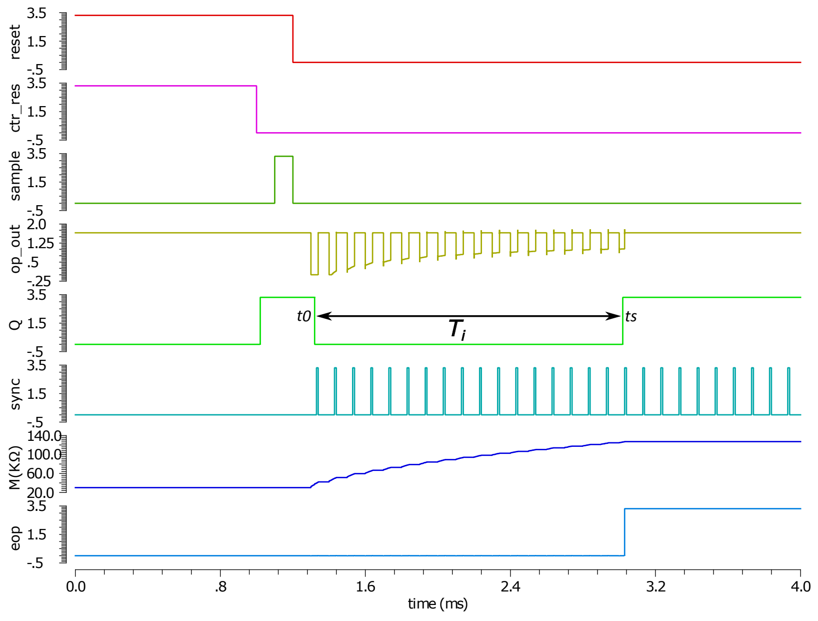
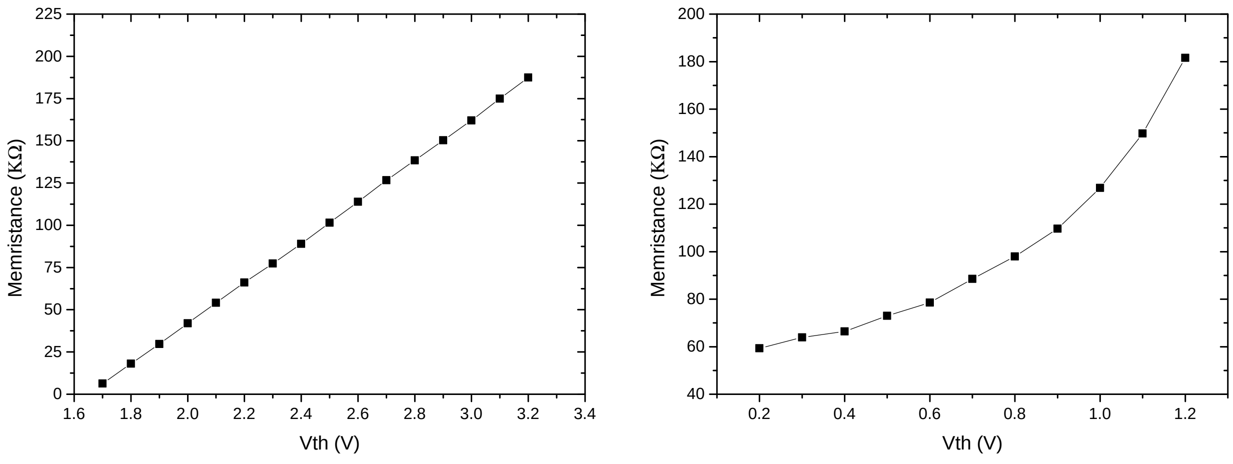


FIGURE 4.20: Simulation of the Incremental Programming Circuit



a **b**

FIGURE 4.21: Programmed Memristance vs reference voltage. **a**, Decremental Programming. **b**, Incremental Programming

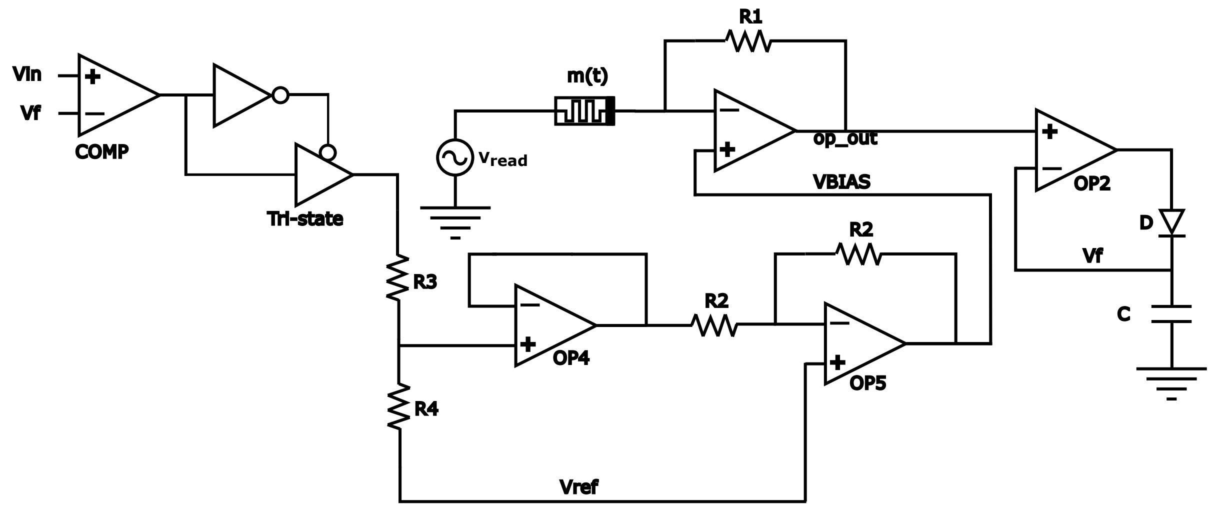


FIGURE 4.22: Schematic Diagram of the Analogue Programming Circuit

4.6.2 Analogue Programming Circuit

In this Section, the operation of an analogue programming circuit is presented. The architecture is a slight modification of the circuit proposed by Berdan *et al.* having an AC voltage source as input signal and which exploits the dynamic resistance modulation of memristors under constant DC bias. The modified circuit is an improvement of Berdan *et al.*'s proposal which also employs feedback for converging the state of the memristor to a given level within two decade, given an analogue input voltage *Vin*.

The schematic of the analogue programming circuit is given in Figure 4.22. In this circuital configuration, the memristor is instead mounted at the input stage of the inverting voltage amplifier with it's positive terminal connected to the Op-amp's virtual ground and its negative terminal connected to the analogue input voltage *Vin* as shown in the Figure 4.22. Similar to the digital programming circuit, the analogue circuit can be split into 3 stages: (i) the inverting amplifier stage, (ii) the peak detector stage which gives information about the output of the inverting amplifier stage, and finally (iii) the voltage divider stage.

Circuit Operating Principle – In the inverting voltage amplifier stage, *m(t)* and *R1* are connected to the operational amplifier to form a gain stage. The voltage at the output of the operational amplifier is:

$$OP_{OUT} = -\frac{R_1}{m(t)} * V_{read} \tag{4.1}$$

This output voltage envelope is continuously tracked by the peak detector which offers information on the value of memristance every $1/f_{read}$ (T_{read}) seconds. The output of the peak detector Vf is then fed to the input of the comparator which compares it

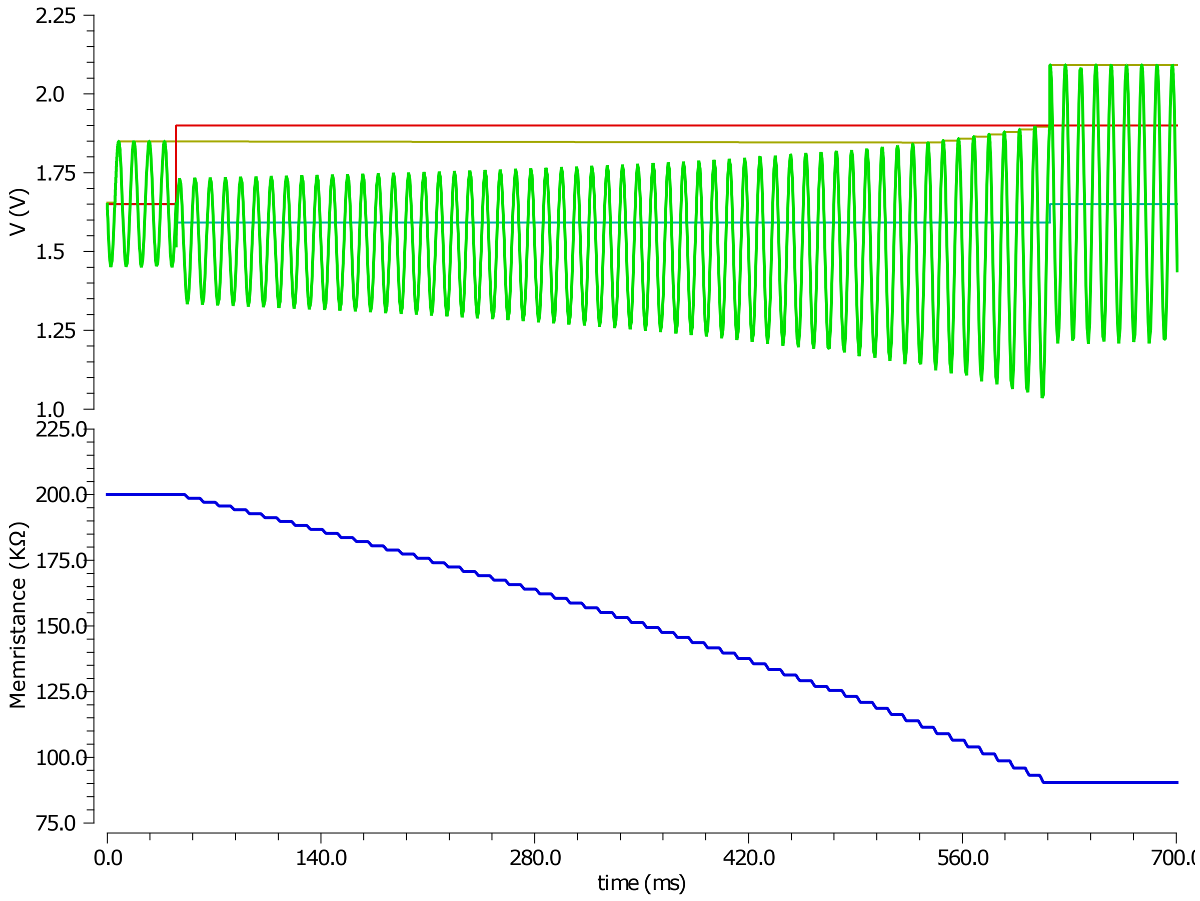


FIGURE 4.23: Simulation Illustrating one programming cycle of a memristor from $200K\Omega$ down to $80K\Omega$

with an input programming voltage Vin. Let us consider the simulation illustrating one programming cycle in Figure 4.23. At the initial state where $Vin = 1.65V$ *(red)*, the output voltage of the peak detector is equal to:

$$V_F = V_{read} * \frac{R_1}{m(t)} \tag{4.2}$$

The comparator *(COMP)* is set to ground, therefore, the voltage divider input is in *tri-state*. The non-inverting input of the *OP1* is therefore set to the potential $VBIAS = Vref$. The AC signal at the inverting amplifying stage with amplitude $Vread$ and frequency $fread$ and a DC offset of $1.65V$ which corresponds to the *(Vref)* is chosen in order not to cause any significant change in memristance over the course of one period. The memristor maintains it's state as long as $V_{IN}<V_F$. An increase in the voltage V_{IN} to a value larger than V_F will cause to output of $COMP$ to swing to Vdd. This provides a potential of $V_{BIAS}<Vref$ to the non-inverting input of OP. The value of V_{BIAS} can easily be set depending on the ratio resistors in the voltage divider. Memristor programming

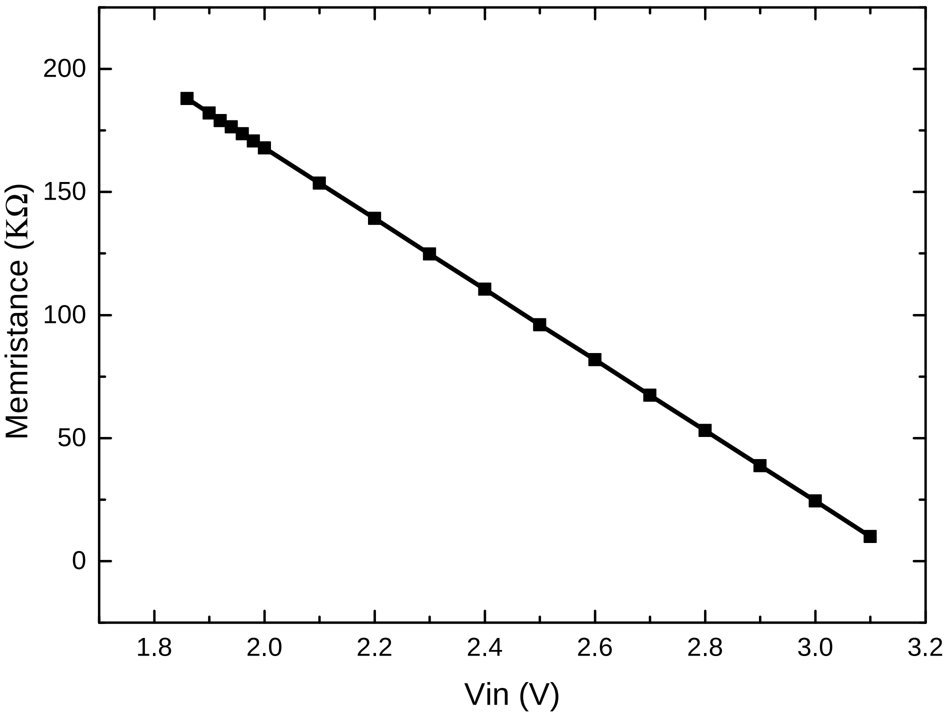


FIGURE 4.24: Results showing programmed memristance value across two decades versus corresponding input programming voltage V_{IN}

is achieved thanks to the DC current $I_{DC} = V_{BIAS}/R_1$ imposed over the AC current supplied V_{READ}. This is possible since the open-loop gain of the operational amplifier is large, so its output is forced to the same potential as V_{BIAS}. This way the memristor is programmed decrementally.

The expression of V_F as the memristance decreases is given by:

$$V_F = \frac{R_1(V_{read} + V_{BIAS})}{m(t)} + V_{BIAS} \tag{4.3}$$

The expression of the attained resistive state at the switching point of the comparator (COMP) where $V_F = V_{IN}$ is given by:

$$m(t_s) = \frac{V_{read} - V_{BIAS}}{V_{IN} - V_{BIAS}} * R_1 \tag{4.4}$$

Once V_F becomes greater than V_{IN} again the comparator output drops to ground, the voltage divider input goes to *tri-state* and the non-inverting input of OP is latched onto V_{ref} again, establishing a change in V_F equivalent to ΔV_{STOP} as illustrated in the simulation diagram of Figure 4.23 More details about the description of the circuit operation can be found here [84].

Simulation – The functionality of the analogue programming circuit was validated

Table 4.2: Analogue Programming circuit Simulation Parameters.

Parameter	Value	Parameter	Value
R_1	$200K\Omega$	$m_{(init)}$	$200K\Omega$
$m_{(vth)}$	$200mV$	V_{read}	$200mV$
f_{read}	$100Hz$	$V_{(BIAS)}$	$20mV$
C	$1pF$	$m_{(ron)}$	200Ω
$m_{(roff)}$	$200K\Omega$	$m_{(D)}$	$10nm$
$m_{(\mu_\nu)}$	$10^{-10}cm^2S^{-1}V^{-1}$		

through simulation using Cadence Spectre. The simulation parameters are given in Table 4.2. The type of memristor used is as well based on the HP model. Fig 4.24 is a graph showing programmed memristive state value across two decades for different voltages V_{IN}. In this simulation, for the different V_{IN}, the device was first initialised to $m(init)$ and then programmed down decrementally in conformity to the illustration of Fig 4.23. The graph shows a good linear relationship between $m(t)$ and V_{IN}.

4.7 Conclusion

Memristor resistance modulation gave a detailed overview of the various programming techniques used in changing the conductance of memristors. The programming techniques discussed in this chapter is regrouped into three categories: (i) pulse-based programming technique which utilises pulses of distinct width and amplitude to read, write and reset a device to an intermediate state, (ii) DC programming technique and (iii) self-terminating programming method.

For each programming method some simulation and experimental results carried out on memristors fabricated at FBK are presented. Moreover, applications related to each programming technique have been proposed.

In the last section of this chapter, two self-terminating memristor programming circuits capable of accurately tuning the memristance of a memristor to correspond to an input reference voltage are presented. The device is tuned accordingly by automatically patterning the input programming pulses, thanks to a change in gain of the operational amplifier within which the memristor forms a gain stage. The circuit not only exploits a digitally assisted form of memristor analogue state tuning using voltage pulses of fixed amplitude and width but also has the advantage of not needing external processing to accurately determine the memristor state.

The functionality of the circuit was validated with some simulation results. For a simulation of the network for memristance across two decade, the decremental programming circuit seems to outperform the incremental circuit in terms of linearity with the input reference voltage vth, as observed in Figure 4.21, this is due to certain factors which we would like to discuss. For simulations carried out in the decremental configuration, it was relatively easier to program the memristor to any point across its full bandwidth. Whereas for the incremental part, we had to start from an initial resistance value R_{init} different from the memristor's R_{ON} state in order to be able to exploit a good percentage of the memristor's dynamic range. This is because if R_{init} is too small compared to the reference resistor $R2$, the output gain becomes excessively high, the op-amps output remains saturated and a change in the memristance will have little effect on the op-amp's output. This phenomenon does not occur in our decremental configuration because the programming cycle departs with a gain of 1, i.e., $R_{init} = R_1$, therefore we are able the exploit the memristor's overall dynamic range.

With the advantage of having a huge versatility, the programming precision can be tune to suit a wide variety of devices and applications. The circuit has been implemented in a CMOS process and our aim for a future work will be to investigate how an automatic programming of a large array of memristive devices could be carried out using this simple circuit. Currently we're working on a development board to test the the functionality of the circuit at a first instance with our memristor emulator [12] and then proceed to testing it with solid-state memristors.

Chapter 5

Memristor-Based Light-to-Resistance Encoder

5.1 Introduction

Basically every CMOS image sensor implementation of a multiple-reset pulse-frequency modulator (PFM) for light-to-frequency conversion would require a digital counter [114, 115]. Embedding a counter in each pixel of an imager would require significant silicon area, turning into a large pixel pitch. The number of pulses generated during a period of time is a linear function of the incident light falling onto the pixel during that time. A higher pulse count will thus correspond to a brighter pixel and vice-versa as observed in the illustration of PFM encoding of visual information shown in Figure 5.2c. The achievable dynamic range depends mainly on the resolution of implemented counter.

In [116], Wang *et al.* proposed an image sensor with an in-pixel light-to-frequency conversion, where a 10-bit counter takes over one-third of the pixel area. We propose to replace the digital counter with an analogue counterpart, made with a single memristor. Thanks to the memristor pulse-based non-linear programmability (See Chapter 4), light-to-resistance (L2R) encoding can be achieved with a light to frequency converter acting as a programming source and driving the memristor with current pulses proportional to the light intensity as depicted in the illustration of the operating principle of the proposed pixel architecture in Figure 5.3b.

In comparison to similar implementation, a memristor-based unit cell for a photodetector readout circuit was implemented in [65] by Yakopcic *et al.* implementing a pixel embedding a memristor which is aimed at directly programming the memristor with the photo-generated current. This approach of direct programming could have unrealistic power requirements to properly program the memristor. Conversely, in our approach the memristor is programmed through a voltage-controlled current-source driven by the pulses generated by the light to frequency block. This way we decouple the optical sensor from the analogue counter (memristor). The amount of current flowing into the memristor can be properly tuned according with the system requirements.

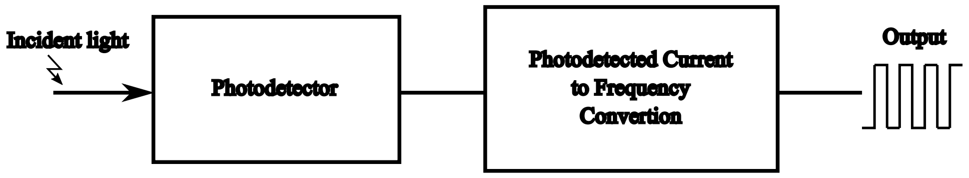


FIGURE 5.1: Functional Block Diagram of the Light to Frequency Conversion steps. The output of the *LTF* conversion is typically a square wave or pulse train whose frequency is linearly proportional to the light intensity

This Chapter which reports a pixel architecture implementing a light-to-resistance encoder with a memristor is organised as follows. In the next Section, I will give a brief introduction to the principle of light-to-frequency conversion before presenting the proposed pixel architecture in Section 5.3. Simulation results of the pixel architecture is given in Section 5.4 before concluding in Chapter 5.5

5.2 Light-to-Frequency Conversion

Light-to-Frequency (*LTF*) conversion can be implied as the a process which converts light intensity to a digital form (*pulse*). A functional block diagram of the process is given in Figure 5.1. The block diagram is composed mainly of a Photodetector front-end which is used to detect impinging light variations, and a current-to-frequency converter outputting the digital pulses. Based on the type of detector implemented, the output pulses can be modulated to suit several applications.

Pulse modulating (*PM*) techniques can be broadly classified into two basic categories, namely: (i) pulse-width modulation (PWM) encoding, and (ii) pulse frequency modulation (PFM) or spiking pixel encoding.

We are particularly interested in the pulse frequency modulation technique since the architecture presented in this Chapter and subsequent Chapters to come is solely based on this technique. Therefore even though it is just going to be a brief introduction to pulse frequency modulation, pulse-width modulation will not be treated as it is out of the scope of this Chapter.

5.2.1 Pulse Frequency Modulation

Pulse-Frequency Modulation (*PFM*) is a modulation method that combines the features of pulse-code modulation for representing both digital and analogue signal inputs. it is similar to Pulse-Width Modulation (*PWM*), where the magnitude of an analogue signal is digitally enoded as the duty cycle of a square wave. Unlike *PWM*, where the duty

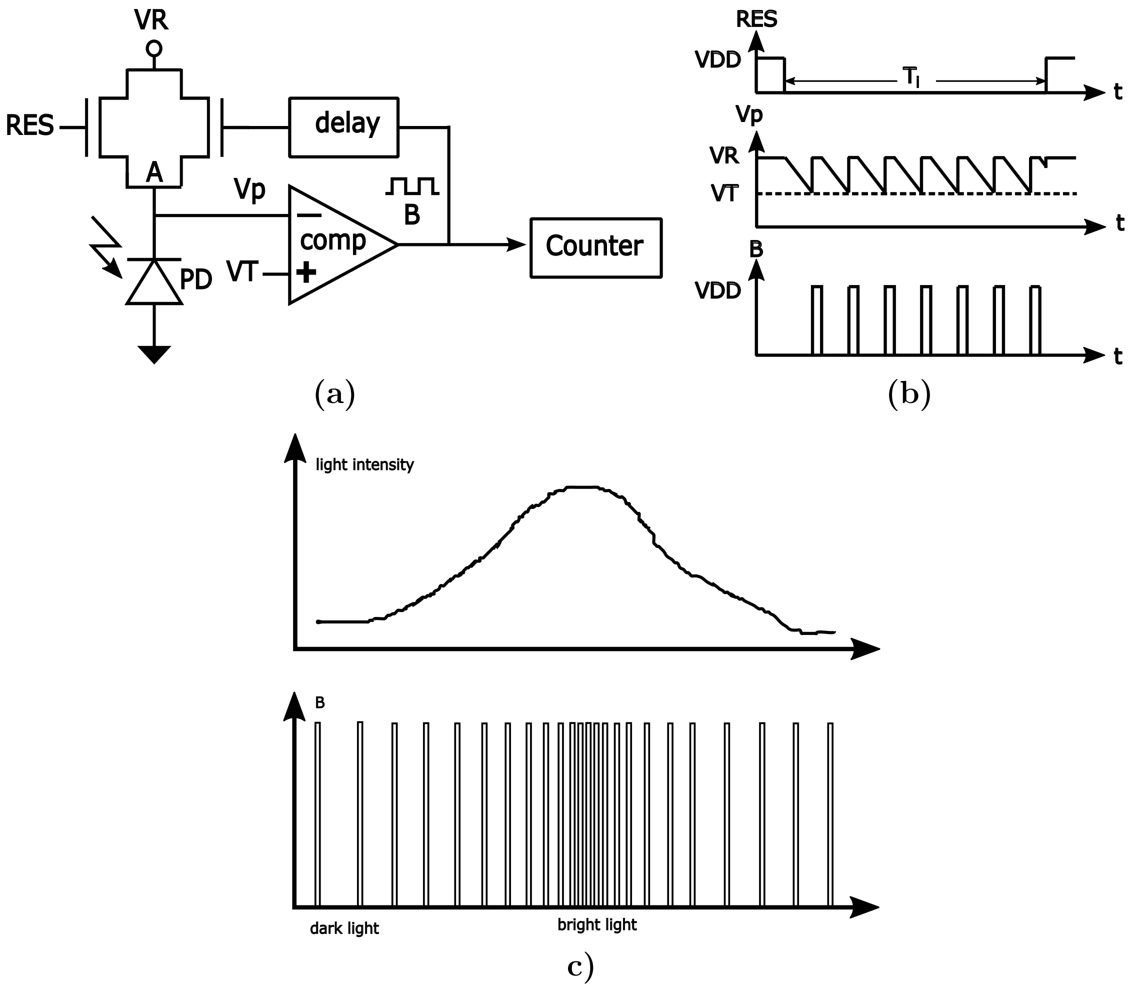


FIGURE 5.2: Operating principle of the L2F conversion. (a), Schematic of multiple-reset *PFM* digital pixel. (b), Illustration of the principle of multiple-reset *PFM* for L2F encoding during an integration time T_i. (c), Illustration of pulse modulation encoding of visual information in *PFM*

cycle of a squqre wave is varied at constant frequency, in PFM, the width of the square signal is fixed while the frequency is varied.

In digital pixel sensors, pulse-frequency modulation is typically employed to achieve high dynamic range by converting the photosensor current into a digital pulse signal through a process known as multiple-reset light-controlled oscillation (LCO).

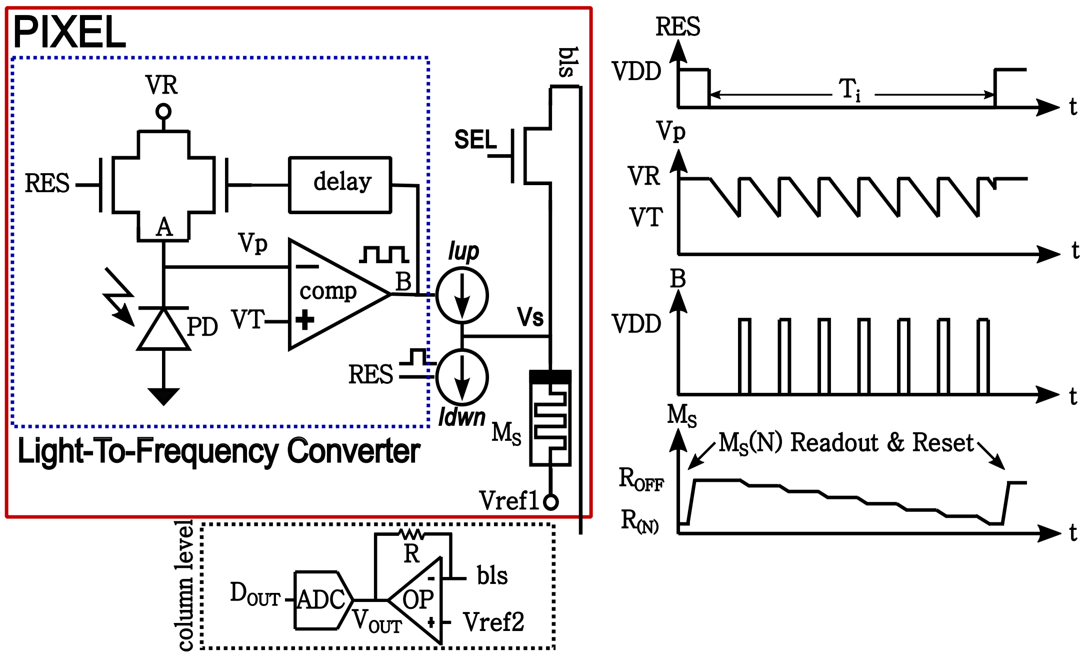
The principle of multiple-reset digital pixel is illustrated in Figure 5.2. To initiate a photo-measurement cycle, the photodiode voltage is pre-charged VR to a defined voltage level VR by applying a short pulse signal to a reset switch (transistor indicated a RES). Subsequently, the photodiode is discharged by the photo-generated current i_{ph}. Incident light generated photo-current causes the voltage across the photo-diode node to drop at a slope proportional to the illumination level. A comparator continuously compares the integration voltage ramp to a, usually fixed, voltage reference VT. When the voltage reaches the threshold of the comparator, the comparator's output flips and resets the photodiode. The exposure information is encoded in the edge timing. The incident light intensity is inversely proportional to the integration time T_i – brighter pixels yield shorter integration times and vice-versa. Light-to-frequency encoding is achieved since the output pulse frequency is linearly proportional to the light intensity. The encoded output pulses during the integration time is counted and the equivalent frequency can be decoded either off-line or in real-time.

Pulse frequency modulation pixel sensors as the advantage of having a high frame rate as well as facilitate real-time pixel level processing as we will see in subsequent Chapter. One of the majoy disavdantages of Pulse frequency modulation is it's complex pixel pixel architecture leading to larger pixel architecture and possibly higher pixel-wise fixed-pattern noise (FPN).

Even though smaller pixel architectures are aready being realised with the latest technology due to transistor-technology scaling down, we are still far from getting the best performance in terms of silicon area. This Chapter offers some novel solution to the issue of complexity of the pixel architecture by implementing certain functions with a memristor.

5.3 L2R Encoder Pixel Architecture

The schematic of the pixel architecture shown in Figure 5.3a consists of a photodiode with a reset transistor, a voltage comparator, a delay stage as feedback and a memristor working as an analogue counter. In the proposed pixel architecture, a L2F converter generates digital pulses at a frequency proportional to the light intensity impinging onto the photodiode (PD). In our implementation, we drive the memristor (M_s) with current pulses, which provide better linearity against a voltage control. The output voltage at



(a) (b)

FIGURE 5.3: Pixel architecture embedding a memristor as an analogue counter. **(a)**, Block diagram of pixel sensor with memristor-based in-pixel analogue counter. **(b)**, Illustration of the principle of multiple-reset pulse-frequency modulation for light-to-resistance encoding during an integration time

node B is therefore converted into a current before driving M_s. Every time the output (B) of the comparator goes high, the memristance is reduced by a certain value depending on the pulse duration which is given by the reset delay of the L2F and by the current amplitude (Iup), which ensures that the memristor is programmed with a proper width and amplitude.

The basic principle of operation is given in Fig. 5.3b. The photodiode is first pre-charged to V_R. After reset, the photodiode voltage starts discharging proportionally with the light intensity. Once the voltage reaches the threshold (V_T) of the comparator, the latter toggles and resets the photodiode, with a certain delay, back to the pre-charged value V_R. This operation repeats at a rate proportional with the light intensity and generates a pulse width equal to the $DELAY$. According with the schematic of Fig. 5.3b, the non-linear decrease in memristance is proportional to the number (N) of pulses generated during the integration time T_i. Hence, light-to-resistance encoding is achieved as a result of the proportionality between the light intensity and the frequency of the programming pulses. At the end of the exposure time, the pixel is read out, shorting the node A to the bit-line $(SEL = H)$. The memristor is connected to the inverting input of OP to form a gain-stage:

$$V_{OUT} = V_{ref2} - \frac{R}{R_M}(V_{ref1} - V_{ref2}) \tag{5.1}$$

V_{OUT} is then converted by a column-level analogue-to-digital converter (ADC). After the readout operation, the memristor is reset back to the R_{OFF} state by means of $RES(Idwn)$ and another exposure time can start. The comparator output pulse frequency is given in Eq. (5.2).

$$f = \frac{1}{T} = \frac{i_{ph} + i_d}{C(V_R - V_{TH})} \tag{5.2}$$

where: i_{ph} is the photo-generated current, i_d is the dark current C is the capacitance at the photodiode node, V_R is the photodiode pre-charge voltage, V_{TH} is the comparator reference voltage. The number of pulses generated during an exposure time Ti is given in Eq. (5.3).

$$N = \frac{T_i}{T} = \frac{T_i(i_{ph} + i_d)}{C(V_R - V_{TH})} \tag{5.3}$$

For each pulse applied to the memristor, its memristance is reduced by $\Delta R_M(t)$; therefore, the expression of the memristance after an exposure time is given in Eq. (5.5).

$$\Delta R_M(t) = R_{M(final)} - R_{M(init)} \tag{5.4}$$

$$R_M = R_{OFF} - N(\Delta R_M(t)) \tag{5.5}$$

Since each variation step in R_M is affected by both the total charge quantity injected and the current magnitude during injection, the light-to-resistance curve is expectedly non-linear. This should be carefully accounted for when deciding the current magnitude, write-in times, and output digital quantization, depending on the type of memristor used – and the typical spreads in its characteristic curves. Read-out should not perturb R_M significantly, that is, the charge injected during the read-out phase (where t_{SEL} is the read-out time during which SEL is high) shouldn't exceed the threshold voltage of the memristor. In our configuration, this is guaranteed by Operational amplifier's (Opamp) virtual ground V_{ref2}. At the gain-stage, the potential difference (PD) across the memristor (V_{ref1}, V_{ref2}) is small enough to avoid corrupting the intended resistance R_M. The relevant trade-off, here, is between a low (PD), which could impair the precision of the output voltage, and a short t_{SEL}, which requires faster read-out circuitry.

5.4 Simulation of the Pixel Architecture

In the proposed L2R pixel architecture, the binary counter has been replaced with an analogue counterpart, made of a single memristor. This turns into a smaller pixel pitch, compared with an all-CMOS solution and analogue non-volatile characteristics. The proposed circuit has been designed and simulated in a $3.3V$, $0.35\mu m$ CMOS process, while the memristor behaviour relies on the TiO_2 model.

The circuit has been validated with the simulation of different light intensities represented by the photo-generated current [117]. The range of The functionality of the pixel architecture has been designed and simulated with the following parameters: $V_R = 2.3V$, $V_{TH} = 2V$, $C = 100fF$, $Ti = 20ms$, $DELAY = 100ns$ and the the following memristor parameters: R_{OFF}/R_{ON}= 1000, $\mu_\nu = 10^{-9}cm^2S^{-1}V^{-1}$, $D = 10nm$.

Figure. 5.4 is a plot of the pixel digital output (D_{OUT}) versus the photo-generated current. The results obtained with a reset delay of $100ns$ are as expected, we observe good linearity for low light intensity but as the pulse frequency increases, there is less linearity. This phenomenon is due to the reset delay being independent to the light intensity. Using a short reset delay will correct this phenomenon. The circuit is flexible and depending on the dynamic range of the memristor, it can be adjusted to suit any application. The issue of having memristors with different physical properties on an array of pixel can be easily taken care of due to the fact that we can control the current integrated into the device for each programming pulse applied. This implies fine tune-up of each pixel by varying the amplitude (Iup) or the width of the programming current pulses. An alternative solution involves software post-processing in order to correct the imager Fixed Pattern Noise (FPN) and gain variations of each pixel signal chain. This is usually done through a Look Up Table (LUT) stored into an external memory.

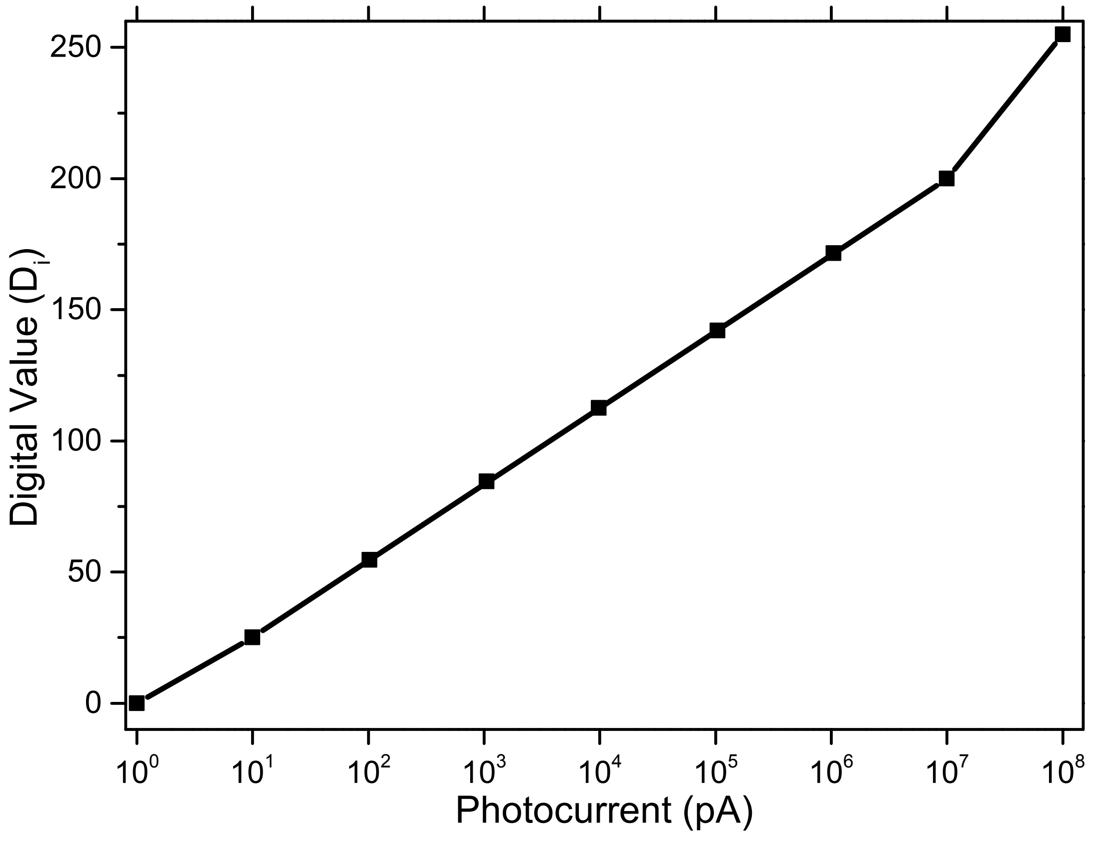


FIGURE 5.4: Digital output versus photo-generated current

Figure. 5.5 is a plot of D_{OUT} versus comparator reference voltage for two different photo-generated current $500pA$ and $20nA$ mimicking low and high illuminating conditions respectively. Under a fixed condition, the memristance is proportional to the voltage reference.

5.5 Conclusion

Presented in the chapter is a pixel architecture which implements a light-to-resistance encoder exploiting the properties of a memristor. A light-to-frequency converter is adopted to drive a memristor with pulses, thus changing its resistance according with the light intensity. In a conventional L2F implementation, a binary counter is needed to store the number of pulses generated within the exposure time (Ti). In the proposed circuit, the binary counter has been replaced with an analogue counterpart, made of a single memristor. This turns into a smaller pixel pitch, compared with an all-CMOS solution and analogue non-volatile characteristics.

In the implemented light-to-resistance converter, each pixel is composed of a memristor acting as an analogue counter during which in an exposure time, the memristor

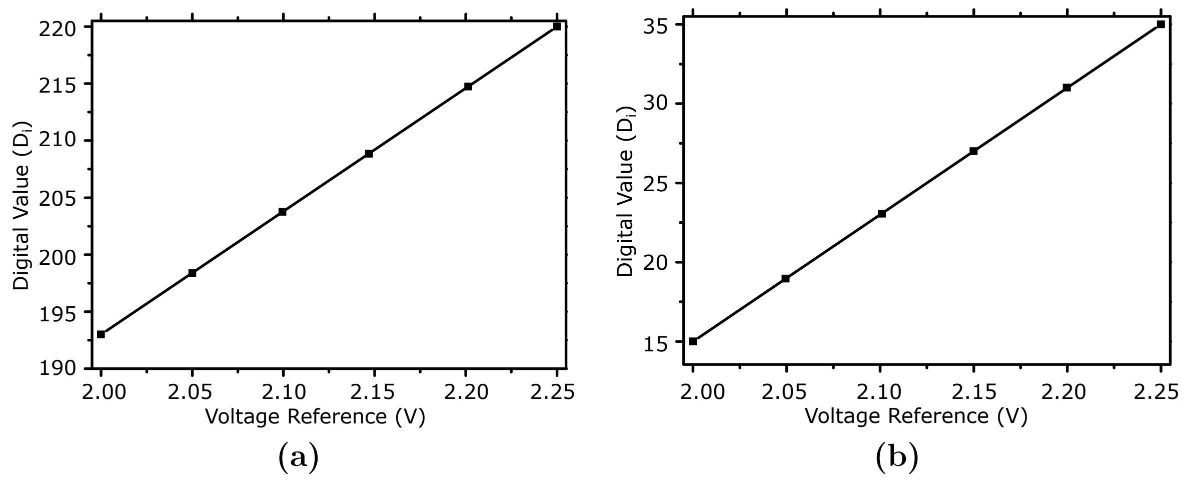


FIGURE 5.5: Digital output versus comparator threshold voltage V_{TH}. **(a)**, Under low illumination for different voltage reference. **(b)**, Under high illumination for different voltage reference.

is programmed by the light-to-frequency converter, thus, light to resistance encoding is achieved. The major advantage of this circuit is the drastic reduction of pixel size and improved pixel performance. One important application of this pixel architecture could be in the field of activity monitoring where some sort of low-pass memory is needed at pixel level for tracking contrast variations as we are going to see in chapter 6. The compactness, programmability and memory effect of the memristor makes it a suitable candidate. This application and the proposed mechanism will also work with the currently available memristors whose characteristics and reproducibility are still to be improved.

The proposed circuit has been simulated in a $3.3V$, $0.35\mu m$ CMOS process, while the memristor behaviour relies on the model presented in Chapter 2.6.

Chapter 6

Memristor Pixel Architecture for Dynamic Background Subtraction

6.1 Introduction

CMOS architectures for vision sensors aimed at processing images in the early-stage to extract salient features from the scene that would otherwise be too costly to achieve through a standard computing approach, have been widely investigated [118, 119, 120, 121, 122, 123]. In this respect, motion detection being the basis for several complex visual tasks, is one of the most important image features. Precise and reliable form of detection of temporal contrast is particularly important in a broad range of applications, including traffic monitoring, human motion capture and video surveillance. The implementation presented in this chapter focuses on temporal contrast, therefore, a precise and reliable detection which should be accurate in shape detection and response to changes in time is highly required. The most widely adopted technique is background subtraction, where an estimation of the background is generated and updated frame by frame [124].

While the analysis of the type of motion and its association to a particular object in the scene is for superior levels of processing, the first clue of an occurring movement in the scene is no doubt the detection of a change in light intensity. Since changes can be expectedly detected at any time in all of the pixels – due to, among other factors, variation of lighting and shadowing, and noise. However potential movement should be considered when a pixel changes quickly enough with respect to its past history. Thus, a sort of low-pass memory must be implemented at the pixel level, tracking contrast variations and generating an alert when the pixel behaviour changes.

Core of the processing is the pixel, containing a light-to-frequency converter outputting digital pulses, proportional to the intensity of light, which are in turn applied to a memristor, changing its resistance accordingly. Two additional memristors are used to store the dynamic boundaries, outside which the behaviour of the photo-generated signal is recognised to be anomalous, i.e., unexpectedly fast changing. The main advantages of using memristors over all-CMOS implementations are a smaller pixel pitch and non-volatility, the latter allowing the image background to be modelled with programmable time constants.

Several memristor-based image processing architectures and their applications in computation have been published in the literature. Most of the applications in computing fundamentally exploit the memristor by combining its retentive memory properties in image processing [125, 126, 127, 128, 129, 130]. Unfortunately not much research has been done in the area of analogue processing with memristors combining them with a photo-detector frontend.

In comparison to this work, in [65], a memristor based unit cell design for a readout integrated circuit (ROIC) is proposed having a photodetector frontend. The proposed circuit similar to this work, operate in 3 different modes: (i) integration, where the photodetectors are exposed to light in order to generate a current proportional to their exposed level of light with the photogenerated current changing the state of the memristor, (ii) read, the memristor state is read out, and finally (iii) erase, the memristor state is reset back to a predefined state. One major disadvantage is in the way the memristor is coupled with the optical sensor. The approach employed is based on direct programming of the memristor with a photogenerated current. As stated in the introduction of the previous Chapter, this kind of technique could have unrealistic power requirements to properly program the memristor, thereby, severely hampering the overall dynamic range of the system.

Another architecture was proposed by Eshraghian *et al.* to integrated photo detector receptor With Memristor Memory cell to perform simultaneous image capture and image matching [131]. This technique is very similar to this work in the sense that the pixel output is a pulse width-modulated signal that transfers illumination levels directly to resistance level. Unlike this work, the memristor is decoupled from the optical sensor by employing a Time-to-Voltage (TTV) converter to produce a digitised signal representing the incident light intensity which is converted to a voltage before being written to memory. Meanwhile in this work, the incident light is converted to a train of current pulses which is more effect and easier to program memristors in a non-linear way.

From an architectural point of view, in our approach, we decouple the optical sensor from the memristor using a multiple-reset PFM technique. This way, the memristor is tuned accordingly by varying the amount of current flowing through it, thereby yielding a higher dynamic range.

This Chapter is organised into three more sections as follows. In the next section, the operating principle of the algorithm for background subtraction Section 6.2. Section 6.3 describes the implementation of such algorithm on pixel level. Simulation results of the pixel architecture are presented in Section 6.4. Finally, I conclude in Section 6.5 with

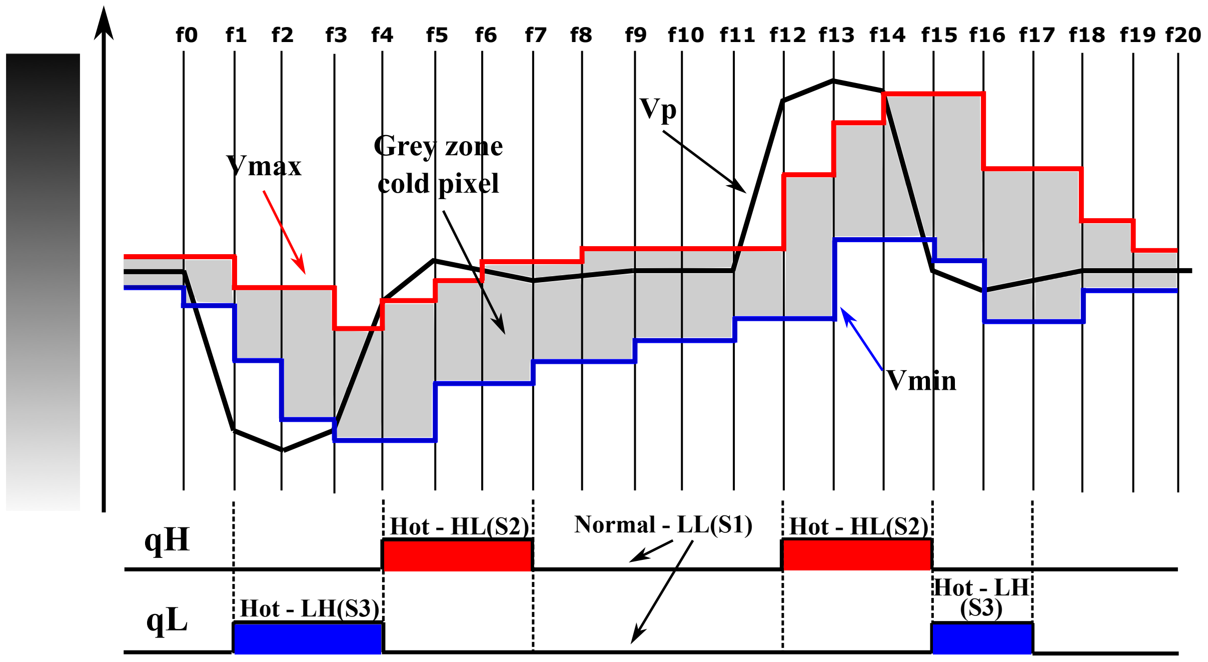


FIGURE 6.1: Algorithm for dynamic background subtraction executed at pixel-level.

some discussion of some practical aspects and some possible limitations related to the proposed pixel architecture.

6.2 Pixel Operating Principle

Before discussing the pixel architecture, it is worth describing the algorithm for adaptive background subtraction [132] working at pixel-level. Let us consider a single pixel of an image sensor, encoding one specific point of the scene. The pixel acquires the light intensity at a frame rate fps and converts it into a voltage $V_S(nT)$, where $T = 1/fps$ is the pixel sampling time and the integer n indicates the frame number. During the sensor operation, the light impinging onto the pixel can change with different dynamics, depending on either the type of motion or the ambient light variations in the scene. By monitoring the signal dynamics and amplitude, each pixel is asked to detect whether anything potentially anomalous may be occurring in the scene.

In order to do this, the image background (B) has to be subtracted from the current image (F_i) and the resulting difference is to be compared with a proper threshold (TH):

$$|F_i - B| > TH. \tag{6.1}$$

Pixels reaching the threshold are labelled as *hot-pixels*, i.e. pixels detecting a potential alert in the scene; differently they are recognized as *cold-pixels*. Taking into account that background is subject to changes, different models can be adopted having different complexity, according to the specific applications:

- Frame difference: the background is assumed to be equal to the past image ($B = F_{i-1}$). It is a simple and straightforward technique although it is not very reliable. In fact, it is very sensitive to the threshold (TH), to the frame rate and to the object speed:
$$|F_i - F_{i-1}| > TH; \tag{6.2}$$

- Simple Moving Average: takes into account the background variations over a certain number of frames. This technique requires n frame buffers, which makes it very memory- and computationally-consuming:
$$B_i = \frac{1}{n}\sum_{j=0}^{n-1} F_{i-j}; \tag{6.3}$$

- Exponential Moving Average: the background consists of an infinite impulse response filter that applies exponentially decreasing weighting factors ($0 < \alpha < 1$):
$$B_i = \alpha F_i + (1-\alpha)B_{i-1}. \tag{6.4}$$
As main advantage, it does not require additional memory and the filter can be tuned by changing the value of the learning rate α.

Considering the hardware implementation aspects and the robustness of the above techniques, we modelled the background with the Exponential Moving Average using two threshold voltages instead of only one as in (6.1). The thresholds define the voltage range inside which the signal is allowed to change safely (*cold-pixel*). Outside this range (above or below), the signal is considered anomalous (*hot-pixel*) and potentially marking an alert. Figure 6.1 shows how the algorithm works. The example refers to a single pixel operation over 20 frames. The black curve indicates the signal voltage V_S acquired by the pixel. The red (V_{max}) and blue (V_{min}) waveforms represent the two threshold voltages delimiting the grey-zone, inside which the signal can freely change without detecting any alert. The two thresholds can be generated by low-pass filtering the signal V_S. Each filter switches between two time constants ($\tau_H < \tau_L$), depending on the following conditions:

$$V_{max} = \frac{1}{1+s\tau_H}V_S\ if\ (V_S > V_{max}) \Rightarrow hot - pixel\ H \tag{6.5}$$

$$V_{max} = \frac{1}{1+s\tau_L}V_S\ if\ (V_S \leq V_{max}) \Rightarrow cold - pixel\ H \tag{6.6}$$

$$V_{min} = \frac{1}{1 + s\tau_H} V_S \; if \; (V_S < V_{min}) \Rightarrow hot - pixel \; L \tag{6.7}$$

$$V_{min} = \frac{1}{1 + s\tau_L} V_S \; if \; (V_S \geq V_{min}) \Rightarrow cold - pixel \; L \tag{6.8}$$

where equations (6.5) and (6.7) represent the *hot-pixel* conditions for V_{max} and V_{min} respectively, while equations (6.6) and (6.8) refer to the *cold pixel* conditions. The behaviour of the two thresholds defines a grey-zone that changes over time according with the signal dynamics. The grey-zone represents the range of voltage inside which V_S can move without any anomalous condition being detected. For example, if V_S changes suddenly, from bright to dark, (e.g. the leaves of a tree moving in the wind), trespassing the top boundary of the grey-zone (V_{max}), a *hot-pixel* is generated. Therefore, while V_{max} tries to quickly reach V_S, V_{min} does the same, but more slowly. Here, the grey-zone rapidly grows larger. After a certain number of frames, both thresholds clamp V_S, completely absorbing the signal variation, so that no *hot-pixel* are generated any further. Here, the grey-zone returns thin, retrieving the maximum pixel sensitivity.

6.3 Pixel Implementation

Equations (6.5)-(6.8) can be implemented using two ideal low-pass filters (LPF), like those depicted in Fig. 6.2, where LPF1 implements equations (6.5) and (6.6), while LPF2 realises (6.7) and (6.8) respectively. Assuming ideal diodes D1-D4 (with no voltage drop) and $R_L > R_H$, each block realises two different first-order RC low-pass filters, with $\tau_H = R_H C$ and $\tau_L = R_L C$, where $R_H >> R_L$. Monitoring events in the scene requires filters with large time constants, that can range from seconds to tens of seconds. This means that R and C should be the orders of magnitude of $M\Omega$ and μF respectively. Each block (LPF1, LPF2) has to be able to automatically switch from one time constant to the other, thus accomplishing the behaviour required by the adaptive algorithm. In order to have an efficient vision sensor architecture, this kind of double-side peak-detection and filtering operation has to be done locally, close to the pixel. In this regard, few custom CMOS implementations have been already proposed [124, 133, 134], using switched-capacitor techniques to emulate the two filters inside each pixel. However, few basic drawbacks arise from this approaches: (a) the analogue memories, needed to store the two threshold voltages, do not have such a long retention time as required by the application; (b) the silicon area, occupied by the capacitors, working as analogue storage elements, is large, hampering the pixel from having a small pitch. In order to address these main issues, we have considered the possibility of partially replacing the filter with one memristive device, exploiting its non-volatile properties and nano-scale dimensions.

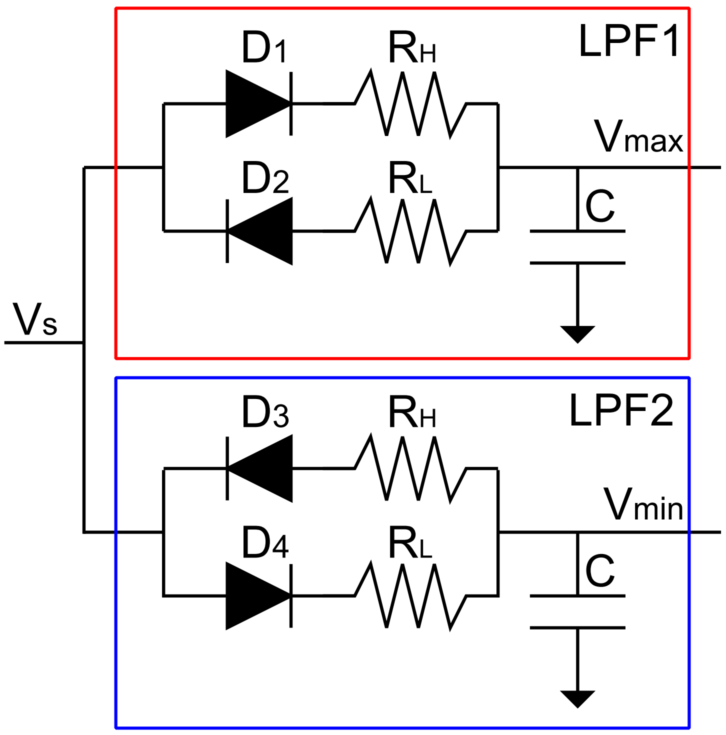


FIGURE 6.2: Two first-order low-pass filters generating the two thresholds of the algorithm for dynamic background subtraction executed at pixel-level given in Fig. 6.1.

Moreover, the value of its resistance can be easily controlled through digital pulses (either voltage or current). According with the operating principle of Fig. 6.1, the proposed pixel relies on three memristive devices (M_S, M_{max} and M_{min}), storing three resistance values which are proportional to the signals V_S and the two thresholds V_{max} and V_{min} respectively. The schematic of the proposed pixel is shown in Fig. 6.3. A Light-To-Frequency (L2F) block converts the light intensity, impinging onto the pixel, into digital pulses, with fixed pulse width (ΔT) and a frequency which is linearly proportional with the photo-generated current (I_{ph}) [17]. During the pixel reset, M_S is set to its highest resistance ($M_{S_L} = R_{OFF}$), ready to be programmed through the L2F digital pulses.

6.3.1 Exposure Time

During the exposure time (T_i), the L2F generates a train of current pulses with amplitude I_1, width ΔT and frequency proportional to the light intensity, feeding M_S as shown in Fig. 6.4. The state of the latter is described in terms of a state variable $w(t)$ by the following equation:

$$w_{M_S}(n) = \mu_\nu \frac{R_{ON}}{D}\Big(nI_1\Delta T\Big); \tag{6.9}$$

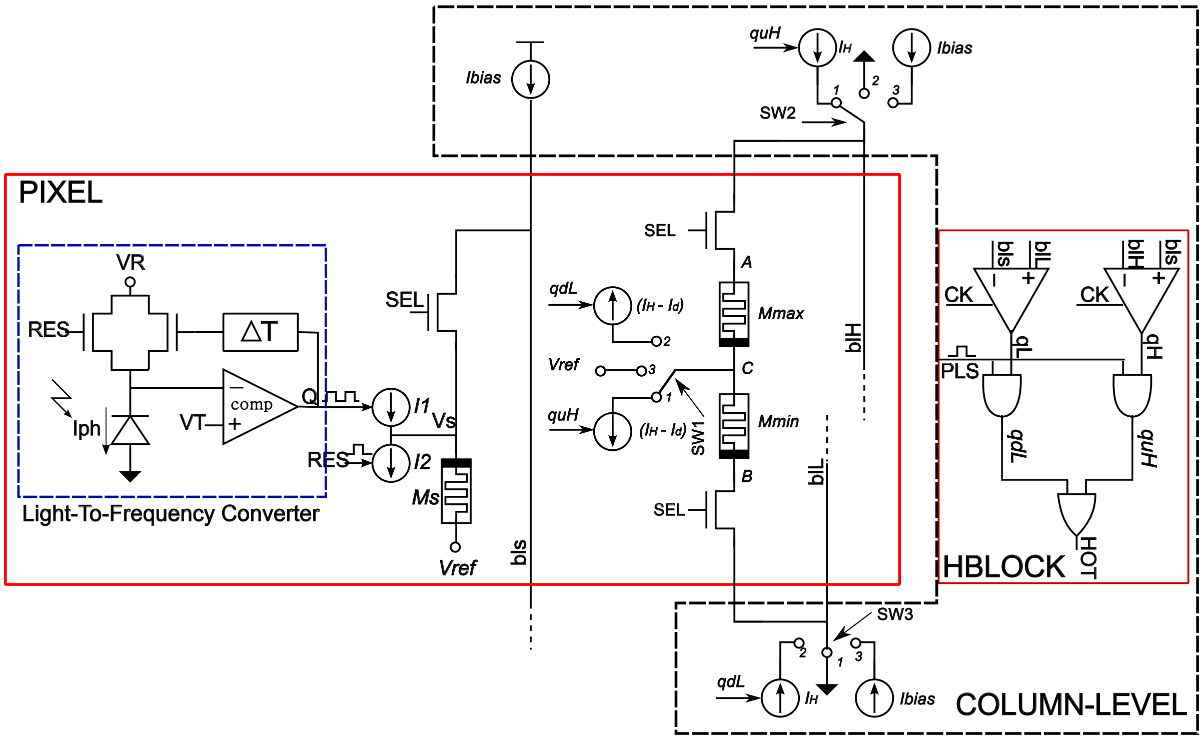


Figure 6.3: Schematic of the proposed pixel embedding three memristors for dynamic background subtraction.

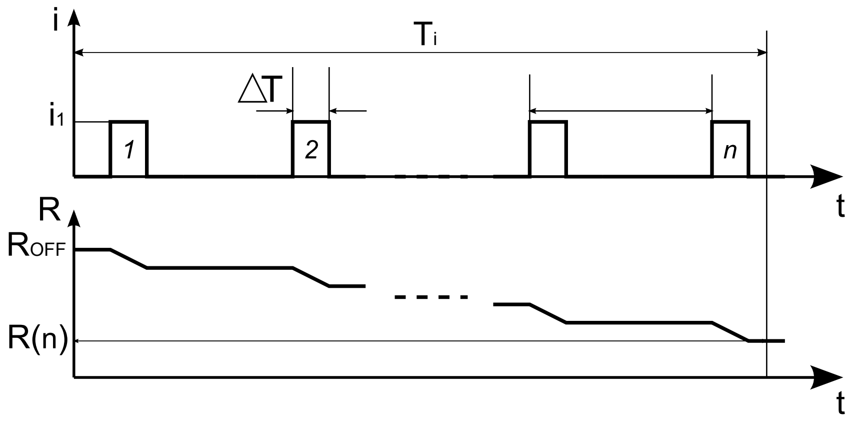


Figure 6.4: Timing diagram of the pixel architecture during the integration time (T_i). The L2F feeds M_S with n digital pulses of current I_1, thus changing the memristance from R_{OFF} to $R(n)$.

TABLE 6.1: Four pixel states which are coded by qH and qL.

S	Constraints	qL	qH	HOT
S1	$(V_{blL} < V_{blS} < V_{blH})$	L	L	L
S2	$(V_{blL} < V_{blS})$ & $(V_{blS} \geq V_{blH})$	L	H	H
S3	$(V_{blL} \geq V_{blS})$ & $(V_{blS} < V_{blH})$	H	L	H
S4	$(V_{blL} \geq V_{blS})$ & $(V_{blS} \geq V_{blH})$	H	H	H

where, R_{ON} is the low resistance, D is the length of the device, μ_ν is the dopant mobility, and n is the number of pulses generated by the L2F during the exposure time. The final value of the resistance for n applied pulses is:

$$R(n) = \Big(R_{ON}\frac{w_{M_S}(n)}{D} + R_{OFF}\Big(1 - \frac{w_{M_S}(n)}{D}\Big)\Big). \tag{6.10}$$

6.3.2 Readout and Hot-Pixel Detection

After the exposure time, M_S is compared with M_{max} and M_{min}. Hence, the pixel is connected to the bit-lines (SEL=H) and the three memristors are biased with the same current I_{bias} in order to force their voltages on the three bit-lines (*blS, blH, blL*). The voltage on *blS* is then compared with those on *blH* and *blL* respectively in order to detect potential *hot-pixel* conditions. SW1, SW2 and SW3 are all set to the position "3", thus shorting the common node C to V_{ref} and biasing M_{max} and M_{min} with the bias current I_{bias}. In the end, the following voltage drops are obtained:

$$V_{blS} = M_S \cdot I_{bias} + V_{ref}; \tag{6.11}$$

$$V_{blL} = M_{min} \cdot I_{bias} + V_{ref}; \tag{6.12}$$

$$V_{blH} = M_{max} \cdot I_{bias} + V_{ref}. \tag{6.13}$$

Hot-pixel detection is accomplished by means of two clocked comparators, placed outside the pixel, at the column-level (HBLOCK). Table 6.1 shows the digital output signals for the different states of the pixel.

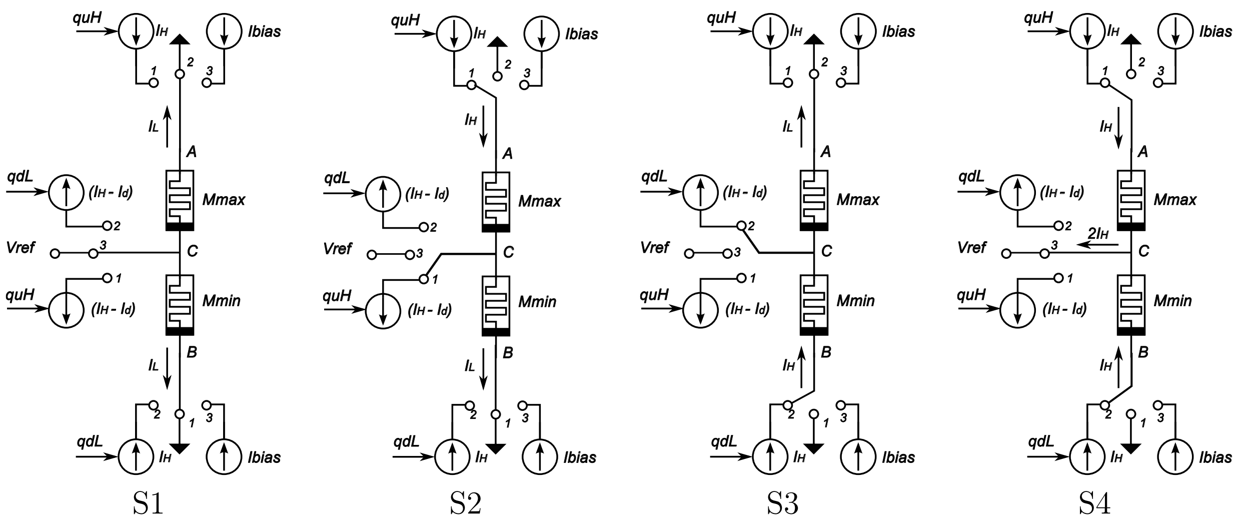


FIGURE 6.5: Memristors control related to the four different pixel conditions (max, min): LL, HL, LH, HH.

6.3.3 Threshold Update

Fig. 6.5 shows how the two memristors (M_{max} and M_{min}) are controlled depending on each of the four states of the pixel: S1, S2, S3 and S4. In order to implement a filter with a fast time constant τ_H, the memristor is pulsed with a current I_{PH} with a pulse width ΔT_P generated by the signal PLS, feeding HBLOCK. Therefore, the time constant of the filter is set by sizing the charge $q_{HOT} = I_H \cdot \Delta T_P$ injected into the device and by "m", the number of applied pulses. On the other side, the slow filter, taking care of the *cold-pixel* conditions, handles a charge $q_{COLD} = I_{PL} \cdot \Delta T_P$, with $q_{COLD} < q_{HOT}$. This means that, for the same number of charge packets provided to the device, the memristance changes less in the case of q_{COLD}. By adopting the circuital configuration shown in Fig. 6.5, the update process for the two thresholds can be done at the same time. Let us assume that the pixel is in the status S2; thus M_{max} is fed with $q_{HOT} = I_H \cdot \Delta T_P$, while M_{min} is fed with $q_{COLD} = (I_H - I_d) \cdot \Delta T_P$, with $I_d = I_H - I_L$. In this case, both thresholds (V_{max} and V_{min}) approach the current signal V_S but at different speed (V_{max} increases quickly, V_{min} increases slowly).

6.4 Simulation of the Pixel Architecture

The CMOS part of the pixel architecture was designed in a 3.3V, 0.35μm CMOS process. The algorithm for adaptive background subtraction has been modelled and simulated using MATLAB [124, 134]. The pixel architecture, shown in Fig. 6.3, was simulated in the four different statuses. The expected behaviour, described in Fig. 6.6, was reproduced electrically through Spectre. The memristor was modelled using Verilog-A according to

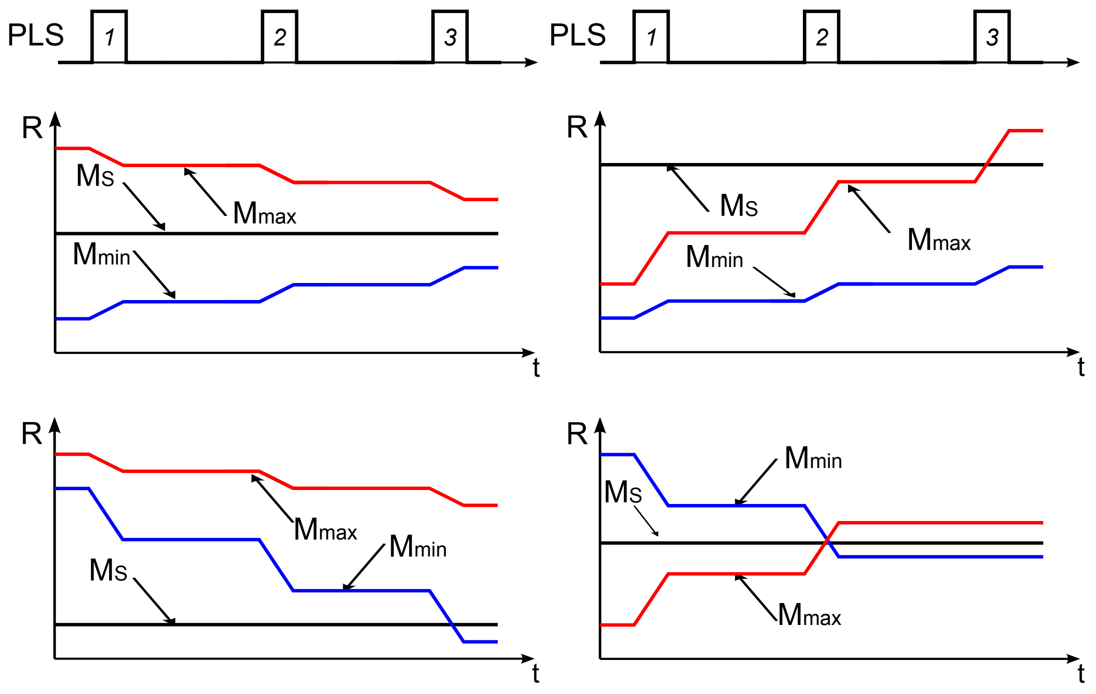


FIGURE 6.6: Expected behaviour of the two thresholds, expressed in memristance values (M_{max}, M_{min}), after each update pulse (PLS) for each of the four pixel status: S1, S2, S3, S4 of Table 6.1. While S1, S2 and S3 are states which typically occur during the sensor operation, S4 takes place at the sensor calibration phase and is generated on purpose.

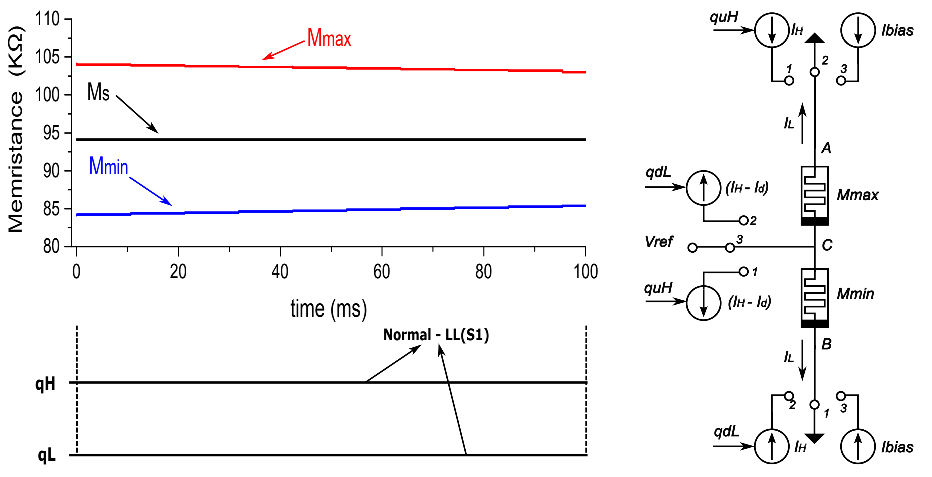


FIGURE 6.7: Electrical simulation of the pixel condition LL of the proposed pixel architecture in Fig. 6.3 embedding three memristors for dynamic background subtraction. **a**, Simulation of the control status S1 for a normal pixel operating condition. The upper and lower threshold try to absorb the temporal variation. Also shown is the hot-pixel (HOT) binary signal for each pixel status which is reasonably low in this case as no alert has been detected. **b**, Memristors control related to the pixel conditions (max, min): LL

the equations extracted from [6]. The memristance value range ($R_{ON} = 200\Omega$, $R_{OFF} = 200K\Omega$) was adopted to cover for a large dynamic range. An exposure time T_i of $10ms$ is chosen to be short enough so as not to drive M_s to the ON state during light to resistance encoding for high frequencies and large enough to detect the low light intensity.

Using the same parameters to validate all the four pixel status, the results are shown in Figure 6.7 – Figure 6.9. M_s is programmed with digital pulses generated by the L2F block during an exposure time (Ti) of 10 ms. The final value of M_s is then compared to $Mmax$ and $Mmin$ before it is reset back to initial R_{OFF} state. $Mmax$ and $Mmin$ are then adjusted accordingly, depending on the pixel condition. Fig. 6.7 is the result of the pixel status S1 of Fig. 6.5. Here the pixel is in the normal working condition and as shown in the plot, $Mmax$ and $Mmin$ try to slowly converge towards Ms in order to absorb the variation. We also notice that the binary signal of the $hot - pixel$ is always low.

For the other pixel statuses: S2, S3 and S4 given in Fig. 6.5 where a HOT pixel has been indicated, we observe two scenarios; one in which, a pixel in a HOT state changes directly to the normal pixel condition; the other in which a pixel in a HOT state (typically S4) will have to transit into another HOT pixel condition (S2 or S3) before going to the normal state (S1).

6.4.1 Direct Transition from one Hot-Pixel state to Cold-Pixel

For pixel statuses S2 and S3, typically, a direct transition to the normal pixel condition occurs. As observed in Fig. 6.8a, for the S2 status, $Mmax$ and $Mmin$ tries to reach Ms at different time constants with $Mmax$ increasing much faster until the pixel is restored to the normal working condition. As shown in Fig. 6.8b, the reverse is the case when the pixel is in the S3 status, having $Mmin$ decreasing faster than $Mmax$ to try and reach Ms. Fig. 6.9 is the simulation result corresponding to the S4 status. In this case, $Mmax$ is increased at the same rate with which $Mmin$ is reduced until the pixel returns to the normal condition. In all cases, the rate of change is controlled by the amplitude of the current pulses applied.

6.4.2 Transition from one Hot-Pixel state to another Hot-Pixel state and then to Cold-Pixel

Although S4 is a typical forbidden status, where an hot-pixel occurs at the same time against both thresholds, $Mmax$ and $Mmin$, it usually takes place at system power-up, during the calibration phase. In this case, the sensor takes still images and the status S4 is set on purpose while the algorithm tries to quickly converge the two thresholds to the cold-pixel conditions. This phase may take several frames until all the pixel reach

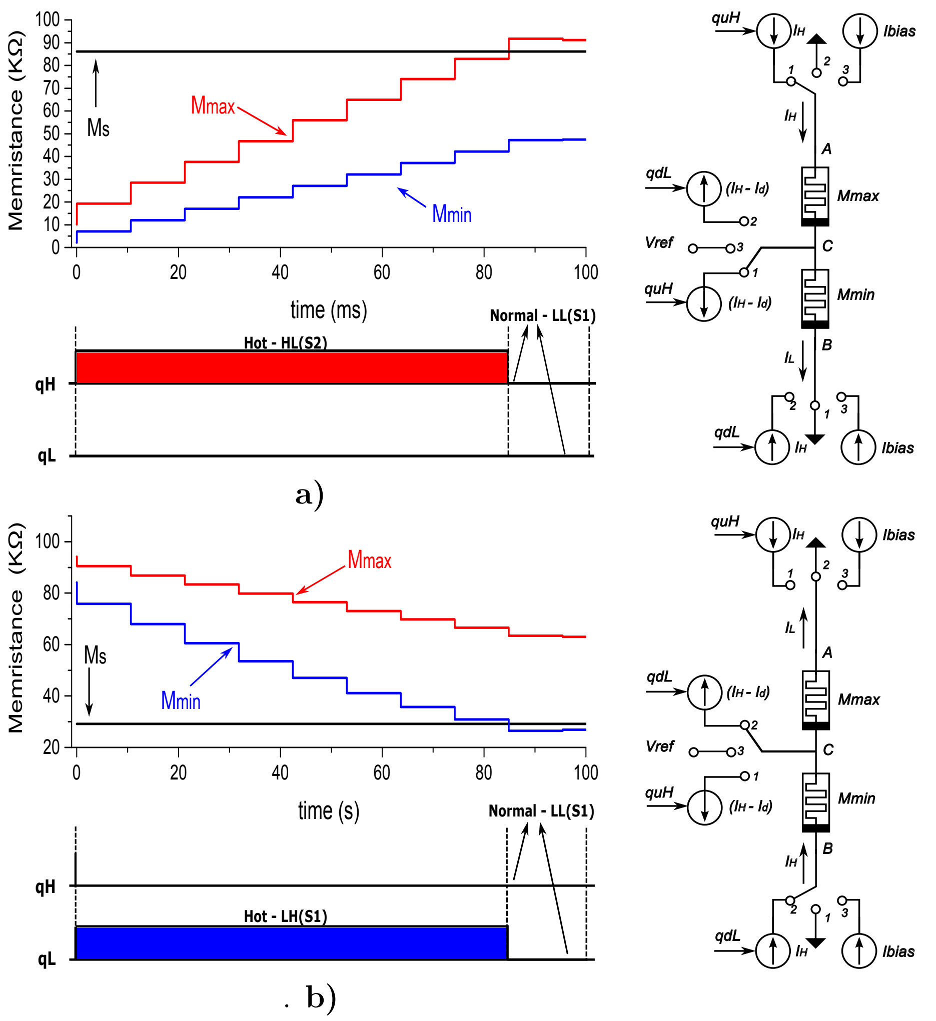


FIGURE 6.8: Electrical simulation of the pixel conditions HL and LH of the proposed pixel architecture in Fig. 6.3 embedding three memristors for dynamic background subtraction. **a**, Simulation of the control status S2 for an anomalous pixel operating condition for a direct transition from HOT to COLD. The upper threshold changes faster then the lower threshold try to absorb the temporal variation. **b**, Simulation of the control status S3 for an anomalous pixel operating condition for a direct transition from HOT to COLD. In this configuration, the reverse operation of S2 is implemented. Also shown is the hot-pixel (HOT) binary signal for each pixel status. The red bar indicates anomalous event ($hot - pixel$ detection) relating to the upper threshold $Vmax$ determined by Mmax while the blue bar indicates anomalous event relating to the lower threshold $Vmin$ determined by Mmin as indicated in the working principle of the algorithm given in Fig. 6.1.

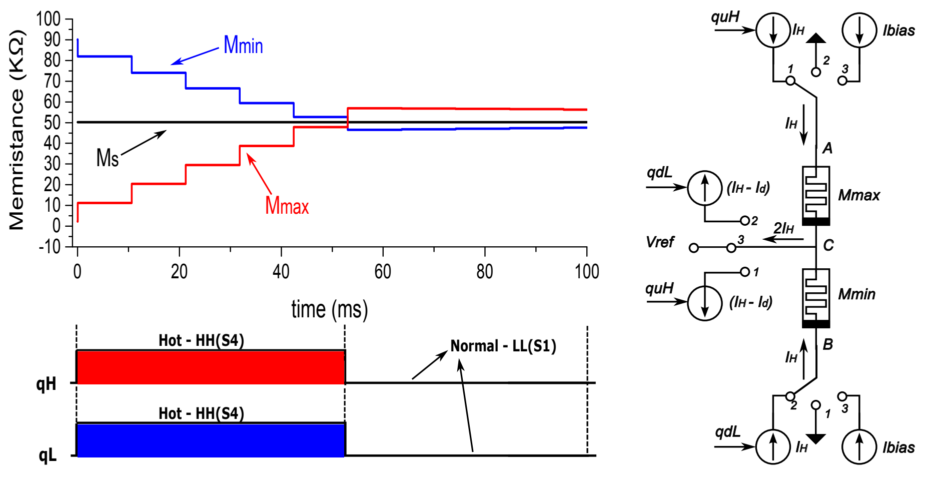


FIGURE 6.9: Electrical simulation for the pixel condition HH of the proposed pixel architecture in Fig. 6.3 embedding three memristors for dynamic background subtraction. Simulation of the control status S4 for an anomalous pixel operating condition for a direct transition from HOT to COLD. Both thresholds are programmed with at the same time constant and as well make the transition from HOT to COLD at the same time.

the cold state. During S4, hot-pixel are not considered as potential alerts. In Fig. 6.10a, the upper bound $Mmax$ stabilizes before the lower bound $Mmin$, leading to a transition from S4 to S2 and then S1. Fig. 6.10b is the output of the situation whereby $Mmin$ stabilizes before $Mmax$; a transition from S4 to S3 and then S1 is observed here.

6.5 Discussion and Conclusion

In discussing some practical aspects and some possible limitations of the proposed pixel architecture, I would like to address first of all, the issue of device-to-device variation with respect to the memristors used during L2R encoding. With the proposed algorithm for dynamic background subtraction, at high frequencies, when the frequency of the programming train of pulses becomes non-linear, inter-pixel mismatch is not significantly affected because each memristor is compared to its previous history and delivers a binary output. In this case the reproducibility of each memristor is what is critical. For device-to-device variations concerning memristors used to store the threshold (V_{max}, V_{min}), this could be accounted for by varying the threshold update current-magnitude and write-in times. Alternatively by software post-processing, e.g. by changing isolated pixels which should detect light intensities unrealistically different from those in their neighbourhood, or convolving the light intensity map with an averaging kernel whose size should be properly chosen according to the expected variability between neighbouring devices, and/or by. The latter solution will likely impair the resolution, to some extent.

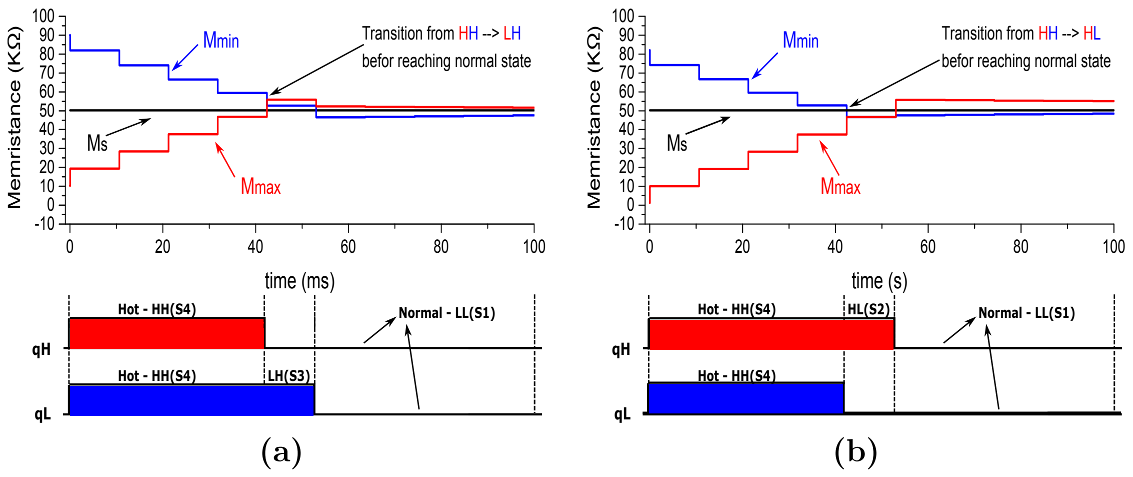


FIGURE 6.10: Electrical simulation for the pixel condition HH corresponding to control status S4 for an intermediary transition state before transition to the normal working condition. **(a)**, Transition from S4 to S2 and then S1. **(b)**, Transition from S4 to S3 and then S1.

However, this could be traded-off with a simpler and smaller pixel schematic – that is, a finer resolution on the hardware side.

Another critical issue could be the possibility of resistance disturbance during read-out where the voltages of the memristors are forced on the bit-lines by biasing them with the same current I_{bias}. Read-out should not perturb the memristors significantly, that is, the charge injected during the read-out phase, $Q_{ro}{\sim}I_{bias}{*}t_{sel}$ (where t_{sel}) is the read-out time during which SEL is high should produce a $\Delta R(t_{sel})$ smaller than ΔR_{ls}, the latter being the minimum resistance variation affecting the least significant bit. The relevant trade-off, here, is between a low I_{bias}, which could impair the precision of the output voltage, and a short t_{sel}, which requires faster read-out circuitry. Moreover, varying I_{bias} generally impacts on the resistance variation, also at constant charge.

In conclusion, this chapter shows how it would be possible to efficiently adopt memristors in combination with CMOS electronics aimed at implementing a vision sensor architecture with efficient distributed processing and memory to perform robust real-time image processing. The present work focuses on adaptive background subtraction as the basic engine for object tracking. Owing to their nanoscale dimensions and non-volatile features, memristors are expected to be the ideal device for a novel parallel computing paradigm, embedding distributed processing and memory capabilities. These characteristics are even more important when on-chip filtering with long time constants or memory capabilities with long retention time are required.

Chapter 7

Memristor-Based Neural Network for Image Processing

7.1 Introduction

Artificial Neural Network (ANN) as defined by Dr. Robert Hecht-Nielsen can be considered as *a computing system made up of a number of simple, highly interconnected processing elements, which process information by their dynamic state response to external inputs* [135]. In other words, ANNs are systems which can be used to associate one or several input(s) with one of several decisions or predictions [136, 137, 138, 139, 140, 141, 142, 143, 144, 145]. Neural networks are typically organised in layers made up of a number of interconnected *node* containing a *transfer function.* The layers could be broadly divided into 3 namely: an input layer, an hidden layer and an output layer. The neural network is fed with patterns through the *input layer*, processing is done in the hidden layer ia a system of weighted *connections* and a decision is outputted via the output layer as shown in Figure D.1 used to associate two *inputs* to two *outputs* [146, 147, 148, 149, 150, 151, 152].

Since the discovery of memristors, interests in neuromorphic architectures have been popular among researchers [153]. The ability of memristors to mimic biological synapses no doubt gives them the potential to vastly improve neuromorphic computing [44, 154, 155, 34, 156]. One major problem of hardware implementation neural networks lies on reconfigurability where once a network has been trained, the weight of each synapse remains fixed and cannot be changed, therefore neural networks are mostly implemented on software. With memristors, we would not encounter this issue, the weights can be stored as a resistance level on the memristor which in turn can be easily reconfigured. In using memristors to store weights, neural networks can be easily implemented on hardware and the network trained by varying the memristor resistance.

The motivation behind this Chapter stems from memristor applications in image processing as we have already seen in previous chapters. As we try to exploit the hybrid network of standard CMOS image processing techniques with memristive devices. In the last three previous Chapters, we have seen how a memristive network can be exploited in this respect. Another area where memristor-based image processing technique could come in handy is in VLSI implementation of neural networks and their applications to signal and image processing, which is an field that has been severely hampered by resource constraints such as power consumption, area, speed, etc., affecting architectural and circuit design solution. In the introduction to this thesis, I based, as part of the motivation for trying to replace existing processing architectures based on CMOS technology with memristor-solutions, I based my arguments that even though CMOS has advanced progressively for several decades against all odds through the so-called Moore's law, the transistor which happens to be the building block of integrated circuit is gradually reaching it's utmost limits in scaling down.

In computation-intensive applications such as image processing, motion detection, pattern formation and recognition, etc., requiring high speed and massive parallel signal processing, the realisation of such powerful and dense networks in an integrated circuit with acceptable size and non-resource hungry by using the commonly encountered elements and conventional CMOS technology becomes increasingly difficult to achieve. On the other hand, a powerful back-end analogue array processor in the form of memristive devices in a crossbar configuration presents the advantages and could help overcome the limitations of an overall implementation with conventional electronic elements.

In this chapter, an implementation of such technique is discussed for a memristor-based neural network targeted to a complex and advanced application such as colour detection for low power Human Machine Interface applications. The implementation is based on smart energy-efficient devices such as a low-power optical embedded system used to capture and process a visual signal in order to extract salient information about the surrounding environment, an approach based on the detection of human skin under various illuminant conditions.

The neural network utilises a skin detection algorithm which works on the rg chromaticity space and classifies any colour signal as skin or non-skin by testing the membership of its rg coordinates to a skin locus [157, 158]. The skin locus is actually a compact region of the rg space including the skin tones of people captured experimentally under different illuminants. For the neural network developed, the skin locus forms part of the datasets acquired from people of different ethnicity by the low-power optical sensor which is an essential element required for effectively training a neural network.

The output of the neural network will tell us if a subject imaged, is skin or not. This neural network output can then be further used in several applications such as colour detection for low power human machine interface applications which is a promising tool for energy-efficient, touchless control of machines. Figure 7.1 shows some typical application of gesture recognition for mobile phone applications. For further reading on

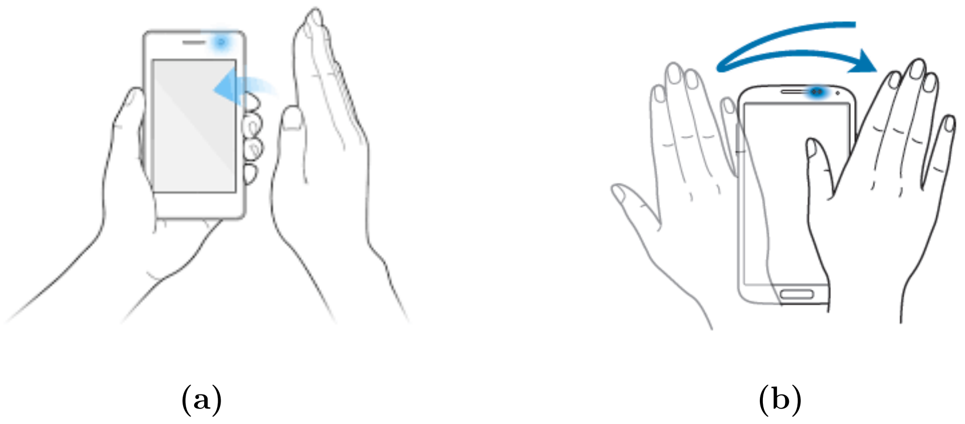


FIGURE 7.1: Example of applications using Hand-gesture technology. **(a)**, an example of hand-gesture recognition for mobile swiping function. **(b)**, Hand gesture recognition using the built-in face-camera.

gesture recognition, refer the following references which this work is based on [159, 160, 161, 162, 163, 164, 165, 166].

This Chapter is further divided into five Sections as follows. In the next Section, I will describe the algorithm in which the colour classification is based on. In Section 7.3, a description the overall architecture of the colour sensor used to acquire the data that was used to train the network is given. Section 7.4 presents the proposed memristor-based neural network architecture. In Section 7.5, the neural network is implemented using Matlab neural network tool to train and validate the network. Finally, I conclude with some discussions in Section 7.6

7.2 Skin Classification Algorithm

Colour being a powerful discriminative and computationally fast feature, plays an important role in discriminating visual appearances of an object. The drawback of its usage is the fact that it is often limited by its strong sensitivity to the spectral profile of the light illuminating the scene. Therefore some kind of colour correction technique is needed for a reliable skin detection and gesture recognition algorithm which is immune to ambient light variations.

The skin classification algorithm is modelled from experimentally acquired data of skin tones in the rg chromaticity space of individuals of different ethnicity under several illuminating conditions by an RGB colour sensor. The acquired Skin tones form a compact set which we refer to as skin locus which allows us to perform binary classification. Since the skin locus has to be modelled with the rg chromaticity, the RGB signal would have to be converted to the normalised rg chromaticity.

RGB conversion to the normalised colour space (r,g) chromaticity is achieved by the following equations:

$$r = \frac{R}{R+G+B}; \tag{7.1}$$

$$g = \frac{G}{R+G+B}. \tag{7.2}$$

By definition, the chromaticity coordinates r and g range over the interval [0,1]. The quality of *r,g* chromaticity coordinates to discard luminance information, thereby ensuring that the invariance against variation of the light intensity due to e.g., the shadowing or the change of camera distance from light sources make them suitable in detection and recognition processing algorithms [167]. This makes the chromaticity space a suitable colour space for skin tone detection used in several applications like facial recognition, hand gesture recognition/detection, etc.

Several algorithms discarding the chromatic dominant of the light have been developed in the past [168], but most of them work on the RGB colour space and often not in real-time [169, 170, 171, 172, 173, 174, 175]. The research carried out in this chapter aims to implement an efficient and fast colour correction algorithm working on the *(r,g)* space.

A colour-based skin detection void of any kind of distortion and which is invariant of the surrounding illumination is required in order to effectively carry out out gesture recognition/detection. There are two kinds of colour distortions which could occur due to a change of illumination. The first distortion could be due to variation in intensity of the illumination caused by shadows or by simply the distance from the illuminant. Whereas, the second colour distortion could be due to a change in the chromatic dominance of the illuminant. In the proposed colour detection system studied treated in this chapter, illuminant invariance is achieved by modelling the skin tones in the *r,g* chromaticity space. The r and g coordinates at a given pixel x are obtained based on Equations 7.1 and 7.2, where R,G,B are the colour channel intensities at x. This discarding of luminance makes the chromaticity invariant to changes in light intensity.

The second colour distortion which is as a result of variations in the illuminant in colour temperature, results in a change of chromatic dominance of the illuminant. The pixel-based colour classification carried out in this work relies on a compact 2D region in the (r, g) space called skin locus [176, 177]. The skin locus compiled from the the chromaticity of several skin tones under several Planck's illuminants. Therefore, the skin locus gives a good representation of skin tones under different illuminanting conditions such as daylight, direct sunlight, candle lights and several fluorescent lights, etc.

The skin locus could therefore be used as an illuminant invariant filter for identifying

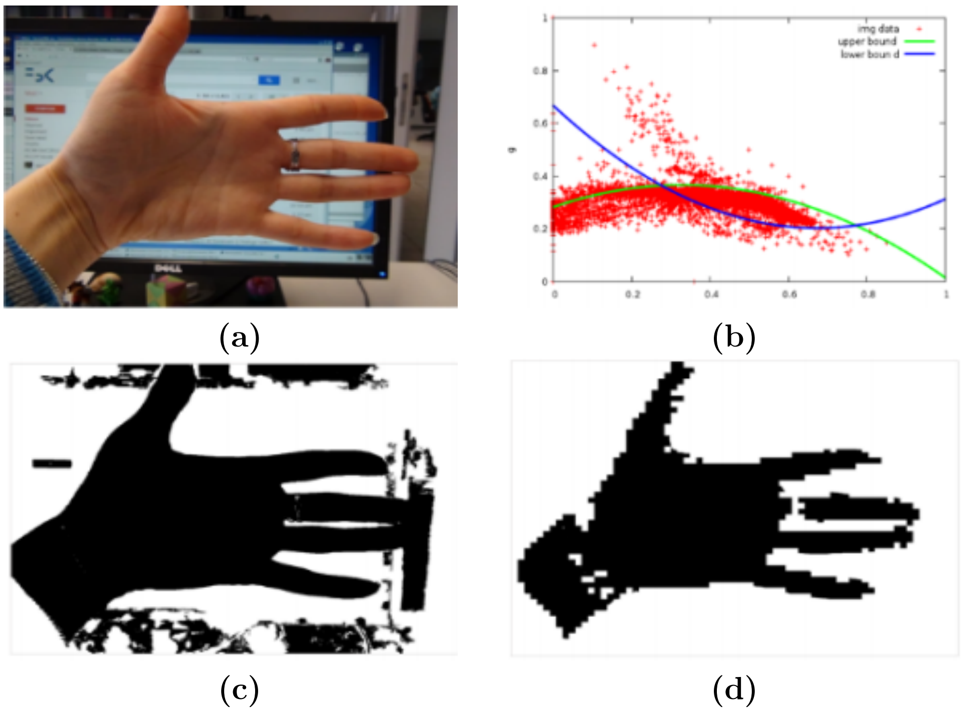


FIGURE 7.2: Illustration of Skin Locus False Positives Detection. **(a)** An Image, **(b)** The skin locus of the user (bounded by the blue and green curves), and the (r,g) coordinates of the image pixels (in red). **(c)** the regions labelled as *skin* by the sensor using the skin locus (simulated data). **(d)** Post processing refinement.

image pixels which are more likely to represent skin [178, 179, 157]. (r, g) coordinates falling outside the skin locus are regarded as "non skin" thus allowing the restriction of the image area possibly indicating skin. This approach is not perfect in the sense that even though the skin locus is sufficient for a satisfactory detection of skin regions, many false positives are often present (see Figure 7.2).

Based on dataset experimentally acquired with the colour sensor on various illuminating conditions, the MATLAB implementation of the overall CNN together with some experimental validation will be presented in subsequent Section.

Firstly, a compact set of the skin tones in (r,g) chromaticity space is modelled from skin tone samples of persons of different ethnicity under different illuminant conditions. This compact set of skin tones are used to create a skin locus which is used for the binary skin classification. The skin locus can be determined by approximating a set of quadratic polynomials d and u. The quadratic polynomials d and u specify the boundaries of the skin locus such that:

$$S = (r, g) \in [0, 1]x[0, 1] : d(r) \leq g \leq u(r) \tag{7.3}$$

If the rg chromaticity of a certain point of the observed scene falls into the skin

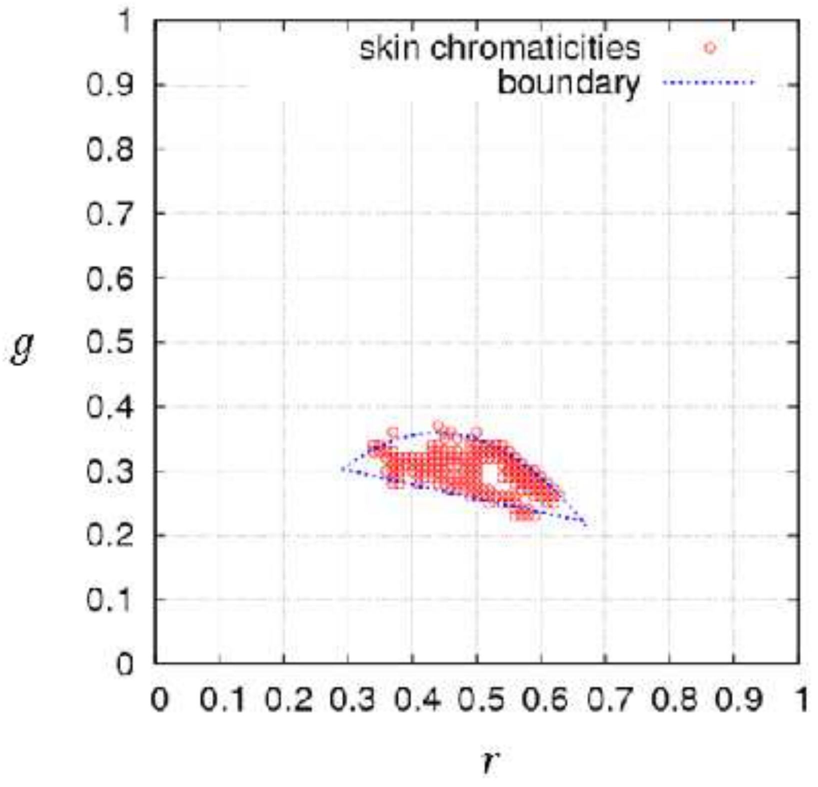


FIGURE 7.3: The skin locus acquired by the RGB colour Sensor. (This image was produced by M. Lecca *et al*. and it is reproduced with permission from M. Lecca [180].)

locus, i.e. d(r) = g = u(r), then that point probably represents skin. According to this principle, the skin locus works as an illuminant-invariant filter and provides a binary classification (skin/non-skin) of an input colour signal. The computation of a skin locus consists of these steps:

- acquisition of human skin tones under different illuminating conditions;
- data filtering, necessary to remove possible noise from the acquired data;
- estimation of the polynomials d and u from the data selected at the previous step.

The skin locus of the RGB colour Sensor whose boundaries have been approximated by the two polynomials given in Equations 7.4 and 7.5 is plotted in Figure 7.3

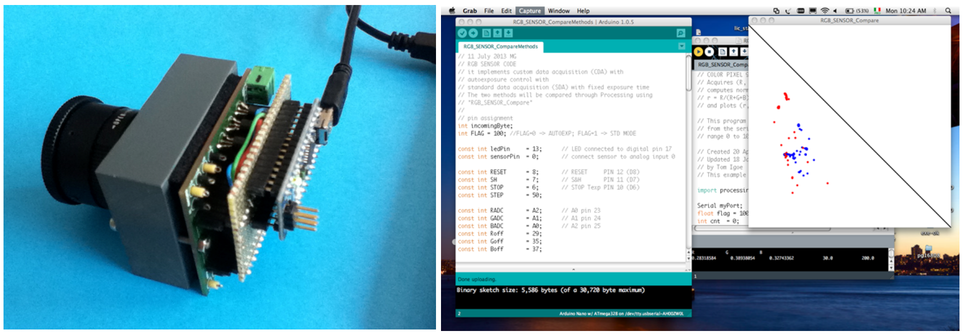
$$u(r) = -2.668r^2 + 2.336r - 0.152 \tag{7.4}$$

$$d(r) = 0.027r^2 - 0.242r + 0.372 \tag{7.5}$$

The choice of rg chromaticity colour space over other colour spaces commonly used for representing human skin tones is motivated by three main reasons, listed as follows:

- tones corresponding to human skin imaged under different illuminants, e.g., sunlight, daylight, candles, neon, tungsten lamps, form in the *rg* space a compact two-dimensional set S, called the skin locus. The name skin locus, is derived from its shape, which follows the Planckian locus [181]. The skin locus can therefore be used as a filter for illuminant-invariant skin classification, i.e, pixels whose *rg* points fall out of the skin locus do not represent skin, while those that fall into the skin locus likely represent skin.
- Conversion from the RGB colour space to the rg coordinates can be efficiently implemented in hardware at pixel level using with just a couple of transistors [182, 183].
- *rg* coordinates possess immunity to changes of the light intensity illuminating the observed scene this is because they do not contain information about signal brightness.

In order to have an improved image classification, some sort of post processing is required to refine the skin classification of the regions detected as *"skin"* by the sensor. The envisioned post processing system computes the samples of users' skin (r, g) distribution with the *(r, g)* distributions of sub regions of portions of the image selected as "skin" by the sensor, retaining only the sub regions having a distribution close to that of the users' skin.


Grab File Edit Capture Window Help
RGB_SENSOR_CompareMethods | Arduino 1.0.5
RGB_SENSOR_CompareMethods

```
// 11 July 2013 MG
// RGB SENSOR CODE
// it implements custom data acquisition (CDA) with
// autoexposure control with
// standard data acquisition (SDA) with fixed exposure time
// The two methods will be compared through Processing using
// "RGB_SENSOR_Compare"
//
// pin assignment
int incomingByte;
int FLAG = 100; //FLAG=0 -> AUTOEXP; FLAG=1 -> STD MODE

const int ledPin     = 13;     // LED connected to digital pin 17
const int sensorPin  = 0;      // connect sensor to analog input 0

const int RESET      = 8;      // RESET     PIN 12 (D8)
const int SH         = 7;      // S&H       PIN 11 (D7)
const int STOP       = 6;      // STOP Texp PIN 10 (D6)
const int STEP       = 50;

const int RADC       = A2;     // A0 pin 23
const int GADC       = A1;     // A1 pin 24
const int BADC       = A0;     // A2 pin 25
const int Roff       = 29;
const int Goff       = 35;
const int Boff       = 37;
```

```
Done uploading.
Binary sketch size: 5,586 bytes (of a 30,720 byte maximum)
```

FIGURE 7.4: Prototype of the RGB Colour Sensor

The post processing takes into account possible variation in light colour when matching *(r, g)* distribution of user skin and sub regions of portions of the image selected as "skin" by the sensor. In principle, the illuminant under which the analysed scene has been captured generally differs from the reference illuminant used for acquiring the skin distribution. Therefore, the development of such matching technique invariant to such changes will depend on the development of a model of the chromaticity variations in response to changes of colour temperature. This chapter aims at investigating / implementing colour correction algorithms working on the *(r, g)* chromaticity space. Such corrections of an image taken under a reference illuminant is also important in the comparison of other features used for gesture recognition such as edge / corner detection [184, 160]. Detailed description of the algorithm and application is presented in [180].

The skin locus given in Figure 7.3 was compiled following the algorithm described in Section 7.2. Skin tones of several volunteers under a set of 15 illuminants whose correlated colour temperatures range over [3000°K, 7000°K] were captured by the RGB sensor. These captured skin tones as well as non skin tones captured under the same conditions make up the dataset used in Section 7.5.

7.3 RGB colour Sensor

One of the major drawbacks in using colour is its strong dependence on light illuminating the observed scene. Therefore, in order to provide a reliable skin detection under different illuminants, colour correction techniques or an illuminant-invariant colour-based model should be implemented [185].

The prototype of the optical system used for data acquisition alongside the data acquisition software is given in Figure 7.4. The RGB sensor consists of a single pixel embedding two analogue shutters, signal processing electronics, and including three-channel (RGB) photodiodes.

The schematic diagram of the pixel architecture is given in Figure 7.5a. With this type of pixel architecture embedding analogue electronics and by exploiting the auto-exposure control in order to execute the RGB normalization at pixel level thereby improving performance. The pixel architecture consists of three photodiodes. The operating principle of the RGB pixel is shown in Figure 7.5b. The three photodiodes precharged to Vr at time T_0 begin to discharge at a slope proportional to the light intensity.

During an exposure time, the three photo-generated RGB signals are analogue(ly) summed with the aid of the ADDER circuit. The ADDER estimates the sum of the three signals (S=R+G+B) on-the fly, during the pixel exposure time and as soon as the output of the ADDER S reaches the voltage threshold TH (Time T_S), the comparator which continuously compares S(t) is then with the threshold TH, triggers the SAMPLE & HOLD, which samples R and G signals. The equations of the computed (r,g) chromaticity is given below.

$$r = \frac{R}{S} = \frac{R}{TH} = \frac{R}{R(T_0) + G(T_0) + B(T_0)}; \tag{7.6}$$

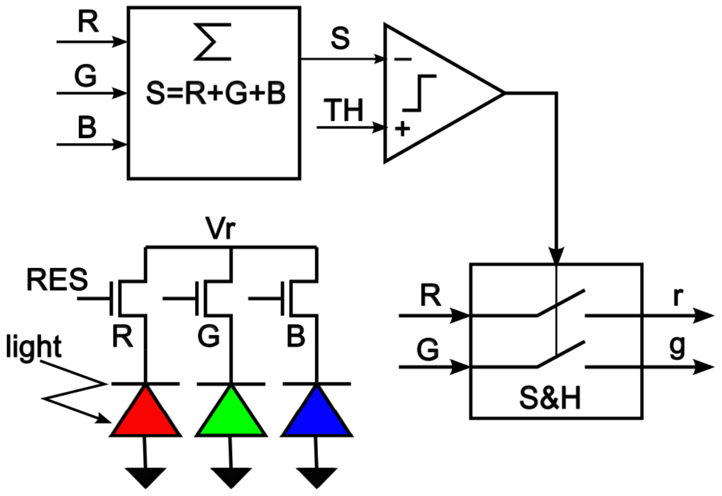
$$g = \frac{G}{S} = \frac{G}{TH} = \frac{G}{R(T_0) + G(T_0) + B(T_0)}. \tag{7.7}$$

The output rg chromaticity could then be delivered to a high level system that executes membership test of computation and skin classification using the algorithm presented in the previous section. Similarly, output rg chromaticity could be fed to a neural network implemented on hardware which is capable of implementing the same algorithm alongside the advantages of having it on hardware.

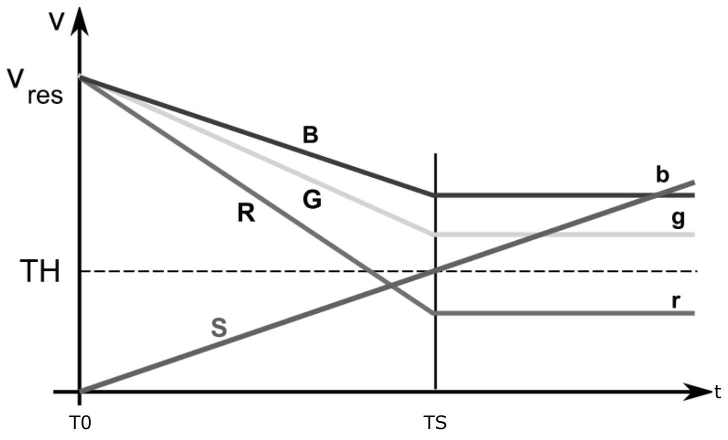
As earlier described, in the proposed solution, the pixel embeds analogue electronics executing RGB normalisation over high-dynamic range, thanks to its auto-exposure control. It consists of three photodiodes, covered with standard RGB Bayer filter and feeding an analogue ADDER. The pixel directly computes the (r,g) chromaticity over high dynamic-range, delivering the analogue values (r,g) on two bit-lines to the neural network for subsequent image processing, which can be easily accomplished, on-chip at column-level over a programmable pixel kernel. The proposed pixel has the following advantages over a standard colour pixel:

1. RGB to (r-g) chromaticity is more reliable than standard technique;
2. auto-exposure control;
3. high-dynamic range performance up to 100dB (measured);
4. low-power performance.

Moreover, with proposed RGB sensor, skin locus computation is more reliable as seen in Figure 7.6. reports a comparison between the skin locus computed by using the



(a)



(b)

FIGURE 7.5: Block diagram of the RGB pixel for RGB to r-g colour space conversion. R, G, B are the voltage drops across the 3 photodiodes. The ADDER estimates the analogue sum of the 3 signal on-the-fly. TH is a threshold voltage which defines the time at which the signals R and G have to be sampled. (This image was produced by M. Lecca *et al*. and it is reproduced with permission from M. Lecca [167].)

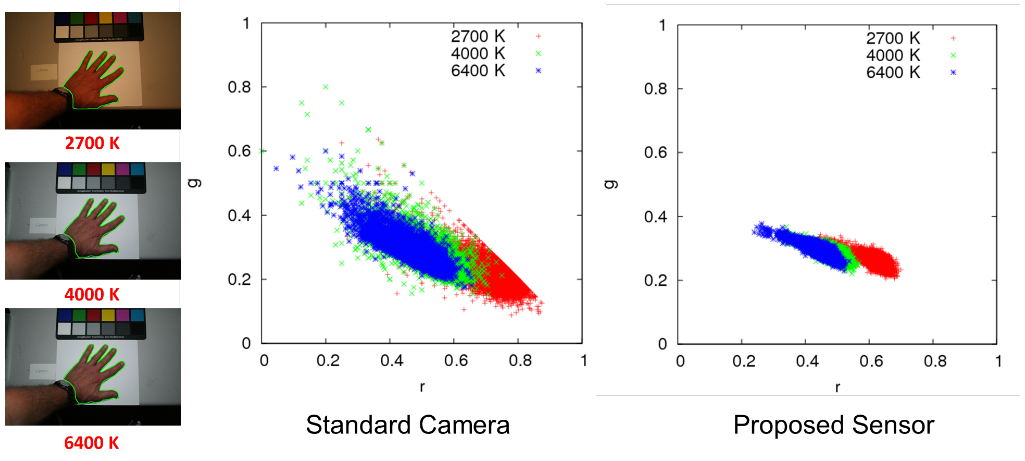


FIGURE 7.6: Comparison of skin-locus generated with a standard imager and with the proposed sensor. The comparison is performed over three standard illuminants (2700K, 4000K, 6400K), The skin-locus in proposed sensor is much more compact, turning into a smaller uncertainty in the skin pixels detection.

data of a standard camera and by using the new sensor. In this experiment, the hand has been captured under three illuminants with different colour temperatures (2700K, 4000K, 6400K) by a standard camera and by the proposed sensor. Once again, the data taken by the sensor are more accurate, so that the skin-locus can be determined with a higher precision without any need of data post-processing.

7.4 Memristor-Based Neural Network Architecture

This design of a novel hardware incorporating post processing unified technology for gesture recognition with energy saving characteristics is what this work aims at achieving. One of the main advantages of custom vision sensors over commercial imagers is their capability of on-chip image pre-processing at low-power consumption, high-dynamic range, pixel auto-exposure control [186, 187, 188]. These characteristical features make makes it crucial in several applications such as surveillance, smart interfaces, gaming, etc., where an efficient scene interpretation is required as compared to the reproduction of high quality image. This is simply achieved by fast extraction and processing of specific low-level visual cues and discarding discarding of irrelevant data similar to the implementation of the background subtraction algorithm presented in Chapter 6.

There are some commercial products using built-in hardware (camera + processor) and running vision algorithms, while others adopt specific hardware for high performance applications. On chip processing of the visual signal is the first step to reducing the computational charge of the processor which as well optimises the complexity of many

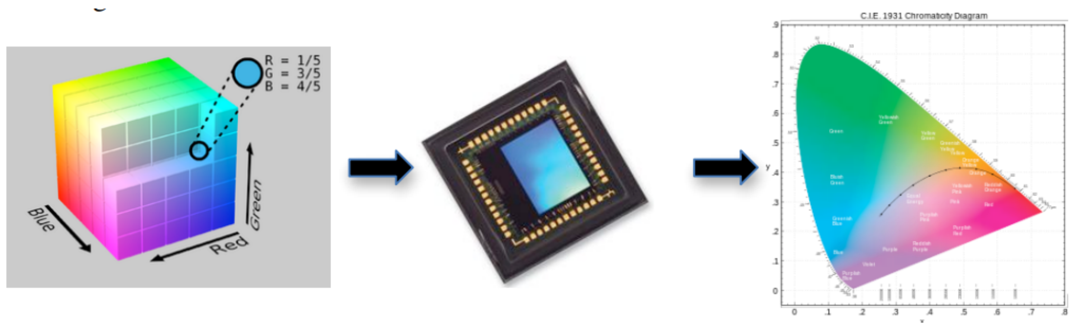


FIGURE 7.7: Proposed CMOS vision sensor performing low-power high dynamic range colour space conversion

image processing algorithms. However available commercial products are power hungry for mobile applications and can only work for a limited amount of time.

A commercial megapixel imager will typically consume about 90mW. These images are usually interfaced with a processors with even larger power consumption. Such a system when battery powered, would discharge the batteries in only a few hours. Having vision technology on a mobile devices for long lasting operations is highly desirable.

To achieve a considerable amount of reduction in power supply, a technique having a computing paradigm radically different from that of a standard system is required. The technique described in this work relies on a custom CMOS vision sensor capable of pre-filtering images at very low-power consumption. The adopted technique is particularly characterised by the following:

1. the sensor is the master of the system. After detecting a salient event in the scene, it wakes up the processor (low-level processing). Its duty cycle is 100%;

2. the processor is normally in idle mode and starts processing data from the sensor only upon sensor request. It performs gesture recognition (high-level processing). Its duty cycle is typically small ($\sim$10%);

3. whenever a gesture has been recognized, the processor decides among several actions: wake-up the smartphone, launch an application, turn-off the smartphone. Its duty-cycle is typically very low (1%).

In this way, the computational load is efficiently balanced between hardware and system post processing. The higher the power consumption of one part the lower is its duty-cycle.

In this framework, skin detection is the salient event scene. The skin tone is modelled in the 2D chromaticity space, in which its (r,g) coordinates are computed on chip, as shown in Figure 7.7.

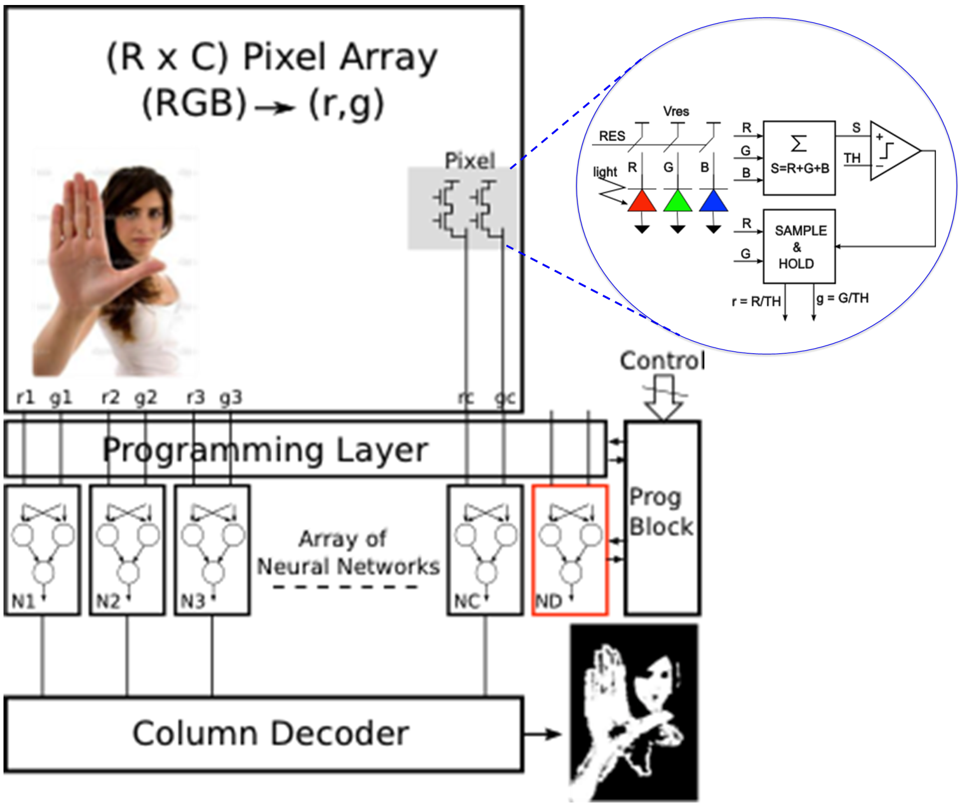


FIGURE 7.8: Proposed Neural Network Architecture for Colour Classification consisting of the RGB colour sensor, the neural network and the memristor programming circuit.

Nevertheless, apart from the preprocessing of *RGB* signal to *r,g* chromaticity, further processing can be done on hardware to further reduce the overhead complexity. In our case, colour classification for hand gesture applications can be implemented on a neural network capable of associating the *(r,g)* chromaticity acting as the neural network input to to possible outcomes, *skin* or *no skin*. The neural network as already explained, will have the advantage of having memristors store its weights, thereby becoming easily reconfigurable.

The proposed neural network architecture is given in Figure 7.8. The pixel architecture consists of an array of *RGB* pixel implementing the conversion to *(r,g)* chromaticity as described in Section 7.3, a column level array of neural networks for parallel processing of each row of the *RGB* pixel matrix, a programming layer used for programming the weights of the neural network and based on the programming circuit described in Chapter 4.6 and a column level decoder for the output binary decision.

The conceptualisation of the desired type and configuration of the neural network is based on real world data of skin (forming our skin locus) and non-skin acquired by the *RGB* sensor architecture and fed into Matlab neural network tool for training and

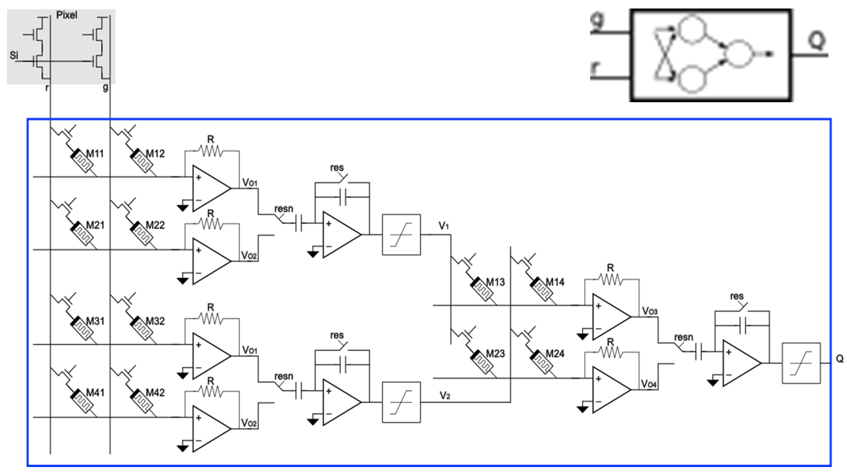


FIGURE 7.9: Hardware Implementation of the two layer Neural Network

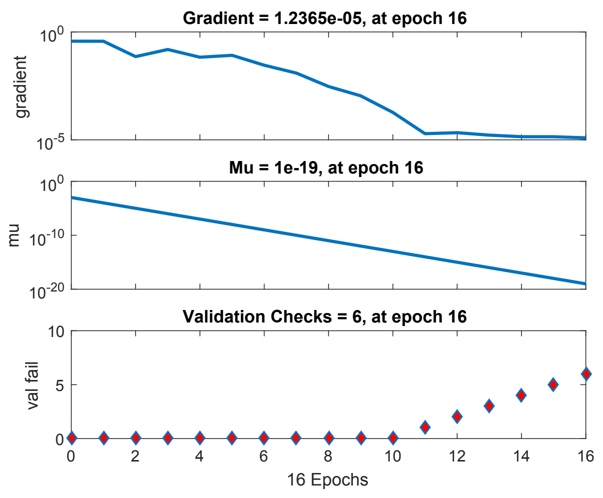
validation (See Section 7.5). At the end of training, an optimised network able to efficiently differentiate skin from non-skin was agreed upon. The network is composed of a two layer, two analogue input one binary output network. The electronic implementation of this network is shown in Figure 7.9. This type of implementation is targeted to a crossbar array of memristor requiring two memristors for each weighted sum. One for the positive weight and the other for the negative weight. As seen in the figure, a difference amplifier is used for the weighted summation of each neuron; the output of the neuron is then fed into the transfer function which makes up an input for the next layer. Details of the architecture is explained in the Matlab implementation (Section 7.5) and also in Appendix D where the idea has been implemented with discreet components on a PCB as a proof of concept. In the next section, a Matlab implementation will be simulated based on the neural network developed from datasets captured by our custom *RGB* sensor.

7.5 Matlab Neural Network Design

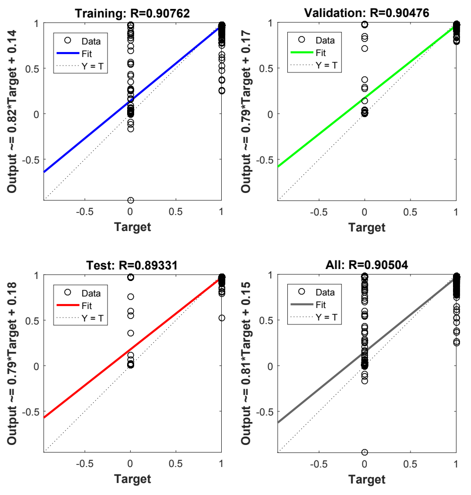
The MATLAB Neural Network design was implemented to predict if the normalised (r,g) chromaticity obtained the RGB sensor correspond to human skin or not, i.e, a binary decision. The Neural Network fitting toolbox with a powerful fitting function present in MATLAB has been used for the design.

Function fitting implies the process of training neural network typically on a set of inputs in order to produce an associated set of target outputs [1]. Once the data has be been fitted by the neural network by forming a generalisation of the input-output

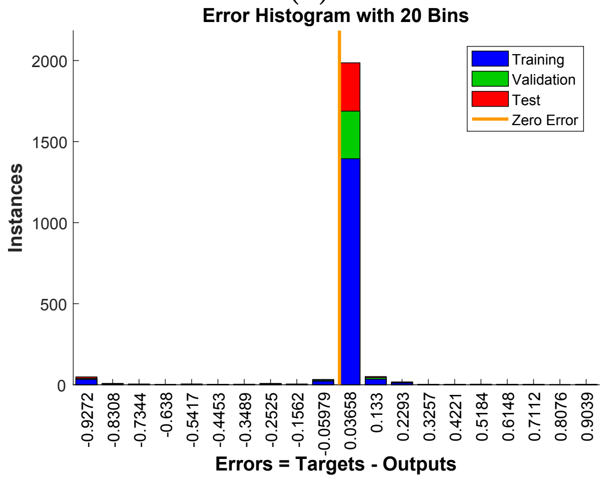
[1]MATLAB tutorial - Fit Data with a Neural Network, exhaustively treats the all aspect of neural network training and validation



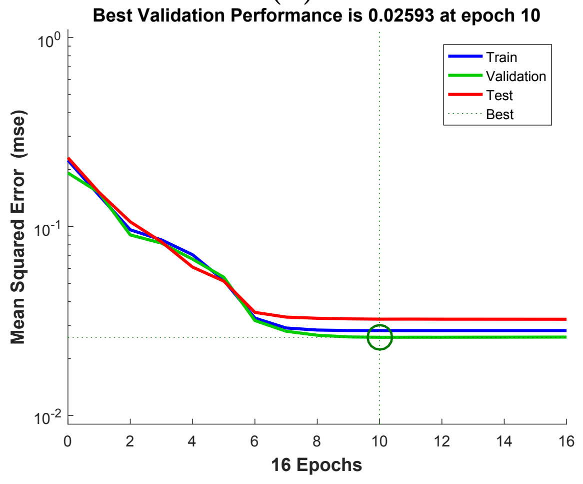
(a)



(b)



(c)



(d)

FIGURE 7.10: Neural Network Training Performance plot. **(a)**, Training steps for 16 epochs. **(b)**, Regression plot. **(c)**, Error histogram plot. **(d)**, Validation performance plot.

relationship, the network can be used to generate outputs for inputs it was not trained on.

Training – Function fitting is basically how our network has been trained. The network was trained with datasets obtained from experiments carried out with the RGB sensor. The training dataset is: (i) a 2 x 2176 matrix defining attributes of the normalised r,g coordinates which was used to plot the skin locus given in Figure 7.3 and also of r,g coordinates of non skin dataset obtained from random samples both under different illuminating conditions. (ii) a 1 x 2176 matrix target dataset is a binary attributes to be estimated (skin/no skin). During training, the input dataset and target dataset are randomly divided into three sets: 70% for training, 15% for validation and the remaining 15% is used as a completely independent test of network generalisation. The Levenberg-Marquardt training algorithm was used for the entire training.

The network was trained and retrained several times following the steps of the function fitting tutorial until an optimum network was achieved without exaggerated overfitting [145, 189, 190, 191, 192, 193, 194, 195, 196]. The final network is a two layer, three neuron network (2 neurons in the hidden layer and 1 output neuron). Figure 7.10 is a plot of the performance of the network. The network outputs with respect to targets for training, validation, and test sets are given in the regression plots. For a good fit where the network outputs are equal to the targets, the data should fall along a 45 degree line. For the training of the network presented, the fit is reasonably good for all data sets, with R values in each case of 0.89 or above. The network can be retrained if even more accurate results were required. This will change the initial weights and biases of the network, and may produce an improved network after retraining.

The error histogram gives an indication of outliers, which are data points where the fit is significantly worse than the majority of data. The blue bars represent training data, the green bars represent validation data, and the red bars represent testing data. In this case, the most error falls in the 0.03658 point. These outliers are also visible on the testing regression plot.

The training steps give an idea of the training phases. In this case, the training continued until the validation error failed to decrease for six iterations (validation stop). The entire training process took 16 iteration. Training dataset are presented to the network during training, and the network is adjusted according to its error. The validation dataset are used to measure network generalisation, and to halt training when generalisation stops improving.

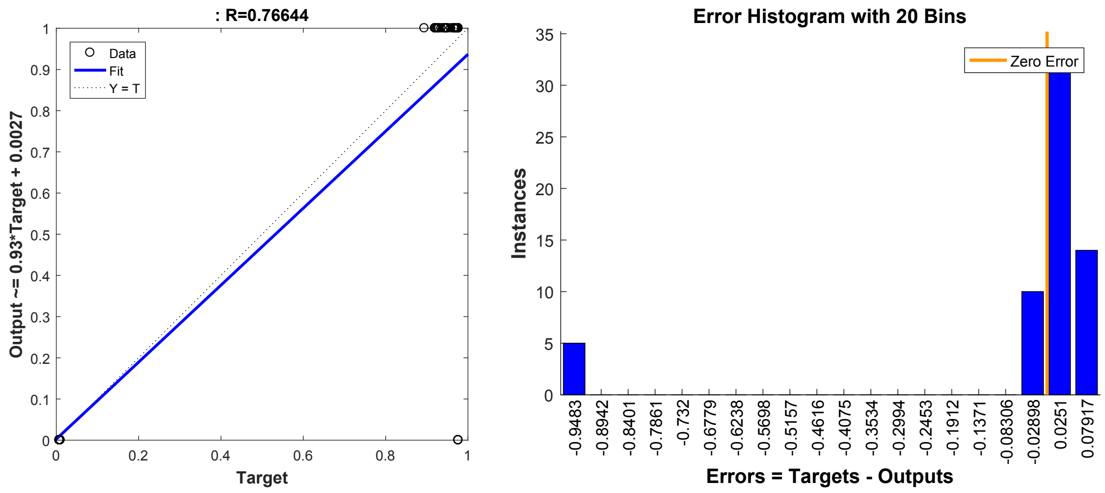
The Performance output plots the training errors, validation errors, and test errors. For the result to be reasonable, the following points have to be taken into consideration:

– The final mean-square error is small.

– The test set error and the validation set error have similar characteristics.

– No significant overfitting should have occurred where the best validation performance occurs. In our example, iteration 10.

Test – After the training and validation phase using the MATLAB tool, the neural network function is generated and tested again with datasets never seen during the training phase. The regression plot and error histogram of this test is given in Figure 7.11. The network outputs with respect to the never seen test sets is quite good, the data falls along a 45 degree line, where the network outputs are equal to the targets with an R value of 0.77. As for the error histogram plot, most error falls in between -0.02898 and 0.07917.



(a) (b)

FIGURE 7.11: Neural Network Test Performance plot. **(a)**, Regression plot. **(b)**, Error histogram plot.

7.5.1 Neural Network Simulator

After training, validation and testing of the Neural Network, the next step is the deployment. Two MATLAB simulators was implemented to emulate the Colour Sensor Neural Network.

– **ImageLoad.m** is a MATLAB script that loads all the images contained in the target folder, feeds them one by one into the simulator and outputs the image which corresponds to the binary decision (skin/no skin).

– **WebcamStream.m** is another MATLAB script similar to the previous but this one instead connects to the webcam of the computer, constantly samples streams of images in real-time and displays the out o the screen along side the original captured image in streams.

MATLAB scripts **ImageLoad.m** and **WebcamStream.m** are given in Code 7.1 and Code 7.2 respectively.

CODE 7.1: ImageLoader.m

```
srcFiles = dir('ImagesB\*.jpg');  % the folder in which images exists
L = length(srcFiles);
Col = 3;
Row = L;
k = 1;
for i = 1 : L
        filename = strcat('ImagesB\',srcFiles(i).name);
        img = imread(filename);
        [m,n,o] = size(img); % returns the image resolution
        R = reshape(img(:,:,1),1,[]);
        G = reshape(img(:,:,2),1,[]);
        B = reshape(img(:,:,3),1,[]);

        %convert image from RGB to rg colour space
        r = double(R)./double((R+G+B));
        g = double(G)./double((R+G+B));
        b = double(B)./double((R+G+B));

        %Create 2xQ matrix of rg components for the Neural network input
        rg = vertcat(r,g);
        Y = myNeuralNetworkFunction(rg);
        % Output 1
        I = reshape(Y,[m,n]);
        J = I;
                for i = 1:m
                        for j = 1:n
                                if J(i,j)>=0.8
                                        J(i,j)=1;
                                else
                                        J(i,j)=0;
                                end
                        end
                end
        figure(1),
        %set(gcf, 'Position', get(0,'Screensize'));
        subplot(Row,Col,k,'align'), imshow(img);
        subplot(Row,Col,k+1,'align'), imshow(I);
        subplot(Row,Col,k+2,'align'), imshow(J);
        k = k+3;
end
```

Code 7.2: WebcamStream.m

```
clear('cam');
cam = webcam();
True = 1;
%cam.Resolution = '320x240'; %Comment if webcam does not support this
%resolution. Default resolution supported will be used.
% Create a loop to acquire 10 frames feeding each to the NN in real time,
% plotting the corresponding binary images
%Acquire 10 frames (1 frame every 3 seconds)
%for idx = 1:100
while True
        % Acquire a single image.
        img = snapshot(cam);
        [m,n,o] = size(img); % returns the image resolution
        R = reshape(img(:,:,1),1,[]);
        G = reshape(img(:,:,2),1,[]);
        B = reshape(img(:,:,3),1,[]);

        %convert image from RGB to rg colour space
        r = double(R)./double((R+G+B));
        g = double(G)./double((R+G+B));
        b = double(B)./double((R+G+B));
        %Create 2xQ matrix of rg components for the Neural network input
        rg = vertcat(r,g);
        Y = myNeuralNetworkFunction(rg);
        % Output 1
        I = reshape(Y,[m,n]);
        J = I;
        for i = 1:m
                for j = 1:n
                        if J(i,j)>=0.8
                                J(i,j)=1;
                        else
                                J(i,j)=0;
                        end
                end
        end
figure(1),
set(gcf, 'Position', get(0,'Screensize'));
subplot(1,3,1,'align'), imshow(img); title('Input image');
subplot(1,3,2,'align'), imshow(I); title('NN Output image');
subplot(1,3,3,'align'), imshow(J); title('threshold @ 0.9');
%imshow(J); title('threshold @ 0.9');
%pause(1);
end
```

The scripts perform the following functions in the order listed:

- Load an image (from either the target folder or from a frame sampled by the webcam)
- Individualise the three dimensional RGB matrix from the image into 3 separate vectors
- normalisation of the R, G and B vectors to r, g and b chromaticity.
- Create 2xQ matrix of r and g components for the Neural network input; where Q = mxn given an image of dimension mxn.

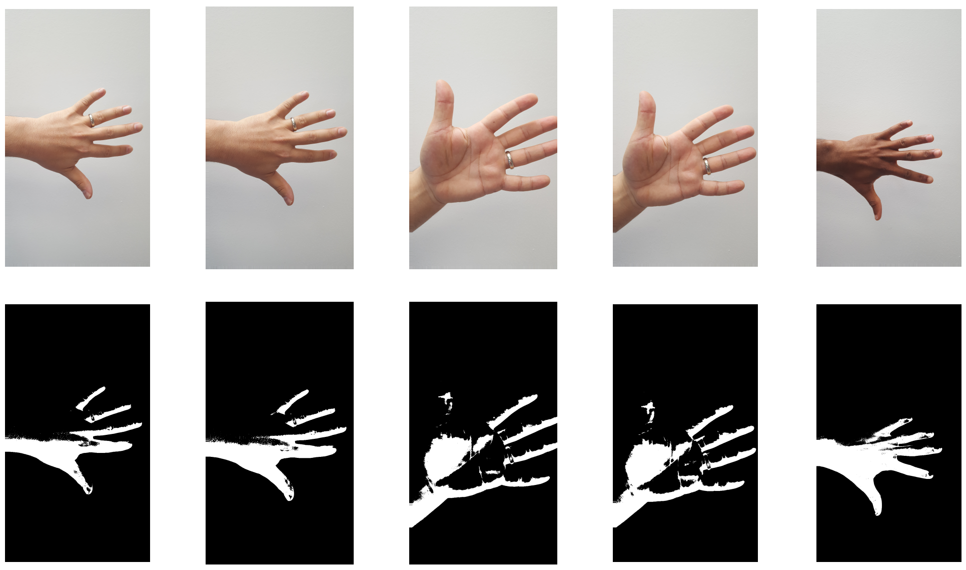


FIGURE 7.12: Implementation of the Matlab Neural Network Simulator. The input images are the top images while the bottom images are the corresponding neural network output images.

- Feeds the 2xQ matrix into the neural network function **myNeuralNetworkFunction.m**.
- The resulting 1xQ output vectors is then reconstructed to an mxn binary image

The Generated MATLAB Neural Network Function used is available in Appendix D.4. Figure 7.12 is a demonstration of how the Neural Network simulator is able to classify human hand from the background and other objects in the scene.

7.6 Conclusion

This chapter focuses on the development of an illuminant invariant colour based algorithm for skin detection that is needed to cope with changes of light. In the proposed system colour correction is necessary to refine the sensor based skin detection as well as to extract other colour-based features useful for hand and gesture recognition (e.g. edge orientation). An implementation of an efficient and fast colour correction algorithm working on the *(r,g)* colour space was presented in this chapter.

A memristor-based architecture for colour classification targeted to mobile phone application has been presented. The architecture based on a vision sensor for scene interpretation embeds a neural network to reduce the overhead processing time as well as power consumption. This approach exploits all the advantages of memristors targeted to a complex and advanced application (colour detection for low power Human Machine

Interface applications). The proposed vision sensor architecture is as well targeted to low-power applications for mobile devices. The vision sensor was used for the acquisition of experimental data which was used to train the neural network in Matlab in order for us to come up with a compact and efficient neural sensor architecture.

The sensors embeds reliable RGB normalisation needed to perform on-chip subsequent reliable skin-colour detection and morphological features extraction. The concept has been proven through the realisation of a colour pixel prototype. Thanks to its low-power performance, the vision sensor is intended to work continuously, triggering the processor only when necessary, i.e. when detecting a relevant amount of skin pixels. This turns into a significant reduction in power consumption, without sacrificing the gesture functionality. In this chapter, low power colour normalisation is implemented at pixel-level, while skin colour detection is accomplished at column-level, reaching a good trade-off between pixel complexity and pitch.

The skin detection algorithm was based on the computation of a skin locus from the *(r,g)* chromaticity obtained from different illuminants. Comparative measurements was done against a standard colour sensor, demonstrating significant advantages of the proposed pixel architecture. Finally, a Matlab simulator of the architecture was implemented to test the functionality of the neural network for colour classification.

Chapter 8

Summary

This chapter summarises and draws conclusion on the overall memristive computing architecture discussed in this thesis according to the flow from Chapter 1 through Chapter 7. Several techniques of memristive computing architecture have been exhaustively treated which is crucial in this rapidly changing field especially the over stretching of the limitations that CMOS poses. The fact that the roadmap based on the continued extension of CMOS technology with respect to Moore's law does not guarantee any more that silicon-based CMOS will extend beyond, coupled with the inherent properties of memristors, motivated me to carry out research on this exciting and emerging topic having a well-suited alternative in emerging technologies like memristors to complement CMOS circuitry as well as in computational tasks. The architectures presented in this thesis indicate memristors' suitability in achieving all of this in the near future.

In this thesis, the fabrication and characterization of results obtained based memristive devices developed during the course of this research have been analysed. In addition, a comparison between the existing memristor device models is analysed to show how the memristor can be used in a multi-state operation. Memristor programming circuit designs have been implemented to demonstrate the writing and reading of information to and from memristive devices which represent the initial steps required in developing electronic systems based on memristors.

This study mainly focusses on memristor modelling, device and characterisation followed by circuit application of memristor-based architectures for several image processing algorithms presented in a logical flow.

Notable circuit application was emphasised in the field of on-chip image processing where a novel in its kind light to resistance encoder circuit was presented a light-to-frequency converter was adopted to drive a memristor with pulses, thus modulating its resistance according with the light intensity. The memristor in this circuit is intended to act as an analogue counter thereby replacing the resources hungry CMOS counted intended for such application. This pixel architecture went on to be used in a more complex pixel architecture used to implement pixel-level adaptive background subtraction algorithm.

The work flow in the treatment of *Memristor-Based Computing Architecture with Advanced Signal Processing Capabilities* evolved throughout this thesis in such a way that the first couple of chapters (Chapters 2 through 4) of the thesis were focused on the history, theoretical and physical properties of the memristor, the device fabrication and experimental characterisation to validate the fundamental fingerprint of memristors, and then the resistance tuning techniques which apart from its memory features, is one of the major characteristics exploited in this thesis. The second part focussed more on circuital application in image processing with a target to mobile applications (Chapters 5 through 7).

8.1 Device, Modelling, Characterisation and Programming

Memristor Fabrication and Characterisation: Chapter 2

Background: Since the announcement of the discovery of the physical memristor, research activities in this field has increased exponentially due to the advantages memristors have over conventional electronic devices. In the course of this thesis, based on the framework of the MaDEleNA project, memristor device, fabrication and characterisation was envisaged in a work package. Discussed in this chapter were the various low cost fabrication steps taken into consideration in the manufacture of memristive devices as well as the characterisation steps yielding promising results.

Methodology: The fabrication steps from the technological side was carried out using new approaches involving material science and micro-fabrication processes which provided the basic building blocks for realising more complex architectures based on memristive components.

Result: The adopted fabrication process for the development of test structures for exploiting the various memristive properties is relatively simple and low cost with respect to the state of the art. Experimental characterisation was carried out with the setup given in Figure 2.8. Having the whole measurement process automated using LabVIEW, a preliminary study of controlled voltage-current curves was performed and a switching resistance effect was found on several structures with some excellent reproducibility. Further experimental measurements involving memristors fabricate with the above mentioned process was carried out and results presented in Chapter 4

Memristor Device Modelling: Chapter 2

Background:: Even though the concept of memristors have existed theoretically for several decades, it was not until 2008 before interests in investigating them stared rising. Due to their volatile nature and different fabrication materials and steps, no generalised model of the memristor exist. In order to simulate memristive systems, a simulation model is required. This means we have to create our own models which should behaves as closely as possible to the physical device. The memristor models, and basic circuit designs served as the first steps in developing electronic systems and computing architectures based on memristors.

Methodology: A behavioural model of the memristor was developed using Verilog-A, a high-level language that uses modules to describe the structure and behaviour of analogue systems and their components. Verilog-A gives the advantages of being simple, efficient, more concise and clear language of describing analogue behaviour in simulators over SPICE models, simulation run time is much shorter. Also, Verilog-A was chosen because it can be easily integrated in CADENCE (a standard CAD software for IC design) which is one of the tools I used during the course of this research, thereby, giving the model the advantage of being integrated in a more complex network with standard electronics. The memristor modelling equations of the memristance function were extracted based on parameters from thin films of titanium dioxide using its state dependent current-voltage relationship used. Here the memristor is modelled using two variable resistor.

Result: A memristor model which can be easily tuned to fit a broad range of memristive devices was developed. A circuit symbol of the simple an accurate behavioural model was generated placed in our CAD library which was used for memristive systems simulation. Based on measurement and experimental results, the memristor model can effectively be tuned by acting on certain physical parameters contained in the model to fit experimental data. An example of model to hardware fitting was also given. All simulations carried out during the course of this work were done using the above mentioned model.

Memristor Emulator: Chapter 3

Background: Due to their ability to be accurately programmed within their resistance range, after the realisation of a solid state device, memristor circuit particularly programmable analogue ICs are still being investigated. Researchers have so far relied on SPICE simulation models of memristive devices for numerical simulation in order to investigate the memristor device properties in programmable analogue electronics. Even though SPICE macromodels are important for the simulation of memristors, they posses certain limitation in terms of the accuracy needed to simulate devices of such dynamical properties. Moreover in order to implement practical electronic circuits based on memristors, SPICE models cannot be used. It is therefore, necessary to have emulators which could be use for real-world application which are practical pending when reliable commercial solid state memristors are available.

Methodology: The emulator composed of off-the-shelf electronic components come in handy at a time where reliable physical devices are yet to be available. The method implemented for the memristor emulator is based on a microcontroller and a digital potentiometer. Communication between the microcontroller and the digital potentiometer is achieved through SPI protocol. The ADC of the microcontroller is used to continuously sample the voltage difference between the terminals of the digital potentiometer's resistance network, the corresponding resistance is t hen calculated by the microcontroller based on the memristor current-voltage relationship and the digital potentiometer is in turn updated with the current resistor value. This technique is particularly of educational value and can be used to initiate students into the basic theory of memristors.

Result The entire process of validating the emulator was designed to be as simple and easily replicable as possible by using a serial monitor to plot data acquired by the microcontroller in real time. The fundamental properties of the memristor have been validated experimentally with the emulator.

Memristor State Tuning: Chapter 4

Background: In order to use memristors in programmable analogue circuitry by exploiting their non-volatile resistance memory, it is important to be able to accurately change their conductance states. Programmability has to be precise and reproducible within an adequate modulation range. Programming of these devices is not straightforward and can sometimes be difficult or tricky while the device reproducibility with respect to the kind of programming technique plays an important role. The accuracy of a memristor-based programmable circuit is therefore strongly dependent on how accurately the memristor can be tuned. In this Chapter, several programming techniques were presented with respect to the state of the art and in the end I presented an analogue and a digital programming solutions for accurately tuning memristors.

Methodology The pulse-based programming method alongside it's DC counterpart were experimented upon. The pulse-based technique is arguably the most popular method in terms of read, write and reset of the memristor. As a proof of concept of the pulse programming technique, memristance rate of change over time is analysed for programming input of different duty cycles of a train of current/voltage pulses. The same operation was carried out based on subjecting the device to a constant DC bias but this time by varying the input voltage.

Result Experimental characterisation related to memristor resistance modulation techniques carried out on FBK memristors validates the theory of changing the state of memristor by accumulating charges. This also shows the non-volatile nature of memristors. Two memristor programming circuits were presented which are capable of accurately tuning the memristance of a memristor to correspond input reference voltage thanks to the automatic patterning of input programming voltages.

8.2 Memristor Circuit Application

Light-to-Resistance Converter: Chapter 5

Background: Light to frequency conversion is a process of converting the intensity of an impinging light into digital pulses by a modulation method for representing an analogue signal using only two levels (1 and 0). Generally, in order for this digital level to be translated into light intensity an encoder in the form of a digital counter would be needed for pulse count. Unfortunately, these counters are area hungry and impractical especially in applications requiring smaller pixel pitch sizes. In memristor-based light to resistance encoder an alternative was proposed, replacing the digital counter with an analogue counterpart with a single memristor. This is of course, thanks to memristors' fine resolution programmability. A pixel architecture implementing a light-to-resistance encoder with a memristor is detailed in this Chapter.

Methodology In the solution presented in this chapter, a multiple reset pulse frequency modulation pixel architecture composed of a photodiode with a reset transistor, a voltage comparator, a delay stage as feedback and a single memristor working as an analogue counter was implemented. The operating principle of the pixel is as follows: the photodiode is first pre-charged to a voltage. After reset, the photodiode voltage starts discharging proportionally with the light intensity. Once the voltage reaches the threshold of a comparator, the latter toggles and resets the photodiode, with a certain delay, back to the pre-charged value V_R. This operation repeats at a rate proportional with the light intensity and generates pulses with a distinct pulse width. The memristor encodes non-linearly each pulse generated by an increase in its conduction. At the end of the exposure time, the memristor's resistance value is read out and reset back to the initial state before another exposure time can start. This way light to resistance encoding is achieved.

Result The circuit has been validated with the simulation of different light intensities represented by the photo-generated current. The proposed circuit has been designed and simulated in a $3.3V$, $0.35\mu m$ CMOS process with our Verilog-A model relying on the TiO_2 model. In the results presented of a plot of the pixel digital output versus the photo-generated current are as expected, we observe good linearity over a wide dynamic range. The solution of having the binary counter being replaced with an analogue counterpart, made of a single memristor transforms the architecture into a smaller pixel pitch, compared with an all-CMOS solution and analogue non-volatile characteristics.

Pixel Architecture for Dynamic Background Subtraction: Chapter 6

Background: Motion detection based on temporal contrast requires a precise and reliable detection which should be accurate in shape detection and response to changes in time is highly required. Background subtraction is one of the most widely adopted technique, where an estimation of the background is generated and updated frame by frame which is important in a broad range of applications, including traffic monitoring, human motion capture and video surveillance.

Methodology Core of the processing is the pixel, containing a light-to-frequency converter outputting digital pulses, proportional to the intensity of light, which are in turn applied to a memristor, changing its resistance accordingly. Two additional memristors are used to store the dynamic boundaries, outside which the behaviour of the photo-generated signal is recognised to be anomalous, i.e., unexpectedly fast changing. In this case the image background is modelled with programmable time constants. Considering the hardware implementation aspects and the robustness of the above techniques, we modelled the background with an exponential moving average using two threshold voltages represented by the two additional memristors. The thresholds define the voltage range inside which the signal is allowed to change safely. Outside this range (above or below), the signal is considered anomalous and potentially marking an alert.

Result In accordance with the operating principle of the pixel architecture for adaptive background subtraction, the presented pixel architecture relies on three memristive devices (M_S, M_{max} and M_{min}), storing three resistance values which are proportional to the photo generated signal and the two thresholds V_{max} and V_{min} respectively. With this proposed pixel architecture, the four possible pixel conditions which can be deduced have been simulated on Cadence spectre. The results also address some practical limitations of such pixel architecture such as device-to-device variation with respect to the memristors and shows that the reproducibility of each memristor is what is critical instead of having exactly identical memristors.

Memristor-Based Neural Network: Chapter 7

Background: Neural networks are mostly implemented on software as a result of the issues that comes with the reconfigurability of hardware implemented networks where once a network has been trained, the weight of each synapse remains fixed and cannot be changed. With a memristor-based neural network, applications regarding in-pixel image processing can be taken into another dimension. Colour detection algorithm is analysed and an application on skin detection targeted to mobile applications is discussed.

Methodology Memristive devices can be used to easily addressed the limitations associated with implementing neural networks on hardware. The proposed neural network architecture for colour classification is composed of an RGB colour sensor, the neural network and the memristor programming circuit. Since colour has a strong dependence on light illuminating the observed scene, an illuminant-invariant colour-based model was implemented based on RGB normalisation to (r,g) chromaticity colour space. The RGB colour sensor has the advantage of embedding analogue electronics and exploits its auto-exposure control in order to execute the RGB normalisation at pixel level thereby improving performance.

Result The proposed neural network architecture has been designed, trained with experimental data and validated on Matlab neural network tool. Results show good fittings of the network and the network is capable of recognising patterns it has never seen before as either skin or no skin.

Appendix A

Application of the Switched Memristor Circuit

A.1 Comparator with Programmable Hysteresis

The memristor-based comparator with hysteresis is also a typical configuration of a comparator with hysteresis [197, 198]. It uses a voltage divider to set up an upper and a lower threshold voltage (V_H, V_L).

Figure A.1 is the proposed memristor-based comparator with hysteresis. This circuit uses a memristor network M_1, M_2 and M_3 which have been previously programmed to set the hysteresis at the desired value. When the output of the comparator goes high $(V_{out} = V_{dd})$, M3 is in parallel with M1 increasing the voltage node A:

$$V_H = \frac{M_1M_2 + M_2M_3}{M_1M_2 + M_1M_3 + M_2M_3} + V_{dd} \tag{A.1}$$

This drives more current into M_2, setting the threshold voltage M_2. The expression of V_H is given in Equation A.1. The input signal will have to drive above V_H to cause the output to transition to logic low.

When the output is at logic low $(V_{out} = 0)$, M_3 is in parallel with M_2 reducing the current into M_2, setting again the threshold voltage to V_L.

$$V_L = \frac{M_2M_3}{M_1M_2 + M_1M_3 + M_2M_3} + V_{dd} \tag{A.2}$$

The input signal will have to drive below VL to cause the output to transition. From Equation A.1 and Equation A.2, we derive the following hysteresis voltage equations:

$$V_H - V_L = \frac{M_1M_2}{M_1M_2 + M_1M_3 + M_2M_3} + V_{dd} \tag{A.3}$$

$$V_{dd} - V_H = \frac{M_1M_3}{M_1M_2 + M_1M_3 + M_2M_3} + V_{dd} \tag{A.4}$$

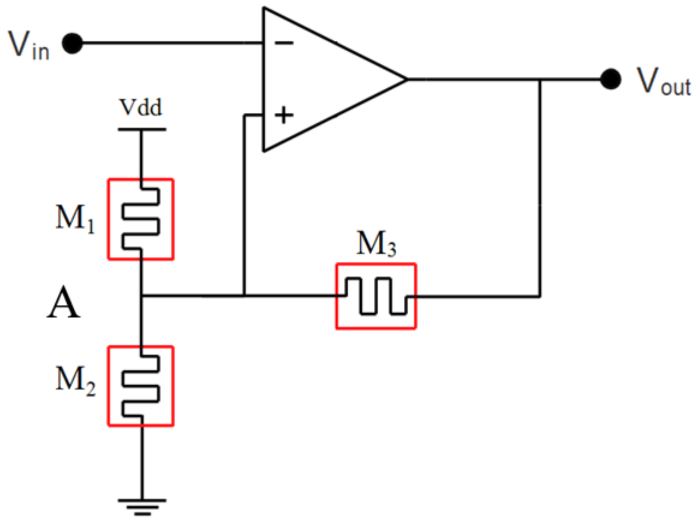


FIGURE A.1: Schematic of the memristor-based comparator with hysteresis. The memristor circuit here is based on the chopped memristor circuit given in Fig. 1. High and low thresholds are set by the 3 memristors in a voltage divider configuration..

Applying programming pulses to either of M_1, M_2 or M_3 will either increase or decrease V_{OH} and V_{OL} [199, 200].

The memristance of M_1, M_2 and M_3 should be chosen to be very high to minimise the current drawn by the op-amp. We chose a model of the memristor with resistance ranging up to several hundreds of Kilo-Ohm [35]. The equations for setting the threshold voltages using the memristors are given in Equation A.1 and Equation A.2. From Equation A.1, Equation A.2, Equation A.3 and Equation A.4, we obtain the following equations:

$$\frac{M_1}{M_3} = \frac{V_H - V_L}{V_L} \tag{A.5}$$

$$\frac{M_1}{M_2} = \frac{V_{dd} - V_H}{V_L} \tag{A.6}$$

Equation A.5 and Equation A.6 can be used to set the hysteresis threshold voltages V_H and V_L. In our experiment, we decided to maintain M_1 at a fixed value and then apply the programming pulses to either M_2 or M_3. Assuming memristors M_1 is set to arbitrary value M and the programming pulses will be applied to M_2 or M_3 while maintaining M_1 unchanged all the time.

$$M_2(t) = \frac{M_1 V_L}{V_{dd} - V_H} \tag{A.7}$$

$$M_3(t) = \frac{M_1 V_L}{V_H - V_L} \tag{A.8}$$

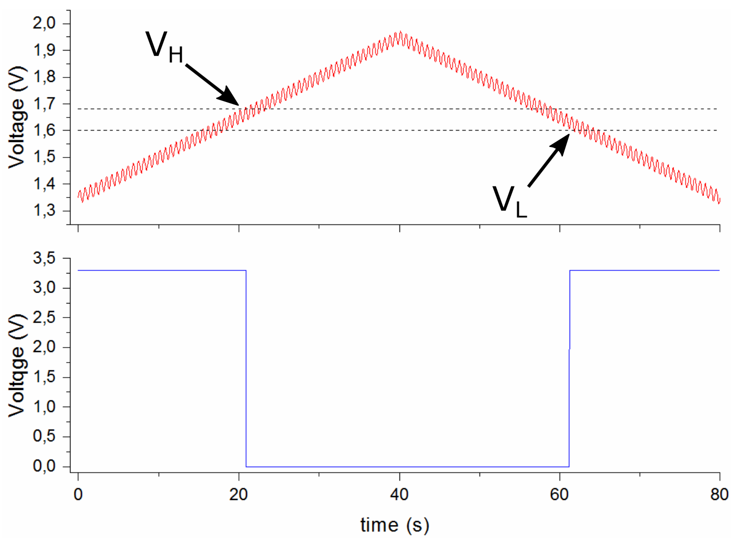


Figure A.2: Output of the chopped memristor-based comparator with hysteresis showing single transition in spite of the noisy triangular waveform signal at the input of the comparator. V_{OH} and V_{OL} are set to 1.68V and 1.6V respectively with an input noise range of 80mV. Depending on the application, the hysteresis range can be adjusted to be large enough to reject noise by changing the state of either or all of M_1, M_2 and M_3.

$$M_1(t) = M_3\frac{V_H - V_L}{V_L} = M_2\frac{V_{dd} - V_H}{V_L} \tag{A.9}$$

From Equation A.7, we have the relationship between the memristance of M_2, M_1 and the threshold voltages. By applying programming pulses to M_2 or M_1 we see the threshold voltages change (Figure A.3).

A.2 Simulation

The comparator hysteresis change for the application of six programming pulses to M_2 and M_3 as given in Figure A.3a and Figure A.3b respectively. Changing the memristance of M_2 shifts the curve away from the voltage reference. Meanwhile, a change in M_3 shifts the hysteresis around the reference voltage. In Figure A.4, V_{out} is the output of the comparator for the corresponding hysteresis curve change in Figure A.3a. We observe a change in the hysteresis due to the application of a programming pulse across M_2. Figure A.5 is also the comparator response corresponding to the hysteresis curve in Figure A.3b. During both operations, a total of 6 positive programming pulses of 0.1V width were applied to the memristors being programmed. For every applied pulse, the memristance is decreased by approximately 10%.

In Figure A.5, a change in M_3 does not bring about any significant change in hysteresis in contrast to a change in M_3 as observed in Figure A.4. Generally, for this type of

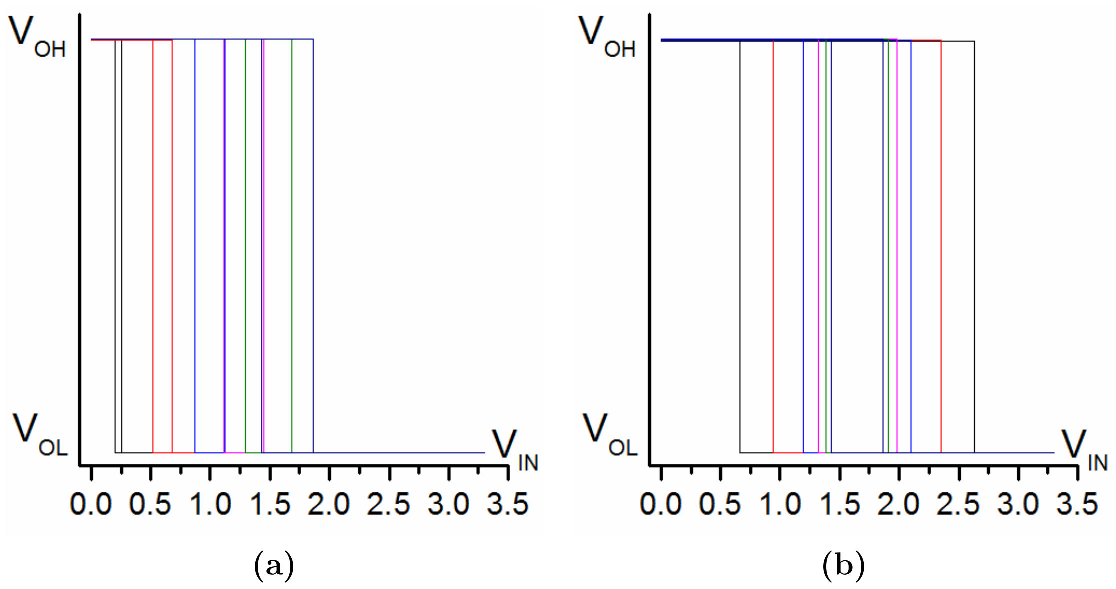


FIGURE A.3: Hysteresis curve plot for different programming pulses applied. **(a)**, Hysteresis curve change for programming pulses applied to M_2. **(b)**, Hysteresis curve change for programming pulses applied to M3.

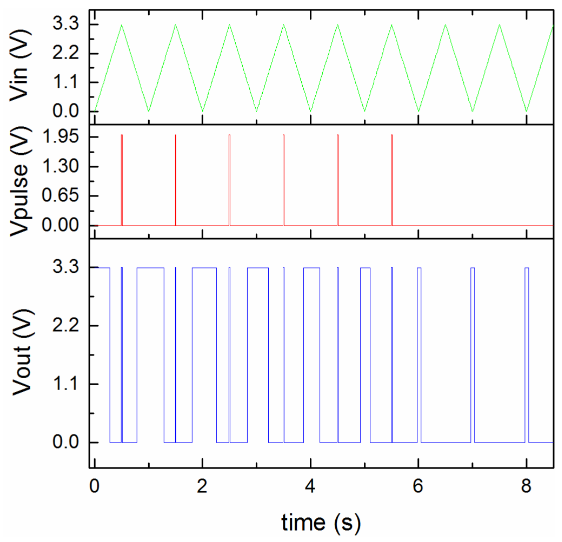


FIGURE A.4: Output of the memristor-based programmable threshold comparator with response to programming pulses applied to M_2. The input signal V_{in} is a periodic triangular waveform.

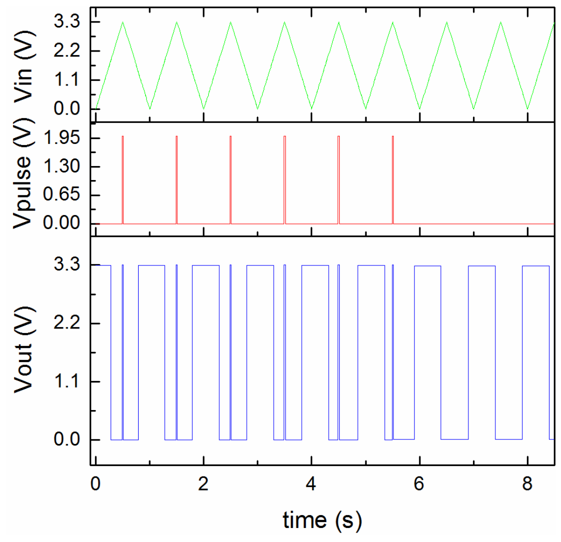


FIGURE A.5: Output of the memristor-based programmable threshold comparator with response to programming pulses applied to M_3. The input signal V_{in} is a periodic triangular waveform.

configuration, M_3 should be relatively larger than M_1 and M_2 so as to minimise the current drawn by the circuit.

A.3 Conclusion

This work aims to demonstrate the advantages of using memristors as part analogue circuits. By adopting a chopped memristor configuration, it is possible to maintain its resistance even with a large current flowing into the device. From our evaluation based on simulation results, given the application of the chopped memristor circuit, the impact of the chopping frequency, the positive and negative levels of the chopping clock over memristance does not pose much of a problem as the desired output is achieved. In the case of the comparator with programmable hysteresis, the results are as expected. The change in memristance with respect to the chopping clock is not perceived at the output of the comparator. This could be because the large resistance values are used in a voltage divider setup. It is worth to note that, over a long period of operation, due to drift effects, the device should be re-programmed to guarantee the performance of the entire circuit. Therefore, constant recalibration of the circuit could be necessary to curtail this problem.

We have presented a comparator using a memristor network to achieve programmable threshold. The basic idea is to chop the memristor so as to achieve a (substantially) stable value of its resistance in spite of the non-zero voltage effectively applied. By applying pulses the memristor, we change its state, thereby, changing the comparator hysteresis.

Appendix B

Emulator Demo PCB Design

B.1 Standalone Emulator PCB Design

This Section is dedicated to the design of a PCB to house the memristor in a demo-like setup. The term standalone is used to denote that unlike the previous experimental setup given in Figure B.1, the compact setup used will be able to carry out all its required functions without the use of laboratory equipments. The setup is designed to be interfaced with an Arduino board similar to the previous configuration. Data acquisition and real-time plotting of results is as well done exactly the same way we have already seen in the previous section.

B.1.1 Functional Description

The functional block diagram of the demo-style memristor emulator embedded in a PCB is given in Figure B.1. This block diagram consists of an on-board voltage regulator, a functional generator and level shifter circuit which has been implemented to scale-up the output of the functional generator to a range suitable for the emulator setup and in order to be able to fully utilise the ADC of the Arduino.

The board has been designed in such a way that it can be stacked with the the Arduino Due board used for the SPI communication with the DigPot and the waveform generator, serial interfacing with the PC, implementation of the memristor mathematical equations and for the analogue and digital signal processing. The procedural function of the Arduino is similar to the one presented in the previous section. The programmable waveform chosen, Analogue Devices AD9833EP, is a low precision programmable waveform generator capable of producing sine, triangular and square wave outputs. The output frequency and phase are software programmable allowing easy tuning. Programming and communication with the AD9833EP is achieved through the Serial Peripheral Interface *(SPI)* via a 3-line with the Arduino as earlier mentioned and as shown in the block diagram of Figure B.1. The 3-wire serial interface comprises of the Serial Data Input *(SDI)*, Serial Clock *(SCK)*, and Chip Select *(CS)*. This is all achieved without

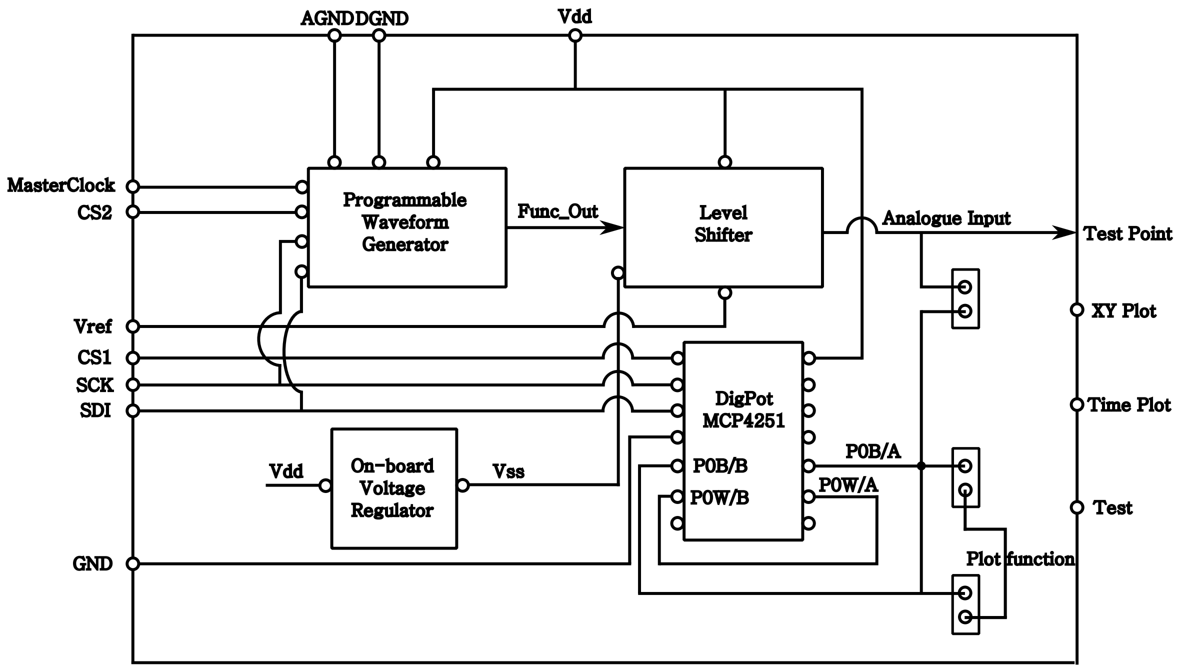


FIGURE B.1: Functional Block Diagram of the memristor Emulator PCB Design

any other external component required. According to the AD9833EP's data-sheet, the maximum output voltage is $0.65V$. This isn't suitable for our application, therefore we had to scale-up the output voltage with the level shifter block to an acceptable range suitable for programming the emulator.

The level shifter block is composed of a simple non-inverting amplifier configuration with a network of four resistors. The MCP4251 8-bit dual digital potentiometer is also integrated on the PCB. As clearly seen in the block diagram, the SPI is shared by both the MCP4251 and the AD9833EP. The two devices use different SPI modes and thus, when programming a given device when the appropriate CS button is pushed, we have to similarly set the corresponding SPI mode.

B.1.2 PCB Design

The board was designed with National Instrument (NI) Multisim and Ultiboard suites. The layout of the PCB is shown in Figure B.2. The PCB is composed of two layers showing visibly, the components mounted, top and bottom layer copper traces, vias and mounted connectors. Although not shown in the figure, the top and bottom layers are covered with solid power planes. Low signal traces have been routed to the bottom layer to ensure a low impedance path for any return currents on the bottom layer ground plane. Vias were placed at each components ground connection for the routing of return

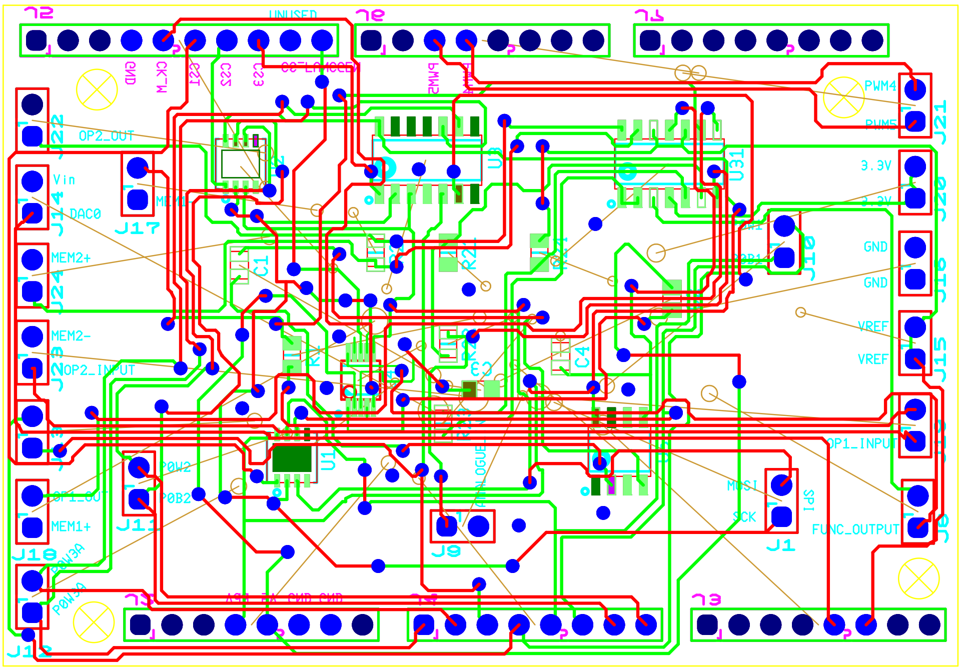


FIGURE B.2: Memristor Emulator PCB design Layout

currents to the bottom plane in order to provide the shortest possible path back to ground. General PCB layout guidelines were followed.

Appendix C

Programming Circuits Hardware Design

C.1 Hardware Design of the Programming Circuits

This section is dedicated to the hardware design of the two programming circuits presented in Section 4.6. The CMOS IC and PCB design are presented. Finally, the measurement setup comprising of the PCB housing the CMOS chip is presented with some measurement result.

C.1.1 CMOS IC Design

Both the analogue and digital programming circuit architectures presented in Section 4.6 were designed and fabricated in a $3.3V$, $0.35\mu m$ CMOS process. See above mentioned related Section for a description of the various constituting blocks of the overall circuit (Digital and Analogue). The chip micrograph of the circuits is shown in Fig. C.1, the digital programming circuit takes up an area of $250\mu m$ x $277\mu m$ whereas, the analogue programming circuit takes up an area of $250\mu m$ x $277\mu m$.

C.1.2 PCB Design

Similar to other PCBs fabricated during the course of my PhD, the board was designed with National Instrument (NI) Multisim and Ultiboard suites. Unlike the Emulator PCD design, due to the complexity and density of this design, a four layer board was adopted with components mounted on both sides of the board. The layout of the PCB given in Figure C.3 clearly shows the components mounted (top and bottom), top and bottom layer copper traces, inner 1 and inner 2 layer traces, vias and mounted connectors. Although not shown in the figure, the top and bottom layers are covered with solid power planes. Low signal traces have been routed to the bottom layer to ensure a low impedance path for any return currents on the bottom layer ground plane. Vias were placed at each components ground connection for the routing of return currents to the bottom plane

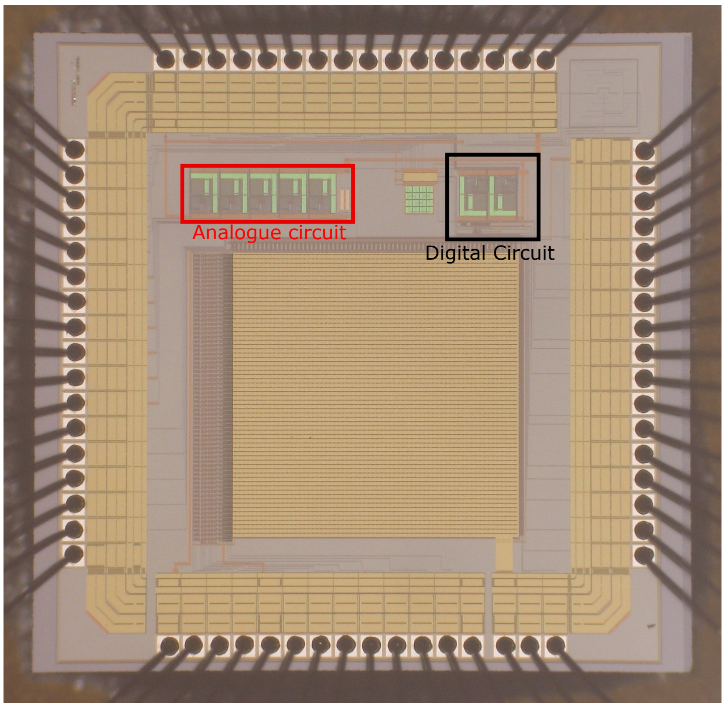


FIGURE C.1: Micrograph of the fabricated chip showing the digital and analogue programming circuits

in order to provide the shortest possible path back to ground. General PCB layout guidelines were followed.

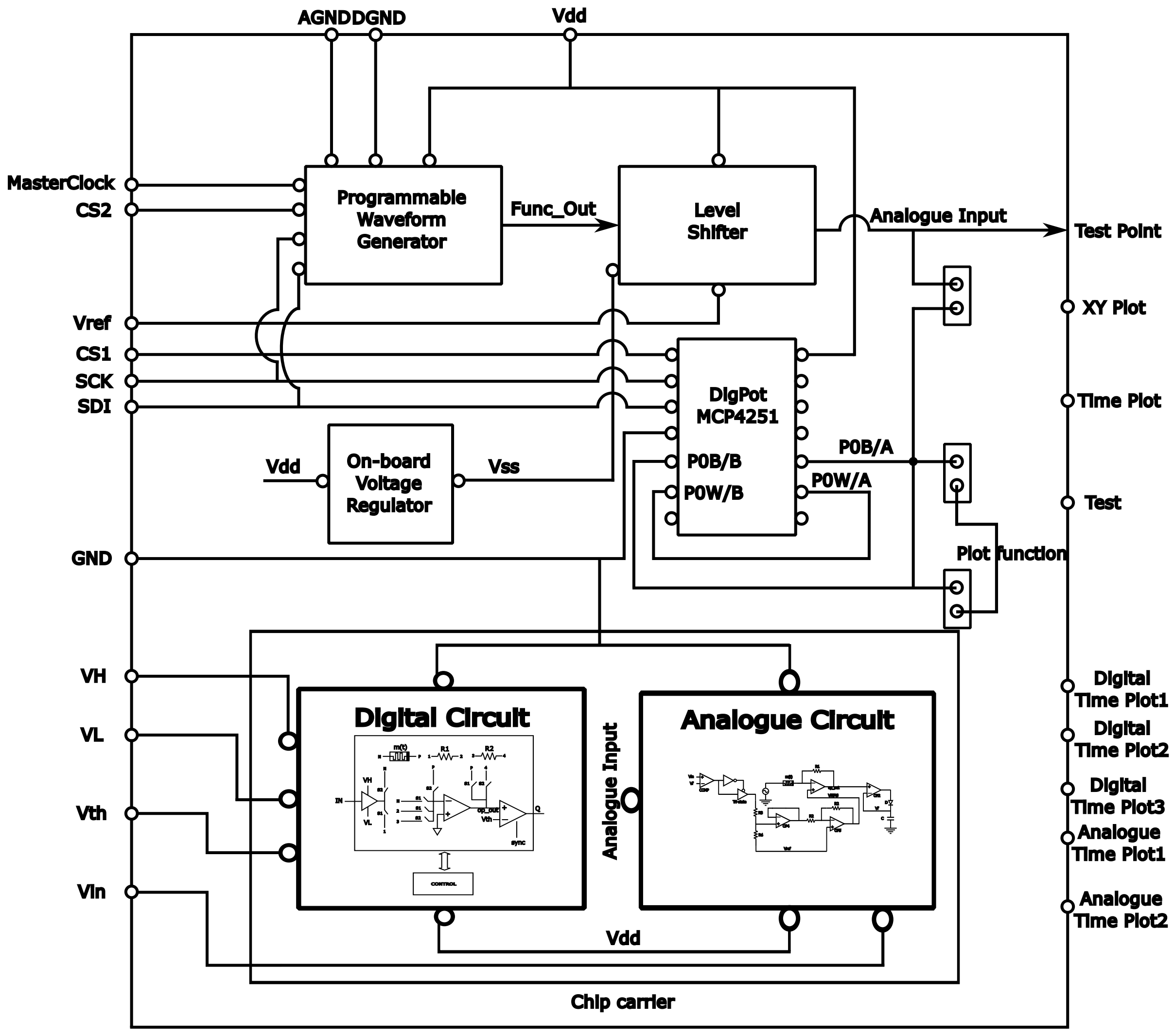


Figure C.2: Programming Circuit Functional Block Diagram

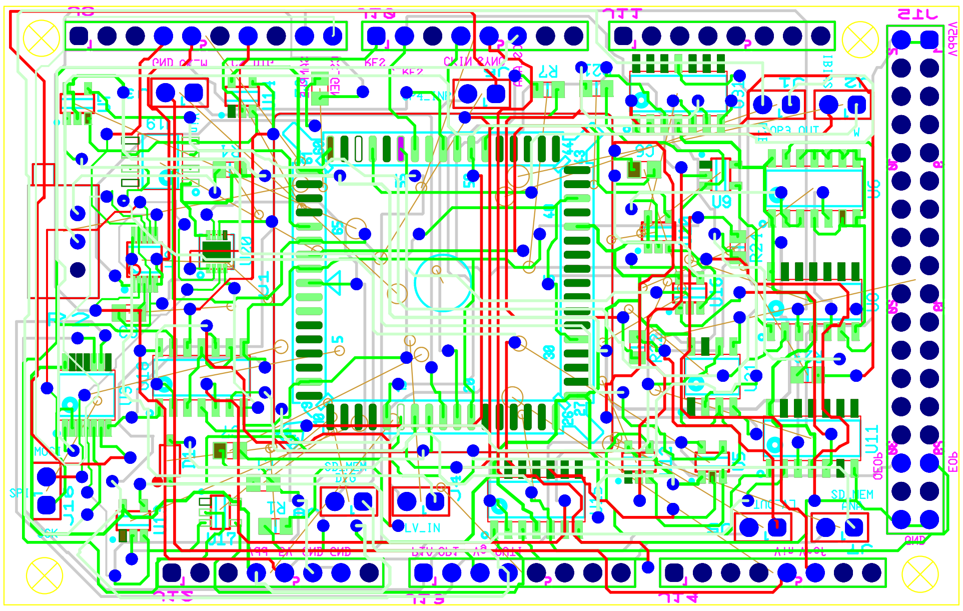


FIGURE C.3: Memristor Programming Circuit PCB Layout

Appendix D

Neural Network PCB Design

D.1 Overview

This document presents the proposed electronics for the design of a printed circuit board (PCB) to be used to implement the two layer Perceptron based on memristive devices. The perception composed of 5 neurons each consisting of CMOS drivers to be interfaced with the memristors, summing block, and transfer function block.

The need of validating our memristor-based neural network for Skin detection in hardware necessitates the building of a compact, non-resource hungry architecture which can be easily reconfigured to suit several other applications. A functional diagram of the proposed architecture is given in Figure D.1. The network can be used to associate an input consisting of two values with one of two decisions or predictions. As a result of this and taking inspiration from the present state of the art, we have come up with a general purpose reconfigurable Perceptron architecture. The architecture is composed of a two layer neural network: an input layer (hidden) and an output layer. The input layer is made up of three neurons while the output layer is made up of two neurons making a total of a five neuron Perceptron network. Each neuron is composed of an addressable input stage, a difference amplifier stage, a transfer function stage and an output stage serving as input to the next layer.

The PCB schematic design of a single neuron is given i Figure D.2. The neuron is composed of CMOS drivers (Blue box), Memristor (shown as resistors here in green box), difference amplifier circuit (red box) and the sigmoid transfer function (yellow box). In1 and In2 are the input signals. A combination of memristors is used to determine the weighted connections. As we already have seen in Chapter 4, memristor's programmability comes in handy here as we can easily tune their memristance over a wide range to achieve different weighted sum. This is why at the input of each neuron, we have the CMOS drivers in order to be able to address each memristor for read, write and reset of their states.

The board was designed with National Instrument (*NI*) Multisim and Ultiboard suites. The layout of the PCB is shown in Figure D.7. The PCB is composed of two

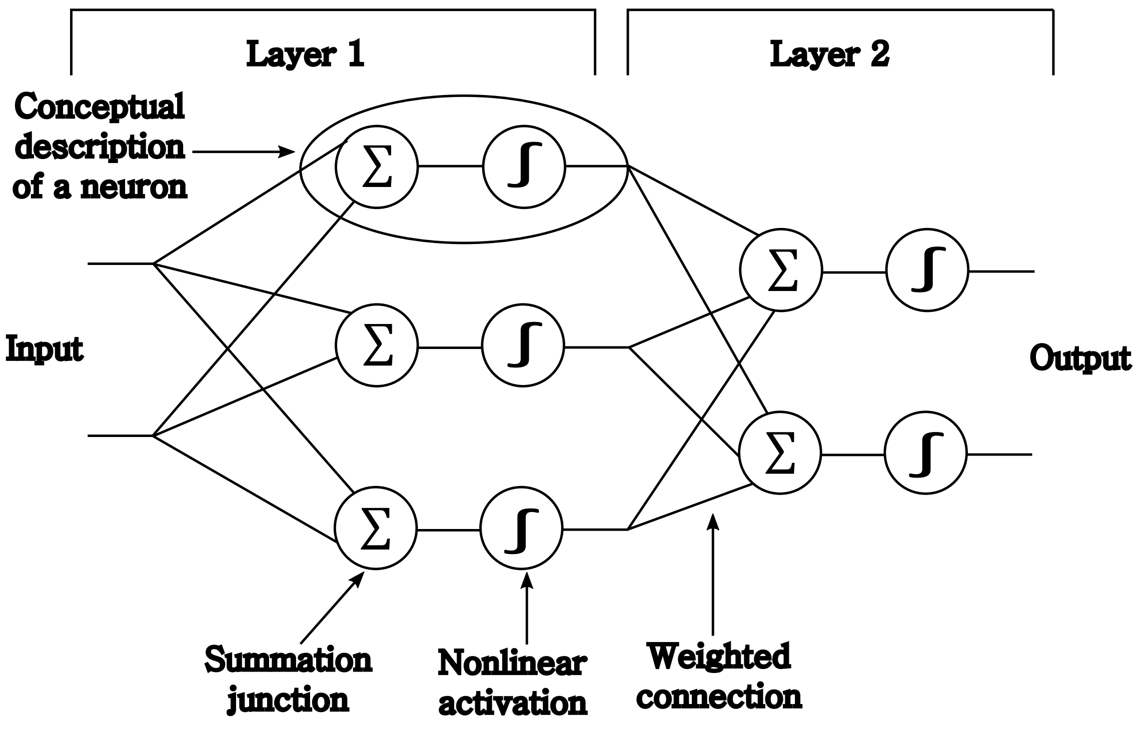


FIGURE D.1: An example of a two-layer neural network that can be used to associate an input consisting of two numbers with one of two decisions or predictions

layers showing visibly, the components mounted, top and bottom layer copper traces, vias and mounted connectors. Although not shown in the figure, the top and bottom layers are covered with solid power planes. The board has mounted connectors to be used to interface the memristive devices. Based on the target application, we can exploit only a couple of neurons present on the board and we will see i subsequent Sections. Small signal traces have been routed to the bottom layer to ensure a low impedance path for any return currents on the bottom layer ground plane. Vias were placed at each components ground connection for the routing of return currents to the bottom plane in order to provide the shortest possible path back to ground. General PCB layout guidelines were followed.

D.2 Component Design Documentation

This entails for each major section of the design flow, the description of each components and how it contributes toward the overall goal of the project.

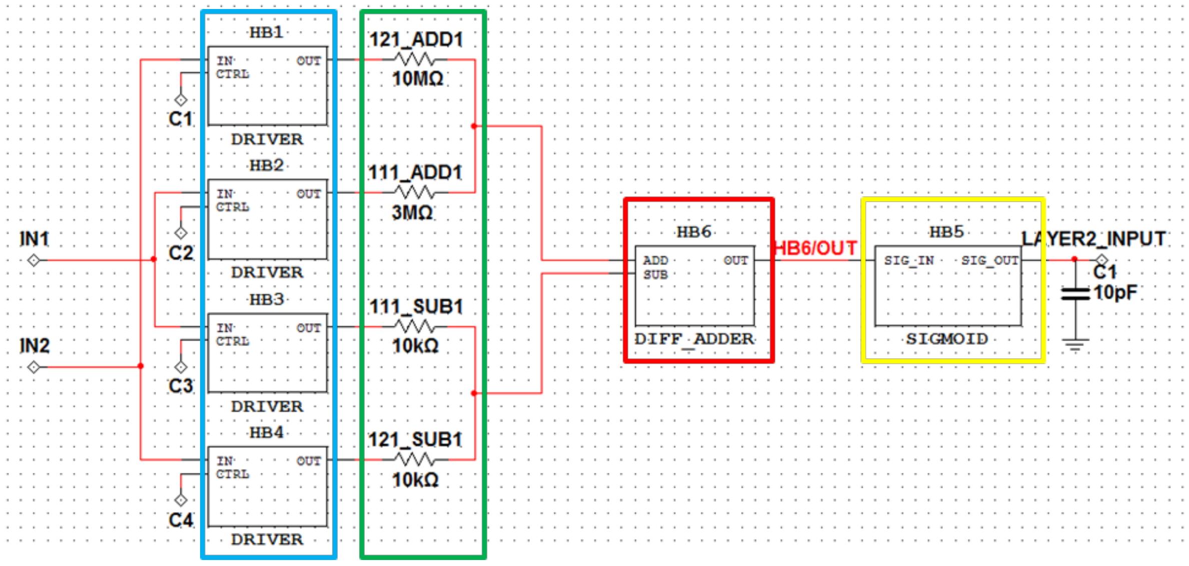


FIGURE D.2: Schematic of a Neuron showing the different stages of the neural network

D.2.1 CMOS Drivers

To respond some certain constraints during read and write operation, the driver is configured as shown in schematic in Figure D.3. The driver consists of 2 input signals (IN for the Perceptron input signal and R/W as read, potentiation or depression voltage). Control signal CTRL is used to select the memristor in which the driver is connected to for read/write (R/W) operation). Control signal OP is used to turn on the switch it's connected to during circuit operation. Signals OP is a complement of CTRL, therefore, they're never closed at the same time, nevertheless they could be both idle (This is the case when one memristor is selected using the appropriate digital control signal, the other memristors are completely disconnected). Signals OP & CTRL are generated internally by the board. CTRL depends on the input combination of the decoder circuit (See Section D.3).

During Perceptron operation, switch OP is closed (switch CTRL remains open), thereby connecting the memristor to the input signal. At the end of the operation, in order to begin reading/potentiation/depression, switch OP turns off (open) and depending on the input combination of the Decoder block either one of the memristors making up the network can be selected through the closing of its corresponding CTRL switch.

NB: Signals R/W is a global signal distributed to all memristors and can be used to read or change the state of the memristors, i.e., it can have different voltage levels depending on the operation that we want to perform (read or write). Before selecting any memristor, we are sure to have set the proper read voltage level in order to avoid corrupting the state of the memristor undesirably. E.g., If the conductivity of memristor

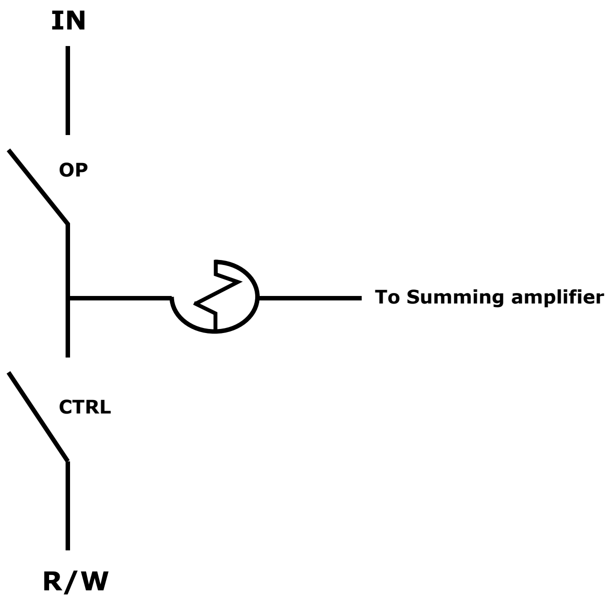


Figure D.3: Neuron Input Drivers: An illustration of how the memristor is connected to the CMOS switches

M_1 has been increased by selecting it and applying a R/W, and memristor M_2 is then selected without turning off or changing R/W to the appropriate Read voltage level, we would have systematically altered the state of memristor M_2 even before reading its value. Therefore, before selecting another memristor, we are sure to have verified the voltage level of R/W.

D.2.2 Summing Amplifying Stage

This stage is direct and obviously clear. The schematic of the difference adder is given in Figure D.4 utilising 3 operational amplifiers. We use this configuration to guarantee the READ/WRITE operation of the memristors by summing the positive and negative weights independently and using a third Op-amp to get the difference. The value of the feedback resistors are set to $200K\Omega$ to account for a good portion of the dynamic range of the memristors. Equation D.1 is the voltage at the output of the ADDER.

$$V_{OUT} = i_{SUB}R2 + i_{ADD}R1 \tag{D.1}$$

V_{OUT} is deliberately inverted here because of the kind of sigmoid function we're implementing. The transfer function in-turn flips the signal back to its intended polarity.

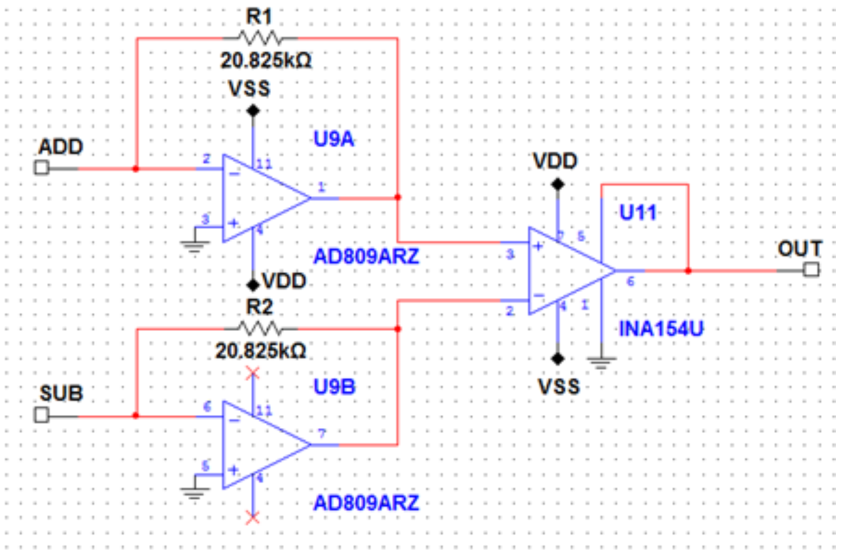


FIGURE D.4: Schematic of the Difference Amplifying stage. Hierarchical Block 6(HB6) of Figure D.2

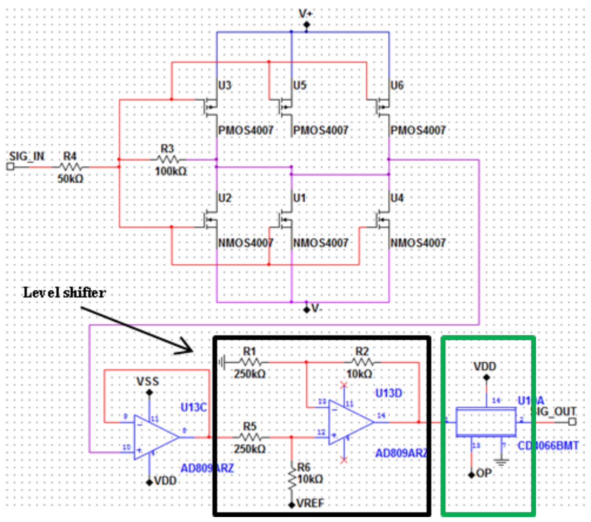
D.2.3 Transfer Function

Figure D.5a shows the schematic of the sigmoid function having 1 input pin (SIG_IN) and 1 output pin (SIG_OUT). The sigmoid receives on the input a voltage ranging between the output swings of the summing stage (VSS - VDD) and outputs a voltage between our input signal dynamic range to the next layer. This is achieved thanks to the in-built level shifter (Black box). A switch (Green box) is connected at the output of the sigmoid function which is controlled by the same signal (OP) in Figure D.3. This makes sure the output of each neuron is disconnected from the next layer during R/W operation. The slope of the sigmoid function can be adjusted by varying the ratio of the input and feed-back resistors R4:R3 as observed in Figure D.5b. The default configuration on the board is 1:4 with R4 $= 30k\Omega$ & R3 $= 120K\Omega$.

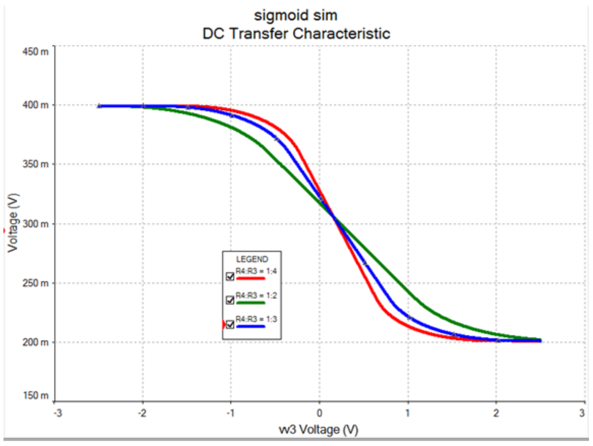
D.3 Decoder

The five neuron, two-layer neural network architecture which permits us to associate an input (IN1 and IN2) with one of two decisions. This implies that for two input values, two possible output, for a five neuron, two-layer neural network, we would require 24 memristors to store the weighted sums for the overall computation. Therefore, we should be able to address each of the memristive components individually in order to be able to change their states.

Addressing each of the 24 memristors has been realised thanks to the cascading three 3—8 decoders as shown in Figure D.6 to form a 5-–24 demultiplexer. The decoder circuit



(a)



(b)

Figure D.5: Sigmoid Transfer Function. **(a)**, Schematic of the Sigmoid Function. **(b)**, Simulation of the Sigmoid Function for different resistor ratios.

Table D.1: Decoder Control Signal Combination.

Select Combination	**Selected memristors**
00000 – 01000	One of 1 – 24 memristors. Used for Reading/Writing
11XXX	No memristor selected. Network completely disconnected

consists of an Enable signal, a 5-bit select combination (A4, A3, A2, A1, A0) and 24 output signals (C1 – C24) connected to switches CTRL of the drivers.

With signal EN set to V_L, OP is automatically set to V_H, this implies that all the memristors are connected to the Perceptron circuitry and the output of each neuron can be measured. Setting EN to V_H disconnects the output of the neurons; all the memristors are also disconnected except the one selected based on the combination of the select signal (A0 – A4). Table D.1 summarises the control of the a 5-bit select combination.

D.4 Generated MATLAB Neural Network Function

Code D.1: myNeuralNetworkFunction.m

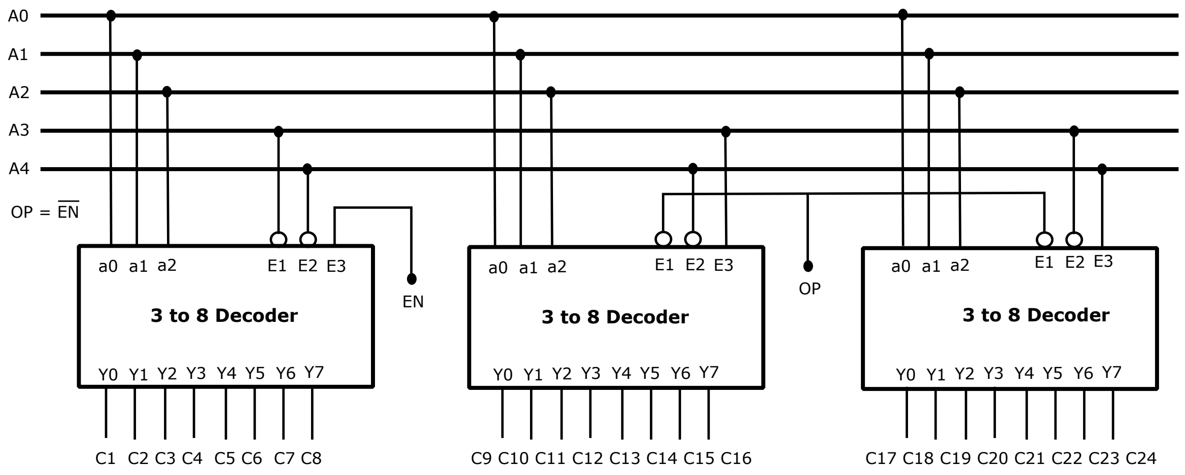


FIGURE D.6: 5–24 Decoder/Demultiplexer block diagram used for addressing the memristors in the neural network

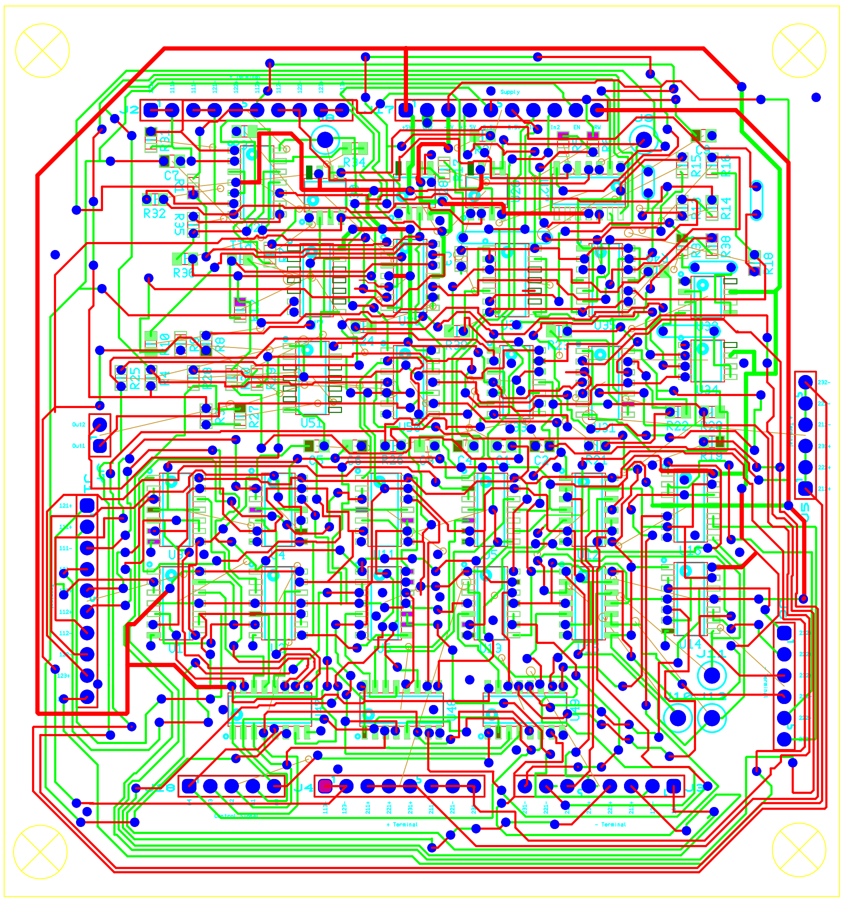


FIGURE D.7: Memristor-based Neural Network PCB design Layout

```
function [y1] = NNBlockE2N(x1)

% ===== NEURAL NETWORK CONSTANTS =====

% Input 1
x1_step1.xoffset = [0.13073;0.18929];
x1_step1.gain = [3.64590928977687;
5.40423692174665];
x1_step1.ymin = -1;

% Layer 1
b1 = [-16.100860162185128;-1.4419282285538586];
IW1_1 = [-8.844988676556925 -30.071703507584989;
16.68538715921623 -28.811472235359169];

% Layer 2
b2 = -0.98520600057947838;
LW2_1 = [-0.96776674572964749 0.96811446907090215];

% Output 1
y1_step1.ymin = -1;
y1_step1.gain = 2;
y1_step1.xoffset = 0;

% ===== SIMULATION ========

% Dimensions
Q = size(x1,2); % samples

% Input 1
xp1 = mapminmax_apply(x1,x1_step1);

% Layer 1
a1 = tansig_apply(repmat(b1,1,Q) + IW1_1*xp1);

% Layer 2
a2 = repmat(b2,1,Q) + LW2_1*a1;

% Output 1
y1 = mapminmax_reverse(a2,y1_step1);
end

% ===== MODULE FUNCTIONS ========

% Map Minimum and Maximum
%Input Processing Function
```

```
function y = mapminmax_apply(x,settings)
y = bsxfun(@minus,x,settings.xoffset);
y = bsxfun(@times,y,settings.gain);
y = bsxfun(@plus,y,settings.ymin);
end

% Sigmoid Symmetric Transfer Function
function a = tansig_apply(n,~)
a = 2 ./ (1 + exp(-2*n)) - 1;
end

% Map Minimum and Maximum Output
%Reverse-Processing Function
function x = mapminmax_reverse(y,settings)
x = bsxfun(@minus,y,settings.ymin);
x = bsxfun(@rdivide,x,settings.gain);
x = bsxfun(@plus,x,settings.xoffset);
end
```

Bibliography

[1] L. Chua. “Memristor - the missing circuit element”. In: *IEEE Trans. Circuit Theory* 18 (1971), pp. 517–519.

[2] Charles A Desoer. *Basic circuit theory.* Tata McGraw-Hill Education, 2009.

[3] D. C. Chang and J. X. Zheng. “Electromagnetic modeling of passive circuit elements in MMIC”. In: *IEEE Transactions on Microwave Theory and Techniques* 40.9 (1992), pp. 1741–1747. ISSN: 0018-9480. DOI: 10.1109/22.156600.

[4] Bharathwaj Muthuswamy and Leon O Chua. “Simplest chaotic circuit”. In: *International Journal of Bifurcation and Chaos* 20.05 (2010), pp. 1567–1580. DOI: 10.1142/S0218127410027076.

[5] Bharathwaj Muthuswamy and Pracheta P. Kokate. “Memristor-Based Chaotic Circuits”. In: *IETE Technical Review* 26.6 (2009), pp. 417–429. DOI: 10.4103/0256-4602.57827. eprint: http://www.tandfonline.com/doi/pdf/10.4103/0256-4602.57827. URL: http://www.tandfonline.com/doi/abs/10.4103/0256-4602.57827.

[6] Dmitri B Strukov et al. “The missing memristor found”. In: *nature* 453.7191 (2008), pp. 80–83. DOI: 10.1038/nature06932.

[7] R. S. Williams. “How We Found The Missing Memristor”. In: *IEEE Spectrum* 45.12 (2008), pp. 28–35. ISSN: 0018-9235. DOI: 10.1109/MSPEC.2008.4687366.

[8] R. Stanley Williams. “How We Found the Missing Memristor”. In: *Memristors and Memristive Systems.* Ed. by Ronald Tetzlaff. New York, NY: Springer New York, 2014, pp. 3–16. ISBN: 978-1-4614-9068-5. DOI: 10.1007/978-1-4614-9068-5_1. URL: https://doi.org/10.1007/978-1-4614-9068-5_1.

[9] Leon Chua. “Resistance switching memories are memristors”. In: *Applied Physics A* 102.4 (2011), pp. 765–783. ISSN: 1432-0630. DOI: 10.1007/s00339-011-6264-9. URL: https://doi.org/10.1007/s00339-011-6264-9.

[10] T. W. Lee and J. H. Nickel. “Memristor Resistance Modulation for Analog Applications”. In: *IEEE Electron Device Letters* 33.10 (2012), pp. 1456–1458. ISSN: 0741-3106. DOI: 10.1109/LED.2012.2207429.

[11] O. A. Olumodeji and M. Gottardi. “Behavioural modelling of memristive devices targeted to sensor interfaces”. In: *AISEM.* IEEE. 2015, pp. 1–4.

[12] Olufemi A Olumodeji and Massimo Gottardi. "Emulating the physical properties of HP memristor using an arduino and a digital potentiometer". In: *Ph. D. Research in Microelectronics and Electronics (PRIME), 2016 12th Conference on*. IEEE. 2016, pp. 1–4.

[13] Olufemi Akindele Olumodeji and Massimo Gottardi. "Arduino-controlled HP memristor emulator for memristor circuit applications". In: *Integration, the VLSI Journal* 58 (2017), pp. 438 –445. ISSN: 0167-9260. DOI: `http://dx.doi.org/10.1016/j.vlsi.2017.03.004`. URL: `http://www.sciencedirect.com/science/article/pii/S0167926017301499`.

[14] O. A. Olumodeji and M. Gottardi. "A pulse-based memristor programming circuit". In: *2017 IEEE International Symposium on Circuits and Systems (ISCAS)*. 2017, pp. 1–4. DOI: `10.1109/ISCAS.2017.8050793`.

[15] O. A. Olumodeji and M. Gottardi. "A Pulse-based Memristor programming Circuit". In: *Circuits and Systems (ISCAS), 2017 IEEE International Symposium on*. IEEE. 2017, pp. 232–235. DOI: `10.1109/PRIME.2015.7251377`.

[16] O. A. Olumodeji and M. Gottardi. "Memristor-based comparator with programmable hysteresis". In: *Microelectronics and Electronics (PRIME), 2015 11th Conference on Ph. D. Research in*. IEEE. 2015, pp. 232–235. DOI: `10.1109/PRIME.2015.7251377`.

[17] Olufemi A Olumodeji, Alessandro Paolo Bramanti, and Massimo Gottardi. "A memristor-based pixel implementing light-to-resistance conversion". In: *Optical Engineering* 55.2 (2016), pp. 020501–020501. DOI: `10.1117/1.OE.55.2.020501`.

[18] Olufemi Akindele Olumodeji, Alessandro Paolo Bramanti, and Massimo Gottardi. "A Memristive Pixel Architecture for Real-Time Tracking". In: *IEEE Sensors Journal* 16.22 (2016), pp. 7911–7918. DOI: `10.1109/JSEN.2016.2606599`.

[19] O. A. Olumodeji, A. P. Bramanti, and M. Gottardi. "Memristor-based pixel for event-detection vision sensor". In: *SENSORS, 2015 IEEE*. 2015, pp. 1–4. DOI: `10.1109/ICSENS.2015.7370688`.

[20] Olufemi A Olumodeji et al. "Estimating illuminant chromaticity with a low-power color pixel". In: *AISEM Annual Conference, 2015 XVIII*. IEEE. 2015, pp. 1–4. DOI: `10.1109/AISEM.2015.7066815`.

[21] Leon O Chua. "Nonlinear circuit foundations for nanodevices. I. The four-element torus". In: *Proceedings of the IEEE* 91.11 (2003), pp. 1830–1859.

[22] James M Tour and Tao He. "Electronics: the fourth element". In: *Nature* 453.7191 (2008), pp. 42–43.

[23] Deyan Lin, Leon Chua, and Shu-Yuen Hui. "The First Man-Made Memristor: Circa 1801 [Scanning Our Past]". In: *Proceedings of the IEEE* 103.1 (2015), pp. 131–136.

[24] Themistoklis Prodromakis, Christofer Toumazou, and Leon Chua. "Two centuries of memristors". In: *Nature materials* 11.6 (2012), pp. 478–481.

[25] OA Ageev et al. "Memristor effect on bundles of vertically aligned carbon nanotubes tested by scanning tunnel microscopy". In: *Technical Physics* 58.12 (2013), pp. 1831–1836.

[26] André Chanthbouala et al. "A ferroelectric memristor". In: *Nature materials* 11.10 (2012), pp. 860–864.

[27] Xiaobin Wang et al. "Spintronic memristor through spin-torque-induced magnetization motion". In: *IEEE electron device letters* 30.3 (2009), pp. 294–297.

[28] Leon O Chua and Sung Mo Kang. "Memristive devices and systems". In: *Proceedings of the IEEE* 64.2 (1976), pp. 209–223.

[29] Yuriy V Pershin and Massimiliano Di Ventra. "Memory effects in complex materials and nanoscale systems". In: *Advances in Physics* 60.2 (2011), pp. 145–227.

[30] Yogesh N Joglekar and Stephen J Wolf. "The elusive memristor: properties of basic electrical circuits". In: *European Journal of Physics* 30.4 (2009), p. 661. URL: `http://stacks.iop.org/0143-0807/30/i=4/a=001`.

[31] Juraj Valsa, Dalibor Biolek, and Zdeněk Biolek. "An analogue model of the memristor". In: *International Journal of Numerical Modelling: Electronic Networks, Devices and Fields* 24.4 (2011), pp. 400–408. DOI: `10.1002/jnm.786`.

[32] Yenpo Ho, Garng M Huang, and Peng Li. "Dynamical properties and design analysis for nonvolatile memristor memories". In: *IEEE Transactions on Circuits and Systems I: Regular Papers* 58.4 (2011), pp. 724–736.

[33] Joseph Blanc and David L Staebler. "Electrocoloration in srti o 3: Vacancy drift and oxidation-reduction of transition metals". In: *Physical Review B* 4.10 (1971), p. 3548.

[34] Victor Erokhin and Marco P Fontana. "Electrochemically controlled polymeric device: a memristor (and more) found two years ago". In: *arXiv preprint arXiv:0807.0333* (2008).

[35] G Baldi et al. "Logic with memory: and gates made of organic and inorganic memristive devices". In: *Semiconductor Science and Technology* 29.10 (2014), p. 104009. DOI: `10.1088/0268-1242/29/10/104009`. URL: `http://stacks.iop.org/0268-1242/29/i=10/a=104009`.

[36] ET Kang, KG Neoh, and KL Tan. “Polyaniline: a polymer with many interesting intrinsic redox states”. In: *Progress in Polymer Science* 23.2 (1998), pp. 277–324. DOI: 10.1016/S0079-6700(97)00030-0.

[37] Tatiana Berzina, Victor Erokhin, and MP Fontana. “Spectroscopic investigation of an electrochemically controlled conducting polymer-solid electrolyte junction”. In: *Journal of applied physics* 101.2 (2007), p. 024501. DOI: 10.1063/1.2422750.

[38] V. Erokhin and M. P. Fontana. “Organic memristive device and its application for the information processing”. In: *2010 17th IEEE International Conference on Electronics, Circuits and Systems*. 2010, pp. 926–929. DOI: 10.1109/ICECS.2010.5724664.

[39] Victor Erokhin et al. “Conducting polymer—solid electrolyte fibrillar composite material for adaptive networks”. In: *Soft Matter* 2.10 (2006), pp. 870–874. DOI: 10.1039/B606893F.

[40] Tatiana Berzina et al. “Role of the solid electrolyte composition on the performance of a polymeric memristor”. In: *Materials Science and Engineering: C* 30.3 (2010), pp. 407 –411. ISSN: 0928-4931. DOI: http://dx.doi.org/10.1016/j.msec.2009.12.010. URL: http://www.sciencedirect.com/science/article/pii/S0928493109003282.

[41] V. Erokhin and M. P. Fontana. “Electrochemically controlled polymeric device: a memristor (and more) found two years ago”. In: *ArXiv e-prints* (July 2008). arXiv: 0807.0333 [cond-mat.soft].

[42] Victor Erokhin et al. “Bio-inspired adaptive networks based on organic memristors”. In: *Nano Communication Networks* 1.2 (2010), pp. 108 –117. ISSN: 1878-7789. DOI: http://dx.doi.org/10.1016/j.nancom.2010.05.002. URL: http://www.sciencedirect.com/science/article/pii/S1878778910000098.

[43] VICTOR EROKHIN, GERARD DAVID HOWARD, and ANDREW ADAMATZKY. “ORGANIC MEMRISTOR DEVICES FOR LOGIC ELEMENTS WITH MEMORY”. In: *International Journal of Bifurcation and Chaos* 22.11 (2012), p. 1250283. DOI: 10.1142/S0218127412502835. eprint: http://www.worldscientific.com/doi/pdf/10.1142/S0218127412502835. URL: http://www.worldscientific.com/doi/abs/10.1142/S0218127412502835.

[44] Victor Erokhin et al. “Material Memristive Device Circuits with Synaptic Plasticity: Learning and Memory”. In: *BioNanoScience* 1.1 (2011), pp. 24–30. ISSN: 2191-1649. DOI: 10.1007/s12668-011-0004-7. URL: https://doi.org/10.1007/s12668-011-0004-7.

[45] V. Erokhin. “Organic memristors : Basic principles”. In: *Proceedings of 2010 IEEE International Symposium on Circuits and Systems*. 2010, pp. 5–8. DOI: 10.1109/ISCAS.2010.5537145.

[46] Victor Erokhin, Tatiana Berzina, and Marco P Fontana. “Hybrid electronic device based on polyaniline-polyethyleneoxide junction”. In: *Journal of applied physics* 97.6 (2005), p. 064501. DOI: 10.1063/1.1861508.

[47] Victor Erokhin and MP Fontana. “Thin film electrochemical memristive systems for bio-inspired computation”. In: *Journal of Computational and Theoretical Nanoscience* 8.3 (2011), pp. 313–330. DOI: 10.1166/jctn.2011.1695.

[48] JA McGeough et al. “Electroforming process and application to micro/macro manufacturing”. In: *CIRP Annals-Manufacturing Technology* 50.2 (2001), pp. 499–514. DOI: 10.1016/S0007-8506(07)62990-4.

[49] C Nauenheim et al. “Investigation of the electroforming process in resistively switching TiO 2 nanocrosspoint junctions”. In: *Applied Physics Letters* 96.12 (2010), p. 122902. DOI: 10.1063/1.3367752.

[50] T Menke et al. “Impact of the electroforming process on the device stability of epitaxial Fe-doped SrTiO 3 resistive switching cells”. In: *Journal of Applied Physics* 106.11 (2009), p. 114507. DOI: 10.1063/1.3267485.

[51] Sung Hyun Jo et al. “Nanoscale memristor device as synapse in neuromorphic systems”. In: *Nano letters* 10.4 (2010), pp. 1297–1301. DOI: 10.1021/nl904092h.

[52] J Joshua Yang et al. “The mechanism of electroforming of metal oxide memristive switches”. In: *Nanotechnology* 20.21 (2009), p. 215201. URL: http://stacks.iop.org/0957-4484/20/i=21/a=215201.

[53] K Szot et al. “TiO2—a prototypical memristive material”. In: *Nanotechnology* 22.25 (2011), p. 254001.

[54] Doo Seok Jeong, Herbert Schroeder, and Rainer Waser. “Coexistence of bipolar and unipolar resistive switching behaviors in a Pt/ TiO2/ Pt stack”. In: *Electrochemical and solid-state letters* 10.8 (2007), G51–G53.

[55] Woo Young Park et al. “A Pt/TiO2/Ti Schottky-type selection diode for alleviating the sneak current in resistance switching memory arrays”. In: *Nanotechnology* 21.19 (2010), p. 195201.

[56] J Joshua Yang, Dmitri B Strukov, and Duncan R Stewart. “Memristive devices for computing”. In: *Nature nanotechnology* 8.1 (2013), pp. 13–24.

[57] Omid Kavehei et al. “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*. The Royal Society. 2010, rspa20090553. DOI: 10.1098/rspa.2009.0553.

[58] Maheshwar Pd Sah et al. “A voltage mode memristor bridge synaptic circuit with memristor emulators”. In: *Sensors* 12.3 (2012), pp. 3587–3604. DOI: 10.3390/s120303587.

[59] Zdenek Biolek, Dalibor Biolek, and Viera Biolkova. “SPICE model of memristor with nonlinear dopant drift”. In: *Radioengineering* 18.2 (2009), pp. 210–214.

[60] Leon Chua. “Memristors: Past, Present and Future”. In: *IEEE PROCEEDINGS*. 2012, pp. 1–2.

[61] *Cadence Verilog-A language reference Manual*. Version 5.0 Open Verilog International. 2006.

[62] Hyongsuk Kim, Maheshwar Pd Sah, and Shyam Prasad Adhikari. “Pinched hysteresis loops is the fingerprint of memristive devices”. In: *arXiv preprint arXiv:1202.2437* (2012).

[63] Yuriy Pershin and Sergey Shevchenko. “Computing with volatile memristors: An application of non-pinched hysteresis”. In: *Nanotechnology* (2016).

[64] Sangho Shin, Kyungmin Kim, and Sung-Mo Kang. “Memristor applications for programmable analog ICs”. In: *Nanotechnology, IEEE Transactions on* 10.2 (2011), pp. 266–274. DOI: 10.1109/TNANO.2009.2038610.

[65] Chris Yakopcic et al. “Memristor-based unit cell for a detector readout circuit”. In: *SPIE Optical Engineering+ Applications*. International Society for Optics and Photonics. 2011, 81651F–81651F. DOI: 10.1117/12.895951.

[66] Dalibor Biolek, Zdenek Biolek, and Viera Biolkova. “SPICE modeling of memristive, memcapacitative and meminductive systems”. In: *Circuit Theory and Design, 2009. ECCTD 2009. European Conference on*. IEEE. 2009, pp. 249–252. DOI: 10.1109/ECCTD.2009.5274934.

[67] Radu Berdan et al. “A memristor SPICE model accounting for volatile characteristics of practical ReRAM”. In: *IEEE Electron Device Letters* 35.1 (2014), pp. 135–137. DOI: 10.1109/LED.2013.2291158.

[68] Alon Ascoli et al. “PSpice switch-based versatile memristor model”. In: *Circuits and Systems (ISCAS), 2013 IEEE International Symposium on*. IEEE. 2013, pp. 205–208. DOI: 10.1109/ISCAS.2013.6571818.

[69] Ádám Rák and György Cserey. “Macromodeling of the memristor in SPICE”. In: *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* 29.4 (2010), pp. 632–636. DOI: `10.1109/TCAD.2010.2042900`.

[70] S Benderli and TA Wey. “On SPICE macromodelling of TiO 2 memristors”. In: *Electronics letters* 45.7 (2009), pp. 377–379. DOI: `10.1049/el.2009.3511`.

[71] Mohammad Javad Sharifi and Yasser Mohammadi Banadaki. “General SPICE models for memristor and application to circuit simulation of memristor-based synapses and memory cells”. In: *Journal of Circuits, Systems, and Computers* 19.02 (2010), pp. 407–424. DOI: `10.1142/S0218126610006141`.

[72] Yu Zhang, Xuliang Zhang, and Juebang Yu. “Approximated SPICE model for memristor”. In: *Communications, Circuits and Systems, 2009. ICCCAS 2009. International Conference on*. IEEE. 2009, pp. 928–931. DOI: `10.1109/ICCCAS.2009.5250371`.

[73] Daniel Batas and Horst Fiedler. “A memristor SPICE implementation and a new approach for magnetic flux-controlled memristor modeling”. In: *IEEE Transactions on Nanotechnology* 10.2 (2011), pp. 250–255. DOI: `10.1109/TNANO.2009.2038051`.

[74] Sangho Shin, Kyungmin Kim, and Sung-Mo Kang. “Compact models for memristors based on charge-flux constitutive relationships”. In: *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* 29.4 (2010), pp. 590–598. DOI: `10.1109/TCAD.2010.2042891`.

[75] Şuayb Yener and Hakan Kuntman. “A new CMOS based memristor implementation”. In: *Applied Electronics (AE), 2012 International Conference on*. IEEE. 2012, pp. 345–348.

[76] Ahmed I Hussein and Mohamed E Fouda. “A simple MOS realization of current controlled memristor emulator”. In: *2013 25th International Conference on Microelectronics (ICM)*. IEEE. 2013, pp. 1–4.

[77] Hasan Sözen and Uğur Çam. “New memristor emulator circuit using OTAs and CCIIs”. In: ().

[78] Carlos Sánchez-López et al. “A floating analog memristor emulator circuit”. In: *Circuits and Systems II: Express Briefs, IEEE Transactions on* 61.5 (2014), pp. 309–313.

[79] Abdullah G Alharbi, Mohammed E Fouda, and Masud H Chowdhury. “Memristor emulator based on practical current controlled model”. In: *2015 IEEE 58th International Midwest Symposium on Circuits and Systems (MWSCAS)*. IEEE. 2015, pp. 1–4.

[80] Hyongsuk Kim et al. “Memristor emulator for memristor circuit applications”. In: *Circuits and Systems I: Regular Papers, IEEE Transactions on* 59.10 (2012), pp. 2422–2431.

[81] Yuriy V Pershin and Massimiliano Di Ventra. “Practical approach to programmable analog circuits with memristors”. In: *Circuits and Systems I: Regular Papers, IEEE Transactions on* 57.8 (2010), pp. 1857–1864.

[82] MegunolinkPro. URL: `http://www.megunolink.com/`.

[83] Leon O Chua. “The fourth element”. In: *Proceedings of the IEEE* 100.6 (2012), pp. 1920–1927.

[84] R Berdan, T Prodromakis, and C Toumazou. “High precision analogue memristor state tuning”. In: *Electronics letters* 48.18 (2012), pp. 1105–1107. DOI: `10.1049/el.2012.2295`.

[85] I. G. Baek et al. “Highly scalable nonvolatile resistive memory using simple binary oxide driven by asymmetric unipolar voltage pulses”. In: *IEDM Technical Digest. IEEE International Electron Devices Meeting, 2004.* 2004, pp. 587–590. DOI: `10.1109/IEDM.2004.1419228`.

[86] Julien Borghetti et al. “A hybrid nanomemristor/transistor logic circuit capable of self-programming”. In: *Proceedings of the National Academy of Sciences* 106.6 (2009), pp. 1699–1703. DOI: `10.1073/pnas.0806642106`. eprint: `http://www.pnas.org/content/106/6/1699.full.pdf`. URL: `http://www.pnas.org/content/106/6/1699.abstract`.

[87] H. Kim et al. “Neural Synaptic Weighting With a Pulse-Based Memristor Circuit”. In: *IEEE Transactions on Circuits and Systems I: Regular Papers* 59.1 (2012), pp. 148–158. ISSN: 1549-8328. DOI: `10.1109/TCSI.2011.2161360`.

[88] Sungho Kim, ShinHyun Choi, and Wei Lu. “Comprehensive Physical Model of Dynamic Resistive Switching in an Oxide Memristor”. In: *ACS Nano* 8.3 (2014). PMID: 24571386, pp. 2369–2376. DOI: `10.1021/nn405827t`. eprint: `http://dx.doi.org/10.1021/nn405827t`. URL: `http://dx.doi.org/10.1021/nn405827t`.

[89] Yenpo Ho, Garng M. Huang, and Peng Li. “Nonvolatile Memristor Memory: Device Characteristics and Design Implications”. In: *Proceedings of the 2009 International Conference on Computer-Aided Design.* ICCAD ’09. San Jose, California: ACM, 2009, pp. 485–490. ISBN: 978-1-60558-800-1. DOI: `10.1145/1687399.1687491`. URL: `http://doi.acm.org/10.1145/1687399.1687491`.

[90] Dragan Mihailovic. “Ultrafast optical switching between hidden states of electronic matter under non-equilibrium conditions”. In: *Conference on Lasers and Electro-Optics*. Optical Society of America, 2016, FTu1L.4. DOI: 10.1364/CLEO_QELS.2016.FTu1L.4. URL: http://www.osapublishing.org/abstract.cfm?URI=CLEO_QELS-2016-FTu1L.4.

[91] C. Xu et al. “Design implications of memristor-based RRAM cross-point structures”. In: *2011 Design, Automation Test in Europe*. 2011, pp. 1–6. DOI: 10.1109/DATE.2011.5763125.

[92] Antonio C Torrezan et al. “Sub-nanosecond switching of a tantalum oxide memristor”. In: *Nanotechnology* 22.48 (2011), p. 485203. URL: http://stacks.iop.org/0957-4484/22/i=48/a=485203.

[93] Yong-En Syu et al. “Atomic-level quantized reaction of HfOx memristor”. In: *Applied Physics Letters* 102.17 (2013), p. 172903. DOI: 10.1063/1.4802821.

[94] Farnood Merrikh-Bayat and Saeed Bagheri Shouraki. “Programming of memristor crossbars by using genetic algorithm”. In: *Procedia Computer Science* 3 (2011). World Conference on Information Technology, pp. 232 –237. ISSN: 1877-0509. DOI: http://dx.doi.org/10.1016/j.procs.2010.12.039. URL: http://www.sciencedirect.com/science/article/pii/S187705091000414X.

[95] Fabien Alibart et al. “High precision tuning of state for memristive devices by adaptable variation-tolerant algorithm”. In: *Nanotechnology* 23.7 (2012), p. 075201. URL: http://stacks.iop.org/0957-4484/23/i=7/a=075201.

[96] M. K. Qureshi, M. M. Franceschini, and L. A. Lastras-Montaño. “Improving read performance of Phase Change Memories via Write Cancellation and Write Pausing”. In: *HPCA - 16 2010 The Sixteenth International Symposium on High-Performance Computer Architecture*. 2010, pp. 1–11. DOI: 10.1109/HPCA.2010.5416645.

[97] C. Yakopcic and T. M. Taha. “Energy efficient perceptron pattern recognition using segmented memristor crossbar arrays”. In: *The 2013 International Joint Conference on Neural Networks (IJCNN)*. 2013, pp. 1–8. DOI: 10.1109/IJCNN.2013.6707073.

[98] C. J. Xue et al. “Emerging non-volatile memories: Opportunities and challenges”. In: *2011 Proceedings of the Ninth IEEE/ACM/IFIP International Conference on Hardware/Software Codesign and System Synthesis (CODES+ISSS)*. 2011, pp. 325–334. DOI: 10.1145/2039370.2039420.

[99] Y. Kim, Y. Zhang, and P. Li. “A digital neuromorphic VLSI architecture with memristor crossbar synaptic array for machine learning”. In: *2012 IEEE International SOC Conference*. 2012, pp. 328–333. DOI: 10.1109/SOCC.2012.6398336.

[100] P. Pouyan, E. Amat, and A. Rubio. “Reliability challenges in design of memristive memories”. In: *2014 5th European Workshop on CMOS Variability (VARI)*. 2014, pp. 1–6. DOI: 10.1109/VARI.2014.6957074.

[101] P. Knag, W. Lu, and Z. Zhang. “A Native Stochastic Computing Architecture Enabled by Memristors”. In: *IEEE Transactions on Nanotechnology* 13.2 (2014), pp. 283–293. ISSN: 1536-125X. DOI: 10.1109/TNANO.2014.2300342.

[102] Ligang Gao et al. “Fully parallel write/read in resistive synaptic array for accelerating on-chip learning”. In: *Nanotechnology* 26.45 (2015), p. 455204. URL: http://stacks.iop.org/0957-4484/26/i=45/a=455204.

[103] J. Rajendran, R. Karri, and G. S. Rose. “Improving Tolerance to Variations in Memristor-Based Applications Using Parallel Memristors”. In: *IEEE Transactions on Computers* 64.3 (2015), pp. 733–746. ISSN: 0018-9340. DOI: 10.1109/TC.2014.2308189.

[104] Hyongsuk Kim et al. “Memristor-based multilevel memory”. In: *Cellular nanoscale networks and their applications (CNNA), 2010 12th international workshop on.* IEEE. 2010, pp. 1–6.

[105] Farnood Merrikh-Bayat and Saeed Bagheri Shouraki. “Memristor-based circuits for performing basic arithmetic operations”. In: *Procedia Computer Science* 3 (2011), pp. 128–132. DOI: 10.1016/j.procs.2010.12.022.

[106] Farshad Merrikh-Bayat, Farnood Merrikh-Bayat, and Nafise Mirebrahimi. “A method for automatic tuning the memristance of memristive devices with the capacity of applying to memristive memories”. In: *Computer Systems and Industrial Informatics (ICCSII), 2012 International Conference on.* IEEE. 2012, pp. 1–6. DOI: 10.1109/ICCSII.2012.6454318.

[107] Jean-Jacques E Slotine, Weiping Li, et al. *Applied nonlinear control.* Vol. 199. 1. prentice-Hall Englewood Cliffs, NJ, 1991.

[108] S. Shin, K. Kim, and S. M. Kang. “Memristor-based fine resolution programmable resistance and its applications”. In: *Communications, Circuits and Systems, 2009. ICCCAS 2009. International Conference on.* IEEE. 2009, pp. 948–951. DOI: 10.1109/ICCCAS.2009.5250376.

[109] Eero Lehtonen et al. “Memristive computing”. In: *University of Turku, Finland* (2012).

[110] Dalibor Biolek, Viera Biolkova, and Zdenek Kolka. "Memristor model for massively-parallel computations". In: *Computing, Communication and Security (ICCCS), 2015 International Conference on*. IEEE. 2015, pp. 1–5. DOI: 10.1109/CCCS.2015.7374183.

[111] Alon Ascoli et al. "The art of finding accurate memristor model solutions". In: *Emerging and Selected Topics in Circuits and Systems, IEEE Journal on* 5.2 (2015), pp. 133–142. DOI: 10.1109/JETCAS.2015.2426493.

[112] C. Kołaciński and D. Obrębski. "Design of CMOS analog integrated readout circuit for NMOS THz detectors". In: *Proceedings of the 20th International Conference Mixed Design of Integrated Circuits and Systems - MIXDES 2013*. 2013, pp. 222–228.

[113] S. Gazabare, R. J. Pieper, and W. Wondmagegn. "Observations on frequency sensitivity of memristors". In: *Proceedings of the 2012 44th Southeastern Symposium on System Theory (SSST)*. 2012, pp. 45–50. DOI: 10.1109/SSST.2012.6195143.

[114] Denis Guangyin Chen et al. "Pulse-modulation imaging—Review and performance analysis". In: *IEEE transactions on biomedical circuits and systems* 5.1 (2011), pp. 64–82. DOI: 10.1109/TBCAS.2010.2075929.

[115] Zeljko Ignjatovic, Danijel Maricic, and Mark F Bocko. "Low Power, High Dynamic Range CMOS Image Sensor Employing Pixel-Level Oversampling ΣΔ Analog-to-Digital Conversion". In: *IEEE Sensors Journal* 12.4 (2012), pp. 737–746. DOI: 10.1109/JSEN.2011.2158818.

[116] Xiuling Wang, Winnifred Wong, and Richard Hornsey. "A high dynamic range CMOS image sensor with inpixel light-to-frequency conversion". In: *IEEE Transactions on electron devices* 53.12 (2006), pp. 2988–2992. DOI: 10.1109/TED.2006.885642.

[117] Paul Schlyter. "Radiometry and photometry in astronomy". In: (2009). [Online; accessed 01-October-2010].

[118] Jaehyuk Choi et al. "A 3.4 μW CMOS image sensor with embedded feature-extraction algorithm for motion-triggered object-of-interest imaging". In: *Solid-State Circuits Conference Digest of Technical Papers (ISSCC), 2013 IEEE International*. 2013, pp. 478–479.

[119] P. Lichtsteiner, C. Posch, and T. Delbruck. "A 128 x 128 120dB 30mW Asynchronous Vision Sensor that Responds to Relative Intensity Change". In: *IEEE ISSCC Dig. Tech. Papers*. 2006, pp. 25–26.

[120] N. Massari, M. Gottardi, and S. A. Jawed. "A $100\mu W$ 64×128 Pixels Contrast-Based Asynchronous Binary Vision Sensor for Wireless Sensor Networs," in: *IEEE ISSCC Dig. Tech. Papers.* 2008, pp. 588–589.

[121] Gyouho Kim et al. "A 467nW CMOS visual motion sensor with temporal averaging and pixel aggregation". In: *Solid-State Circuits Conference Digest of Technical Papers (ISSCC), 2013 IEEE International.* IEEE. 2013, pp. 480–481.

[122] Takuro Ohmaru et al. "6.5 25.3μW at 60fps 240$\times$ 160-pixel vision sensor for motion capturing with in-pixel non-volatile analog memory using crystalline oxide semiconductor FET". In: *Solid-State Circuits Conference-(ISSCC), 2015 IEEE International.* IEEE. 2015, pp. 1–3.

[123] A. Berkovich et al. "A 30 μW 30 fps 110x110 Pixels Vision Sensor Embedding Local Binary Patterns". In: *Solid-State Circuits, IEEE Journal of* 50.9 (2015), pp. 2138–2148.

[124] N. Cottini et al. "A $33\mu W 64, \times, 64$ Pixel Vision Sensor Embedding Robust Dynamic Background Subtraction for Event Detection and Scene Interpretation". In: *Solid-State Circuits, IEEE Journal of* 48.3 (2013), pp. 850–863.

[125] Chuan Kai Kenneth Lim, A Gelencser, and T Prodromakis. "Computing image and motion with 3-d memristive grids". In: *Memristor Networks.* Springer, 2014, pp. 553–583.

[126] Jing Zhou et al. "Image segmentation with threshold based on memristors". In: *Electronics Information and Emergency Communication (ICEIEC), 2013 IEEE 4th International Conference on.* IEEE. 2013, pp. 41–44.

[127] Jing Zhou, Jun Jie Wu, and Yu Hua Tang. "Edge Detection of Binary Image Based on Memristors". In: *Advanced Materials Research.* Vol. 791. Trans Tech Publ. 2013, pp. 2066–2070.

[128] Jing Zhou et al. "A memristor-based architecture combining memory and image processing". In: *Science China Information Sciences* 57.5 (2014), pp. 1–12.

[129] Shukai Duan et al. "Hybrid memristor/RTD structure-based cellular neural networks with applications in image processing". In: *Neural Computing and Applications* 25.2 (2014), pp. 291–296.

[130] Elham Zamanidoost et al. "Manhattan rule training for memristive crossbar circuit pattern classifiers". In: *Intelligent Signal Processing (WISP), 2015 IEEE 9th International Symposium on.* IEEE. 2015, pp. 1–6.

[131] Kamran Eshraghian, Kyoungrok Cho, and Peter Graham Foster. *Image matching, data compression and tracking architectures.* US Patent 8,982,260. 2015.

[132] Z. Smilansky. *Miniature Autonomous Agents for Scene Interpretation.* 7,489,802 B2, US Patent Application. 2009.

[133] N. Cottini et al. "A CMOS Ultra-Low Power Vision Sensor with Image Compression and Embedded Event-Driven Energy-Management". In: *IEEE Trans. Circuits and Systems - Emerging Technologies* 1.2 (2011), pp. 1–10.

[134] Michele Benetti, Massimo Gottardi, and Zeev Smilansky. "A 80μW 30fps 104$\times$ 104 all-nMOS pixels CMOS imager with 7-bit PWM ADC for robust detection of relative intensity change". In: *ESSCIRC (ESSCIRC), 2013 Proceedings of the.* IEEE. 2013, pp. 303–306. DOI: 10.1109/ESSCIRC.2013.6649133.

[135] Maureen Caudill. "Neural Networks Primer, Part I". In: *AI Expert* 2.12 (Dec. 1987), pp. 46–52. ISSN: 0888-3785. URL: http://dl.acm.org/citation.cfm?id=38292.38295.

[136] J. J. Hopfield. "Artificial neural networks". In: *IEEE Circuits and Devices Magazine* 4.5 (1988), pp. 3–10. ISSN: 8755-3996. DOI: 10.1109/101.8118.

[137] Donald F. Specht. "Probabilistic neural networks". In: *Neural Networks* 3.1 (1990), pp. 109 –118. ISSN: 0893-6080. DOI: http://dx.doi.org/10.1016/0893-6080(90)90049-Q. URL: http://www.sciencedirect.com/science/article/pii/089360809090049Q.

[138] Kurt Hornik. "Approximation capabilities of multilayer feedforward networks". In: *Neural Networks* 4.2 (1991), pp. 251 –257. ISSN: 0893-6080. DOI: http://dx.doi.org/10.1016/0893-6080(91)90009-T. URL: http://www.sciencedirect.com/science/article/pii/089360809190009T.

[139] Stephen Grossberg. "Nonlinear neural networks: Principles, mechanisms, and architectures". In: *Neural Networks* 1.1 (1988), pp. 17 –61. ISSN: 0893-6080. DOI: http://dx.doi.org/10.1016/0893-6080(88)90021-4. URL: http://www.sciencedirect.com/science/article/pii/0893608088900214.

[140] Erkki Oja. "Principal components, minor components, and linear neural networks". In: *Neural Networks* 5.6 (1992), pp. 927 –935. ISSN: 0893-6080. DOI: http://dx.doi.org/10.1016/S0893-6080(05)80089-9. URL: http://www.sciencedirect.com/science/article/pii/S0893608005800899.

[141] Wolfgang Maass. "Networks of spiking neurons: The third generation of neural network models". In: *Neural Networks* 10.9 (1997), pp. 1659 –1671. ISSN: 0893-6080. DOI: http://dx.doi.org/10.1016/S0893-6080(97)00011-7. URL: http://www.sciencedirect.com/science/article/pii/S0893608097000117.

[142] Ken ichi Funahashi and Yuichi Nakamura. "Approximation of dynamical systems by continuous time recurrent neural networks". In: *Neural Networks* 6.6 (1993), pp. 801 –806. ISSN: 0893-6080. DOI: http://dx.doi.org/10.1016/S0893-6080(05)80125-X. URL: http://www.sciencedirect.com/science/article/pii/S089360800580125X.

[143] Věra Kůrková. "Kolmogorov's theorem and multilayer neural networks". In: *Neural Networks* 5.3 (1992), pp. 501 –506. ISSN: 0893-6080. DOI: http://dx.doi.org/10.1016/0893-6080(92)90012-8. URL: http://www.sciencedirect.com/science/article/pii/0893608092900128.

[144] Jürgen Schmidhuber. "Deep learning in neural networks: An overview". In: *Neural Networks* 61 (2015), pp. 85 –117. ISSN: 0893-6080. DOI: https://doi.org/10.1016/j.neunet.2014.09.003. URL: http://www.sciencedirect.com/science/article/pii/S0893608014002135.

[145] Howard B. Demuth et al. *Neural Network Design*. 2nd. USA: Martin Hagan, 2014. ISBN: 0971732116, 9780971732117.

[146] J. de Villiers and E. Barnard. "Backpropagation neural nets with one and two hidden layers". In: *IEEE Transactions on Neural Networks* 4.1 (1993), pp. 136–141. ISSN: 1045-9227. DOI: 10.1109/72.182704.

[147] Vugar E. Ismailov. "On the approximation by neural networks with bounded number of neurons in hidden layers". In: *Journal of Mathematical Analysis and Applications* 417.2 (2014), pp. 963 –969. ISSN: 0022-247X. DOI: http://dx.doi.org/10.1016/j.jmaa.2014.03.092. URL: http://www.sciencedirect.com/science/article/pii/S0022247X14003412.

[148] Tsong-Lin Lee. "Back-propagation neural network for long-term tidal predictions". In: *Ocean Engineering* 31.2 (2004), pp. 225 –238. ISSN: 0029-8018. DOI: http://dx.doi.org/10.1016/S0029-8018(03)00115-X. URL: http://www.sciencedirect.com/science/article/pii/S002980180300115X.

[149] M. Kiani Deh Kiani et al. "Application of artificial neural networks for the prediction of performance and exhaust emissions in SI engine using ethanol- gasoline blends". In: *Energy* 35.1 (2010), pp. 65 –69. ISSN: 0360-5442. DOI: http://dx.doi.org/10.1016/j.energy.2009.08.034. URL: http://www.sciencedirect.com/science/article/pii/S0360544209003685.

[150] K.S Reddy and Manish Ranjan. "Solar resource estimation using artificial neural networks and comparison with other correlation models". In: *Energy Conversion and Management* 44.15 (2003), pp. 2519 –2530. ISSN: 0196-8904. DOI:

http://dx.doi.org/10.1016/S0196-8904(03)00009-8. URL: http://www.sciencedirect.com/science/article/pii/S0196890403000098.

[151] Adnan Sözen, Erol Arcaklioğlu, and Mehmet Özalp. "Estimation of solar potential in Turkey by artificial neural networks using meteorological and geographical data". In: *Energy Conversion and Management* 45.18 (2004), pp. 3033 –3052. ISSN: 0196-8904. DOI: http://dx.doi.org/10.1016/j.enconman.2003.12.020. URL: http://www.sciencedirect.com/science/article/pii/S0196890404000172.

[152] Guang-Bin Huang. "Learning capability and storage capacity of two-hidden-layer feedforward networks". In: *IEEE Transactions on Neural Networks* 14.2 (2003), pp. 274–281. ISSN: 1045-9227. DOI: 10.1109/TNN.2003.809401.

[153] Ronald Tetzlaff. *Memristors and memristive systems.* Springer, 2013.

[154] Andy Thomas. "Memristor-based neural networks". In: *Journal of Physics D: Applied Physics* 46.9 (2013), p. 093001. URL: http://stacks.iop.org/0022-3727/46/i=9/a=093001.

[155] Kyungah Seo et al. "Analog memory and spike-timing-dependent plasticity characteristics of a nanoscale titanium oxide bilayer resistive switching device". In: *Nanotechnology* 22.25 (2011), p. 254023. URL: http://stacks.iop.org/0957-4484/22/i=25/a=254023.

[156] Mirko Prezioso et al. "Training and operation of an integrated neuromorphic network based on metal-oxide memristors". In: *Nature* 521.7550 (2015), pp. 61–64. DOI: 10.1038/nature14441.

[157] Maricor Soriano et al. "Skin detection in video under changing illumination conditions". In: *Pattern Recognition, 2000. Proceedings. 15th International Conference on.* Vol. 1. IEEE. 2000, pp. 839–842. DOI: 10.1109/ICPR.2000.905542.

[158] Maricor Soriano et al. "Adaptive skin color modeling using the skin locus for selecting training pixels". In: *Pattern Recognition* 36.3 (2003), pp. 681–690. DOI: 10.1016/S0031-3203(02)00089-4.

[159] Jong Yih Kuo et al. "Hand Gesture Recognition Using Standard Deviation of Color Block and Thinning". In: *Computational Intelligence, Modelling and Simulation (CIMSim), 2013 Fifth International Conference on.* IEEE. 2013, pp. 247–252. DOI: 10.1109/CIMSim.2013.47.

[160] SN Karishma and V Lathasree. "Fusion of skin color detection and background subtraction for hand gesture segmentation". In: *International Journal of Engineering Research and Technology* 3.2 (2014), pp. 1835–1839. ISSN: 2278-3091.

[161] Saikat Basak and Arundhuti Chowdhury. “A vision interface framework for intuitive gesture recognition using color based blob detection”. In: *International Journal of Computer Applications* 90.15 (2014). DOI: 10.5120/15799-4611.

[162] Michael Donoser and Horst Bischof. “Real time appearance based hand tracking”. In: *Pattern Recognition, 2008. ICPR 2008. 19th International Conference on*. IEEE. 2008, pp. 1–4. DOI: 10.1109/ICPR.2008.4761485.

[163] Heung-Il Suk, Bong-Kee Sin, and Seong-Whan Lee. “Hand gesture recognition based on dynamic Bayesian network framework”. In: *Pattern Recognition* 43.9 (2010), pp. 3059 –3072. ISSN: 0031-3203. DOI: http://dx.doi.org/10.1016/j.patcog.2010.03.016. URL: http://www.sciencedirect.com/science/article/pii/S0031320310001366.

[164] S. K. Kang, M. Y. Nam, and P. K. Rhee. “Color Based Hand and Finger Detection Technology for User Interaction”. In: *2008 International Conference on Convergence and Hybrid Information Technology*. 2008, pp. 229–236. DOI: 10.1109/ICHIT.2008.292.

[165] Siddharth S. Rautaray and Anupam Agrawal. “Vision based hand gesture recognition for human computer interaction: a survey”. In: *Artificial Intelligence Review* 43.1 (2015), pp. 1–54. ISSN: 1573-7462. DOI: 10.1007/s10462-012-9356-9. URL: https://doi.org/10.1007/s10462-012-9356-9.

[166] A. Bellarbi et al. “Hand gesture interaction using color-based method for tabletop interfaces”. In: *2011 IEEE 7th International Symposium on Intelligent Signal Processing*. 2011, pp. 1–6. DOI: 10.1109/WISP.2011.6051717.

[167] Michela Lecca, Leonardo Gasparini, and Massimo Gottardi. “Ultra-low power high-dynamic range color pixel embedding RGB to rg chromaticity transformation”. In: *SPIE Photonics Europe*. International Society for Optics and Photonics. 2014, pp. 914107–914107. DOI: doi:10.1117/12.2051528.

[168] A. Gijsenij, T. Gevers, and J. van de Weijer. “Computational Color Constancy: Survey and Experiments”. In: *IEEE Transactions on Image Processing* 20.9 (2011), pp. 2475–2489. ISSN: 1057-7149. DOI: 10.1109/TIP.2011.2118224.

[169] Michela Lecca and Stefano Messelodi. “Illuminant Change Estimation via Minimization of Color Histogram Divergence.” In: *CCIW*. Springer. 2009, pp. 41–50.

[170] Michela Lecca and Stefano Messelodi. “Linking the von Kries model to Wien’s law for the estimation of an illuminant invariant image”. In: *Pattern Recognition Letters* 32.15 (2011), pp. 2086 –2096. ISSN: 0167-8655. DOI: http://dx.doi.org/10.1016/j.patrec.2011.08.005. URL: http://www.sciencedirect.com/science/article/pii/S016786551100256X.

[171] J. Berens and G. D. Finlayson. "Log-opponent chromaticity coding of colour space". In: *Proceedings 15th International Conference on Pattern Recognition. ICPR-2000*. Vol. 1. 2000, 206–211 vol.1. DOI: 10.1109/ICPR.2000.905304.

[172] Edoardo Provenzi et al. "Mathematical definition and analysis of the Retinex algorithm". In: *J. Opt. Soc. Am. A* 22.12 (2005), pp. 2613–2621. DOI: 10.1364/JOSAA.22.002613. URL: http://josaa.osa.org/abstract.cfm?URI=josaa-22-12-2613.

[173] Alessandro Rizzi, Carlo Gatta, and Daniele Marini. "From retinex to automatic color equalization: issues in developing a new algorithm for unsupervised color equalization". In: *Journal of Electronic Imaging* 13.1 (2004), pp. 75–84.

[174] Daniel Weinland, Remi Ronfard, and Edmond Boyer. "A survey of vision-based methods for action representation, segmentation and recognition". In: *Computer Vision and Image Understanding* 115.2 (2011), pp. 224 –241. ISSN: 1077-3142. DOI: http://dx.doi.org/10.1016/j.cviu.2010.10.002. URL: http://www.sciencedirect.com/science/article/pii/S1077314210002171.

[175] Dharani Mazumdar, Anjan Kumar Talukdar, and Kandarpa Kumar Sarma. "A colored finger tip-based tracking method for continuous hand gesture recognition". In: *Int. J. Electron. Signals Syst* 3 (2013), pp. 71–75. ISSN: 2231- 5969.

[176] Anil K Jain and Stan Z Li. *Handbook of face recognition*. Springer, 2011. ISBN: 9780857299314. DOI: 10.1007/978-0-85729-932-1.

[177] Rehanullah Khan et al. "Color based skin classification". In: *Pattern Recognition Letters* 33.2 (2012), pp. 157 –163. ISSN: 0167-8655. DOI: http://dx.doi.org/10.1016/j.patrec.2011.09.032. URL: http://www.sciencedirect.com/science/article/pii/S0167865511003151.

[178] Maricor Soriano et al. "Using the skin locus to cope with changing illumination conditions in color-based face tracking". In: *IEEE Nordic Signal Processing Symposium*. Vol. 38. Kolmarden, Sweden: IEEE. 2000, pp. 383–386. DOI: 10.1.1.468.8008.

[179] Maricor Soriano et al. "Adaptive skin color modeling using the skin locus for selecting training pixels". In: *Pattern Recognition* 36.3 (2003), pp. 681 –690. ISSN: 0031-3203. DOI: http://dx.doi.org/10.1016/S0031-3203(02)00089-4. URL: http://www.sciencedirect.com/science/article/pii/S0031320302000894.

[180] Michela Lecca et al. "Always-on low-power optical system for skin-based touchless machine control". In: *JOSA A* 33.6 (2016), pp. 1015–1024. DOI: 10.1364/JOSAA.33.001015.

[181] Moritz Störring, Hans Jørgen Andersen, and Erik Granum. “Physics-based modelling of human skin colour under mixed illuminants”. In: *Robotics and Autonomous Systems* 35.3 (2001), pp. 131–142. DOI: 10.1016/S0921-8890(01)00122-1.

[182] Massimo Gottardi, Nicola Massari, and Syed Arsalan Jawed. “A 100μW 128times 64 Pixels Contrast-Based Asynchronous Binary Vision Sensor for Sensor Networks Applications”. In: *IEEE Journal of Solid-State Circuits* 44.5 (2009), pp. 1582–1592. DOI: 10.1109/JSSC.2009.2017000.

[183] Leonardo Gasparini et al. “An ultralow-power wireless camera node: Development and performance analysis”. In: *IEEE Transactions on Instrumentation and Measurement* 60.12 (2011), pp. 3824–3832. DOI: 10.1109/TIM.2011.2147630.

[184] Deng-Yuan Huang, Wu-Chih Hu, and Sung-Hsiang Chang. “Gabor filter-based hand-pose angle estimation for hand gesture recognition under varying illumination”. In: *Expert Systems with Applications* 38.5 (2011), pp. 6031 –6042. ISSN: 0957-4174. DOI: http://dx.doi.org/10.1016/j.eswa.2010.11.016. URL: http://www.sciencedirect.com/science/article/pii/S0957417410012534.

[185] Michela Lecca. “On the von Kries Model: Estimation, Dependence on Light and Device, and Applications”. In: *Advances in Low-Level Color Image Processing.* Ed. by M. Emre Celebi and Bogdan Smolka. Dordrecht: Springer Netherlands, 2014, pp. 95–135. ISBN: 978-94-007-7584-8. DOI: 10.1007/978-94-007-7584-8_4. URL: https://doi.org/10.1007/978-94-007-7584-8_4.

[186] Seokjun Park et al. “7.2 243.3 pJ/pixel bio-inspired time-stamp-based 2D optic flow sensor for artificial compound eyes”. In: *Solid-State Circuits Conference Digest of Technical Papers (ISSCC), 2014 IEEE International.* IEEE. 2014, pp. 126–127. DOI: 10.1109/ISSCC.2014.6757366.

[187] Gyouho Kim et al. “A 467nW CMOS visual motion sensor with temporal averaging and pixel aggregation”. In: *Solid-State Circuits Conference Digest of Technical Papers (ISSCC), 2013 IEEE International.* IEEE. 2013, pp. 480–481. DOI: 10.1109/ISSCC.2013.6487823.

[188] M. Gottardi, N. Massari, and S. A. Jawed. “A 100 μ W 128 *times* 64 Pixels Contrast-Based Asynchronous Binary Vision Sensor for Sensor Networks Applications”. In: *IEEE Journal of Solid-State Circuits* 44.5 (2009), pp. 1582–1592. ISSN: 0018-9200. DOI: 10.1109/JSSC.2009.2017000.

[189] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. “Imagenet classification with deep convolutional neural networks”. In: *Advances in neural information processing systems.* 2012, pp. 1097–1105.

[190] Wojciech Zaremba, Ilya Sutskever, and Oriol Vinyals. "Recurrent Neural Network Regularization". In: *CoRR* abs/1409.2329 (2014). URL: http://arxiv.org/abs/1409.2329.

[191] Andrea Vedaldi and Karel Lenc. "MatConvNet: Convolutional Neural Networks for MATLAB". In: *Proceedings of the 23rd ACM International Conference on Multimedia*. MM '15. Brisbane, Australia: ACM, 2015, pp. 689–692. ISBN: 978-1-4503-3459-4. DOI: 10.1145/2733373.2807412. URL: http://doi.acm.org/10.1145/2733373.2807412.

[192] Gabriel Lera and Miguel Pinzolas. "Neighborhood based Levenberg-Marquardt algorithm for neural network training". In: *IEEE Transactions on Neural Networks* 13.5 (2002), pp. 1200–1203. DOI: 10.1109/TNN.2002.1031951.

[193] G. E. Hinton and R. R. Salakhutdinov. "Reducing the Dimensionality of Data with Neural Networks". In: *Science* 313.5786 (2006), pp. 504–507. ISSN: 0036-8075. DOI: 10.1126/science.1127647. eprint: http://science.sciencemag.org/content/313/5786/504.full.pdf. URL: http://science.sciencemag.org/content/313/5786/504.

[194] Michael Meissner, Michael Schmuker, and Gisbert Schneider. "Optimized Particle Swarm Optimization (OPSO) and its application to artificial neural network training". In: *BMC Bioinformatics* 7.1 (2006), p. 125. ISSN: 1471-2105. DOI: 10.1186/1471-2105-7-125. URL: http://dx.doi.org/10.1186/1471-2105-7-125.

[195] Nitish Srivastava et al. "Dropout: A Simple Way to Prevent Neural Networks from Overfitting". In: *Journal of Machine Learning Research* 15 (2014), pp. 1929–1958. URL: http://jmlr.org/papers/v15/srivastava14a.html.

[196] Steve Lawrence, C Lee Giles, and Ah Chung Tsoi. "Lessons in neural network training: Overfitting may be harder than expected". In: *AAAI/IAAI*. Citeseer. 1997, pp. 540–545.

[197] John K Fiorenza et al. "Comparator-based switched-capacitor circuits for scaled CMOS technologies". In: *IEEE Journal of Solid-State Circuits* 41.12 (2006), pp. 2658–2668. DOI: 10.1109/JSSC.2006.884330.

[198] Reut Wizenberg et al. "Applications of solid-state memristors in tunable filters". In: *Circuits and Systems (ISCAS), 2014 IEEE International Symposium on*. IEEE. 2014, pp. 2269–2272. DOI: 10.1109/ISCAS.2014.6865623.

[199] R Berdan et al. "Memristive devices as parameter setting elements in programmable gain amplifiers". In: *Applied Physics Letters* 101.24 (2012), p. 243502. DOI: 10.1063/1.4770315.

[200] Tsung-Wen Lee and Janice H Nickel. “Memristor resistance modulation for analog applications”. In: *IEEE electron device letters* 33.10 (2012), pp. 1456–1458. DOI: 10.1109/LED.2012.2207429.