Charge Trap Transistors as an Analog Memory for Neuromorphic Systems

Ataollah Saeed Monir

Department of Electrical & Computer Engineering

McGill University

Montréal, Québec, Canada

Aug 15, 2023

A thesis presented for the degree of Masters of Electrical Engineering

Abstract

The von Neumann architecture features a clear separation between memory and processing units, but despite enabling versatile computers, it faces inherent issues. A key challenge is the compute-memory bottleneck due to limited CPU-memory bandwidth, slowing overall processing, particularly for memory-intensive tasks. Memory-processor separation requires complex interconnects, and power consumption is critical as complex tasks demand energy. Data transfer between the memory and the processor contributes to power inefficiencies.

Researchers, therefore, have been striving to develop a system that uses a different architecture. Neuromorphic systems are one of the suggestions that mimic the sophisticated and highly efficient processing capabilities of the human brain. The idea is to combine the computing and memory elements and perform the operations at the same place instead of them being separate, similar to what happens in the human brain with neurons, to overcome the limitations imposed by the von Neumann bottleneck.

Currently, many in-memory computing systems rely on emerging nonvolatile memory devices, including resistive memory, phase-change memory, etc. However, incorporating these new materials and technologies into the standard fabrication process increases cost and extends the development cycle. One of the memories used for this purpose is the charge trap transistor (CTT), which is a complementary metal-oxide-semiconductor (CMOS)-only device. Any logic transistor from advanced node technologies with high-k dielectric can be employed as a CTT. By trapping and de-trapping electrons, respectively, in and from the oxygen-vacancy modulated traps within the Hafnium oxide material, the threshold voltage of the CTT can be altered.

Understanding physics-driven principles of CTTs significantly influences the attainable accuracy, processing speed, and power efficiency of the network. However, the existing body of research in this domain remains limited. Furthermore, the absence of simulation models that can seamlessly integrate into conventional integrated circuit computer-aided design tools hinders the capture of programming and erasure effects as they correlate with applied stress voltages.

In this thesis, neuromorphic systems and related literature are discussed. Then, there is a more focused discussion on analog memories and their use in neuromorphic systems. Afterwards, CTTs and their properties as analog memory are examined. The next chapter focuses on the trapping behaviour of CTTs and studies how they behave during programming. Various practical tests are done to understand CTT behaviour in different programming scenarios. From this empirical work, a new model is introduced that explains the intricate charge-trapping behaviour in CTTs and a Verilog-A code using this model was also tested to capture these effects. Moving forward, a published architecture centered around CTTs is highlighted. This framework serves both machine learning applications and the ambitious goal of emulating brain functions. This architecture was tested for a digit recognition application and had very promising results. In the end, the conclusions section sum up the discussions from the thesis and discuss possible future works.

Abrégé

L'architecture de von Neumann présente une séparation claire entre la mémoire et les unités de traitement, mais bien qu'elle permette des ordinateurs polyvalents, elle est confrontée à des problèmes inhérents. L'un des principaux défis réside dans le goulot d'étranglement de la mémoire de calcul dû à la bande passante limitée de la mémoire du processeur, qui ralentit le traitement global, en particulier pour les tâches gourmandes en mémoire. La séparation mémoire-processeur nécessite des interconnexions complexes, et la consommation d'énergie est critique car les tâches complexes nécessitent de l'énergie. Le transfert de données entre la mémoire et le processeur contribue aux inefficacités énergétiques.

Les chercheurs se sont donc efforcés de développer un système utilisant une architecture différente. Les systèmes neuromorphiques sont l'une des suggestions qui imitent les capacités de traitement sophistiquées et hautement efficaces du cerveau humain. L'idée est de combiner les éléments de calcul et de mémoire et d'effectuer les opérations au même endroit au lieu de les séparer, comme ce qui se passe dans le cerveau humain avec les neurones, pour surmonter les limitations imposées par le goulot d'étranglement de von Neumann.

Actuellement, de nombreux systèmes informatiques en mémoire s'appuient sur des dispositifs de mémoire non volatile émergents, notamment la mémoire résistive, la mémoire à changement de phase, etc. Cependant, l'intégration de ces nouveaux matériaux et technologies dans le processus de fabrication standard augmente les coûts et prolonge le cycle de développement. L'une des mémoires utilisées à cette fin est le transistor piège de charge (CTT), qui est un dispositif complémentaire à métal-oxyde-semi-conducteur (CMOS) uniquement. N'importe quel transistor logique issu de technologies de nœuds avancées avec un diélectrique à k élevé peut être utilisé comme CTT. En piégeant et en dépiégeant les électrons, respectivement, dans et depuis les pièges modulés par manque d'oxygène à l'intérieur du matériau d'oxyde de hafnium, la tension de seuil du CTT peut être modifiée.

Comprendre les principes physiques des CTT influence de manière significative la précision, la vitesse de traitement et l'efficacité énergétique du réseau. Cependant, le corpus de recherche existant dans ce domaine reste limité. De plus, l'absence de modèles de simulation pouvant s'intégrer de manière transparente aux outils de conception assistée par ordinateur de circuits intégrés conventionnels entrave la capture des effets de programmation et d'effacement car ils sont en corrélation avec les tensions de contrainte appliquées.

Dans cette thèse, les systèmes neuromorphiques et la littérature connexe sont discutés. Ensuite, il y a une discussion plus ciblée sur les mémoires analogiques et leur utilisation dans les systèmes neuromorphiques. Ensuite, les CTT et leurs propriétés en tant que mémoire analogique sont examinés. Le chapitre suivant se concentre sur le comportement de trapping des CTT et étudie leur comportement lors de la programmation. Divers tests pratiques sont effectués pour comprendre le comportement de CTT dans différents scénarios de programmation. À partir de ce travail empirique, un nouveau modèle est introduit qui explique le comportement complexe de piégeage de charge dans les CTT et un code Verilog-A utilisant ce modèle a également été testé pour capturer ces effets. À l'avenir, une architecture publiée centrée sur les CTT est mise en évidence. Ce cadre sert à la fois aux applications d'apprentissage automatique et à l'objectif ambitieux d'émuler les fonctions cérébrales. Cette architecture a été testée pour une application de reconnaissance de chiffres et a donné des résultats très prometteurs. À la fin, la section des conclusions résume les discussions de la thèse et discute des travaux futurs possibles.

Acknowledgements

As I come to the close of my Master's thesis journey at McGill University, I want to take a moment to express my heartfelt gratitude to the individuals and organizations that have rendered this experience both meaningful and attainable.

My family – my parents and my little sister – hold my first and deepest gratitude. Their unwavering support, patience, and understanding have carried me through the highs and lows of this academic pursuit, making it possible for me to get here.

A special note of appreciation goes to Professor Vaisband, my supervisor, for his belief in me and for providing me with this opportunity. Your guidance has not only shaped the trajectory of my research but has also influenced my approach to learning and personal growth, I appreciate your trust in me since day one and helping me to grow and become the person I am today, both in my personal and academic life.

To my friends in the lab, you have become my second family. Your encouragement, late-night discussions, and shared challenges have indelibly etched this journey into my memory. Your companionship has served as my anchor, guiding me through the intricate pathways of research and its trials.

The guidance and resources offered by Blumind, under the guidance of Navid Rezazadeh, have significantly enriched my practical comprehension. Your support has not only refined my professional skills but has also equipped me with the tools to connect academic knowledge with real-world applications.

Reflecting on this journey, certain moments stand out – like the New Year's Eve spent in the lab working on the simulations with Rezvan, and 7 hour-long meetings until 4 am with my supervisor, where we confronted challenges together.

I appreciate everyone working and living in the lab with me, it was a hell of an experience, to say the least!

Contents

List of Figures

List of Tables

List of Acronyms

AdExp	adaptive-exponential.
CIM	compute-in-memory.
CMOS	complementary metal–oxide–semiconductor.
CTT	Charge Trap Transistor.
DSC	differential current sensor.
H&H	Hodgkin-Huxley.
HCI	hot-carrier injection.
HRS	high resistance state.
I&F	integrate-and-fire.
IC	integrated circuit.
LIF	leaky integrate-and-fire.
LRS	low resistance state.
MIM	metal-insulator-meta.
MNIST	modified national institute of standards and technology.

NBTI	negative bias temperature instability.
NN	neural network.
PBTI	positive bias temperature instability.
PCM	Phase change memory.
RRAM	Resistive Random-Access Memory.

Chapter 1

Introduction

Over the past four decades, transistor size has greatly reduced, resulting in notable performance and functionality improvements. Nevertheless, pushing transistor size to its manufacturing limit presents challenges such as increased parasitic elements, limited interconnect scaling, and rising non-recurring engineering costs that need to be addressed. Therefore, as Moore's law approaches its limits [3], [4], the exploration of brain-inspired computing architectures, such as neuromorphic engineering, becomes crucial to enhance overall system performance. Neuromorphic systems aim to replicate the flexibility and computational efficiency observed in biological neural processing systems by implementing spiking neural network architectures with tightly co-located processing and memory [5], [6]. The human brain consists of 100 billion neurons, where the memory and control units are not separated from each other. Thus, all of the operations are performed in one place, inside the neuron. Co-locating the memory and control units, removing the bandwidth and power limitations, and making it possible to be highly scalable.

1.1 Background of Neuromorphic Circuits

Neuromorphic systems offer several key benefits over traditional von Neumann systems, leading to their increasing prominence in the field of computing. Traditional von Neumann systems [7], [8] often consume excessive energy due to the separation of memory and processing units and the need for frequent data movement. In contrast, neuromorphic systems closely mimic the parallelism and event-driven computation observed in the human brain, allowing for efficient processing and reduced energy consumption. By leveraging specialized hardware designs and distributed computing, neuromorphic systems excel in tasks that require low power consumption, making them particularly well-suited for resource-constrained environments and applications [9].

Furthermore, neuromorphic systems offer enhanced real-time processing capabilities. In von Neumann architectures, the central processing unit acts as a bottleneck, limiting the system's ability to handle time-critical tasks efficiently [10]. Neuromorphic systems, on the other hand, utilize distributed computing and parallel processing, allowing for simultaneous computations across a network of artificial neurons. This parallelism supports real-time processing, enabling the system to respond rapidly to time-sensitive inputs. As a result, neuromorphic systems find applications in domains such as robotics, sensory processing, and real-time decision-making, where prompt and dynamic responses are crucial [11], [12].

In addition to the advancements in neuromorphic engineering, a particularly intriguing concept that aligns with brain-inspired computing is compute-in-memory (CIM) [13]. CIM is a cutting-edge computing paradigm that aims to emulate efficient information processing of the brain by tightly integrating computation and memory within the same unit [14], [15]. This brain resemblance system offers several advantages, including low power consumption and the ability to support numerous applications involving a massive number of neurons. One of the primary benefits of CIM is its low power profile. This efficiency is crucial for emerging technologies such as Internet of Things (IoT) devices, where power constraints are a major consideration [16], [17].

Furthermore, CIM systems exhibit the potential to support a multitude of applications that involve a vast number of neurons. The human brain consists of billions of interconnected neurons, enabling complex cognitive tasks. CIM systems, with their ability to perform computations within memory units, can handle large-scale neural simulations and modeling more effectively [18]. This capability opens up opportunities in various domains, including neuroscience research, artificial intelligence (AI), and cognitive computing. CIM systems can accelerate simulations of neural networks, facilitating advancements in understanding brain functions and enabling the development of advanced AI algorithms [15–17].

Various approaches to neuromorphic designs exist, with some being lightweight while lacking biological intuition [19–21], and others, highly detailed but not scalable [22].

Another possibility is a chipset-based system-level electrical model of the brain connectivity network. Each chiplet, a small integrated circuit (IC), can house hundreds to thousands of neurons, each with 100 to 1,000 synapses. Communication on each chiplet occurs at short-range using standard on-chip routing techniques. Mid-range communication involves interactions among chiplets within the same cluster and can be achieved through protocols like BoW [23] or SuperCHIPS [24]. Currently, researchers have only scratched the surface concerning connectivity-related brain diseases (*e.g.*, epilepsy [25, 26]). As a result, an ultra-large-scale neural network that emulates the brain holds tremendous potential in this domain. Connectivity networks derived from magnetic resonance imaging can be integrated into a neuromorphic emulator, where the network's connection strength is represented by synaptic weights within the emulator's architecture. This neuromorphic emulator allows real-time monitoring of activity within the ”uploaded” brain map. By adjusting the synaptic weights, valuable insights can be gained into the activity patterns associated with connectivity-related brain diseases.

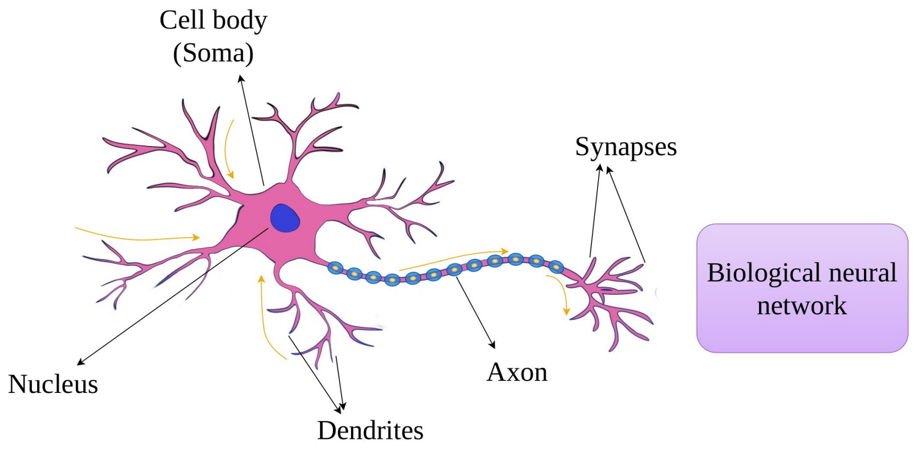
Before discussing the analog memories used in electrical engineering, the neuroscience side of the brain is briefly discussed here. The core building blocks of neuromorphic systems are neurons (soma), synapses, dendrites, and axons [6], as shown in Fig 1.1(a). Neurons are comprised of three primary components: a central cell body known as the soma, along with two distinct types of branched, treelike structures that extend from the soma called dendrites and axons. Depending on the inputs, neurons decide when to fire and send a spike to their connected neurons. On the other hand, the primary body of an artificial (electrical) neuron is termed a node or unit. These nodes are interconnected physically through wires that simulate the connections found between biological neurons [27], [28], [6].

In the brain, neurons receive information from other neurons in the form of electrical impulses, which enter the dendrites through connection points known as synapses.This information is received by the dendrites and processed in the soma. The resulting output signal is transmitted down the axon to reach the synapses of other neurons in the form of a series of impulses [6, 27–29].

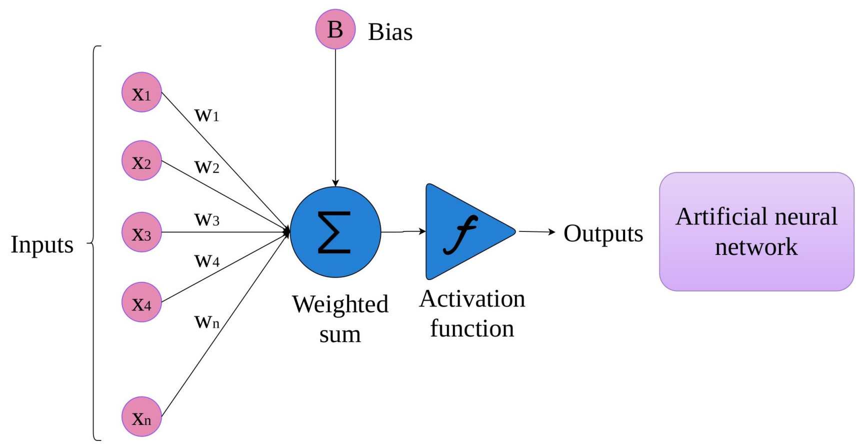
Biological Network	**Artificial Network**
Soma	Node(Unit)
Dendrites	Input
Synapse	Weight
Axon	Output

Table 1.1: Comparison of Biological and Artificial Networks.

Before discussing existing neuron models and each part of the neuron, first, the operation of a neuron is explained here. Neurons are connected to each other on a large scale, and these connections are considered synapses [30]. When neuron A is connected to neuron B's dendrites with a synapse, the weight and strength of the synapse is increased with the activity of neuron A, and if this activity decreases, the connection is weakened, and neuron A's activity now has less effect on the activity of neuron B. Considering the fact that most neurons are active during the day, their connection's strength changes every day



(a)



(b)

Figure 1.1: (a) Biological neural network with neurons (soma), synapses, dendrites, and axons. (b) Artificial natural network with inputs, weights, bias and activation function

(this continues even during REM sleep), and this is how a human learns during the day, stores daily events in its memory and etc. Most of the illnesses regarding the human brain are caused by damage in these connections, for example, seizures or epilepsy [25, 26, 31, 32]. The artificial neuron is considered to be one or a mass of neurons that operate exactly the same way, each input connection is weighted usually with an analog memory, and then the weighted inputs are summed inside the neuron and, depending on the activation function and the neuron model, the neuron decides to fire or not as shown in Fig. 1.1(b) [2].

Neuron models, such as the integrate-and-fire (I&F) [33], [34] and leaky integrate-and-fire (LIF) models [35], [36] , are instrumental in understanding neural coding approaches. The choice of neuron model directly impacts the network's information representation and processing capabilities. Several neuron models, including the Hodgkin-Huxley (H&H) [33], [37], Izhikevich [38], and adaptive-exponential (AdExp) [39], [40] models, provide varying trade-offs between biophysical accuracy, versatility, and implementation efficiency. Each model grasps different behaviours of the neurons, and depending on the application, different models can be used [41]. For example, using a complete and complex model like (H&H) would be highly bio resemblance while adding a lot of complexity to the system, and a simple LIF model could be enough for the architecture if only simple behaviours of the neuron are expected.

Biological synapses are highly compact structures that facilitate memory and plasticity functions [42], allowing neurons to connect with a large number of synapses per neuron, ranging from 100 to 10,000 [43]. Achieving an optimal balance between versatility and efficiency is crucial for synapses, as they often dominate the area of neuromorphic processors, sometimes by more than one order of magnitude [22].

To enable large-scale integrations, designers face the challenge of either moving synaptic resources off-chip, which increases system power and latency [44], or sacrificing the key feature of synaptic plasticity [45], [46]. However, preserving embedded online learning is essential for several reasons. Firstly, it enables low-power autonomous agents to gather knowledge and adapt to new features in real-time, particularly in uncontrolled environments where new training data is presented on the fly [47], [48]. Secondly, from a computational efficiency perspective, neuromorphic designs without synaptic plasticity rely on off-chip optimizers, limiting their deployment in power-constrained and resource-constrained applications, not only during inference but also in the training phase. Lastly, exploring biophysically realistic silicon synapses that incorporate spike-based plasticity mechanisms may offer insights into their operations in the brain and support cognition [49].

1.2 Analog Memories Investigated for Neuromorphic Systems

Neuromorphic systems aim to mimic the structure and functionality of the human brain, including its ability to process and store information. While digital memories, such as those based on transistors and non-volatile memory technologies, are commonly used in conventional computing systems, neuromorphic systems often incorporate analog memories due to their potential advantages in terms of energy efficiency and parallel processing [18].

Here, a number of popular analog memories which are being used are described.

1.2.1 Memristors

The memristor, a portmanteau of ”memory resistor”, is considered the fourth basic circuit element alongside the resistor, capacitor and inductor [50]. It is a nonlinear passive two-terminal electronic component which is defined by the relation between the magnetic flux ϕ and the electric charge q. It can neither store nor generate any power because it is a passive element. The exact definition of a memristor is given by its frontiersperson as [51]: ”*any 2-terminals device, exhibiting a pinched hysteresis loop, which always passes through the origin in the voltage-current plane when driven by any periodic input current source, or voltage source, with zero DC component. If the input is a current source, it is called a current-controlled memristor. If it is a voltage source, it is called a voltage-controlled memristor*”. Indeed, the resistance of a memristor is contingent on the charge that passed through the circuit. When the current flows in one direction, the resistance escalates, but it decreases when the current flows in the opposite direction. However, the resistance cannot go below zero. Remarkably, when the current ceases, the memristor retains the resistance value it had prior, essentially ”remembering” the last current that flowed through it [52]. The memristor holds the potential to serve as a crucial component, acting as a stateful logic element and an artificial neuron/synapse in both von Neumann and neuromorphic computing paradigms [53].

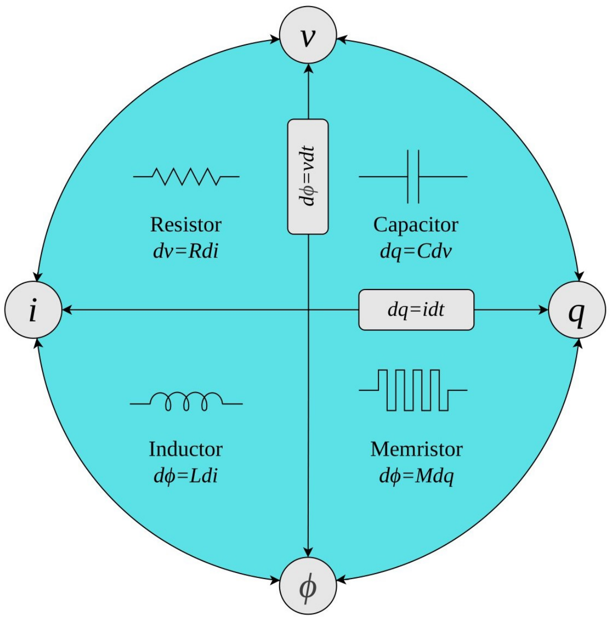


Figure 1.2: The four fundamental two-terminal circuit elements: resistor, capacitor, inductor and memristor.

The most basic mathematical definition of a current-controlled memristor for circuit analysis is the differential form

$$v = R(w)i \ , \tag{1.1}$$

$$\frac{dw}{dt} = i \ , \tag{1.2}$$

where w is the state variable of the device and R is a generalized resistance that depends upon the device's internal state. Here, the state variable corresponds to the charge. In 1976, Chua and Kang expanded this concept to encompass a wider range of nonlinear dynamical systems, which they referred to as memristive systems, described by the equations [54].

$$v = R(w, i)i \ , \tag{1.3}$$

$$\frac{dw}{dt} = f(w, i) \ , \tag{1.4}$$

where w can be a set of state variables and R and f can, in general, be explicit functions of time. Chua and Kang showed that the current-voltage (i-v) characteristics of some devices and systems, including thermistors, Josephson junctions, neon bulbs, and even the H&H model of the neuron, could be accurately modelled using memristive equations. However, despite these findings, there was no direct link between the mathematical equations and the physical properties of any practical system. As a result, nearly forty years later, the concept of memristive systems had not gained widespread adoption [54].

1.2.2 Phase Change Memory

Phase change memory (PCM) is a non-volatile memory technology that uses phase change materials, such as chalcogenide alloys or transition metal dichalcogenides, to store data [55,56]. PCM makes effective use of the difference in resistivity between the crystalline phase (characterized by low resistivity) and the amorphous phase (characterized by high resistivity) of the phase change material. In PCM, the "set" state refers to the low-resistance condition, while the "reset" state pertains to the high-resistance condition. This inherent property of PCM is crucial in enabling its application in various fields, particularly in electronic memory devices and neuromorphic computing systems. PCM cells consist of a small volume of material between two electrodes. By applying electrical currents or voltages, the phase change material located between the top electrode and heater can be switched between amorphous and crystalline states, representing binary data [55]. This current crowding at the contact point between the ”heater” and the phase change material leads to the formation of a programmed region, visually represented by the mushroom boundary as shown in Fig. 1.3. In recent times, PCM devices have emerged as a promising contender for neuromorphic computing, primarily because of their multi-level storage capabilities [57], [58]. PCM offers fast read and write speeds, high endurance, and excellent data retention [59].

PCM is a candidate for future memory architectures due to its fast switching speeds in the nanosecond range, surpassing many other non-volatile memory technologies. It also has high endurance with millions of write cycles possible. PCM is scalable to an extent, allowing for miniaturization and high-density memory arrays. This makes it suitable for advanced computing applications such as in-memory computing and neuromorphic computing.

Although PCM has significant potential, it still faces some challenges. One of the main limitations is the scalability of each cell size, as a certain volume of phase change material is required for reliable operation. Additionally, the power consumption during write operations is a big concern, as high currents are necessary. The cost of production and integration with existing memory technologies are also obstacles to its widespread adoption. PCM requires additional masks and materials to be added, which adds complexity to the process and increases its price. Nevertheless, PCM is still an active area of research and development, with the potential to revolutionize computing by providing fast, high-density, and non-volatile storage capabilities.

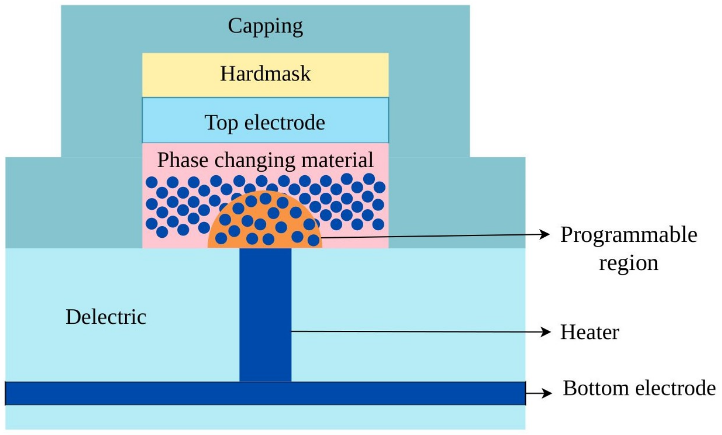


Figure 1.3: Phase change memory (PCM) structure

1.2.3 Floating-Gate Transistors

Floating-gate transistors are a type of transistor commonly used in non-volatile memory devices like Flash memory. [60]. Electrical charge is stored in an insulated and electrically isolated floating gate. This process traps charge in the floating gate, called hot electron injection. The transistor's state is determined by the presence or absence of charge, representing binary data. [61]. Floating gate transistors are based on two metal gates - a floating gate and a control gate. Charge trapping within this device occurs within the metal-oxide-metal capacitor that is formed by the two gates.

The structure and operation modes of a floating gate transistor are depicted in Fig. 1.4. Three kinds of operations can be performed: erase, write(program), and read. The erase operation is carried out by emptying the floating gate of its negative charge, which involves the removal of electrons contained within. This process is facilitated by field electron emission (Fowler-Nordheim tunnelling). To achieve this, a high voltage, typically around 20 volts [62], is applied to the substrate, while the control gate receives a zero voltage. On the other hand, writing(programming) entails negatively charging the floating gate, and also utilizing field emission. During this process, the substrate receives a zero voltage, and a high voltage is applied to the control gate. Lastly, the read operation is performed by applying a reference voltage to the control gate of the transistor. If the floating gate is charged negatively, the transistor is turned off, resulting in no current flow in the channel between the drain and the source. This scenario typically corresponds to a logical ”0” (zero) stored in the cell. Conversely, if the gate is not charged, the transistor is conducting, equivalent to a logical ”1” (one). [61]

Floating-gate transistors have several advantages in non-volatile memory applications. They can store data without power and have high endurance, allowing for millions of read and write cycles. Their compact design enables high-density memory arrays. Floating-gate transistors have certain limitations that affect their performance. The programming and erasing operations of these transistors tend to be slow due to the requirement of high

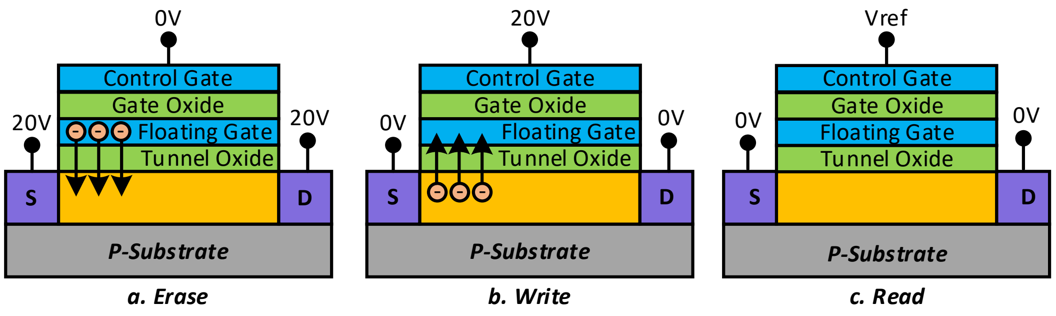


Figure 1.4: Operations using a floating gate memory.

voltages and the interference between adjacent cells. As the technology reduces in size, the reliability of these transistors can be impacted by issues such as electron leakage and charge loss.

Despite these limitations, floating-gate transistors have certain limitations, they have played a critical role in the advancement of non-volatile memory devices. These transistors can store, charge, and retain data, making them an essential component in Flash memory. As a result, they have enabled the creation of high-capacity and durable storage solutions.

1.2.4 Resistive Random-Access Memory

Resistive Random-Access Memory (RRAM) is a non-volatile memory technology that stores data by changing the resistance of a material [63, 64]. RRAM memory cells consist of a resistive switching memory cell, which features a metal-insulator-metal (MIM) structure. This structure consists of an insulating layer(I) sandwiched between two metal(M) electrodes, as shown in Fig. 1.5. When an external voltage pulse is applied to the RRAM cell, it triggers a transition of the device from a high resistance state (HRS), or OFF state representing logic value '0', to a low resistance state (LRS), or ON state usually representing logic value '1'. This transition can also occur in the reverse direction, allowing the device to switch between the two states [65], [66].

RRAM offers several advantages as a memory technology. It provides fast read and writes speeds, comparable to or better than traditional memory technologies like Flash memory. RRAM cells also have high endurance, with millions of read and write cycles possible. Additionally, RRAM has good scalability potential, enabling high-density memory arrays.

However, there are challenges associated with RRAM. Precise control of the resistive switching process is required for reliable performance, as variations in cell characteristics can affect data storage. Power consumption during write operations is also a consideration, as high voltages are often needed for resistive switching.

Despite these challenges, RRAM holds promise as a potential future memory technology. Its fast operation, high endurance, and scalability make it an attractive alternative to existing non-volatile memory technologies. Ongoing research and development efforts are focused on improving the performance and reliability of RRAM for a wide range of computing and storage applications.

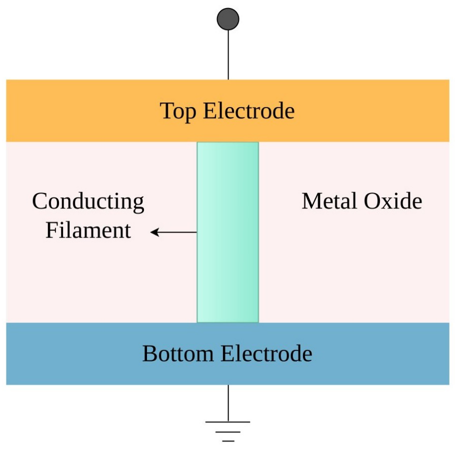


Figure 1.5: Resistive Random-Access Memory (RRAM)

1.3 Charge Trap Transistors

Two decades ago, the ongoing decrease in transistor dimensions led to the use of extremely thin gate oxides, resulting in a significant issue of high gate leakage through silicon. To address this problem, alternative solutions such as high dielectric constant materials like hafnium oxide (HfO_2) were proposed. However, the presence of inherent oxygen vacancies in HfO_2 posed a challenge by acting as charge-trapping centers, causing a gradual increase in the threshold voltage (V_t) of transistors. [67] This electron trapping effect leads to a substantial shift in the drain current (I_D) vs. gate-source voltage (V_{gs}) curves, as illustrated in Fig. 1.6. The change in V_t is influenced by various factors, and even slight variations in these parameters can result in significant alterations [68], which will be discussed in Chapter 2.

By applying appropriate, logic-compatible voltages higher than the nominal (0.8V),

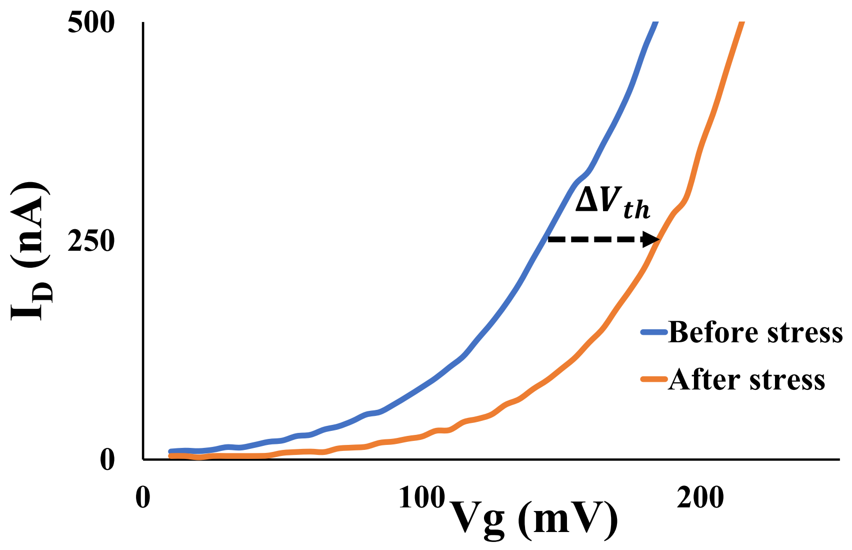
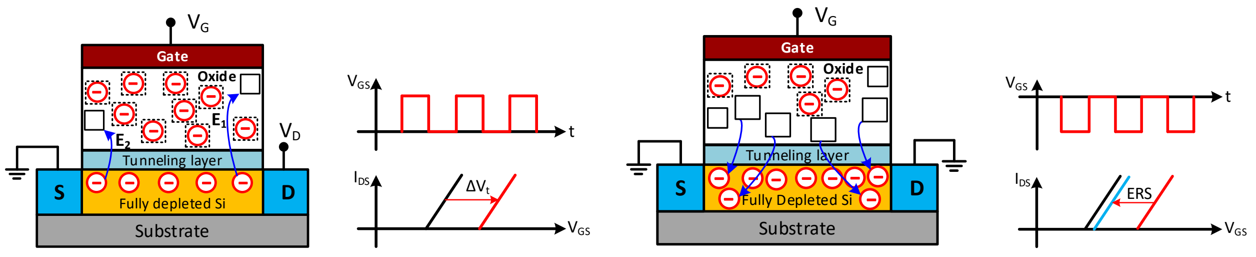


Figure 1.6: Effect of charge trapping on the drain current of the transistor. The currents were measured at $V_{ds} = 100\ mV$.

enhanced charge trapping in the high-k gate dielectric of HKMG logic transistors occurs, leading to significant and stable threshold voltage shifts (ΔV_t) [69]. These shifts can then be used as a non-volatile data storage mechanism. To understand how CTTs work, first, it should be noted that working with CTTs consists of three operation modes, programming, erasing and inference. During programming, high voltage pulses arrive at the gate while drain voltage is present, causing the charges to get trapped in the vacancies and altering the inference current of the CTT as demonstrated in Fig. 1.7 (a). During Erase, A negative voltage is applied to the gate, while $V_{ds} = 0\ V$, causing the electrons to get de-trapped and go back to the channel as shown in Fig.1.7 (c). During inference, the device is biased at a low voltage (in the sub-threshold region), and based on the amount of trapping and the initial current of the device, the drain current is set to the desired value.



(a) **(b)** **(c)** **(d)**

Figure 1.7: Program and erase phenomena within the CTT device. (a) Schematic illustration of programming (PRG), where the charge is driven by positive V_{GS} pulses and constant V_{DS} from the channel into the vacant traps in the high-k oxide, and (b) the corresponding voltage and current diagrams, illustrating applied gate voltage and the resulting shift in threshold voltage. (c) Schematic illustration of erase (ERS), where the charge is driven by negative V_{GS} pulses (V_{DS} = 0 V) from the traps in the high-k oxide back into the channel, and (d) the corresponding voltage and current diagrams, illustrating applied gate voltage and resulting backward shift in threshold voltage [1].

Although CTTs may look similar to floating gate transistors, however, charge trap transistors include a single metal gate, similar to the standard CMOS process, and the charge is trapped within the metal-oxide-silicon structure of the device. Furthermore, the floating gate device is not logic voltage compatible (10 to 20 V) and requires level shifters that occupy a significant area.

When compared to other analog synapse alternatives, the CTT offers a variety of advantages:

CMOS-Only Technology: Unlike resistive/phase-change RAM or flash technology, the CTT requires no additional materials or processes. Consequently, the development cost is significantly reduced. This technology allows for the utilization of advanced-node logic transistors as CTTs directly from the fabrication and easy porting of existing IP designs from an older node to a newer one.

Low Voltage Requirement: Operating exclusively on logic-compatible voltage below 3 V, the CTT enjoys a notable advantage over flash-based analog synapses. [70]

It is clear that CTTs have the potential for analog synapses in neuromorphic computing systems. [1,71–76], catering to swift and energy-efficient cognitive tasks. The objective of this work is to utilize CTTs as an analog memory in a proposed neuromorphic system. Firstly, the device's characteristics are analyzed using different measurements on multiple devices in Chapter 2. Subsequently, an architecture is proposed and simulated on the MNIST dataset using CTTs and an LIF neuron in Chapter 3.

Chapter 2

CTTs as an Analog Memory

In this chapter, we have explored the behavior of CTTs as analog memory. All of the measurements and plots are done by the author using the devices and facilities provided by Blumind [77] and are in preparation to be submitted at the time of thesis submission.

2.1 CTT Behavior

CTTs hold great potential as analog memory, however, understanding their behavior has a crucial role in using the CTTs. It is important to understand what affects the current of the device and what factors should be considered while designing a circuit using CTTs. Device variation, charge trapping behavior, effects of the parasitics, the layout of the circuit, de-trapping of the electrons and many other things should be considered for using CTTs. In this section, a deeper look has been taken into the physics of the device and the use of it as an analog memory.

2.1.1 Physics Behind Trapping and de-trapping

HfO_2 is commonly used as the gate dielectric in high-k/metal gate CMOS technologies, and it is known to have charge traps related to oxygen vacancies [67]. Oxygen vacancies are defects caused by oxygen diffusion from HfO_2, leaving behind positively charged vacancy defects. It has been observed that bias stress-induced charge trapping and defect generation in HfO_2 are significantly accelerated by temperature [68]. Charge trapping in HfO_2 is typically considered a source of device and circuit variability and an undesired feature, but in this concept, it is actually useful.

The positioning of the traps in the device depends on the device's relative bias points. In this section, we explore charge trapping in two specific locations: bulk-oxide traps and interfacial traps. One extensively studied case involving such logic devices is the negative or positive-bias temperature instability (NBTI or PBTI) [78], [79]. NBTI or PBTI can cause a shift in the device's threshold voltage (Vth) due to a negative or positive gate bias. For nMOSFETs, PBTI is more pronounced because positive VGS is typically employed during most operations. PBTI leads to Vth shift by trapping charges in the bulk-oxide traps.

Although PBTI can induce Vth shift, it is not as significant as the proposed PRG method, which primarily traps charge in the bulk oxide but benefits from enhanced resistive self-heating caused by the large drain current [79]. The PRG method proves to be almost three times more efficient in charge trapping with good retention. Research has shown that the self-heating enhanced CTT PRG method can retain 70% of the trapped charge at 105°C for ten years [80].

Another approach to utilize charge trapping for device Vth shift is through hot-carrier injection (HCI), which necessitates a very high VDS to generate interfacial traps between the oxide and the drain-side channel [81]. HCI has demonstrated the ability to significantly shift Vth with excellent retention (less than 10% charge loss in 10 years at 125°C) [82]. However, HCI can only be de-trapped through high-temperature and long-time annealing [83]. In contrast, bulk-oxide trapping can be reversed without heating (since resistive self-heating during negative VGS is not feasible due to the device being in the off-state) by applying high VGS, with the drain floated or shorted to the source. [84]

2.2 Experimental Results on Trapping Behavior

As discussed in the previous sections, many factors can affect charge-trapping behavior. Previous research has shown that even a small change in these factors can result in a shift in the threshold voltage. To address these effects, numerous measurements have been conducted in this study. All measurements were done using SLVT transistors with dimensions of $W = 430\ nm$ and $= 20\ nm$, in 22nm FDSOI technology. Firstly, the effect of different factors has been reported in this study. Secondly, a simplified model has been introduced that considers the programming conditions and predicts the amount of threshold voltage shift. The test

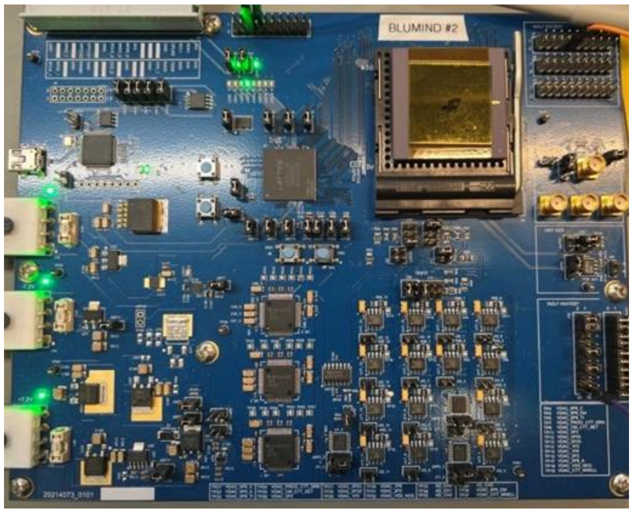


Figure 2.1: Experimental test setup

board has been depicted in Fig 2.1

2.2.1 Effect of Gate Voltage on Programming CTTs

Gate pulses play a crucial role in CTT devices, serving two primary functions. Firstly, these pulses activate the channel, facilitating the flow of electrons through it. Concurrently, they induce a high vertical field that aids in the injection and subsequent entrapment of electrons within the gate's high-k dielectric. The effectiveness of the trapping process is directly influenced by the magnitude of the applied pulses, as it governs the strength of the electric field across the channel.

Research has demonstrated that increasing the magnitude of these pulses enhances the trapping efficiency [85], and recent investigations have revealed that a power law relationship has good accuracy in describing the voltage dependence across a broad voltage range [86]. As

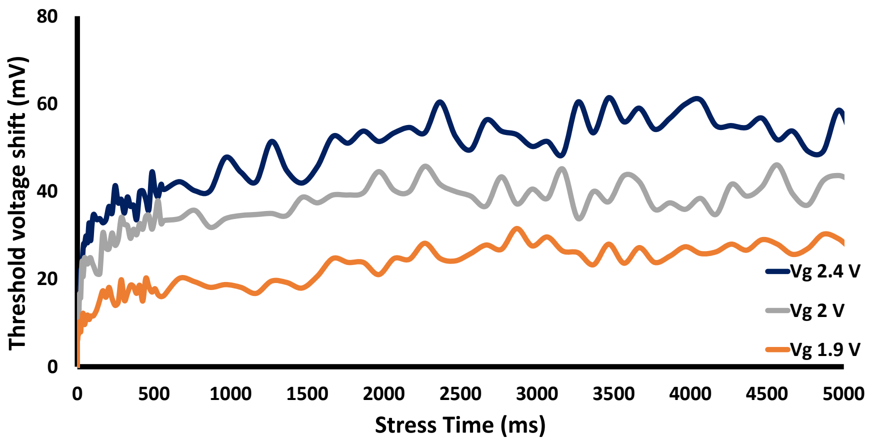


Figure 2.2: Effect of V_g on ΔV_t. ΔV_t was measured after each stress pulse at $V_{gs} = 125\ mV$ and $V_{ds} = 100\ mV$. During programming V_d was set to $900\ mV$.

shown in Fig. 2.2, changing the applied drain voltage affects the threshold voltage drastically.

2.2.2 Effect of Drain Voltage on Programming CTTs

The temperature within the charge trap transistor's channel is influenced by the drain voltage. The channel's temperature and current both rise in response to an increase in the drain voltage. Because of the increased temperature, there are more available traps for electrons to fall into. Therefore, raising the drain voltage makes more traps available (generates traps), which facilitates the storage of charge in the transistor. This result is illustrated in Fig. 2.3

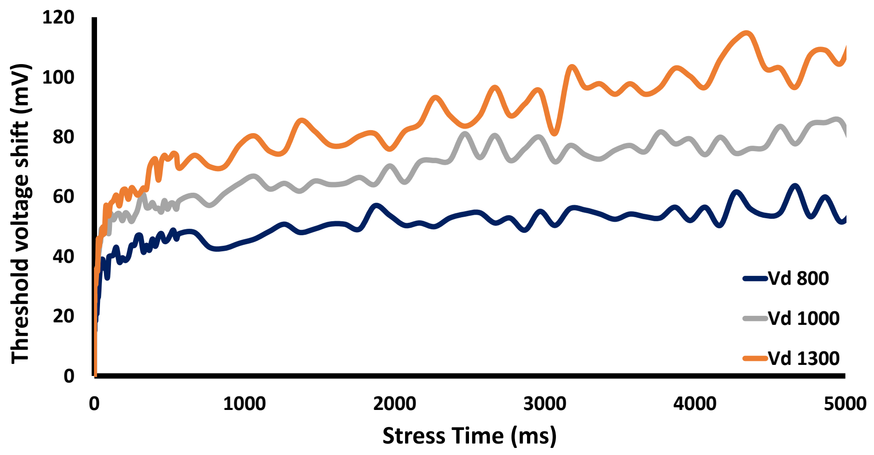


Figure 2.3: Effect of V_d on ΔV_t.ΔV_t was measured after each stress pulse at $V_{gs} = 125\ mV$ and $V_{ds} = 100\ mV$. During programming V_g was set to 2100 mV.

2.2.3 Effect of stress time on Programming CTTs

During programming, the device is under stress for a certain amount of time, increasing this stress time increases the probability of electrons getting trapped in the traps. The effect of stress time is shown in Fig. 2.4.

It is notable that the electrons start to get trapped when the device is under stress, and after a while, the traps get saturated, and there are no more traps available for the electrons, this is evidence that under those programming conditions, the traps are saturated and for the device to have more threshold voltage shift, the conditions programming conditions should change (i.e. increase in the V_d or V_g to make more traps available for the electrons).

To see if it is important to apply one continuous pulse or multiple small pulses, the following experiment was done, in this test, similar devices were programmed with the same voltages, however, one batch of devices was under stress with 1 ms pulses and the other

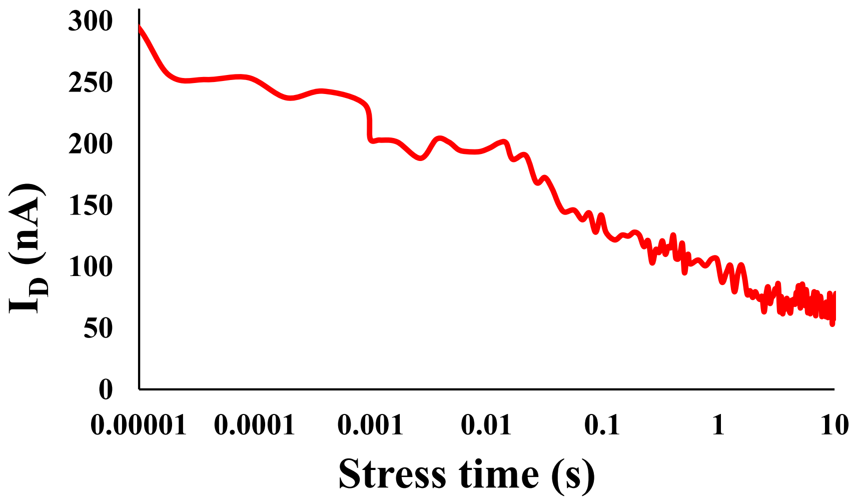


Figure 2.4: Effect of stress time on the current of the device during inference. The current was measured after each programming pulse at $V_{gs} = 125\ mV$ and $V_{ds} = 100\ mV$.

batch with 10 ms pulses. As shown in Fig. 2.5, the final results follow closely among two batches and merge into each other. The small differences in the result could be attributed to de-trapping during measurements since it takes 10 ms to measure the current of the device, and with increasing the number of pulses, the de-trapping increases.

2.2.4 Weight Levels Quantization

As it was illustrated in the previous section, programming CTTs has a stochastic nature, meaning that although the behavior of the device can be modeled and predicted to an extent, many other factors still affect the amount of programming done in the device. For instance, the parasitic of the device can affect the load that the device sees during programming and change the amount of electrons getting trapped in the device. The layout of the device

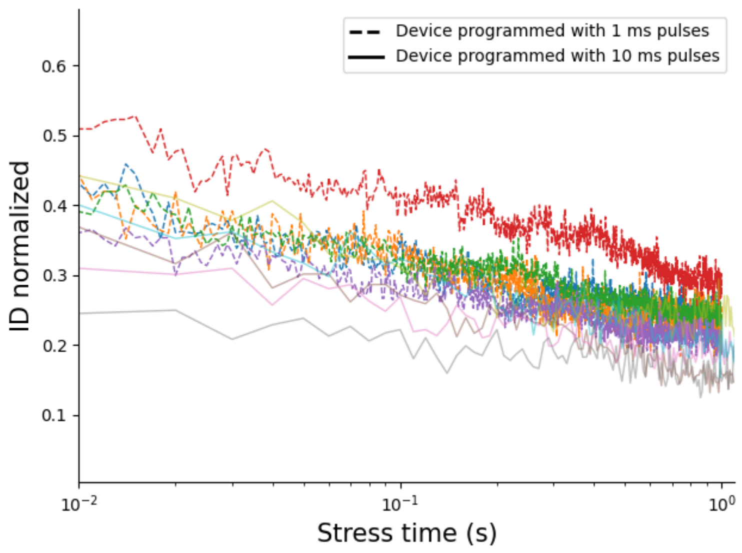


Figure 2.5: Comparison of devices programmed with series of 1 ms and 10 ms pulses. The normalized current of the device is being reported here to remove the effect of device's the initial current.

affects the R_{th} of the device and changes the effect of drain voltage in electron energy levels, and the nature of trapping is also dependent on the probability of electrons getting trapped, and it is not guaranteed to happen [84].

As illustrated in Fig.2.6, even when all of the devices are programmed in identical conditions, the amount of shift is still different. Therefore having a closed-loop verification is necessary for using CTTs as analog memories, which increases the programming time significantly. This brings the question of how accurate the programming requires to be. If high precision is required for the current of the device, this will result in very long programming (i.e., write) time which becomes impossible in large systems with large numbers of devices. Therefore, a robust network is required to accompany the CTTs and

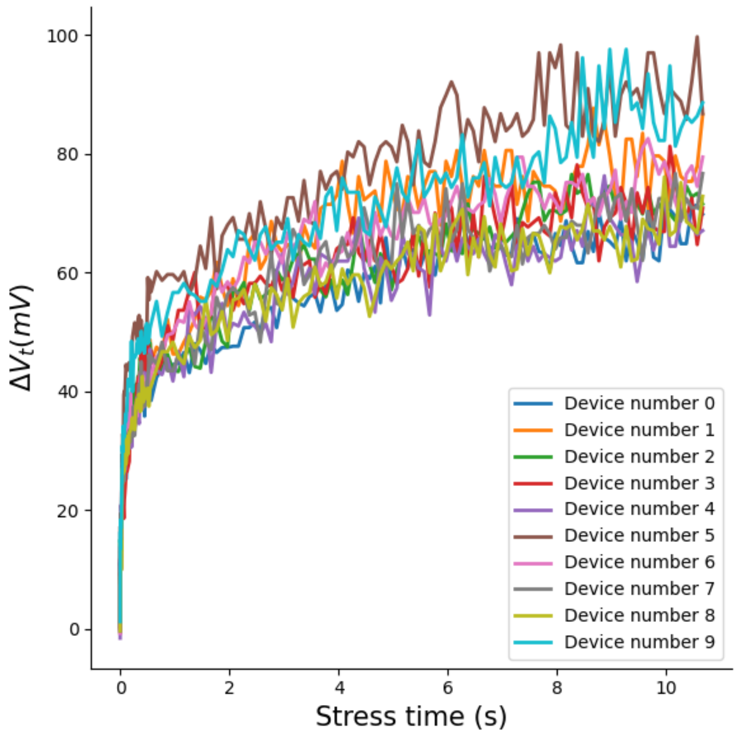


Figure 2.6: Stochastic nature of trapping. Ten different devices with the same dimensions and similar layout have been programmed under the same conditions.

have a middle ground in programming accuracy and write time and make the designer able to set the weights to the required level in a reasonable time.

Our work in [1], considered 12 distinct weight levels, which need to be verified, and the greatest threshold voltage shift to be 110 mV, which is consistent with published and experimental results. Further information will be covered in Chapter 3

2.3 Simplified Model

Many factors affect the charge-trapping behavior of CTTs. General charge trapping in transistors has been extensively investigated [87], and previous works have also explored charge trapping in CTTs [68, 71, 85], however, modeling the trapping behavior for using it as analog memory in the circuit design has not been done before. Multiple expressions have been introduced to determine the amount of ΔV_t in transistors. All of these expressions take trap density and the distribution of the traps into account in some sort. (2.1) is being used in this work which has been used before in various researches [86]

$$\Delta V_t = \Delta V_{t_{max}} \left(1 - e^{-(t/\tau_0)^{\beta}}\right), \tag{2.1}$$

where $\Delta V_{t_{max}}$ is the maximum threshold voltage shift, t is the stress time, τ_0 represents a characteristic time scale for the trapping dynamics. A smaller value of τ_0 indicates faster trapping dynamics, while a larger value implies slower dynamics, and β is a measure of the width or spread of the trap distribution and quantifies the width of this distribution. When β is close to 1, it suggests a narrow distribution, indicating that the capture cross-sections associated with different traps have discrete or distinct values. On the other hand, smaller values of β indicate a broader distribution of time constants. Due to the stochastic nature of trapping, the values for τ_0 and β are technology, process, and layout dependent. Based on the experimental results done in this work and fitting the data into the mentioned equation, the value of β is in the range of 0.2 and 0.3, which agrees with the values reported in [85, 86, 88]. The value of τ_0 in this work is calculated to be in the range of 0.2 to 1.1 s which is also in the range of previously reported works [85].

To calculate the shift on ΔV_t as demonstrated in (2.1), it is required to know the $\Delta V_{t_{max}}$ for each pair of V_{ds} and V_{gs}. Changing the V_{ds} and V_{gs} affects the electric field and the temperature, causing a change in the number of traps and, consequently, the amount of the shift, as discussed in Section 2.2. A series of stress pulses described in the following was applied to the device to determine the maximum shift amount on different stress voltages.

Forty-nine different sets of stress voltages were used in this measurement, and each session consisted of multiple stress pulses with the same voltage, applied with different pulse widths for a total of 10.6 s to ensure the traps were saturated in each condition(in all of the measurements 90% of the traps were saturated in less than a second but still the pulses were given to ensure saturation). The device's V_t was measured before and after each session, and between each stress pulse, the device's current was logged to keep track of the trend in trapping. Each measurement was done on multiple devices to avoid the noises caused by the device variation and parasitics on the results, and the average numbers are being reported here. Based on these experiments and previous works [89], it was noted that trappings begin to be significant and noticeable around $V_{gs} = 1.8$ V and $V_{ds} = 0.9$ V. During these measurements, the V_{gs} were set to [1.8 V$-$2.2 V], and V_{ds} were set to [0.9 V$-$1.3 V]. To avoid the effect of de-traping [74], the currents were measured for

$\sim$ 10 ms, and the next stress voltages were applied. The current of the transistor was measured at V_{gs} = 125 mV and V_{ds} = 100 mV and they ranged from 100 nA to 350 nA before stress. The dimensions of all tested devices were W = 430 nm and L = 20 nm.

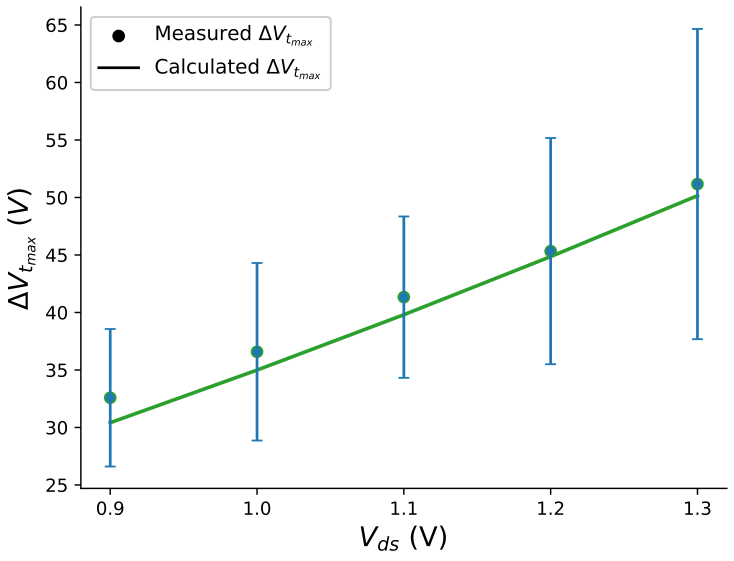


Figure 2.7: Effect of V_d on ΔV_t. $\Delta V_{t_{max}}$ was measured after a total of 10.6 s of stress time with different V_d and $V_g = 2.0\ V$.

Using the mentioned measurements, this research introduced an equation to determine the value of $\Delta V_{t_{max}}$ based on stressing voltages. $\Delta V_{t_{max}}$ is dependent on (i) the electric field across the channel during programming (V_{gs}). The average result for $V_{gs} = 2.0\ V$ with error bars are shown in Fig.2.7 and the overall plot for all of the V_{gs} is shown in Fig.2.8 (ii) the current of the transistor during programming ($I_{D_{sh}}$), which affects the temperature of the channel and (iii) the V_{ds} during programming that also affects the temperature and self-heating effect. The average result for $V_{ds} = 1.2\ V$ with the error bars is shown in Fig.2.9 and the overall plot for all of the V_{ds} is shown in Fig.2.10.

In this work [90], (2.2) is suggested to take into account these effects and calculate the $\Delta V_{t_{max}}$ in the programming voltage range to be used in (2.1).

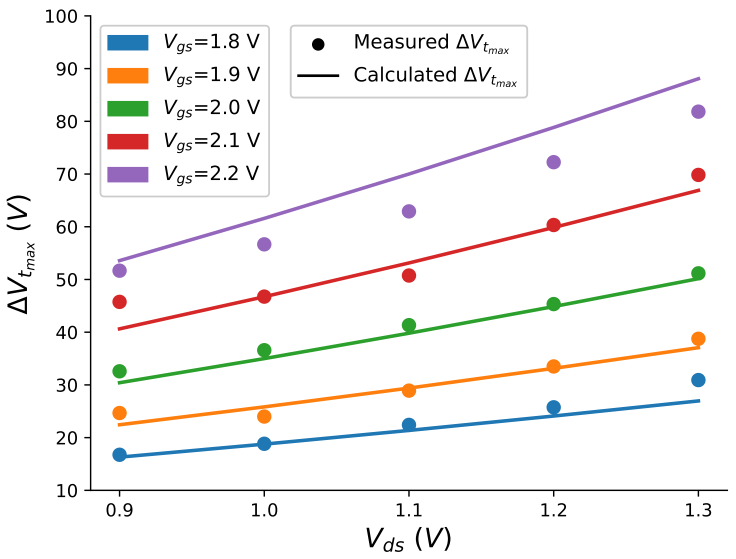


Figure 2.8: Effect of V_d on $\Delta V_{t_{max}}$. $\Delta V_{t_{max}}$ was measured after a total of 10.6 s of stress time. The current of the device before and after programming was measured at $V_{gs} = 125\ mV$ and $V_{ds} = 100\ mV$.

$$\Delta V_{t_{max}} = d \cdot \Delta T \cdot V_{gs}^{m}, \tag{2.2}$$

where d is a fitting parameter to scale the voltage and is found to be around 13.12. ΔT is the temperature rise in the channel during stress time, which is calculated using R_{th} and the power dissipated in the channel during stress time ($V_{ds} \times I_{D_{sh}} \times R_{th}$). The current of the devices was measured during stress, and it ranged from 700 µA to 910 µA. Therefore, the final equation for $\Delta V_{t_{max}}$ would be:

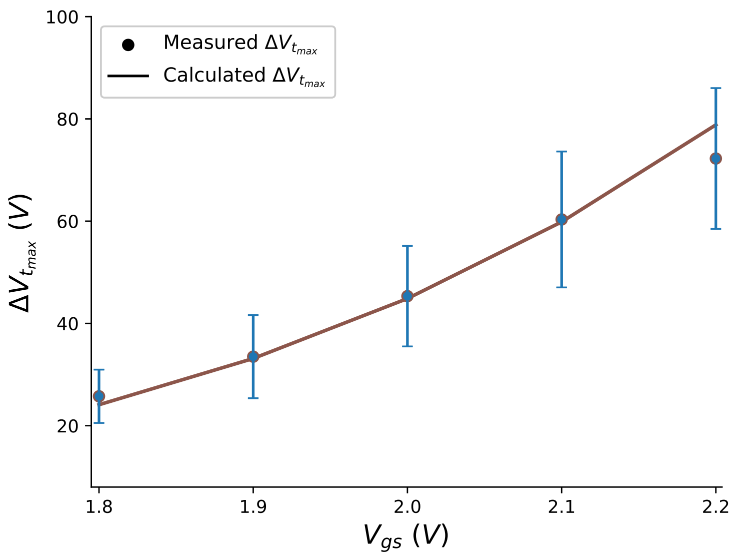


Figure 2.9: Effect of V_g on ΔV_t. $\Delta V_{t_{max}}$ was measured after a total of 10.6 s of stress time with different V_d and $V_d = 1.2\ V$.

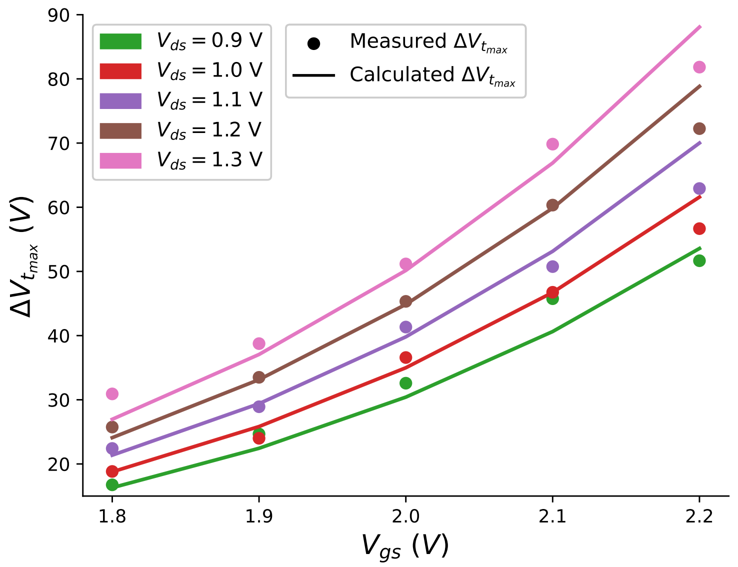


Figure 2.10: Effect of V_g on $\Delta V_{t_{max}}$. $\Delta V_{t_{max}}$ was measured after a total of 10.6 s of stress time. The current of the device before and after programming was measured at $V_{gs} = 125\ mV$ and $V_{ds} = 100\ mV$.

$$\Delta V_{t_{max}} = d \cdot V_{ds} \cdot I_{D_{sh}} \cdot R_{th} \cdot V_{gs}^{m}, \tag{2.3}$$

this expression takes the input voltages into account and based on the experimental results provides an estimation of the amount of trapping. It should be noted that the error rate of the linear model for the effect of V_{ds} on $\Delta V_{t_{max}}$ (shown in Fig.2.8) was less than one percent compared to the averaged measured numbers. During measurements, different devices were used to decrease the effects of parasitics by using the average amounts of trapping. Increasing the stress voltages higher than the reported range was causing the effect of de-trapping to be dominant during measuring, causing the value of $\Delta V_{t_{max}}$ to drop and making the measurements inaccurate. It is possible to apply higher voltages to reduce the stress time required to reach the same ΔV_t; however, due to the nature of trapping, the trapping behavior becomes more dependent on the technology, process and layout of the device, making it hard to predict. Furthermore, while applying the pulses, the voltages should not go higher than a certain value since these devices' nominal operating point is 0.8 V, and it is not recommended for very high voltages, and the device will break down under them [68].

As an illustration, four measurements are compared to the model's predictions and depicted in Fig. 2.11. It is important to acknowledge that while the linear relationship exhibited minimal error with the average results presented here, trapping behavior is stochastic and technology-dependent. The accuracy of trapping may vary based on factors such as layout, technology, etc., for each separate device, so verifying the programming is essential when aiming for high accuracy in systems.

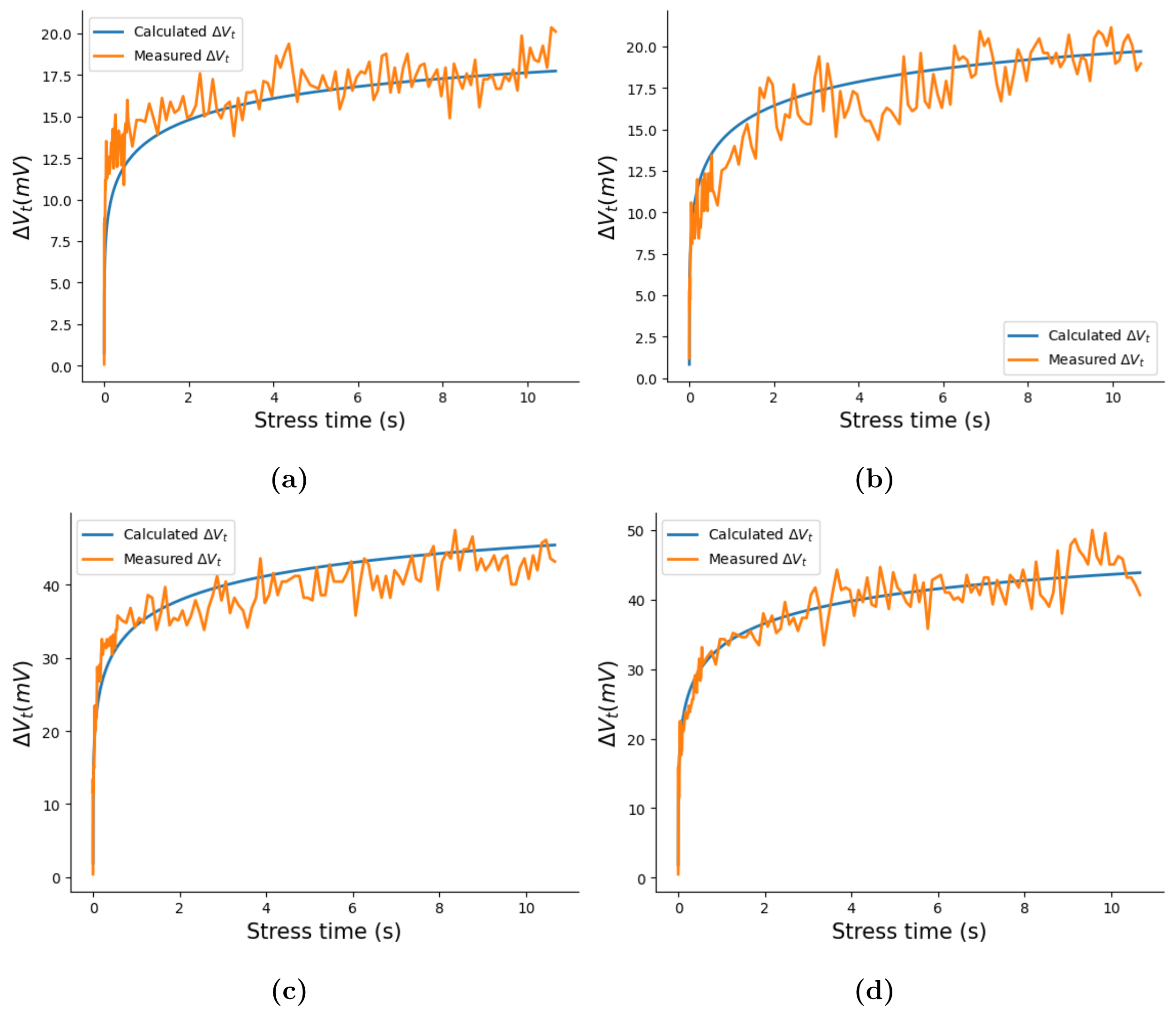


Figure 2.11: Comparison of the simplified model versus some stress pulses (a) A device under $V_{gs} = 1.8$ V and $V_{ds} = 0.8$ V (b) A device under $V_{gs} = 1.8$ V and $V_{ds} = 0.9$ V (c) A device under $V_{gs} = 2.1$ V and $V_{ds} = 1$ V (d) A device under $V_{gs} = 2$ V and $V_{ds} = 1.2$ V

2.4 Trapping Behaviour in CAD Tools

Charge trapping in CTTs is not being captured in the available CAD [91] tools, therefore making it impossible for designers to use CTTs in their designs. As shown in Fig 2.12, applying programming voltages does not change the current of the transistor for inference.

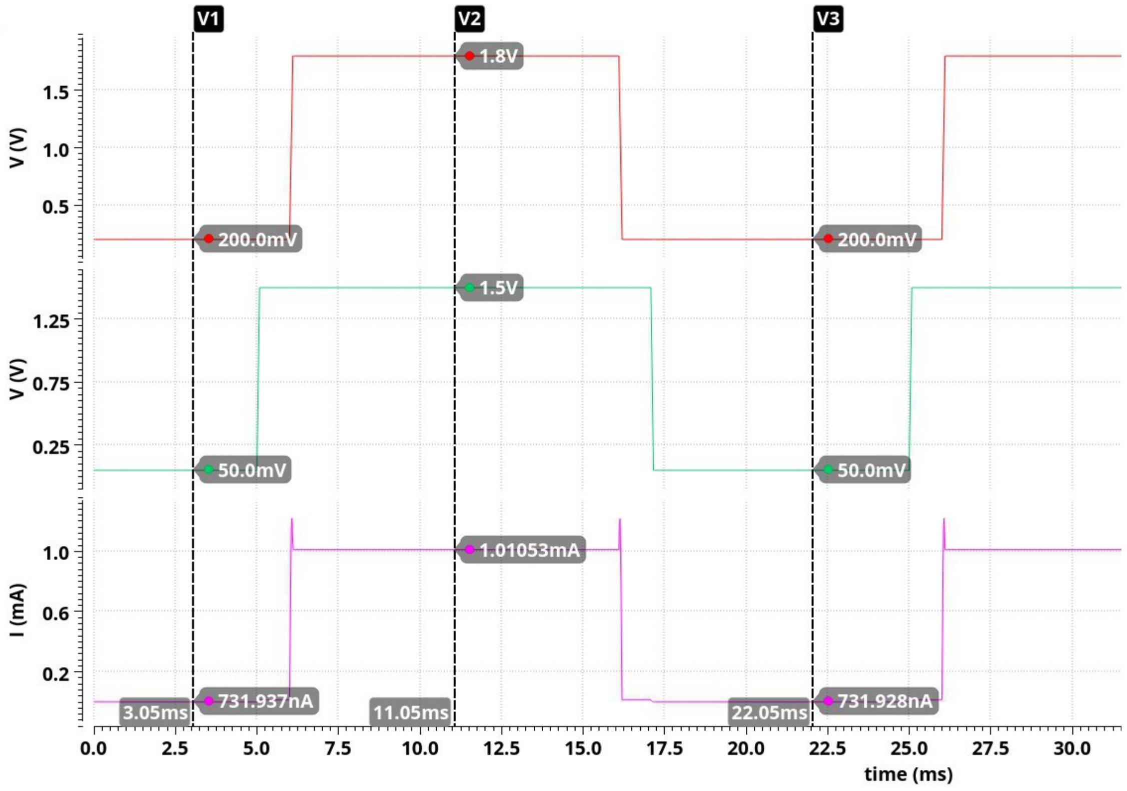


Figure 2.12: Applying programming pulses does not change the threshold voltage of the transistor

A model is proposed here to use 2.3 in Verilog-A and add this feature to the simulation tools. Using this model, the designer is enabled to have a CTT device as an analog memory. In the proposed Verilog-A model, the gate and drain voltages are the inputs of the model, and based on the stress time, the model predicts a shift in V_t and applies it to the gate of the transistor as a negative DC source [76] as shown in Fig 2.13. The model is replacing the DC voltage in the circuit and is calculating the amount of shift based on the applied voltages to the gate and drain of the device as shown in Fig. 2.14.

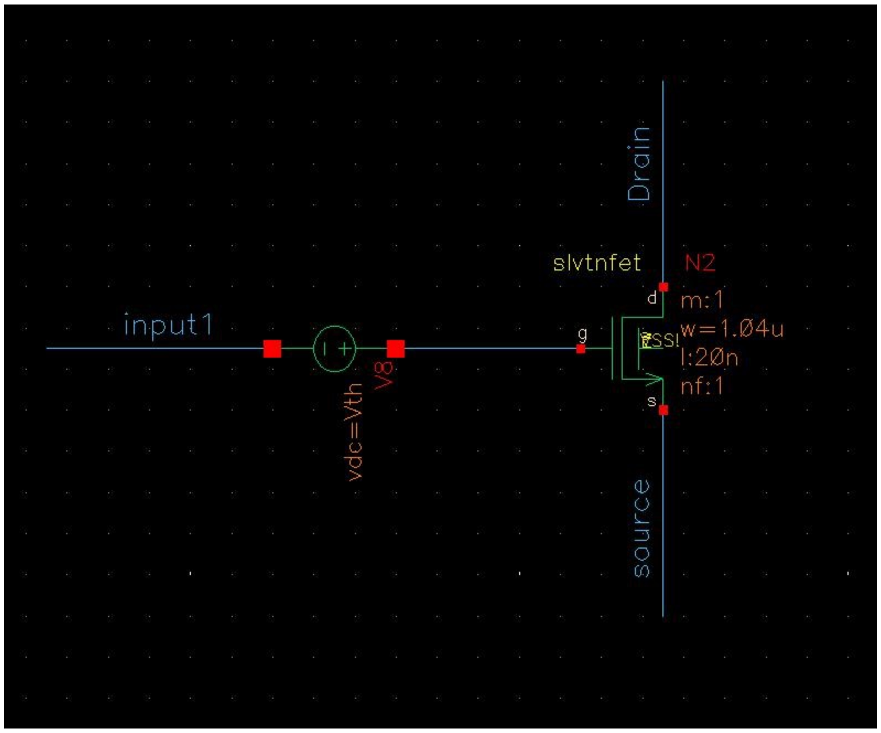


Figure 2.13: Adding a DC voltage source to demonstrate the effect of V_t change.

Considering that CTTs are mainly used in the subthreshold region, the channel current equation in this region is defined by [92]

$$I_D = I_{D0} \cdot e^{q(V_{gs}-V_t)/\eta kT}, \tag{2.4}$$

where I_{D0} is the reference current of the transistor in a specific temperature, V_{gs} in the gate

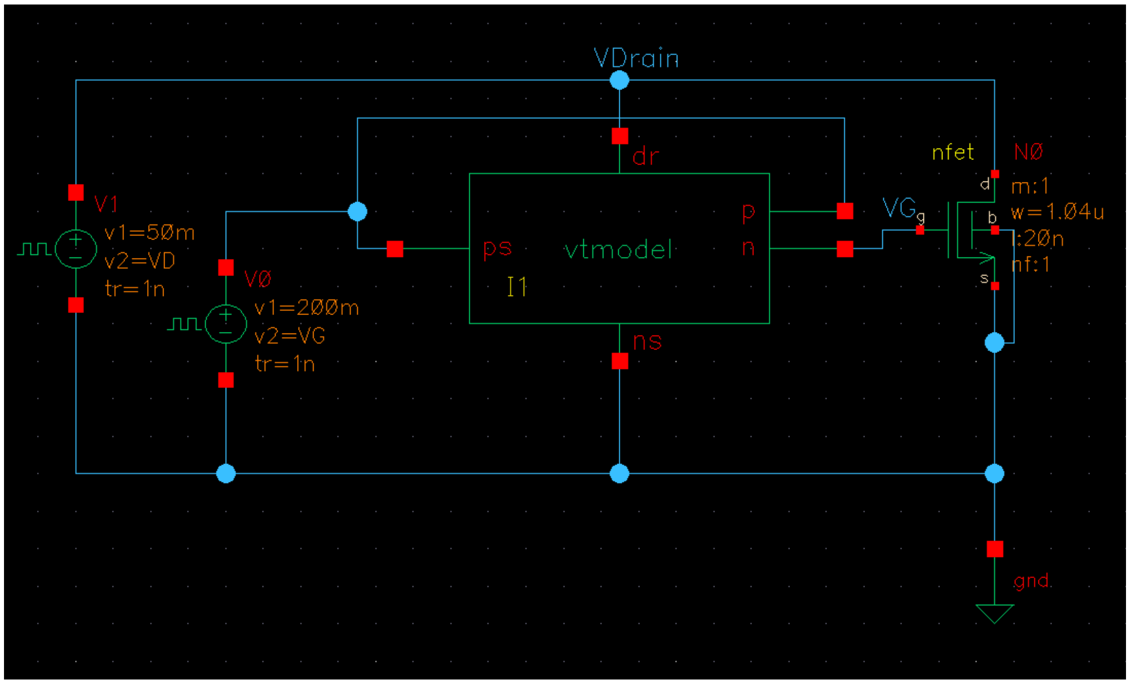


Figure 2.14: Connecting the model to the transistor to add the trapping behaviour of the device.

voltage and V_t is the threshold voltage of the device. η is the subthreshold slope factor. The transistor's programming can be indicated by decreasing the gate voltage instead of increasing the V_t [76]. The output current can be adjusted accordingly by integrating this model into the transistor's gate. The output of the model in comparison to the measured numbers ΔV_t with the same programming voltages has been shown in Fig. 2.15. This approach allows us to simulate the predicted change in V_t based on the actual programming pulses.

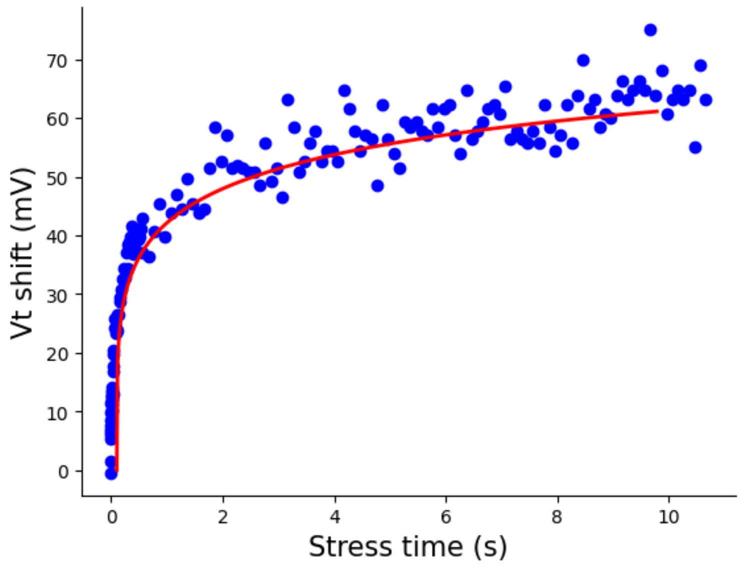


Figure 2.15: Output of the Verilog-A model vs. measured data in the chip. Both model and device have been stressed with $V_{ds} = 1.3$ V and $V_{gs} = 2.1$ V.

Chapter 3

CTTs in neuromorphic systems

This chapter discusses using CTTs as analog memory in neuromorphic systems. Depending on the applications, a couple of architectures can be utilized using CTTs. CTTs have been used in machine learning (ML) applications previously [71,76,93], and we have introduced a novel architecture that supports both ML applications and brain resemblance systems with a large number of neurons due to its scaleability [1] which has been explained here.

This publication was a team effort, and all team members were responsible for different research components. Other than the design of the CTT array, the other circuit blocks design and the software training model were done by other team members. The author was responsible for determining the general specifications of design, the hardware realization of the ML model, characterizing the weight levels in hardware considering the noises from all of the blocks, indicating an approach for utilizing input data, performing the simulations and providing the hardware results, while the other member conducted training in software using this information and matched the software and hardware results in a collaborative effort. Some of the final results of the paper are being reported here for the purpose of clarification and more details are in the paper [1].

3.1 CTT-Based Scalable Neuromorphic Architecture

This research introduces a CTT based neuromorphic architecture designed to be highly scalable and energy efficient for large neural network applications. The CTT acts as a compute-in-memory device, as explained in Chapter 2. The proposed architecture employs simple, small-area circuit blocks, aiming to create a fundamental building block (neuron) that can be replicated efficiently to form a vast network supporting both ML and biological applications at a significant scale. While this architecture demonstrates its capabilities with binary classification, it can be adapted for more complex ML tasks as well. The architecture utilizes differential current mode techniques to minimize noise impact and enhance system robustness.

The proposed CTT-based neuromorphic architecture, depicted in Fig. 3.1, comprises three key components: the array of synapses, the neuron, and the control circuitry responsible for CTT weight adjustment. Within the architecture, the array of synapses employs CTTs to perform a multiplication operation between incoming spike signals originating from fan-in neurons, and the synaptic weights stored in the CTT devices. After the CTT array, the neuron's mixed-signal circuits undertake accumulation of the products received from the CTT array and generate an outgoing spike, effectively performing a multiply-and-accumulate operation. In this context, various neuron models exist in the literature, with a tradeoff between biophysical resemblance to the human brain and complexity (in terms of area, power, and scalability). For this architecture, the LIF model is employed.

The architecture relies on the number of pulses approach for communication (as opposed to pulse amplitude or pulse width modulation). In the brain, pulses exhibit similar amplitude and pulse width, making the number of pulses approach more suitable. Additionally, this approach proves efficient and avoids the need for additional circuits such as time-to-analog converters used in pulse width modulation [2].

The proposed architecture was designed and simulated using GF 22 nm FDSOI technology. The CTT array is a significant component, where synapses are represented using twin-cell CTT devices. Each synapse is modeled using two CTTs: a target device (denoted by the superscript '+') and a reference device (denoted by the superscript '-'). The conductance of a twin-cell within the array is defined as $G_{\text{twin-cell}} = G^{+} - G^{-}$. The CTT array architecture assumes 100 synapses per neuron, represented by 100 twin cells (200 CTTs). The conductance of each CTT in a twin cell is changed based on the amount of programming (weight) on that device. During inference (activation), when a fan-in neuron fires, a spike arrives at the corresponding synapse, resulting in both CTTs within the twin-cell passing drain current according to a programmed weight. The current

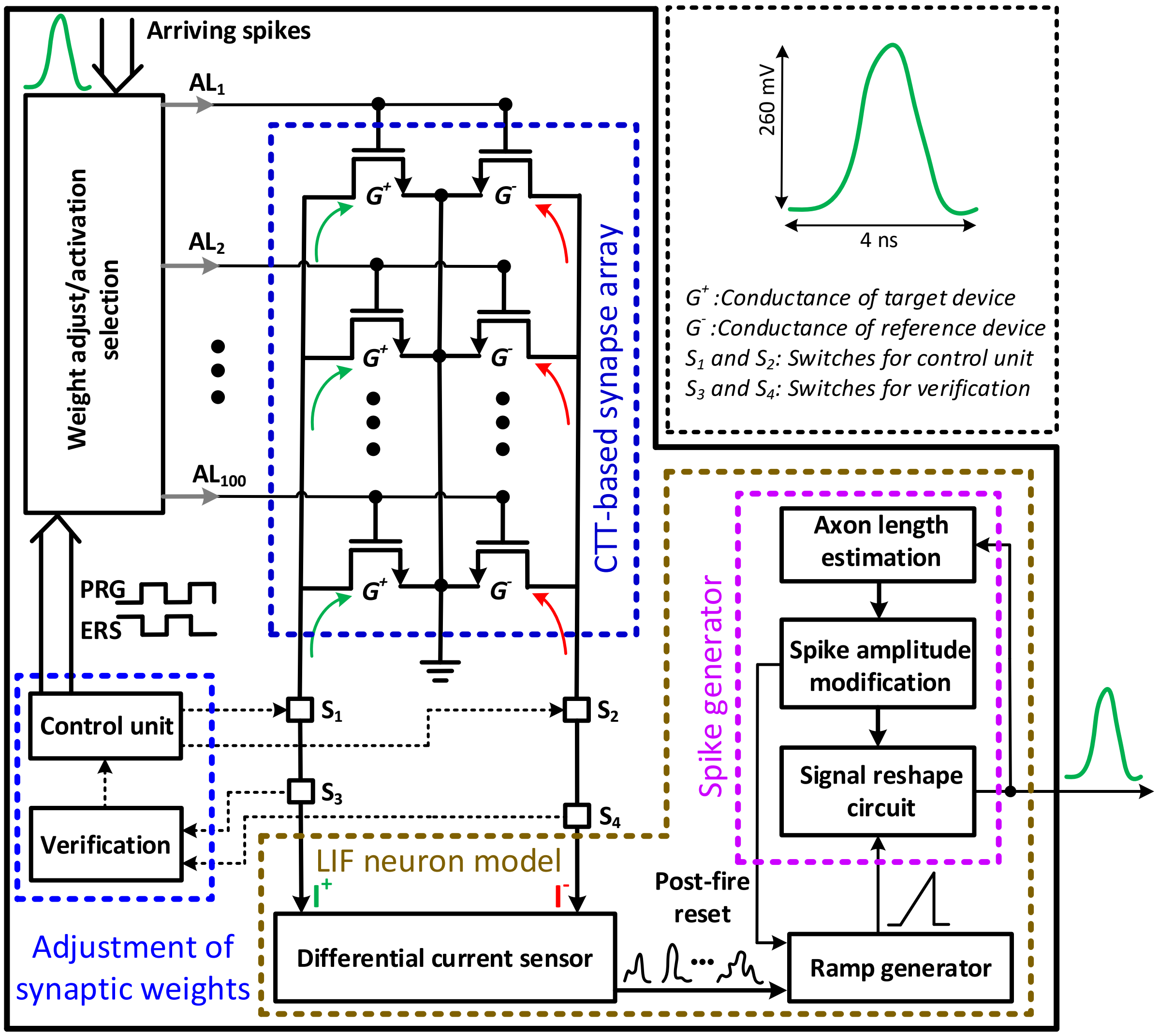

Arriving spikes
AL1
AL2
AL100
Weight adjust/activation selection
G+
G-
CTT-based synapse array
PRG
ERS
Control unit
Verification
S1
S2
S3
S4
Adjustment of synaptic weights
LIF neuron model
I+
I-
Differential current sensor
Post-fire reset
Ramp generator
Spike generator
Axon length estimation
Spike amplitude modification
Signal reshape circuit
260 mV
4 ns
G+ :Conductance of target device
G- :Conductance of reference device
S1 and S2: Switches for control unit
S3 and S4: Switches for verification

Figure 3.1: A block diagram of the proposed neuromorphic architecture. "AL" inputs represent connections (axons) from fan-in neurons that carry the incoming spikes. The proposed architecture includes three main components, array of CTTs (synapses), LIF neuron circuits, and control circuitry for weight adjustment. Incoming spikes are weighted according to the adjusted threshold voltages of the CTTs and produced charge is accumulated within the neuron circuit blocks. When a threshold is reached, the neuron produces an outgoing spike [1].

difference between the target and reference devices is captured in the differential current sensor (DCS). DCS circuit is designed to accommodate a wide range of input currents due to the unknown timing of incoming spikes and their respective weights. The subtractor unit follows the current sensor, producing the differential twin-cell current $I_{diff} = I^{+} - I^{-}$. Then the output current is stored in a capacitor which represents the membrane potential of a human brain, and the LIF neuron connected to the capacitor decides when to spike based on the voltage on the capacitor.

The layout of the designed circuit is provided in 3.2, with the CTT array occupying a silicon area of 6.5 $\mu m \times 15.5$ μm, and the circuits of the neuron model occupying 5 $\mu m \times$ 15.5 μm. The complete architecture of the neuron without weight adjustment circuitry occupies a silicon area of 11.5 $\mu m \times 15.5$ $\mu m = 178.25$ μm^2.

To evaluate the proposed architecture, a machine learning classification task was conducted in both hardware and software to recognize the digits '0' and '1'. As mentioned in Chapter 3, the programming of the Crossbar Tile Transistors (CTTs) was modeled as a DC voltage source. For each CTT, twelve different weight levels were chosen, ranging from 0 mV to 110 mV of programming. Each reference was programmed to the highest weight level (110 mV), and the software simulation determined the weight of each target cell. To model the weight level of each twin cell in software, the voltage of the capacitor was measured when a series of sample input spikes were applied to twin cells with each weight level. For hardware simulations of the entire system, the MNIST dataset for digits '0' and

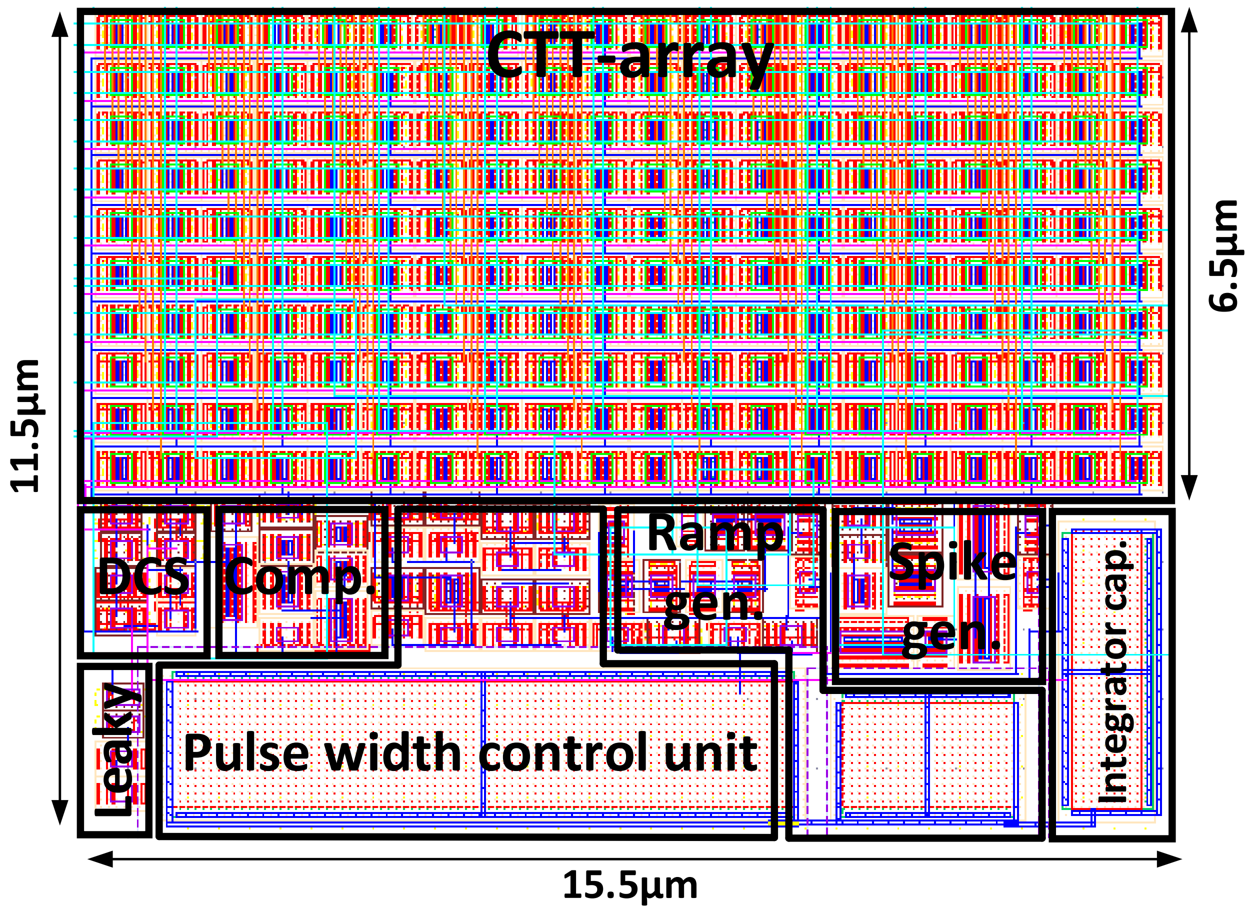


Figure 3.2: Layout of designed circuits, including the CTT array and all of the neuron circuit blocks.

'1' were encoded in the number of input pulses, where a black pixel was represented with 16 pulses and a white pixel with 0 pulses. The simulation was performed using an Oceanscript code in Cadence, and the final results were analyzed using Python.

The binary classifier neuron exhibited a 99.2% accuracy, a 98.9% recall, and a 99.4% precision on the test set in software. The optimized weights from the software and the inference data set were used in simulations of the hardware producing an accuracy, recall,

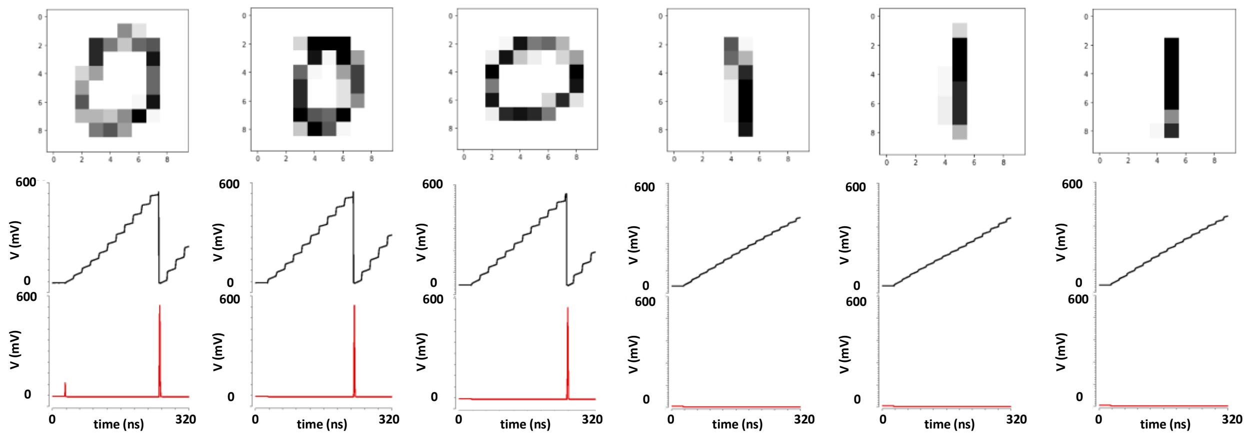


Figure 3.3: An example of the neuron classification simulation for six different images from the test data set (three for each digit). The image, V_c, and V_{spike} are shown. The neuron spikes for images containing the digit '0' and does not spike for images containing the digit '1'.

and precision of, respectively, 99.2%, 99.5%, and 98.8%, closely following the performance of the software simulation. A sample of the output of the circuit has been shown in Fig. 3.3 The architecture's parameters and characteristics were compared to prior work, showing scalability, power efficiency, and energy-per-synaptic operation of 8 pJ.

The proposed architecture demonstrates scalability, robustness, and energy efficiency, with the CTT array supporting a compute-in-memory approach, the differential circuit design ensuring noise immunity, and the Gaussian spike-based architecture enabling energy-efficient activation.

Parameter	**HICANN'10** [94]	**SpiNNaker'13** [21]	**ROLLS'15** [22]	**TrueNorth'15** [19]	**DYNAPs'17** [46]	**Loihi'18** [20]	**ODIN'19** [95]	**This work**
Approach	Mixed-signal (above-threshold)	Digital	Mixed-signal (subthreshold)	Digital	Mixed-signal (subthreshold)	Digital	Digital	Mixed-signal (Sub/above-threshold)
Technology	0.18 µm	0.13 µm	0.18 µm	28 nm	0.18 µm	14 nm FinFET	28 nm FDSOI	22 nm FDSOI
Time constant	Accelerated	Biological to accelerated	Biological	Biological	Biological	N/A	Biological to accelerated	Biological to accelerated
Number of neurons/mm^2	10.5	267	5	2.6k	34	2.5k	3k	5.6k
Synapses/mm^2	2.3k	N/A	2.5k	674k	2.1k	282k to 2.5M	741k	560k
Synaptic weight storage	4-bits SRAM	Off-chip	Capacitor	1-bit SRAM	12-bits CAM	1- to 9-bit SRAM	3+1 bits SRAM	CTT twin-cell
Supply voltage	1.8 V	1.2 V	1.8 V	0.7-1.05 V	1.3-1.8 V	0.5-1.2 V	0.55-1 V	0.8 V
Energy per synaptic operation	N/A	>26.6 nJ	>77 fJ	26 pJ (0.775 V)	N/A	>23.6 pJ (0.75 V)	12.7 pJ	8 pJ

Table 3.1: Comparison of the proposed architecture to other approaches. Literature data compiled from [2].

Chapter 4

Conclusions

In this thesis, we have explored the concept of Charge Trapping Transistor (CTT) as an analog memory and in neuromorphic systems. CTT has demonstrated tremendous potential in addressing the challenges of memory storage and retrieval in advanced neuromorphic computing architectures.

CTTs exhibit good characteristics, such as superior energy efficiency and scaleable, and since it is the usual CMOS devices used in every circuit, it is cheaper than other memories. However, We have established that programming CTTs can be a problem, especially when the number of devices goes up the programming time of the device can be a huge problem.

Therefore, this thesis focuses on the programming challenges of CTTs and it presents a comprehensive study of the behavioural modeling of CTTs within the programming voltage range. A detailed exploration of CTT behavior under various programming voltages provides valuable insights into its characteristics and operation.

Experiments were performed on multiple devices provided by Blumind [77], collecting results using an FPGA board and using the average numbers to report results trying to better understand the problems of CTTs when it comes to programming.

Based on the derived behavioural model, a Verilog-A model is developed for CTT using the Cadence Virtuoso tool [91]. This Verilog-A model is a practical and versatile representation of CTT behavior, enabling circuit designers to incorporate CTTs into their designs seamlessly.

Moreover, this research covered a novel CTT-based neuromorphic architecture with versatile applications, including brain emulation and various machine learning tasks. The architecture embraces a compute-in-memory approach, enabling a low-power, low-area, and scalable design. One of the notable advantages of CTTs is their standard CMOS transistor nature, eliminating the need for additional fabrication steps, thus streamlining the integration process. Communication within the architecture is efficiently achieved through Gaussian spikes (similar to the human brain), which are exchanged among neurons to perform computations. The system's circuit design, simulations, and layout demonstrate its anticipated functionality and robustness.

As a binary classifier, the proposed architecture is validated with a single neuron, showcasing exceptional accuracy (99.2%), recall (99.5%), and precision (98.6%). While this validation serves as a proof-of-concept, previous research has also shown impressive results on similar tasks. The neuron will undergo further testing on more complex applications and under different process, voltage, and temperature (PVT) variations.

In conclusion, CTTs hold remarkable promise for advancing analog memory technology. Their potential to store and manipulate analog signals opens up new horizons for applications in neuromorphic computing, edge AI, and beyond. However, several critical challenges remain to be addressed. The programming time, crucial for real-time processing, requires optimization without compromising reliability. Moreover, ensuring a consistent programming window after multiple erase and program cycles is essential for maintaining accurate data storage. Additionally, managing erasing issues during high-voltage programming is vital to prevent unintended data loss.

Bibliography

[1] M. Karimi, A. S. Monir, R. Mohammadrezaee, and B. Vaisband, "Ctt-based scalable neuromorphic architecture," *IEEE Journal on Emerging and Selected Topics in Circuits and Systems*, vol. 13, no. 1, pp. 96–107, 2023.

[2] C. Frenkel, D. Bol, and G. Indiveri, "Bottom-up and top-down neural processing systems design: Neuromorphic intelligence as the convergence of natural and artificial intelligence," *arXiv:2106.01288v1*, pp. 1–25, Jun. 2021.

[3] G. E. MOORE, "Progress in digital integrated electronics," *SPIE milestone series*, vol. 178, pp. 179–181, 2004.

[4] G. E. Moore, "Cramming more components onto integrated circuits," *Proceedings of the IEEE*, vol. 86, no. 1, pp. 82–85, 1998.

[5] J. Liu and C. Wang, "A survey of neuromorphic engineering–biological nervous systems realized on silicon," in *2009 IEEE Circuits and Systems International Conference on Testing and Diagnosis*, pp. 1–4, IEEE, 2009.

[6] J.-Q. Yang, R. Wang, Y. Ren, J.-Y. Mao, Z.-P. Wang, Y. Zhou, and S.-T. Han, "Neuromorphic engineering: from biological to spike-based hardware nervous systems," *Advanced Materials*, vol. 32, no. 52, p. 2003610, 2020.

[7] J. Von Neumann, "First draft of a report on the edvac," *IEEE Annals of the History of Computing*, vol. 15, no. 4, pp. 27–75, 1993.

[8] M. D. Godfrey and D. F. Hendry, "The computer as von neumann planned it," *IEEE Annals of the History of Computing*, vol. 15, no. 1, pp. 11–21, 1993.

[9] S. Wang, D. W. Zhang, and P. Zhou, "Two-dimensional materials for synaptic electronics and neuromorphic systems," *Science Bulletin*, vol. 64, no. 15, pp. 1056–1066, 2019.

[10] B. J. Shastri, A. N. Tait, T. F. de Lima, M. A. Nahmias, H.-T. Peng, and P. R. Prucnal, "Principles of neuromorphic photonics," *arXiv preprint arXiv:1801.00016*, 2017.

[11] D. Liu, H. Yu, and Y. Chai, "Low-power computing with neuromorphic engineering," *Advanced Intelligent Systems*, vol. 3, no. 2, p. 2000150, 2021.

[12] Y. Chai, "In-sensor computing for machine vision," 2020.

[13] S. Hamdioui, L. Xie, H. A. Du Nguyen, M. Taouil, K. Bertels, H. Corporaal, H. Jiao, F. Catthoor, D. Wouters, L. Eike, *et al.*, "Memristor based computation-in-memory architecture for data-intensive applications," in *2015 Design, Automation & Test in Europe Conference & Exhibition (DATE)*, pp. 1718–1725, IEEE, 2015.

[14] F. Staudigl, F. Merchant, and R. Leupers, "A survey of neuromorphic computing-in-memory: Architectures, simulators, and security," *IEEE Design & Test*, vol. 39, no. 2, pp. 90–99, 2021.

[15] S. Yu, H. Jiang, S. Huang, X. Peng, and A. Lu, "Compute-in-memory chips for deep learning: Recent trends and prospects," *IEEE circuits and systems magazine*, vol. 21, no. 3, pp. 31–56, 2021.

[16] D. Ielmini and H.-S. P. Wong, "In-memory computing with resistive switching devices," *Nature electronics*, vol. 1, no. 6, pp. 333–343, 2018.

[17] H. A. Du Nguyen, J. Yu, L. Xie, M. Taouil, S. Hamdioui, and D. Fey, "Memristive devices for computing: Beyond cmos and beyond von neumann. in 2017 ifip," in *IEEE International Conference on Very Large Scale Integration (VLSI-SoC)*, pp. 1–10.

[18] D. Ielmini and G. Pedretti, "Device and circuit architectures for in-memory computing," *Advanced Intelligent Systems*, vol. 2, no. 7, p. 2000040, 2020.

[19] F. Akopyan, J. Sawada, A. Cassidy, R. Alvarez-Icaza, J. Arthur, P. Merolla, N. Imam, Y. Nakamura, P. Datta, G.-J. Nam, B. Taba, M. Beakes, B. Brezzo, J. B. Kuang, R. Manohar, W. P. Risk, B. Jackson, and D. S. Modha, "Truenorth: Design and tool flow of a 65 mw 1 million neuron programmable neurosynaptic chip," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 34, no. 10, pp. 1537–1557, 2015.

[20] M. Davies, N. Srinivasa, T.-H. Lin, G. Chinya, Y. Cao, S. H. Choday, G. Dimou, P. Joshi, N. Imam, S. Jain, Y. Liao, C.-K. Lin, A. Lines, R. Liu, D. Mathaikutty, S. McCoy, A. Paul, J. Tse, G. Venkataramanan, Y.-H. Weng, A. Wild, Y. Yang, and H. Wang, “Loihi: A neuromorphic manycore processor with on-chip learning,” *IEEE Micro*, vol. 38, pp. 82–99, Jan./Feb. 2018.

[21] E. Painkras, L. A. Plana, J. Garside, S. Temple, F. Galluppi, C. Patterson, D. R. Lester, A. D. Brown, and S. B. Furber, “Spinnaker: A 1-w 18-core system-on-chip for massively-parallel neural network simulation,” *IEEE Journal of Solid-State Circuits*, vol. 48, no. 8, pp. 1943–1953, 2013.

[22] N. Qiao, H. Mostafa, F. Corradi, M. Osswald, F. Stefanini, D. Sumislawska, and G. Indiveri, “A reconfigurable on-line learning spiking neuromorphic processor comprising 256 neurons and 128k synapses,” *Frontiers in Neuroscience*, vol. 9, Apr. 2015.

[23] S. Ardalan, R. Farjadrad, M. Kuemerle, K. Poulton, S. Subramaniam, and B. Vinnakota, “An open inter-chiplet communication link: Bunch of wires (bow).,” *IEEE Micro*, vol. 41, pp. 54–60, Jan./Feb. 2021.

[24] S. Iyer, S. Jangam, and B. Vaisband, “Silicon interconnect fabric: A versatile heterogeneous integration platform for ai systems,” *IBM Journal of Research and Development*, vol. 63, pp. 5:1–5:16, Sep. 2019.

[25] S. Lariviere, Y. Weng, R. Vos de Wael, J. Royer, B. Frauscher, W. Zhengge, A. Bernasconi, N. Bernasconi, D. Schrader, Z. Zhang, and B. Bernhardt, "Functional connectome contractions in temporal lobe epilepsy: Microstructural underpinnings and predictors of surgical outcome," *Epilepsia*, vol. 61, pp. 164–173, May 2020.

[26] R. Rodríguez-Cruces, B. Bernhardt, and L. Concha, "Multidimensional associations between cognition and connectome organization in temporal lobe epilepsy," *NeuroImage*, vol. 213, p. 116706, Mar. 2020.

[27] O. Eluyode and D. T. Akomolafe, "Comparative study of biological and artificial neural networks," *European Journal of Applied Engineering and Scientific Research*, vol. 2, no. 1, pp. 36–46, 2013.

[28] J. Tang, F. Yuan, X. Shen, Z. Wang, M. Rao, Y. He, Y. Sun, X. Li, W. Zhang, Y. Li, *et al.*, "Bridging biological and artificial neural networks with emerging neuromorphic devices: fundamentals, progress, and challenges," *Advanced Materials*, vol. 31, no. 49, p. 1902761, 2019.

[29] P. Lorrentz, *Artificial neural systems: Principle and practice.* Bentham Science Publishers, 2015.

[30] A. K. Jain, J. Mao, and K. M. Mohiuddin, "Artificial neural networks: A tutorial," *Computer*, vol. 29, no. 3, pp. 31–44, 1996.

[31] S. L. Moshé, E. Perucca, P. Ryvlin, and T. Tomson, "Epilepsy: new advances," *The Lancet*, vol. 385, no. 9971, pp. 884–898, 2015.

[32] M. A. Arbib, *The handbook of brain theory and neural networks.* MIT press, 2003.

[33] L. F. Abbott and T. B. Kepler, "Model neurons: from hodgkin-huxley to hopfield," in *Statistical Mechanics of Neural Networks: Proceedings of the Xlth Sitges Conference Sitges, Barcelona, Spain, 3–7 June 1990*, pp. 5–18, Springer, 2005.

[34] L. F. Abbott, "Lapicque's introduction of the integrate-and-fire model neuron (1907)," *Brain research bulletin*, vol. 50, no. 5-6, pp. 303–304, 1999.

[35] C. D. Geisler and J. M. Goldberg, "A stochastic model of the repetitive activity of neurons," *Biophysical journal*, vol. 6, no. 1, pp. 53–69, 1966.

[36] B. K. Roy and D. R. Smith, "Analysis of the exponential decay model of the neuron showing frequency threshold effects," *The Bulletin of mathematical biophysics*, vol. 31, pp. 341–357, 1969.

[37] M. Nelson and J. Rinzel, "The hodgkin-huxley model," *The book of genesis*, pp. 29–49, 1995.

[38] E. Izhikevich, "Simple model of spiking neurons," *IEEE Transactions on Neural Networks*, vol. 14, pp. 1569–1572, Nov. 2003.

[39] R. Brette and W. Gerstner, “Adaptive exponential integrate-and-fire model as an effective description of neuronal activity,” *Journal of neurophysiology*, vol. 94, no. 5, pp. 3637–3642, 2005.

[40] W. Gerstner and R. Brette, “Adaptive exponential integrate-and-fire model,” *Scholarpedia*, vol. 4, no. 6, p. 8427, 2009.

[41] E. Izhikevich, “Which model to use for cortical spiking neurons?,” *IEEE Transactions on Neural Networks*, vol. 15, no. 5, pp. 1063–1070, 2004.

[42] G. Indiveri and S.-C. Liu, “Memory and information processing in neuromorphic systems,” *Proceedings of the IEEE*, vol. 103, no. 8, pp. 1379–1397, 2015.

[43] C. Koch, *Biophysics of computation: information processing in single neurons.* Oxford university press, 2004.

[44] M. Horowitz, “1.1 computing’s energy problem (and what we can do about it),” in *2014 IEEE international solid-state circuits conference digest of technical papers (ISSCC)*, pp. 10–14, IEEE, 2014.

[45] F. Akopyan, J. Sawada, A. Cassidy, R. Alvarez-Icaza, J. Arthur, P. Merolla, N. Imam, Y. Nakamura, P. Datta, G.-J. Nam, *et al.*, “Truenorth: Design and tool flow of a 65 mw 1 million neuron programmable neurosynaptic chip,” *IEEE transactions on computer-aided design of integrated circuits and systems*, vol. 34, no. 10, pp. 1537–1557, 2015.

[46] S. Moradi, N. Qiao, F. Stefanini, and G. Indiveri, “A scalable multicore architecture with heterogeneous memory structures for dynamic neuromorphic asynchronous processors (dynaps),” *IEEE Transactions on Biomedical Circuits and Systems*, vol. 12, pp. 106–122, Aug. 2017.

[47] F. Sandin, A. I. Khan, A. G. Dyer, A. H. M. Amin, G. Indiveri, E. Chicca, and E. Osipov, “Concept learning in neuromorphic vision systems: What can we learn from insects?,” *Journal of Software Engineering and Applications*, vol. 7, pp. 387–395, 2014.

[48] G. Indiveri and Y. Sandamirskaya, “The importance of space and time for signal processing in neuromorphic agents: the challenge of developing low-power, autonomous agents that interact with the environment,” *IEEE Signal Processing Magazine*, vol. 36, no. 6, pp. 16–28, 2019.

[49] G. Indiveri, E. Chicca, and R. J. Douglas, “Artificial cognitive systems: From vlsi networks of spiking neurons to neuromorphic cognition,” *Cognitive Computation*, vol. 1, pp. 119–127, 2009.

[50] L. Chua, “Memristor-the missing circuit element,” *IEEE Transactions on circuit theory*, vol. 18, no. 5, pp. 507–519, 1971.

[51] L. Chua, *Everything You Wish to Know About Memristors but Are Afraid to Ask*, pp. 89–157. Cham: Springer International Publishing, 2019.

[52] R. Marani, G. Gelao, and A. G. Perri, "A review on memristor applications," *arXiv preprint arXiv:1506.06899*, 2015.

[53] D. S. Jeong, K. M. Kim, S. Kim, B. J. Choi, and C. S. Hwang, "Memristors for energy-efficient new computing paradigms," *Advanced Electronic Materials*, vol. 2, no. 9, p. 1600090, 2016.

[54] D. B. Strukov, G. S. Snider, D. R. Stewart, and R. S. Williams, "The missing memristor found," *Nature*, vol. 453, pp. 80–83, May 2008.

[55] H.-S. P. Wong, S. Raoux, S. Kim, J. Liang, J. P. Reifenberg, B. Rajendran, M. Asheghi, and K. E. Goodson, "Phase change memory," *Proceedings of the IEEE*, vol. 98, no. 12, pp. 2201–2227, 2010.

[56] G. W. Burr, M. J. Breitwisch, M. Franceschini, D. Garetto, K. Gopalakrishnan, B. Jackson, B. Kurdi, C. Lam, L. A. Lastras, A. Padilla, *et al.*, "Phase change memory technology," *Journal of Vacuum Science & Technology B*, vol. 28, no. 2, pp. 223–262, 2010.

[57] G. W. Burr, R. M. Shelby, A. Sebastian, S. Kim, S. Kim, S. Sidler, K. Virwani, M. Ishii, P. Narayanan, A. Fumarola, *et al.*, "Neuromorphic computing using non-volatile memory," *Advances in Physics: X*, vol. 2, no. 1, pp. 89–124, 2017.

[58] P. Fantini, “Phase change memory applications: the history, the present and the future,” *Journal of Physics D: Applied Physics*, vol. 53, no. 28, p. 283002, 2020.

[59] I. Chakraborty, A. Jaiswal, A. Saha, S. Gupta, and K. Roy, “Pathways to efficient neuromorphic computing with non-volatile memory technologies,” *Applied Physics Reviews*, vol. 7, no. 2, 2020.

[60] J.-S. Lee, “Nano-floating gate memory devices,” *Electronic Materials Letters*, vol. 7, pp. 175–183, 2011.

[61] J. Boukhobza and P. Olivier, “2 - flash memories: Structure and constraints,” in *Flash Memory Integration* (J. Boukhobza and P. Olivier, eds.), pp. 15–33, Elsevier, 2017.

[62] G. Forni, C. Ong, C. Rice, K. McKee, and R. J. Bauer, “Flash memory applications,” *Nonvolatile Memory Technologies with Emphasis on Flash: A Comprehensive Guide to Understanding and Using NVM Devices*, pp. 19–62, 2007.

[63] J. Simmons and R. Verderber, “New conduction and reversible memory phenomena in thin insulating films,” *Proceedings of the Royal Society of London. Series A. Mathematical and Physical Sciences*, vol. 301, no. 1464, pp. 77–102, 1967.

[64] S. Yu, *Resistive random access memory (RRAM).* Morgan & Claypool Publishers, 2016.

[65] F. Zahoor, T. Z. Azni Zulkifli, and F. A. Khanday, “Resistive random access memory (rram): an overview of materials, switching mechanism, performance, multilevel cell

(mlc) storage, modeling, and applications," *Nanoscale research letters*, vol. 15, pp. 1–26, 2020.

[66] V. Gupta, S. Kapur, S. Saurabh, and A. Grover, "Resistive random access memory: a review of device challenges," *IETE Technical Review*, vol. 37, no. 4, pp. 377–390, 2020.

[67] C. Kothandaraman, X. Chen, D. Moy, D. Lea, S. Rosenblatt, F. Khan, D. Leu, T. Kirihata, D. Ioannou, G. LaRosa, J. B. Johnson, N. Robson, and S. S. Iyer, "Oxygen vacancy traps in hi-k/metal gate technologies and their potential for embedded memory applications," in *2015 IEEE International Reliability Physics Symposium*, pp. MY.2.1–MY.2.4, 2015.

[68] F. Khan, E. Cartier, C. Kothandaraman, J. C. Scott, J. C. S. Woo, and S. S. Iyer, "The impact of self-heating on charge trapping in high- k -metal-gate nfets," *IEEE Electron Device Letters*, vol. 37, no. 1, pp. 88–91, 2016.

[69] F. Khan, E. Cartier, J. C. S. Woo, and S. S. Iyer, "Charge trap transistor (ctt): An embedded fully logic-compatible multiple-time programmable non-volatile memory element for high- k -metal-gate cmos technologies," *IEEE Electron Device Letters*, vol. 38, no. 1, pp. 44–47, 2017.

[70] X. Gu, Z. Wan, and S. S. Iyer, "Charge-trap transistors for cmos-only analog memory," *IEEE Transactions on Electron Devices*, vol. 66, no. 10, pp. 4183–4187, 2019.

[71] Y. Du, L. Du, X. Gu, J. Du, X. S. Wang, B. Hu, M. Jiang, X. Chen, S. S. Iyer, and M.-C. F. Chang, "An analog neural network computing engine using cmos-compatible charge-trap-transistor (ctt)," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 38, no. 10, pp. 1811–1819, 2019.

[72] S. Qiao, S. Moran, D. Srinivas, S. Pamarti, and S. S. Iyer, "Demonstration of analog compute-in-memory using the charge-trap transistor in 22 fdx technology," in *2022 International Electron Devices Meeting (IEDM)*, pp. 2.5.1–2.5.4, 2022.

[73] S. Nouri and S. S. Iyer, "An 8t envsram macro in 22nm fdsoi standard logic with simultaneous full-array data restore for secure iot devices," in *Proceedings of the IEEE International Solid-State Circuits Conference*, pp. 434–436, March 2023.

[74] S. Dayo, A. Saeed Monir, M. Karimi, and B. Vaisband, "Statistical weight refresh system for ctt-based synaptic arrays," in *Proceedings of the ACM Great Lakes Symposium on VLSI*, pp. 213–214, June 2023.

[75] Y. Du, L. Du, X. Gu, J. Du, X. S. Wang, B. Hu, M. Jiang, X. Chen, S. S. Iyer, and M.-C. F. Chang, "An analog neural network computing engine using cmos-compatible charge-trap-transistor (ctt)," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 38, no. 10, pp. 1811–1819, 2019.

[76] S. Moran, *Analog In-Memory Multiply-and-Accumulate Engine Fabricated in 22nm FDSOI Technology*. PhD thesis, UCLA, 2022.

[77] "Blumind." `https://blumind.ai/`.

[78] K. O. Jeppson and C. M. Svensson, "Negative bias stress of mos devices at high electric fields and degradation of mnos devices," *Journal of Applied Physics*, vol. 48, no. 5, pp. 2004–2014, 1977.

[79] F. Khan, E. Cartier, C. Kothandaraman, J. C. Scott, J. C. Woo, and S. S. Iyer, "The impact of self-heating on charge trapping in high-k-metal-gate nfets," *IEEE Electron Device Letters*, vol. 37, no. 1, pp. 88–91, 2015.

[80] J. Viraraghavan, D. Leu, B. Jayaraman, A. Cestero, R. Kilker, M. Yin, J. Golz, R. R. Tummuru, R. Raghavan, D. Moy, *et al.*, "80kb 10ns read cycle logic embedded high-k charge trap multi-time-programmable memory scalable to 14nm fin with no added process complexity," in *2016 IEEE Symposium on VLSI Circuits (VLSI-Circuits)*, pp. 1–2, IEEE, 2016.

[81] C. Hu, S. C. Tam, F.-C. Hsu, P.-K. Ko, T.-Y. Chan, and K. W. Terrill, "Hot-electron-induced mosfet degradation-model, monitor, and improvement," *IEEE Journal of Solid-State Circuits*, vol. 20, no. 1, pp. 295–305, 1985.

[82] S. Ma, M. Donato, S. K. Lee, D. Brooks, and G.-Y. Wei, "Fully-cmos multi-level embedded non-volatile memory devices with reliable long-term retention for efficient storage of neural network weights," *IEEE Electron Device Letters*, vol. 40, no. 9, pp. 1403–1406, 2019.

[83] G. Pobegen, S. Tyaginov, M. Nelhiebel, and T. Grasser, “Observation of normally distributed energies for interface trap recovery after hot-carrier degradation,” *IEEE electron device letters*, vol. 34, no. 8, pp. 939–941, 2013.

[84] Z. Wan, *Scalable and Analog Neuromorphic Computing Systems*. PhD thesis, UCLA, 2020. ProQuest ID: Wan_ucla_0031D_18592. Merritt ID: ark:/13030/m5sf83qp.

[85] F. Khan, M. S. Han, D. Moy, R. Katz, L. Jiang, E. Banghart, N. Robson, T. Kirihata, J. C. S. Woo, and S. S. Iyer, “Design optimization and modeling of charge trap transistors (ctts) in 14 nm finfet technologies,” *IEEE Electron Device Letters*, vol. 40, no. 7, pp. 1100–1103, 2019.

[86] A. Kerber and E. A. Cartier, “Reliability challenges for cmos technology qualifications with hafnium oxide/titanium nitride gate stacks,” *IEEE Transactions on Device and Materials Reliability*, vol. 9, no. 2, pp. 147–162, 2009.

[87] T. Grasser, *Bias temperature instability for devices and circuits*. Springer Science & Business Media, 2013.

[88] S. Zafar, A. Kumar, E. Gusev, and E. Cartier, “Threshold voltage instabilities in high-/spl kappa/ gate dielectric stacks,” *IEEE Transactions on Device and Materials Reliability*, vol. 5, no. 1, pp. 45–64, 2005.

[89] X. Gu, *Charge-Trap Transistors for Neuromorphic Computing.* PhD thesis, UCLA, 2018.

[90] A. Saeed Monir, N. Rezazadeh, J. Gosson, and B. Vaisband, "Behavioral model of charge-trap transistors." IEEE Transactions on Electron Devices (TED) (to be submitted).

[91] "Cadence Virtuoso, 2022." `https://www.cadence.com/en_US/home.html`.

[92] M. Lundstrom, *Fundamentals of Nanotransistors.* Lessons from Nanoscience: a Lecture Note Series, World Scientific Publishing Company Pte. Limited, 2018.

[93] X. Gu and S. S. Iyer, "Unsupervised learning using charge-trap transistors," *IEEE Electron Device Letters*, vol. 38, no. 9, pp. 1204–1207, 2017.

[94] J. Schemmel, D. Brüderle, A. Grübl, M. Hock, K. Meier, and S. Millner, "A wafer-scale neuromorphic hardware system for large-scale neural modeling," in *Proceedings of the IEEE International Symposium on Circuits and Systems*, pp. 1947–1950, May/Jun. 2010.

[95] C. Frenkel, M. Lefebvre, J.-D. Legat, and D. Bol, "A 0.086-mm^2 12.7-pj/sop 64k-synapse 256-neuron online-learning digital spiking neuromorphic processor in 28-nm cmos," *IEEE Transactions on Biomedical Circuits and Systems*, vol. 13, pp. 145–158, Nov. 2019.