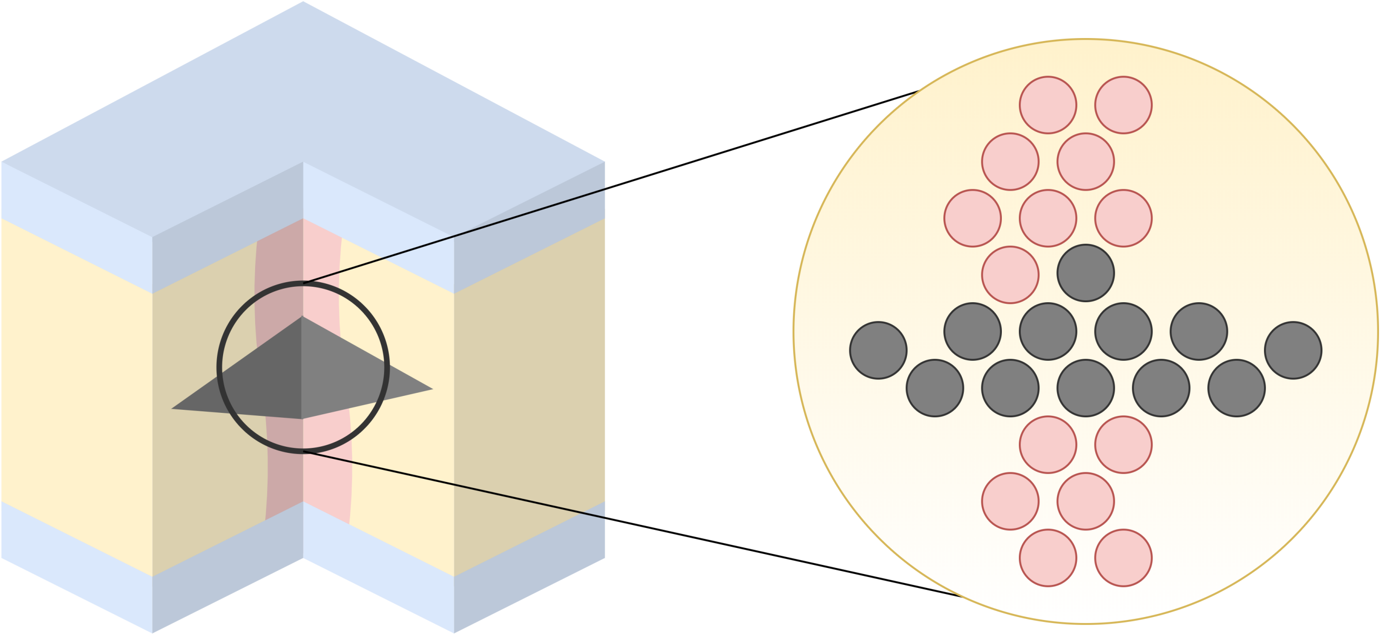
Modeling the physics of RRAM defects

A model simulating RRAM defects on a macroscopic physical level

T. Hol



Modeling the physics of RRAM defects

A model simulating RRAM defects on a macroscopic physical level

by

T. Hol

to obtain the degree of Master of Science
at the Delft University of Technology,
to be defended publicly on Friday August 28, 2020 at 13:00.

Student number:	4295528	
Project duration:	September 1, 2019 – August 28, 2020	
Thesis committee:	Prof. dr. ir. S. Hamdioui	TU Delft, supervisor
	Dr. Ir. R. Ishihara	TU Delft
	Dr. Ir. S. Vollebregt	TU Delft
	Dr. Ir. M. Taouil	TU Delft
	Ir. M. Fieback	TU Delft, supervisor

Abstract

Resistive RAM, or RRAM, is one of the emerging non-volatile memory (NVM) technologies, which could be used in the near future to fill the gap in the memory hierarchy between dynamic RAM (DRAM) and Flash, or even completely replace Flash. RRAM operates faster than Flash, but is still non-volatile, which enables it to be used in a dense 3D NVM array. It is also a suitable candidate for computation-in-memory, neuromorphic computing and reconfigurable computing.

However, the show stopping problem of RRAM is that it suffers from unique defects, which is the reason why RRAM is still not widely commercially adopted. These defects differ from those that appear in CMOS technology, due to the arbitrary nature of the forming process. They can not be detected by conventional tests and cause defective devices to go unnoticed. Therefore, new tests need to be developed that properly include the physics of a defective device in a RRAM model.

Device-aware testing (DAT) is the state-of-the-art solution to this problem. By accounting for the unique physics of an RRAM device, DAT is able to detect unique RRAM defects. However, DAT bases its results on relatively compact electrical models, which do not account for randomness present in e.g. the forming of the filament and local temperature fluctuations. Meanwhile, many low-level physical models exist already that can model this randomness and provide accurate insights into the physical specifics of RRAM. These models do, however, hardly ever analyze the effects of defects.

The contribution of this work is to expand and improve one of the state-of-the-art physical models to analyse manufacturing defects on a low, near atomic-level scale. For the first time, the characteristics of a defect can be described in the physical shape of the defect, rather than only the electrical consequences of a black box device. This enables deep level analysis and characterization of defects, the results of which improves DAT to detect even more unique defects.

The model is applied to four types of RRAM-related defects: oxygen vacancy density fluctuation, oxide thickness variation, electrode roughness, and contamination by impurities. The effect of the defects on the conductivity of the device are observed and explained, and their unique non-linear behavior is confirmed by simulation.

Finally, a discussion is presented which criticizes the reproducability of the referenced defect-free model, but also shows the potential of this work's model to be improved. Dynamic defects are not yet included, but the model does provide a static characterization of unique RRAM defects, improving the quality of DAT.

Preface

After a year of collecting information, understanding theoretical physics, but mostly attempting different implementations, this thesis is finally finished. The great majority of the time spent on this work was trying again and again to generate realistic results, taking advice from so many sources and great people - and then still, failing again and again. Thus, eventually a decision was made to report the parts of the model that produced the most realistic results and draw a conclusion.

Furthermore, the better half of the thesis process had to be done at home, due to the currently still active global pandemic. This definitely did not aid productivity and contributed to the unfortunately unfinished nature of this work.

Nevertheless, it is now done, and I have a lot of people to thank for keeping up with me through the last year: first of all, my supervisor Moritz, who was always present to provide me with valuable feedback, help, pointers, and papers. Then, my thesis advisor Said Hamdioui, for always keeping my eyes on the important parts. Thanks to Ryoichi Ishihara, who helped me understand the concept of phonons and tunneling, and getting me in touch. Also, many thanks to Gennadi Bersuker and Luca Larcher, who, even though they didn't have the required information, still helped me as much as they could. Finally, I thank my parents for never giving up on me, even though I often wanted to, and all my friends at the Delft choir and orchestra for providing me with the music everyone needs in life.

T. Hol
Delft, August 2020

Contents

1
Introduction

1.1. Motivation

Almost 50 years ago, Chua theorized the existence of the memristor as the fourth fundamental electronic component next to resistors, capacitors and inductors [24]. It was considered to be just a theoretical expansion of the already existing passive elements, until not too long ago HP Labs claimed to be the first to create a device that exhibits the characteristics of a memristor [86]. Since then, the applications of memristors, named "memristive devices", have been explored extensively [85, 95, 101]. These applications include computation-in-memory [34, 60, 104], synaptic neuromorphic computing for neural networks [23, 49], reconfigurable computing [25, 107] and reconfigurable physical unclonable functions [17]. One of the most extensively explored applications of memristive devices, however, is the implementation as Non-Volatile Memory (NVM) [53, 64, 68, 101].

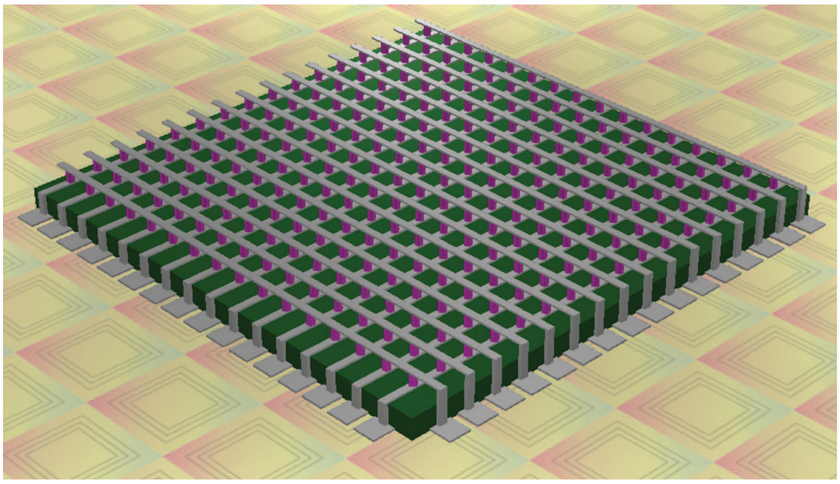


Figure 1.1: A schematic representation of the layout of a memristive crossbar memory [101].

Dense crossbar structures of these devices [53] are speculated to be able to contend with existing 3D NAND flash, given their comparable feature size, but faster read and write times [68, 101]. A schematic representation of a crossbar memory is displayed in Figure 1.1. This layout makes the devices usable as non-volatile Random Access Memory (RAM). The most prominent three of these technologies [101] are Spin-Transfer-Torque Magnetic RAM (STT-MRAM) [52], Phase-Change RAM (PCRAM) [98] and Resistive RAM (ReRAM or RRAM)[97]. The first two exhibit resistance switching, but are not true memristors. However, the last one, RRAM, behaves like a memristor and can therefore also be used for applications apart from NVM as cited before.

RRAM is based on the formation and rupture of a conductive filament through an oxide, a mechanism which was known for a long time already [45] and was also the mechanism behind HP's first memristive device [86]. A notable advantage of RRAM over other emerging NVM technologies is the fact that it can be integrated in the back end of line (BEOL) process of existing CMOS technology [97] and is therefore the leading candidate for large-scale commercial use. The manufacturing process of RRAM involves a controlled dielectric breakdown of an oxide [97]. Applying a reverse bias to this filament reoxidizes part of the filament, whereas applying a forward bias recreates the dielectric breakdown in the oxidized part. This changes the conductive properties of this device and allows it to switch between a High Resistance State (HRS) and a Low Resistance State (LRS). Unfortunately, the formation of a filament occurs randomly and abruptly, causing much variability in the resistances of the two states of the device [19, 29, 79, 81, 105]. Mitigation tactics have been invented to deal with the variability by making use of clever forming techniques [41, 65, 66]. However, these forming techniques are not perfect. Advanced forming techniques can effectively narrow the spread of HRS resistances, but are often too slow and complex to be applied on a large scale in a commercial context.

Another issue with the manufacturing of RRAM is that variations in the manufacturing process can also create defects. The existing fault models, strategies of describing the effect of defects, have previously sufficed to check for defective devices [70]. Inserting linear elements such as resistors in series or in parallel to a device [5] can model the effects of shorts or opens caused by process-induced defects. However, RRAM defects have unique effects because the formation of a filament can fail in unexpected ways [93]. Therefore, new fault models need to be developed. Examples are the undefined state (US) fault [30, 44], where it is unclear whether a cell carries a 1 or a 0, or deep state (DS) fault [50], where a too thick or too thin filament causes a device to be unable to transition under regular applied write voltages. It has been proven that these defects can not be modelled by simple linear elements [31, 32]: since an RRAM behaves like a memristor [97], its resistance changes according to its applied voltage, the duration is was applied, and the previous shape of the filament. The lack of in-depth knowledge about these kinds of defects causes them to go unnoticed, resulting in test escapes [30].

1.2. State of the Art

In the state of the art, steps have already been made to model the defects in RRAM devices, not by linear elements, but on a device-aware level, with the goal to develop more effective tests. Fieback et al. propose the principle of Device-Aware Testing (DAT), where the physical characteristics of a device defect are observed and incorporated in a dedicated defect model. In this way, DAT-models look beyond linear characterizations of defects and introduce non-linear alternatives. These alternatives are based mostly on observations on an electrical level, but lack a general connection to the actual physical processes that occur on a molecular level. Such a physical model is necessary to explain the non-linear characteristics that DAT-models use.

Fortunately, many models already exist that simulate the forming and resistive switching process of RRAM devices [57], some attempting to model atomic physical RRAM processes [27, 28, 109], some connecting these processes to electrical characteristics [1, 43, 69, 76, 81, 82] and some connecting these electrical characteristics to a larger context [20, 42, 62, 84], such as a memory array of several devices [62], or neuromorphic computing [23]. In short, these three levels of model abstraction can be called microscopic, macroscopic and compact [57]. With a few exceptions [1, 16, 31], all of these models assume a perfectly manufactured RRAM device.

Microscopic models are too complex for simulating the full resistive switching process, but provide valuable information for macroscopic models. On the other hand, compact models can easily simulate multiple switching cycles, but lack a direct connection to the atomic physics of microscopic models. Existing RRAM models that incorporate manufacturing defects and process variations [1, 16, 30] do successfully reproduce the electrical behavior of defective devices, but do not explain the reason of this behavior. Thus, a macroscopic, general defect model is needed, that can provide the necessary connection between electrical characteristics of compact models and physical processes of microscopic models.

1.3. Contribution of this work

This work fills the gap between RRAM physics and RRAM electrical behavior, in the case of a defective device. It connects electrical characteristics, that can be measured, to the underlying physical processes, that are more difficult to observe. The model combines a detailed analysis of these physical processes with an optimized MATLAB framework. Thus it gives a thorough, physics-based insight on the effects of defects and process variations on the operation of RRAM devices.
In short, the contribution of this work consists of:

- A complete analysis of the most relevant RRAM physical processes that enable resistive switching.

- An overview of the common RRAM manufacturing process, the defects that occur in it, and a description of these defects to connect them to a physical model.
- A physical RRAM model that connects the defects to electrical device characteristics to accurately reproduce the behavior of defective RRAM devices.
- An open-source MATLAB framework implementing the model that provides a large range of parameters and properties to modify and automatically sweep over.
- A validation and subsequent argumentation of the physical model, criticizing its viability and reproducability.
- An insight in the effects of oxygen thickness variation, electrode roughness and arbitrarily-shaped impurities, on the electrical characteristics of RRAM devices backed by physics.

1.4. Outline

The remainder of this thesis is organised as follows. First, background information, that explains all the basics necessary for understanding the specifics of the model implementation, will be provided in Chapter 2. Then in Chapter 3, existing research into RRAM modeling - with and without inclusion of defects and process variation - is explored. In Chapter 4, an in-depth explanation of this work's model implementation is given. Chapter 5 validates parts of the model and provides argumentation to explain the invalidated parts. After that, in Chapter 6 the model is put to use to observe the effect of manufacturing defects. Finally, this work is concluded with a discussion in Chapter 7 and a conclusion in Chapter 8.

2 Background

This chapter will provide the background information that is required to understand the relevance of memory in computing, emerging memory technologies and RRAM modeling. First, the general context of memory in computing is illustrated. Then, the principles of emerging non-volatile memory (NVM) technologies and RRAM design and operation are outlined, together with its opportunities and challenges in the implementation as a memory array. After that an insight is given in the manufacturing process of RRAM. The chapter concludes with an analysis of defects that may occur during the manufacturing process of an RRAM devices.

2.1. Memory in computing

Before considering the specifics of RRAM, it is a good idea to take a step back and consider the context. First, we look at the purpose and position of memory in a computing system. Then the problem of the memory gap is considered, followed by its mitigation strategy, memory hierarchy. Eventually an overview is given of existing, present day memory technologies.

On a broad scale, a computing system can be summarized into four parts [26]:

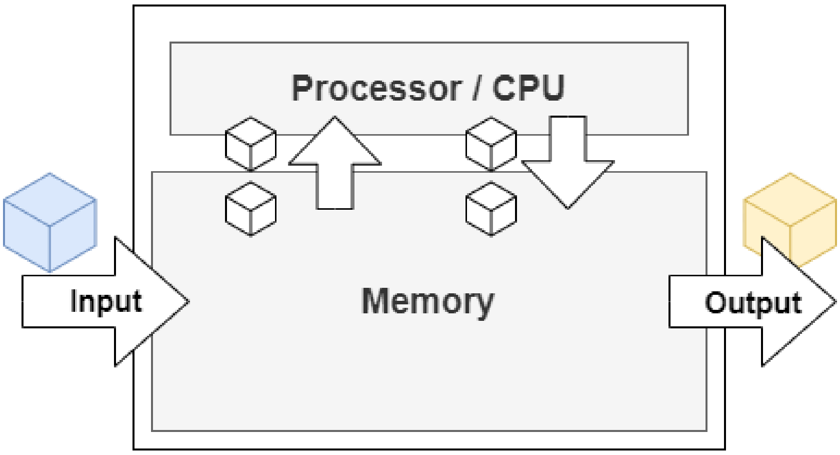


Figure 2.1: The four basic parts of a computer system: input, output, Processor/CPU and memory [26].

Figure 2.1 provides a complete overview of these four basic parts.

- **Input:** a user provides instructions and information to the computing system, telling it what it needs to do.
- **Output:** after processing the provided information, the system returns results to the user.
- **Processor or CPU:** (central processing unit) acts as the controller of the full system, performing operations on the provided information.

- **Memory:** saves the provided and the to be returned information, as well as intermediary results during processing.

The processor receives instructions from the user in assembly code via a compiler. This is why the processor is considered the most important part of a computer, but it cannot function without a memory. The memory is used as a storage for program instructions and intermediate data. Therefore, it gets accessed frequently during operation of the processor. The processor is, however, historically [96, 100] faster than the memory, an issue that is called the Memory Wall or the Memory Gap.

The Memory Gap

In 1965 Gordon E. Moore, now chairman emeritus of Intel Corp., quantified the rapid growth of semiconductor technology by an exponential function that is now known as Moore's Law [83]. His prediction became a self-fulfilled prophecy as processors became continuously faster and smaller: every two years, their speed increased by at least 50%, thanks to technology scaling and strategies such as instruction level parallelization. However, as computing power grew exponentially, the speed of memory could not keep up because much less speedup strategies were invented for memory. This caused the issue called the Memory Gap [96], named after the still growing gap between memory and processing speed. In modern processors, storing and loading data is still the bottleneck in computer operation, and by far the largest area of it is covered in memory. The memory gap can however be mitigated by making use of several different types of memory, that have optimized properties for different uses. When these memories are sorted according to their optimizations, they form the memory hierarchy.

Memory hierarchy

For a software programmer, the ideal memory is infinitely large and has instant access to requested data. Of course, realistically, memory size and access time have limits, but by exploiting the strengths of the different types of memory and data locality, these limits can be stretched. Data locality is the principle of data existing close to each other for computing purposes, either in time (temporal locality) or in space (spatial locality) [26]. If it is not necessary to keep data close, it can be stored further away from the CPU and retrieved when it is needed.

Multiple levels can be defined in this way, where, moving further away from the CPU, data is accessed less often and speed is less of an issue, which means that the level can be focused more on storage capacity. This system of levels is called memory hierarchy. An example of memory hierarchy in a computer system is given in Figure 2.2. Registers and caches are implemented closer to the processor because of their quick access times. A level below that, DRAM is used to store more data in the Random Access Memory which needs to be accessed less frequently. Then, at the base there is the main memory, a non-volatile, large data storage. The main memory uses Flash technology or disk technology.

If the processor needs a certain piece of information, it asks the cache. If the cache does not have the information (a cache miss), the cache asks the RAM for the data. If the RAM does not have the information, it asks the flash memory, which then in turn asks the main memory. The data moves up along the hierarchy and the relevant information is stored in the cache. In this way, the bottom of the hierarchy can be optimized for an affordable, large capacity, whereas the top of the hierarchy is optimized for speed and accessibility. These optimizations are a consequence of the technologies used for the types of memory in the levels.

Types of memory

There are numerous methods of storing data on a chip, but in the last few decades, the most common technologies have been SRAM, DRAM, Flash and magnetic storage [3, 11]. These technologies range from fast but expensive, to cheap but slow. Another trait of memory is volatility: if the power to a memory cell is removed, volatile memory will lose its state, while non-volatile memory (NVM) will retain its state. A list of the four memory technologies follows below:

- **SRAM** or Static Random Access Memory (SRAM) is implemented as two cross-coupled inverters with an additional two transistors acting as selectors, using a total of 6 transistors [3]. SRAM exists as the registers and the cache on the central processing unit. It is the fastest, but also most expensive memory implementation. SRAM is volatile memory.

- **DRAM** or Dynamic Random Access Memory uses just a single transistor and a small capacitor to store charge. It is smaller and thus cheaper per unit than SRAM, but the charging of the DRAM cell takes more

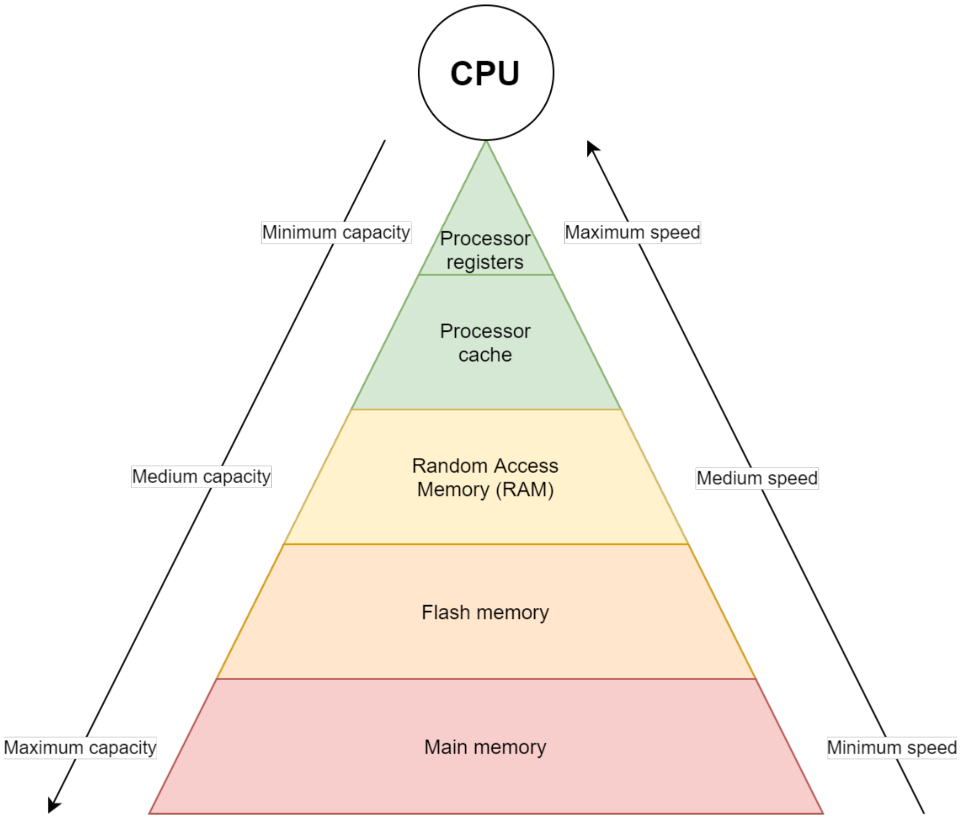


Figure 2.2: The five different levels of memory hierarchy in a computer system [26].

time than setting an SRAM [3]. DRAM exists in separate units in a computer system and is colloquially referred to as just "Random Access Memory" or RAM. DRAM is also volatile.

- **Flash** uses a single transistor and saves data by charging a transistor gate surrounded by dielectric, or floating gate in order to change its threshold voltage [11]. It is slower than DRAM, but has the added benefit of being non-volatile. In the past, flash was mostly used in small external memories such as USB storages and digital camera's [11], but today the technology is also present in solid state disks (SSDs) [110].

- **Magnetic storage** is the oldest method of storing data, first developed in the 1950s [3]. It uses magnetic fields, read and written on a spinning disk by a moving head, or on a moving tape. This allows it to store several terabytes of data of non-volatile data on a small space, making it very cheap compared to higher level memories which can separately store less than a terabyte. This affordability does come at the cost of speed, since the turning of the medium takes time. Historically, magnetic storage has always been used for low-frequency but high-capacity storage in e.g. servers or data banks [3].

By cleverly applying these memory types, it is possible to reduce the gap between memory and processor performance. However, the demand for larger and faster memories grows, and CMOS technology is hitting walls that prevent it from becoming faster and smaller [61]. Fortunately, new technologies are still emerging that can fit between levels of the memory hierarchy and close the gaps [72]. One of these promising technologies is the use of resistive switching to build non-volatile memory [101].

2.2. Emerging memory technologies and RRAM design

In the last decade, efforts to further close the memory gap have brought forth several new technologies that can fill the level between DRAM and Flash memory, or even replace it [101]. One of these technologies is Resistive RAM (RRAM) and is the main focus of this work. In this section, a brief overview will be given on emerging non-volatile memory technologies, followed by a more elaborate description of RRAM operation.

Emerging non-volatile memory technologies

The three most prominent types of emerging memories are Spin-Transfer-Torque Magnetic RAM (STT-MRAM) [52], Phase-Change RAM (PCRAM) [98] and Resistive RAM (RRAM or RRAM)[97]. All of them use resistive switching to switch between high and low resistance states. A consequence of this is hysteresis, which means that once a boundary voltage is crossed, the device quickly switches to another resistance, recording the history of the applied voltage. The following list shortly describes the principles behind the three technologies.

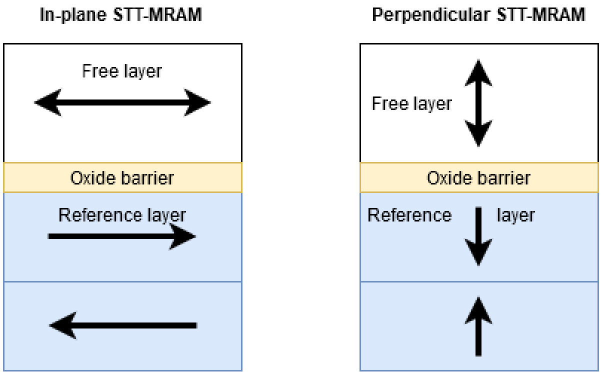


Figure 2.3: The two common layouts of STT-MRAM: in-plane (left) and perpendicular (right). The reference layer consists of two layers of opposite polarity to increase stability [52].

- **STT-MRAM** [52], as displayed in Figure 2.3, uses two magnetic layers separated by a very thin (10Å) oxide layer called a magnetic tunnel junction (MTJ). One of the layers, called the free layer, is used to store data by changing its magnetic state. The other layer is called the reference layer and retains a stable magnetic state. The reference layer often consists of two layers of opposite magnetic field direction to increase the stability of the device. The relative orientation of the magnetic fields of the free layer and the reference layer influences the resistance of the device, called the tunneling magneto-resistance (TMR) [106]. A state is written to the device using the spin-transfer torque (STT) effect [13]. The direction of the fields can be either parallel to the plane in which the device exists (in-plane) or perpendicular to it (perpendicular).

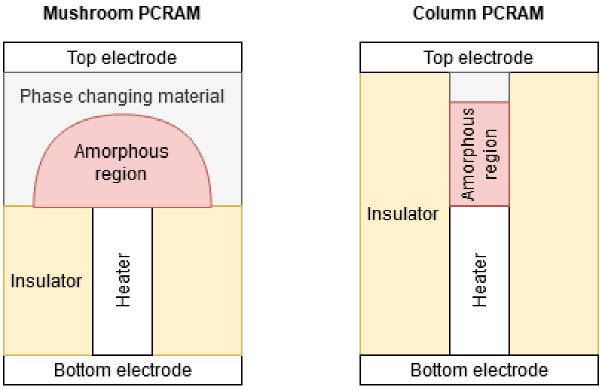


Figure 2.4: The two common layouts of PCRAM: mushroom-shaped (left) and column-shaped (right)

- **PCRAM** [98], displayed in Figure 2.4, uses the change in phase of a semiconductor material to switch between resistances and store data in that manner. The phase-changing material is put in between two electrodes, one of which has a narrow connection to the material called the heater. By applying a large current pulse, the semiconductor material melts, changing from a crystalline phase into an amorphous phase and increasing the device resistance significantly. Applying a longer pulse of medium magnitude anneals the amorphous material back into a crystalline state, reducing its resistance again. The material can be wider than the heater, causing a mushroom-shaped amorphous region when melted, or confined to a column.

- **RRAM** [97] uses a controlled dielectric breakdown to create a conductive filament to switch between resistance states. The device is a layer of oxide between two electrodes. The filament can be ruptured and regrown to switch between resistance states. Since this work concerns RRAM, the device will be explained more extensively in the next section.

Principles of RRAM

Resistive RAM (abbreviated as either RRAM or ReRAM) is manufactured by the formation of a filament in an insulator between two electrodes, called a metal-insulator-metal stack (MIM stack) [97]. The length and width of the filament determine the resistance of the device. The filament can then be ruptured and regrown again to change and switch the resistance of the device. This phenomenom has already been known and researched 60 years ago [45] but the resistive switching was not robust enough for use in memory systems [97]. In the late 1990s to 2000s, resistive switching memories regained interest [6] followed quickly by a world premiere RRAM NVM implementation by Samsung Electronics [4]. HP Labs [86] then discovered that the RRAM device exhibits memristive behavior and invented the world's first nano-scale memristor, opening up a range of new possibilities such as computation-in-memory [60, 104] and arrays for reconfigurable computing [23, 25, 107].

There are two technologies available to create a conductive filament: either by the dielectric breakdown of a metal oxide and the drift of oxygen ions (OxRAM), or by the fast diffusion of metal ions into a solid electrolyte to create a conductive bridge (CBRAM) [94, 97, 101]. Figure 2.5 shows the two available RRAM technologies. OxRAM and CBRAM have similar characteristics, but the most notable difference is that OxRAM has a smaller difference between resistance states, but a larger endurance [97]. This work will only consider OxRAM and describe the forming, reading and writing procedures of this technology.

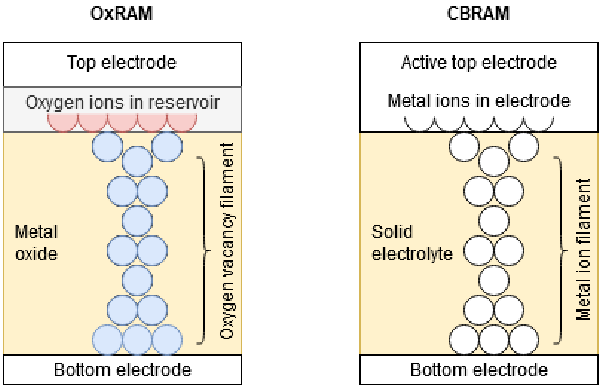


Figure 2.5: The two available RRAM technologies: oxygen vacancy powered RRAM (OxRAM, left) and conductive bridge RRAM (CBRAM, right)

The oxide layer of an OxRAM device is a metal oxide, which means that it consists of metal atoms and oxygen atoms, arranged in a certain molecular structure [97]. This structure is not perfect [10] and at some points, oxygen atoms are missing, causing positively charged oxygen vacancies. These vacancies can capture and emit electrons and thus conduct current. An externally applied electric field can push more oxygen ions away from their positions in the lattice, further decreasing the resistance of the oxide. This process is called dielectric breakdown, as it breaks down the insulating properties of the oxide, or dielectric. For the purpose of OxRAM, however, this is the mechanism that enables resistive switching.

The full operation process of an OxRAM device, consisting of initial forming and subsequent switching, can be described in six phases: Unformed device, forming the filament, low resistance state (LRS), the RESET process, high resistance state (HRS) and the SET process. The phases of operation are displayed in Figure 2.6. The letters A to F on the I-V curve correspond to the device states on the right and will be explained next.

A **Unformed device:** Directly after the MIM stack is manufactured, depending on the process [7, 51, 99], the unformed or "virgin" device already has a few oxygen vacancies present in the metal oxide. These natural vacancies exist on the boundaries between grains in the polycrystalline structure of the metal oxide [10] which already conduct current and cause leakage paths [9, 59, 91]. This leakage current is, however, still very small.

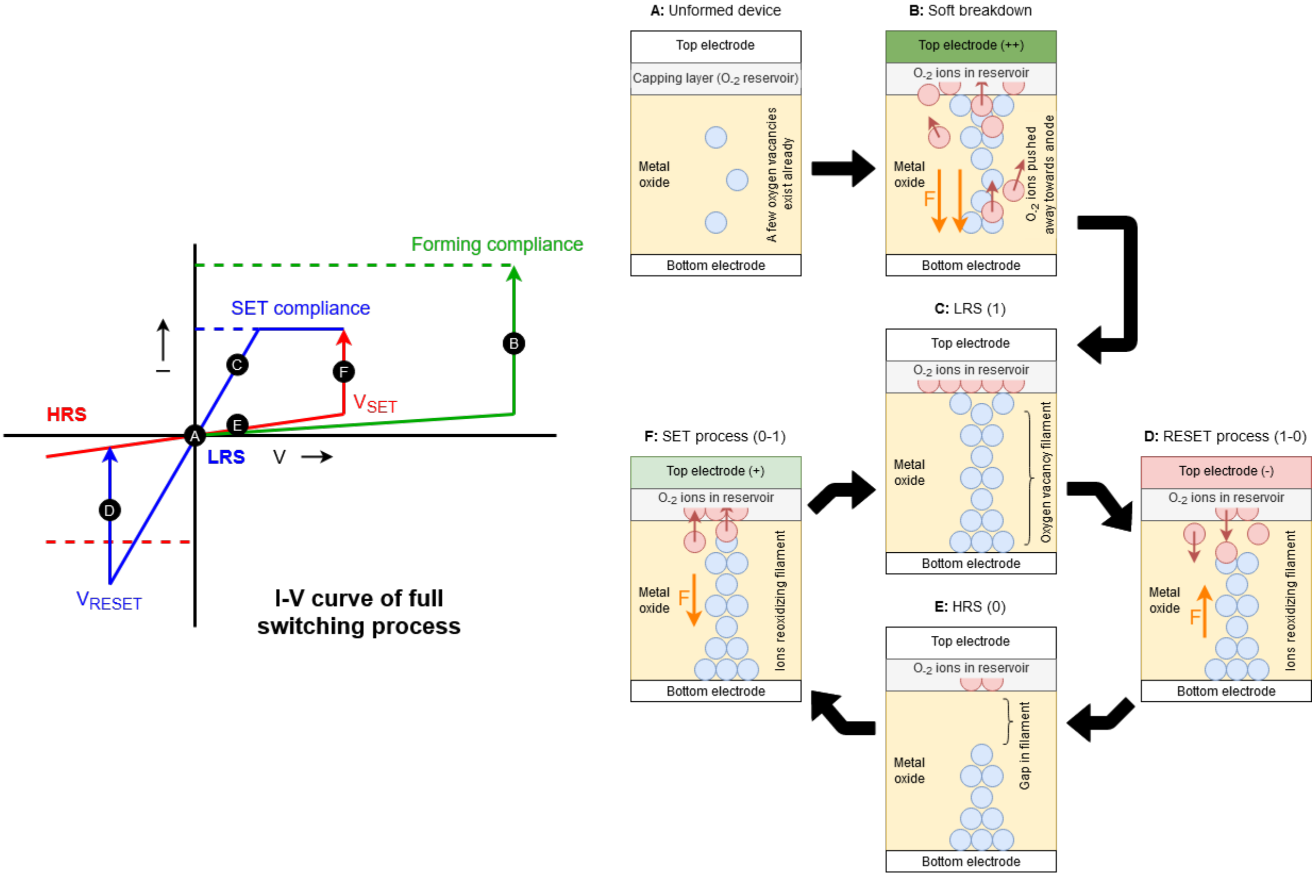


Figure 2.6: The full resistive switching process of an RRAM device of the OxRAM type, using bipolar operation. The letters A to F on the I-V curve on the left correspond to the respectively labeled figures of the device state on the right. [97]

B **Soft breakdown:** To start the forming process, characterized by the soft dielectric breakdown of the oxide, a relatively large voltage is applied to the MIM stack. This creates a large electric field inside the oxide, pushing negatively charged oxygen ions out of their positions in the oxide lattice [97]. The positively charged holes, or vacancies, left by the ions, can conduct current [59, 91]. The oxygen ions pushed out of their positions drift through interstitial positions to a capping layer (the light gray layer storing oxygen ions, underneath the top electrode in Figure 2.6), which acts as an oxygen reservoir.

The current flowing through the newly formed vacancies heats up the filament, which in turn speeds up the generation of new vacancies, which again conduct more current [76]. This positive feedback loop causes a sudden jump in current, caused by the abrupt forming of the filament in the oxide. As soon as the current crosses a compliance value, the forming process is completed and . The abruptness of the dielectric breakdown is a source of variability for the RRAM device [19, 29, 37] and forming strategies that minimize this variation are often a subject of research [38, 41, 65, 66, 77, 81, 103]

C **LRS:** After forming, the device is ready for use and exists in low resistance state. This state is interpreted as a logical 1, and can be read by applying a relatively small voltage and measuring the resulting current.

D **RESET process:** To reset the device, a relatively large reverse bias voltage is applied, pushing the oxygen ions from the reservoir back into the oxide. The ions drift towards the vacancies they left behind during the forming of the filament and reoxidize them, creating a gap in the filament. Only part of the filament needs to be oxidized to set the device to a logical 0 [97].

E **HRS:** The device is now in its high resistance state, interpreted as a logical 0. The filament is not entirely gone, but now a gap exists between the filament and one electrode. This gap increases the resistance of the device by several orders of magnitude [97]. Reading the 0 with the device in HRS works similar to in LRS, by applying a small voltage and measuring the resulting current.

F **SET process** Now, a relatively large forward bias voltage is applied, larger than a read voltage, but smaller than a forming voltage. The voltage needs to be large enough to regrow the filament, but not as large as the forming voltage because part of the filament already exists. The heat of the (HRS) filament combined with the applied electric field once again push out the oxygen ions towards the reservoir in the anode, similar to the final steps of dielectric breakdown in phase B. After a SET compliance current is reached, the filament is rebuilt and the device returns to the LRS state in phase C.

An RRAM device can have two modes of operation, depending on the type of materials used to manufacture the device. One is unipolar operation, where setting and resetting the device is only determined by the magnitude of the applied voltage. The other is bipolar operation, where setting and resetting depends both on the magnitude and the polarity of the applied voltage. The set and reset operations have reverse polarities in this case [97]. The electrical I-V characteristics of both modes of operation are displayed in Figure 2.7. This work will only consider devices that use bipolar operation.

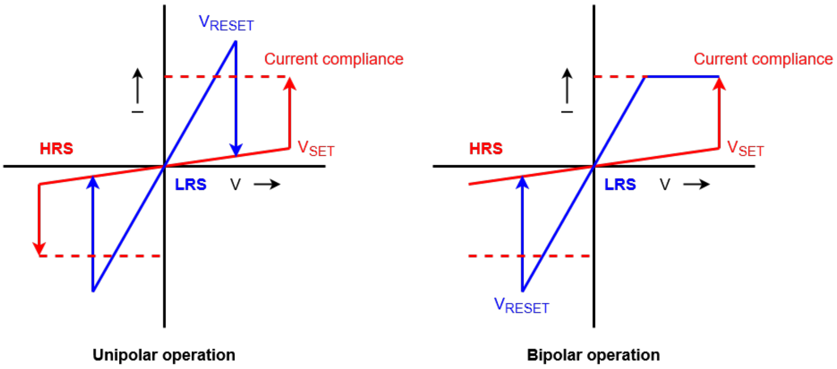


Figure 2.7: The I-V characteristics of two types of RRAM operation: unipolar (left) where the SET and RESET voltages are independent of polarity, and bipolar (right) where the SET and RESET voltages have opposite polarities [97]

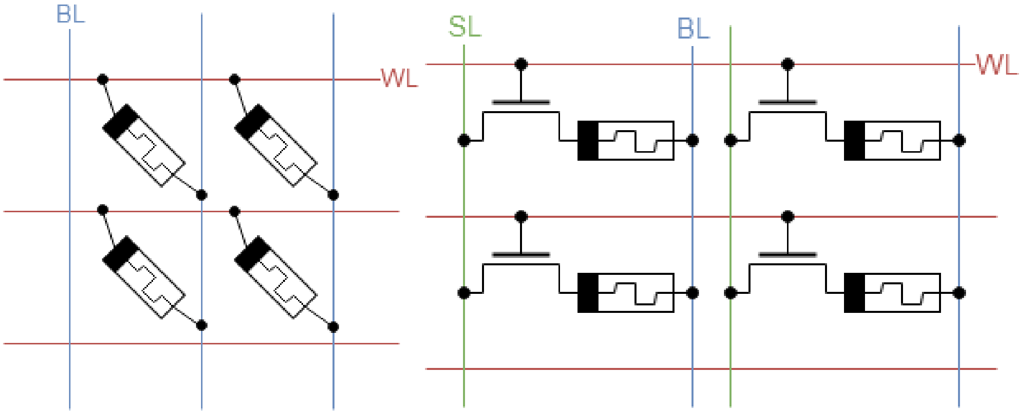
A unique capability of RRAM is the fact that it acts like a memristor [24, 86] and can thus be used for more applications than just non-volatile memory [95], such as computation-in-memory [60, 104], arrays for reconfigurable computing [23, 25, 107], hardware security [17] and neuromorphic computing [49].

Implementation of emerging NVM in a memory array

The advantages of the emerging NVM technologies discussed and compared in Section 2.2 make them a good candidate for high density non-volatile memory arrays with the potential to contend with existing Flash technology. Whereas previously the focus was mostly on the devices by itself, this section will provide an overview of the most common structures used to implement the devices into a larger context to create a memory array. These methods are: one-resistor (1R) or cross-point arrays, one-diode-one-resistor (1D1R) and one-transistor-one-resistor (1T1R) arrays [101, 102]. It should be noted that the “resistor” in context refers to the resistance switching element, i.e. the NVM device. Figure 2.8 shows the schematics of 1R and 1T1R arrays.

- **1R arrays** or cross-point arrays (Figure 2.8a) use just the NVM device, sandwiched between a crossbar structure of vias. The word lines (WL), used to select a row from the array, run perpendicular to the bit lines (BL), used to select a bit or column from a row. An advantage of this structure is that it is simple and small, since a memory cell consists of just the NVM device. The disadvantage is that selecting a single device is not trivial [101] and other devices on the same word line are influenced by a read or write signal, causing unintended writes [31, 102]. Furthermore, without a selector blocking the path to a memory cell, current can run from the bit line, through other cells, back to the word line. This issue is called sneak path and can also cause incorrect reads, as multiple cells contribute to the measured resistance [50].

- **1D1R arrays** have a diode connected in series with the NVM device [63]. This solves some of the problems of 1R arrays: current can no longer flow back up another device, eliminating the problem of sneak path. However, the structure is larger and allows for less memory array density. Furthermore, since the current through the device can only flow in one direction, the device must be compatible with unipolar operation.



(a) 1R design (b) 1T1R design

Figure 2.8: The two most common array design for NVM devices: one-resistor (1R) and one-transistor-one-resistor (1T1R) [101, 102]. The figures show how the memory cells are connected to the bit lines (BL), word lines (WL) and, if applicable, the select lines (SL).

- **1T1R arrays**, finally, solve most of the problems of 1R arrays by adding a transistor in series with the NVM device that acts as a selector (Figure 2.8b). The selector can be activated by sending a signal to the word line (WL), which opens the NMOS transistor and allows a current to run through the NVM device. 1T1R arrays are more robust, but the requirement of an additional transistor make them more expensive: the structure includes a CMOS transistor which requires it to be connected to the substrate on the bottom of the chip, reducing the density of the memory array and making implementation as 3D arrays more difficult.

2.3. RRAM manufacturing process

The manufacturing of RRAM arrays can be summarized in three steps [31]: front end of line (FEOL), back end of line (BEOL) and the forming process. The FEOL is the first part of the manufacturing process and builds the MOSFET transistors. The BEOL is the second step and adds all the metal layers of vias that connect the transistors in the FEOL. The RRAM device itself exists in the BEOL, as a metal oxide between two metal layers. Finally, the forming process concerns step B in Figure 2.6, where the conductive filament is formed.

FEOL

The front end of line manufacturing process builds the substrate that holds the transistors of a chip. This part of the process has been important in the production of chips for decades. As feature sizes got smaller and smaller, following Moore's Law [83] managing process variations has become more and more of a barrier to overcome [55, 56]. These variations include historical variations, that have been an issue and will still cause issues, and emerging variations, that were historically insignificant but now play a role. Examples of historical variations are patterning proximity effects, line-edge roughness, polish variations and variations in the gate dielectric [56]. Emerging variations are the results of downscaling the transistors down to a few tens of nanometers. Before downscaling, these variations were insignificant because they were averaged over a larger feature size, but in the last decade, transistors have become so small that new issues emerged [56]. Examples are random dopant fluctuation, variation associated with implants and anneals and granularity variations [56]. Even though these defects have little effect on RRAM devices - since these are manufactured in the BEOL - it is worth noting that the transistors that coexist with RRAM are also imperfect.

BEOL

The back end of line manufacturing process builds several metal layers in a pattern that connects the transistors below. The RRAM devices are also fabricated in these layers. The process is still equal to the original process, so many defects are already known [30]. Misalignment of the layers or small particles can cause poor connections that increase the resistance of a wire. Furthermore, line-edge roughness causes irregular shapes to form which affect the wire resistance and capacitance [87].

The first step in the fabrication of the actual RRAM devices is the patterning of the bottom electrode (BE). The process leaves a rough surface that causes variability between devices. This roughness can be mitigated by a chemical-mechanical polishing step [21], but this can still leave polish variations. The second step is the deposition of the oxide layer. The thickness of the oxide and the amount of vacancies in it are also a source of variability and need to be carefully controlled. The molecular structure of the oxide also plays a role: it can be either poly-crystalline (multiple, separate, unaligned grains of crystalline structures) or amorphous (no or very little crystalline structure) [39]. The disadvantage of poly-crystalline structures is that the grain boundaries, which are randomly distributed over the oxide, influence the variability of the devices [9]. The advantage is however that the difference between HRS and LRS resistances is much larger than in an amorphous oxide [40] which makes it easier to distinguish between the logical states 1 and 0. After that, a capping layer is deposited on the oxide, which acts as an oxygen reservoir, capturing the drifting oxygen ions during switching operation. This improves the quality of switching [29]. The capping layer can be engineered to be a more efficient reservoir, or just be a part of the top electrode. In the last step of the fabrication process, the top electrode (TE) is deposited, a process which is similar to the deposition of the BE, and experiences the same types of variations and defects.

Forming

The final step of the manufacturing process happens after the device is already completely fabricated and is also the step that is prone to the most variability. Even though the MIM stack is already built, the forming step is still considered part of the manufacturing process, because it determines the characteristics of the RRAM devices [30]. Ideally, a conductive filament is thin enough to be easily ruptured, but not too thin to still be able to conduct [18, 19, 66, 73, 74]. To achieve this, several special forming strategies have been invented, often called “soft forming” [41, 66, 81]. The three most common forming strategies are:

- **Single pulse**: (Figure 2.9) A single pulse of the forming voltage V_{form} is applied for t_{form} seconds. A finite rise and fall time is necessary to avoid current overshoot effects [36]. This effect exists with other pulse-based forming strategies as well, requiring the pulses to have a trapezoidal shape.

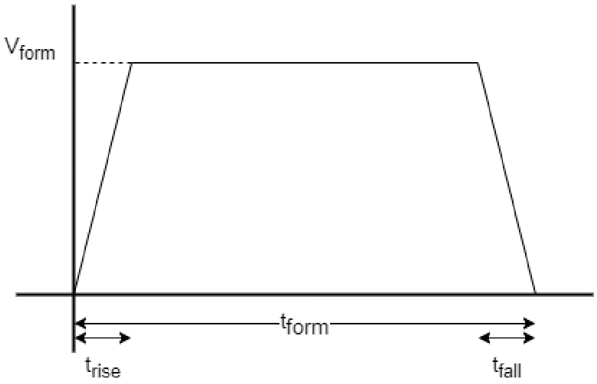


Figure 2.9: Single pulse forming: voltage (y-axis) vs. time (x-axis)

- **Incremental form (IF)** or **Ramp forming**: (Figure 2.10) Multiple pulses of length t_{pulse} are applied sequentially, increasing in voltage until a limit V_{form}.

- **Incremental form and verify (IFV)**: (Figure 2.11) Similar to IF, multiple pulses are applied, but after every pulse, the resistance of the device is measured using a small voltage. This strategy can be used to very accurately control the filament shape narrow the spread of resistances in RRAM devices.

In comparison, IFV forming results in the best quality filament, while single pulse forming is the least complex to implement. One should take into account that for commercialization, complexity can be a showstopper: the IFV process requires a read in between pulses, which can only consider one device at a time and therefore must be implemented serially. A full IFV process for a single device takes around 3.6 ms [41]. Forming a memory array of 1GB thus takes up more than 42 days. So even though IFV produces a more controllable filament, the commercially viable choice is a one-way system that can form multiple devices in parallel.

After the filament has been formed, the RRAM device is ready for use.

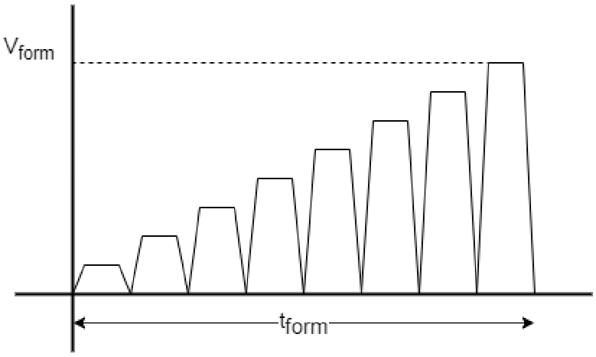


Figure 2.10: Incremental forming: voltage (y-axis) vs. time (x-axis)

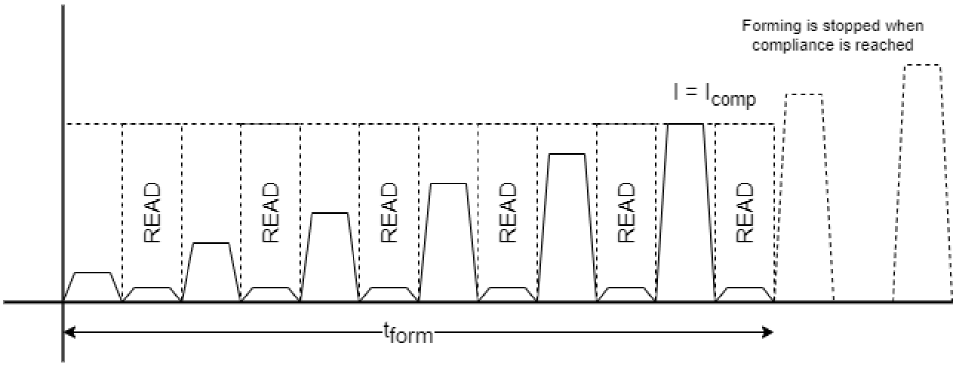


Figure 2.11: Incremental Form and Verify: voltage (y-axis) vs. time (x-axis). As soon as the measured current crosses a compliance value (I_{comp}) the forming process is stopped.

2.4. Defects

Now that the manufacturing process of an RRAM device is clear, a few categories of defects can be extracted to be translated into a physical model. Based on the process variations that cause manufacturing defects, the model will be adapted to investigate its effects.

It should be noted that much literature uses the term "defects" to describe lattice defects [8, 9, 12, 22, 91]. To prevent any confusion, this work will refer to lattice defects only as vacancies and use the term "defect" for a process variation induced deviation from the intended operation of the device. For clarity: this implies that a "defect-free" MIM stack *still contains oxygen vacancies*, because even though these vacancies are considered defective in its use as an insulator, they are a crucial part in its use as an RRAM device.

The process variations in the manufacturing process of the RRAM device that cause variations between devices and subsequently defects can be summarized into four categories [16, 81]: vacancy density variation, oxide thickness variation, electrode roughness and impurities. An overview of the four categories is given in Table 2.1. The table lists the variation that causes defects (defect source), its realization in a model and the (hypothetical) consequences of said variation. The following paragraphs elaborate on the contents of the overview table.

Vacancy density variation

Local variation in the density of vacancies is already known in transistors as random dopant fluctuation (RDF) [56]. Vacancies tend to exist on the boundaries of crystal grains (GBs) which are spread randomly through the surface of the oxide [9, 67]. Multiple GBs in a single MIM stack imply a larger vacancy density and could cause multiple or thicker filaments to grow, decreasing LRS resistance. Similarly, a smaller vacancy density could cause an underformed or even missing filament. Implementation of vacancy density variation in a model is achieved by modifying the density of vacancies (N_{V+}) in the initial lattice.

Defect source	Realization and units in modeling	Hypothetical consequences
Vacancy density	N_{V+} [cm^{-3}]	Overforming (density too high) underforming or no forming (density too low)
Oxide thickness	t_{ox} [nm]	Overforming (oxide too thin) or underforming (oxide too thick)
Electrode roughness	σ_{el} [nm]	Overforming (electrodes locally too close together) or underforming (GB located on locally far-spaced electrodes)
Impurities	material, size, shape, position	Overforming, underforming or no forming (depending on impurity characteristics)

Table 2.1: An overview of the four different defect sources resulting from process variation in the manufacturing process of RRAM devices [16, 56].

Oxide thickness variation

Process variations [55, 56] during the manufacturing of the MIM stack can also cause the metal oxide to be thicker or thinner than intended. A thicker oxide conducts less current and has a smaller internal electric field, causing an underformed filaments. Likewise, a thinner oxide conducts more current and holds a larger electric field, causing overforming [1, 16]. Oxide thickness variation can be implemented by varying the thickness of the oxide (t_{ox}) in the initial lattice.

Electrode roughness

The surface of electrodes are also not ideally flat, but show local fluctuations in thickness due to roughness [16, 56, 81]. This variation was already researched before RRAM [48, 54, 87] and can be modeled as a sinusoidal deviation [54] or a stochastic deviation [16, 48, 81]. Electrode roughness could cause the oxide to locally be quite thin, causing overforming of the filament. Both sinusoidal and stochastic implementations of electrode roughness can be characterized by a standard deviation (σ_{el}) from the planar oxide-electrode surfaces in the initial lattice.

Impurities

Finally, any random shape of any material could be contaminating any part of the stack [16]. The severity of the defect is determined by the properties of the impurity. The effects of impurities on the formation of a filament and subsequent RRAM operation are as of yet largely unresearched. Impurities can be implemented in a model by changing the characteristics of regions of the initial lattice. The shape and properties of said regions depend on the material, size, shape and position of the impurity.

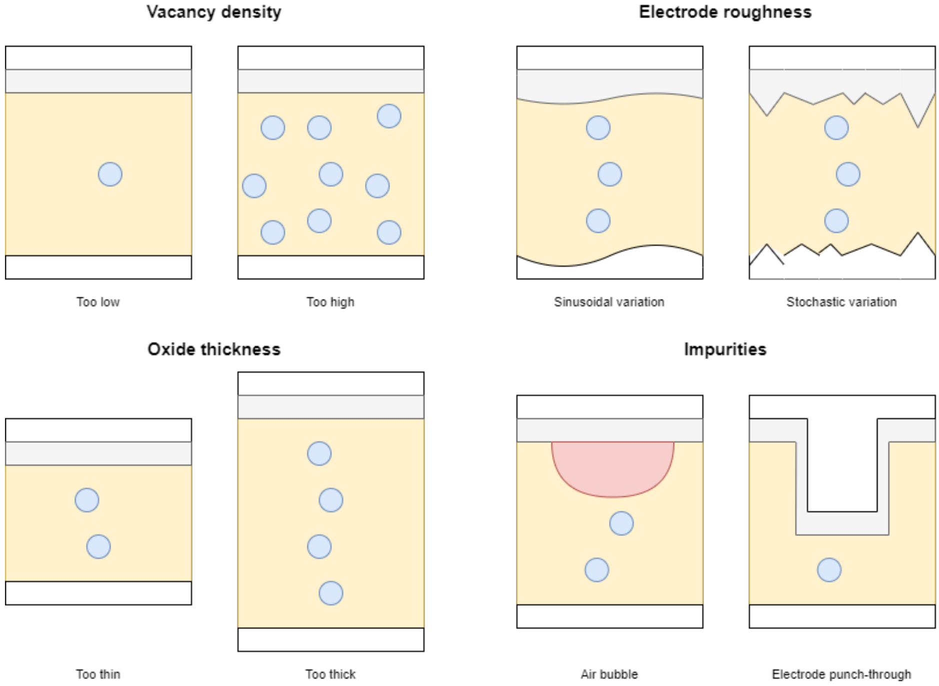


Figure 2.12: A schematic overview of the four different categories of defect sources: vacancy density, oxide thickness, electrode roughness and impurities.

3
State of the art

This chapter provides an overview of the many existing models that describe various aspects of RRAM operation to research their effects. The focus of this chapter lies on the modeling of defects, but the many studies that look at the variability of defect-free devices are also taken into account. First, a few existing studies on RRAM defect modeling are listed and explained. Since these models have shortcomings in terms of physical accuracy, in this work, a physical defect-free model is adapted to include defects. For that purpose, a classification of defect-free RRAM models is given, after which a few models are listed that fall under said classification. Finally, the defect-free models are compared and one is chosen from the listed models, based on a number of requirements that make it the best fit to include defects.

3.1. RRAM defect modeling

In this section, three existing studies on RRAM defect modeling are listed and explained. Even though the inherent variability of RRAM devices has been a subject of research for over a decade, the effects of defects have only been a main focus in the past few years. The three studies that do take RRAM defects into account are Chaudhuri and Chakrabarty ([16]), Fieback et al. ([32]) and Abbaspour et al. ([1]).

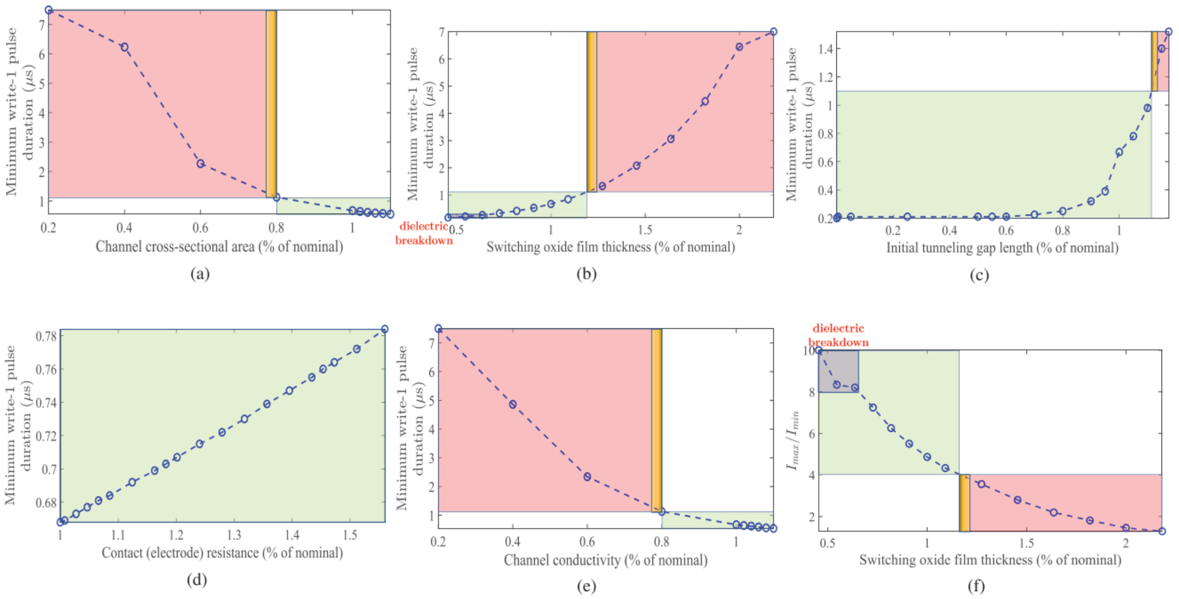
Chaudhuri and Chakrabarty, 2018

Chaudhuri and Chakrabarty ([16]) present a physics-based classification and analysis of memristor fault origins. Its motivation is the disconnection between existing fault models and the underlying physics of RRAM devices. A one-dimensional electrical model [108] is used, only modeling the position of the boundary between doped (filament) and undoped (without filament) oxide. The fault-free operation of the model is demonstrated and verified. Then, 5 process variation-induced fault origins are identified and their variation range is divided in two categories: catastrophic (causing stuck-at faults) and benign (no effect). Plots of the process variations and their categorizations are displayed in Figure 3.1.

Using the electrical model, the variations are observed and categorized. After that, the effect of breaks, voids and punch-throughs on the resistance of the electrodes is investigated. While this does give a good indication of the effects of these defects, this work draws its conclusions from adapting parameters of an electrical model. This electrical model is unable to model the shape of the filament which could also be influenced by defects. Therefore, a lower level model is needed to verify the defective behavior of this compact model.

Fieback et al., 2018

Fieback et al. ([32]) Proposes a new form of defect modeling and subsequent test development, motivated by the shortcomings of cell-aware testing, a traditional method of simulating defects by addition of linear circuit components next to the device under test. Device-aware testing (DAT) considers instead the physical characteristics of the defective device itself, rather than describing its consequences with electrical components (e.g., linear resistors) that surround a defect-free device. The DAT-process consists of three steps: defect modeling, fault modeling and test development, which help describe defects that are unique to RRAM (and STT-MRAM, which is also considered in this work) as detectable faults, to prevent test escapes. The process of DAT is displayed in Figure 3.2.



(a) (b) (c)

(d) (e) (f)

Figure 3.1: The results of Chaudhuri and Chakrabarty ([16]), a physics-based classification and analysis of RRAM fault origins. The graphs a) to f) show the effects of the labelled process variations on the minimum write-1 duration. A green shading indicates benign variation, a red shading indicates catastrophic variation, and an orange shading indicates the transition region from benign to catastrophic.

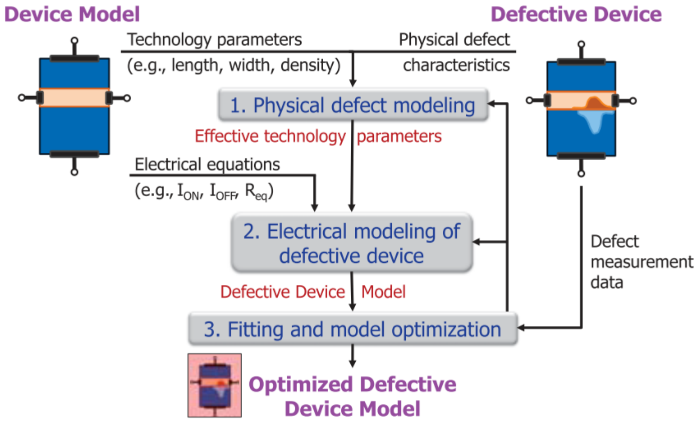

Device Model
Defective Device
Technology parameters
(e.g., length, width, density)
Physical defect
characteristics
1. Physical defect modeling
Effective technology parameters
Electrical equations
(e.g., I_{ON}, I_{OFF}, R_{eq})
2. Electrical modeling of defective device
Defect measurement data
Defective Device Model
3. Fitting and model optimization
Optimized Defective Device Model

Figure 3.2: The main principle behind Device Aware Testing (DAT) as a flowchart, as proposed in Fieback et al. ([32]). The three steps of defect modeling, electrical modeling and then fitting are shown to lead to an optimized defective device model.

The model that is used for RRAM defects directly translates technology parameters (e.g. oxide thickness, conductive filament dimensions) to electrical parameters (e.g. SET/RESET voltage threshold, HRS/LRS resistance) through a series of equations, fitted to experimental data. The technology parameters are modified to reflect defects, after which implementation context is added by putting the device in a Cadence Spectre netlist. With the new defective model, tests are developed that can sensitize the unique RRAM faults.

DAT is however more based on the electrical properties of defects, and does not take into account the lower level physical properties such as the shape of the filament.

Abbaspour, 2018

Abbaspour et al. ([1]) Presents a physical model to investigate the full resistive switching process of OxRAM. The model is three-dimensional and uses a kinetic Monte Carlo method to simulate the forming and dissolution of a vacancy-rich filament. This model is described in further detail in Section 3.3.

The main focus of this work is not defect analysis, but in the application of the model, the effect of oxide thickness on the forming operation is investigated, which is a possible source of defects. The resulting effect

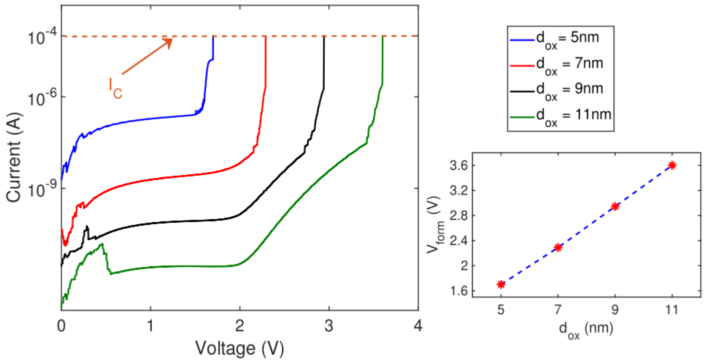


Figure 3.3: The effect of different oxide thicknesses on the forming process, resulting from the kMC model by Abbaspour et al. ([1]). A forming voltage threshold was extracted from the I-V curves and plotted on the bottom right.

on the forming I-V curve is displayed in Figure 3.3. Although, while this model does use a lower level physical implementation than the other two models described above, it lacks a specific focus on defective behaviour.

Looking at the current state of the art, it seems that in-depth research of the physical effects of defects in RRAM is still missing. Chaudhuri and Chakrabarty and Fieback et al. do focus on the effect of these defects, but base their defect simulations on modifications of parameters in fitted equations. Meanwhile, many models like Abbaspour et al. exist [1, 43, 57, 69, 76, 82] that describe the formation of a filament using physical processes, but leave defects as a side focus. A connection between low-level physical descriptions of the RRAM resistive switching process and high-level defect analysis could provide a valuable insight into the physical effect of defects, and rigor to the fittings and assumptions made in higher level models.

Creating such a model from scratch is not feasible - and unnecessary indeed, given the large number of (defect-free) models that are already available in the state of the art. Rather, an existing model should be elaborated to include defects that can be tuned to research their effects. Such a model must fit a number of requirements:

1. **The full resistive switching process must be modeled.** Forming, SET and RESET all contribute to the creation of defects. If the effect of physical defects on electrical characteristics is to be shown, then the model must be able to simulate electrical operation.

2. **It must be possible modify to the shape of the device in detail.** In order to test the effects of defects such as electrode roughness or impurities, the shape and material of various parts of the model must be adaptable.

3. **The shape and position of the filament cannot be assumed.** Defects can alter the shape and position of the conductive filament, which might contribute to the creation of defects. Therefore, a pre-definition of a filament shape - often as a trapezoid in the middle of the oxide - cannot be assumed.

4. **The model must be reproducable.** Given the information provided in the work describing the model, it must be reproducable. Unreferenced dedicated software or missing information prevents the construction of a working and verified model.

With these requirements in mind, a defect-free model can now be chosen from those available in the state of the art.to be modified to include defects. To provide orientation, a classification of models is necessary.

3.2. Classification of defect-free RRAM models

Given the large number of defect-free RRAM models in the state of the art, it is helpful to classify them depending on their scale and intended use. Lanza et al. have proposed a classification into three groups based on the level of abstraction of the physical processes involved in the operation of RRAM. These three classes are Microscopic, Macroscopic and Compact. A symbolic representation of the abstraction level and simulation scope of the classes is displayed in Figure 3.4.

The description of these classes are:

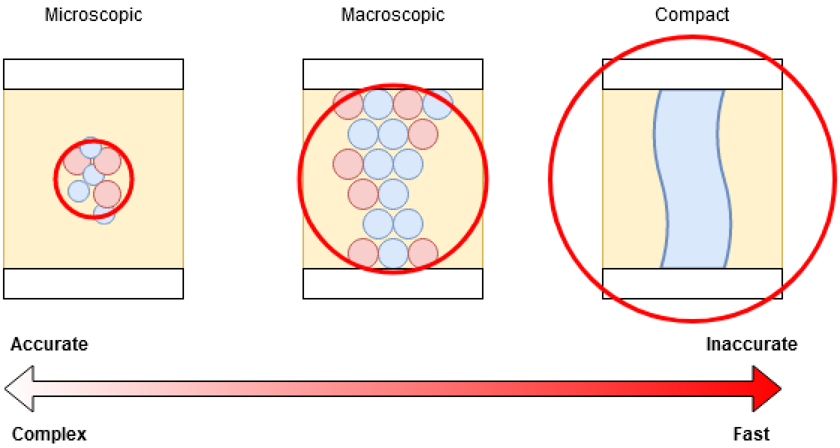


Figure 3.4: The classification of state-of-the-art resistive device models [57]: Microscopic, Macroscopic and Compact. Models on the left of the scale are more complex and small-scale, but physically accurate. Models on the right of the scale are less accurate on a physical level, but relatively fast and simple.

- **Microscopic models:** this type of model, also called an atomistic model, looks at atomic-level physics and interactions and attempts to describe them from scratch, a method called ab initio modeling. They consist of three main components:

  1. The relevant phenomena from atomic physics and material science that have proven to enable resistive switching: such as crystal lattice structures, electron localization functions [27] and polaronic interactions [109];
  2. An electron and ion charge transport model;
  3. A kinetic Monte Carlo engine to simulate the evolution of said phenomena over time.

  Other than models of a higher level of abstraction, microscopic models try to use physical characteristics obtained from measurements to describe phenomena in as much detail as possible. Thanks to the deep level description of the relevant processes, the model can be calibrated to experimental data to simulate the complex physics of resistive switching and explain them with processes and models that are already familiar. The results obtained from this type of model can then be used to calibrate models of a higher level.

  Pros: The most accurate model on an atomic scale. Describes RRAM physical processes using theoretical physics and uses it to attempt to explain measured characteristics.

  Cons: The detailed description of the physical processes on a small scale (just a few hundred atoms [27, 28, 109]) is too complex to simulate the context of RRAM operation. Increasing the scale to cover a full device is computationally intensive and impractical for its purpose.

- **Macroscopic models:** To simulate the electrical characteristics of resistive switching, the atomistic physical processes must be approximated by less computationally intensive alternatives, increasing the scale of the model. This allows the model to simulate the full operation process of a single RRAM device, based on results obtained from both experimental data and microscopic models. This combination of precise knowledge of physical processes in an RRAM device and calibration with electrical measurements from experiments makes it a powerful tool to simulate the full resistive switching process.

  Lanza et al. calls this group the kinetic Monte Carlo/Finite Element Method (kMC/FEM) group, because both methods play an important role in this class of models. The finite element method is used in the calculation of potential profiles and temperature through Poisson and Fourier equations, and kinetic Monte Carlo is used to determine the evolution of the FEM state over time, similar to microscopic models but on a larger scale.

  Pros: A combination of knowledge from the results of microscopic models with calibration to electrical measurements to fit the device's characteristics makes a macroscopic model a good tool to simulate the full RRAM switching process in detail. It provides information about electrical characteristics such as forming, set and reset voltage, and also shows the physical shape of a filament.

Cons: Approximations of physical specifics from microscopic models might not be completely accurate compared to realistic situations. On the other hand, the finite element method requires a large list of points to model the molecular lattice, making simulation of large RRAM arrays impractical.

- **Compact models:** Also called semi-empirical models, this class of models mostly uses the electrical characteristics of an RRAM device. They run in SPICE-like simulation environments and rely on the simplification of the conductive filament to, e.g., an Ohmic conductor, a quantum-point contact (QPC), or a space-charge region. Using these simplifications, the models are fitted to experimental data and macroscopic models. Some models even neglect the physical operation of an RRAM device and focus only on the reproduction of experimental data [84].

  Pros: The simplification of the physics of an RRAM device to only its electrical characteristics allows compact models to simulate many devices in parallel. Furthermore, their compatibility with SPICE allows them to be included in a complete circuit that also accounts for, e.g., interconnect and sense amplifiers.

  Cons: The physical details of the resistive switching process are lost in the simplification to a compact model. It is therefore not fit for studying specifics such as, e.g., the forming of a filament.

In conclusion, the three classes of RRAM models have different levels of abstraction which makes them useful for different purposes. Microscopic models are best for explaining physical processes involved in resistive switching with theoretical physics. Macroscopic models are best for bridging the gap between physics and electrical characteristics. Finally, compact models are best for looking at RRAM devices in context.

Now that a model classification is established, the requirements of the model that were listed in Section 3.1 can be used to choose an appropriate model class for this work. Table 3.1 compares the model classes with the strengths and weaknesses if each model class in mind.

From Table 3.1 it can be concluded that to find a defect-free model that fits the requirements from Section 3.1, a macroscopic model is needed. Microscopic and compact models can instead be used as references to retrieve information and verification required to build the macroscopic model.

3.3. Macroscopic models

This section will list the existing state of the art RRAM models, specifically those that are considered macroscopic models, and briefly address their main focus, method, conclusion and usability for this work. The usability is based on an argumentation given the requirements listed in the previous section.

Guan et al., 2012 [43]

Main focus	To study the variation in switching parameters in metal-oxide-based RRAM.
Method	A physics-based model capable of simulating a large number (>1000) of switching cycles. The model is 2D to improve performance, but does take into account the physics in the third dimension. Trap Assisted Tunneling is the dominant conduction method in HRS. In LRS, the shortest path through a filament is found and its resistance determined. Generation and recombination of vacancies is determined by probabilities.
Conclusion	A correlation is shown between the I-V curves observed at the device terminals and the stochastic vacancy dynamics in the oxide, and it is verified by measurements. Also shows the variation in resistance of HRS and LRS over multiple cycles.
Usability	This model appears to be simple to reproduce and is not computationally intensive. It does, however, make some assumptions, such as using only two dimensions, which might obfuscate interesting characteristics of filament forming in a defective device.

Table 3.1: A comparison of the three model classes presented in [57], using the requirements listed in Section 3.1. Red shading implies that the class does not meet the requirement. No shading implies that the class partially meets the requirement. Green shading implies that the class always meets the requirement.

	1. Full RS process	**2. Shape details**	**3. Unknown filament shape**	**4. Reproducability**
Microscopic	Only parts of the RS process are modelled in a very small scope. Electrical characteristics such as applied voltage over time are not considered.	The shape of the model space can be detailed down to the shape of the atoms involved.	The filament shape - if even considered - is not a product of a stochastic process but pre-defined [27, 109].	Microscopic models are often reproducible but require an in-depth understanding of theoretical physics to implement correctly.
Macroscopic	Simplifications of physical processes, linked to RS by microscopic models, allow the simulation of one or more full RS cycles.	The FEM implies that the model space can be modified within the defined resolution of the finite elements. Within this resolution, any detailing is possible.	Due to the use of kMC methods, the evolution of the shape of the filament is defined only by stochastic events over time.	Whereas ease of reproducibility varies between availiable macroscopic models, they are mostly based on a few clearly documented processes.
Compact	The simplicity of compact models allows them to simulate many RS cycles for multiple devices.	Large-scale shape details such as oxide thickness can usually be modified, but the effect of small-scale variations such as electrode roughness or impurities must be based on assumptions.	The shape of the filament is always predefined and can only vary in dimensions, such as thickness or gap width.	Compact models are based on a few explicit equations, but also rely heavily on empirical data.

Raghavan et al., 2014 [81]

Main focus	To investigate the impact of grain boundaries, process-induced vacancy distribution, metal-dielectric interface roughness and multi-layer high-κ films on the variability of forming.
Method	A Monte Carlo model with a 2D cell-based percolation model. Vacancies are generated based on a generation rate λ_G. The model only considers the generation of vacancies and the influence of process variations on it, but does not consider the current through it.
Conclusion	A link was found between process variations, such as electrode roughness and vacancy distribution, and their effect on forming time. It is shown that the presence of grain boundaries reduce the power required for forming, but increase the forming variability.
Usability	The study of process variations is a useful aspect of this work, but it does not include a charge transport model and the effect of current on vacancy generation. Furthermore, only generation rates are considered, so resetting the model is not possible. Therefore, while suitable for showing physical effects of defects, it is not useful for electrical analysis.

Sadi et al., 2015 [82]

Main focus	To study the switching process of silicon-rich (SiO_x) RRAM devices.
Method	The model couples a kMC engine and an electron transport solver to the statistical 3D TCAD simulator "GARAND", which solves Poisson equations to retrieve electric field profiles. The temperature is calculated by a more advanced in-house time-dependent heat differential equation solver.
Conclusion	The results show that resistive switching is an intrinsic property of silicon oxide. The model is proposed to facilitate efficient RRAM designs in terms of performance.
Usability	Even though this model shows promising results and is based on well documented equations, the simulator "GARAND" is no longer available. The reproducibility of this model is therefore not ideal.

Menzel et al., 2015 [69]

Main focus	To model resistive switching in electrochemical metallization cells (ECM), or Conductive Bridge RAM (CBRAM).
Method	A kMC model that describes the atomic process of RS by four different rates: ion hopping, ion reduction, oxidation, and nucleation. The potential distribution is calculated as a 2D finite element model using COMSOL Multiphysics, accounting for the potential drop at the electrode interfaces. The electron-transfer process for the tunneling current is calculated using a linear equation. Also, the influence of mechanical stress on the shape of the filament is studied.
Conclusion	It concludes that the size and shape of the filament in a CBRAM device depends on the applied voltage and mechanical stress. The work also mentions compact, 1D models by the same authors, of which the results were now further verified by this kMC model.
Usability	The model is documented clearly, with provided visualizations of the filament shape. It does, however, consider CBRAM instead of OxRAM, whereas the focus of this work lies on OxRAM.

Padovani et al., 2015 [76]

Main focus	To describe the forming and switching operations of HfO_x RRAM at a microscopic level, and their dependence on electrical conditions and device characteristics.
Method	As a charge transport model, a previously published model for Trap Assisted Tunneling is used. The potential and temperature profiles are calculated with a Poisson's equation solver. The kMC engine considers three events: generation, drift and recombination.
Conclusion	It demonstrates the dynamics of the forming process and the effect of external temperature and current compliance on the switching process. In conclusion, it presents a model linking microscopic material properties to electrical characteristics.
Usability	Even though this model is named "microscopic" by the authors, it is in fact a macroscopic model as it links the physical processes of RRAM to its electrical characteristics. Being closely based on microscopic models makes it accurate, but possibly too complex to implement. However, this model has multiple, clear references amongst which are earlier iterations that it is based on [8, 10, 58, 90, 92].

Abbaspour et al., 2018 [1]

Main focus	To present an alternative model for investigating the switching cycle of OxRAM.
Method	It uses a 3D kMC structure and only allows vacancies to form at the anode/oxide boundary by exchanging oxygen to the anode, neglecting mobile interstitial oxygen ions. Instead, vacancies are considered the mobile species. The conduction method is Trap Assisted Tunneling. The kMC engine uses three types of events: generation, recombination and vacancy drift. Using a voltage ramp, the switching process of forming, setting and resetting is simulated. It also investigates the effect of different oxide thicknesses.
Conclusion	In conclusion, it reports the development of a model that could be used to simulate hundreds of switching cycles fitted to experimental data.
Usability	The model is clearly documented, and appears simple enough to reproduce. It does, however, simplify away interstitial oxygen ions that are the actual driving force behind recombination.

3.4. Comparison and defect-free model choice

From the six state of the art models, one model will be used as a reference for this work's model to investigate the effects of defects. The choice is based on the usability of the model.

The choice of reference model will be the model by Padovani et al. [76], because of its reproducibility and physical accuracy. The other models are still good candidates, but either lack details on the model implementation, or are simplified in some way. Even though these simplifications are most often verified with experiments, they do not take into account the effects that defects might cause. Therefore it is a good idea to choose a model with as much connection with the underlying physical processes as possible.

4

Model implementation

This chapter details the implementation of the RRAM model simulating the effect of defects on the forming of the filament and subsequent resistive switching. The model is implemented in and optimized for MATLAB, but the description presented in this chapter can fit any framework.

First, an overview is provided, describing the general structure of the reference model in [76] and the general structure of the model of this work, in Section 4.1. Then, the general modules of the model are explained in Section 4.2.

After that, the chapter moves on to the implementation of actual theoretical RRAM physics. The first module is the 3D Poisson equation solver, explained in Section 4.3. Then, the charge transport model, or Trap Assisted Tunneling (TAT) solver is described in Section 4.4. The last part of the reference model, the kinetic Monte Carlo (kMC) engine, is described in Section 4.5.

This concludes the implementation of the reference model. Section 4.6 then explains the additional module, that modifies the initial model lattice state to include manufacturing defects. The complete implementation is available as MATLAB code at [47].

4.1. Overview of the model structure

This section gives an overview of both the reference model in [76] and this work's MATLAB model.

Model structure: Padovani et al.

Figure 4.1 shows the general structure of the model from Padovani et al.. It describes the steps of the looping process the model takes from start to end. The two orange boxes indicate where results from microscopic models and measurements were used to provide information to this macroscopic model. The simulation steps are displayed in the yellow boxes and are briefly elaborated below.

- **Start:** the simulation starts.
- **Vacancies and ions distribution:** before entering the calculation loop, the initial lattice is built by distributing a number of vacancies and ions in a confined 3D space. Padovani's model uses an oxide of $10 \times 10 \times 10\,\text{nm}$. The vacancies are spread uniformly and randomly throughout the oxide, according to vacancy densities measured in [78].
- **3D field map:** a Poisson equation solver is used to calculate the 3D electric field map from the externally applied voltage and the local charge of vacancies and ions in the oxide.
- **Current calculation:** the current through the oxide is calculated, using a TAT solver (see also: Section 4.4) from another work [59, 91].
- **$I_G > I_{CC}, t_{SIM} > t_F$ (Check exit condition):** stop the simulation if the current exceeds a compliance (I_{CC}) or the simulation time exceeds a maximum t_F.
- **3D power dissipation and temperature:** the TAT solver produces a power dissipation map, which is used to calculate the lattice temperature using the heat equation and a Poisson equation solver.

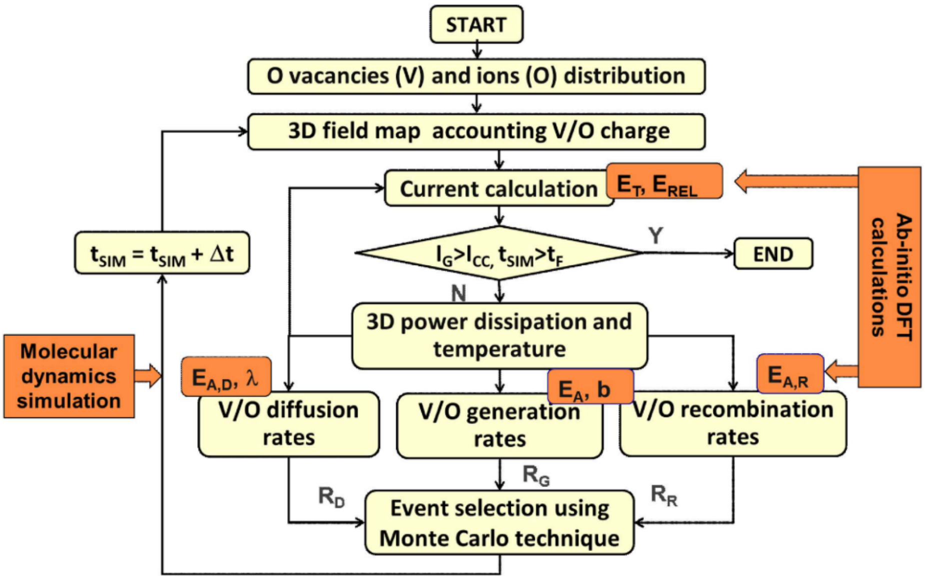

START
O vacancies (V) and ions (O) distribution
3D field map accounting V/O charge
Current calculation
E_T, E_{REL}
Ab-initio DFT calculations
$t_{SIM} = t_{SIM} + \Delta t$
$I_G > I_{CC}$, $t_{SIM} > t_F$
Y
END
N
3D power dissipation and temperature
Molecular dynamics simulation
$E_{A,D}$, λ
E_A, b
$E_{A,R}$
V/O diffusion rates
V/O generation rates
V/O recombination rates
R_G
R_D
R_R
Event selection using Monte Carlo technique

Figure 4.1: The model structure overview from Padovani et al. ([76]). The light yellow boxes represent steps of the model operation. The black arrows represent the flow of information in the operation of the model. The orange arrows and boxes represent places in the model where results from microscopic simulations are used.

- **V/O diffusion/generation/recombination rates:** event rates are calculated for three types of events: the diffusion, generation and recombination of vacancies and ions.
- **Event selection using Monte Carlo technique:** a kMC engine (see also: Section 4.5) randomly selects one of the events based on their rates and executes it.
- **$t_{SIM} = t_{SIM} + \Delta t$ (Time step):** based on the results of the kMC engine, a time step is added to the simulation time.

This structure will be used as a reference to construct the base defect-free model in MATLAB for this work. Unfortunately, the original work and its references mention only the physical details, more practical information such as used frameworks and optimizations is missing [10, 58, 76, 90, 92]. This must therefore be filled in.

Model structure: this work

Figure 4.2 shows the general structure of the model implemented in this work, which will also be explained in this chapter. The looping process is identical to the process described in the reference work [76] (Figure 4.1) but the structure in Figure 4.2 also mentions the more practical components of the model that connect the physics modules and handle input, output and parameter sweeps.

The yellow boxes represent components that were reproduced directly from the reference model, which will henceforth be referred to as *physics modules*. The blue boxes represent the software that connects the physics modules and manages inputs and outputs, the *general modules*. The general modules act as “glue” to connect the physics modules into a full model. Finally, the red box is the *defect injector*, the main addition to the model by this work.

The following list will briefly address each module of the structure and the interactions between the modules.

- **Model handler:** (Section 4.2) accepts a set of instructions that describe the simulation the model is about to perform, i.e. a variable and a set of values to create a sweep, or simply a single property. It then initiates and controls the base model object to retrieve output variables such as I-V curves, grid layouts and temperature profiles.
- **Base model object:** (Section 4.2) stores the lattice array and collects or distributes data to and from the lattice elements, to send to the physics modules according to the process specified in Figure 4.1. Every

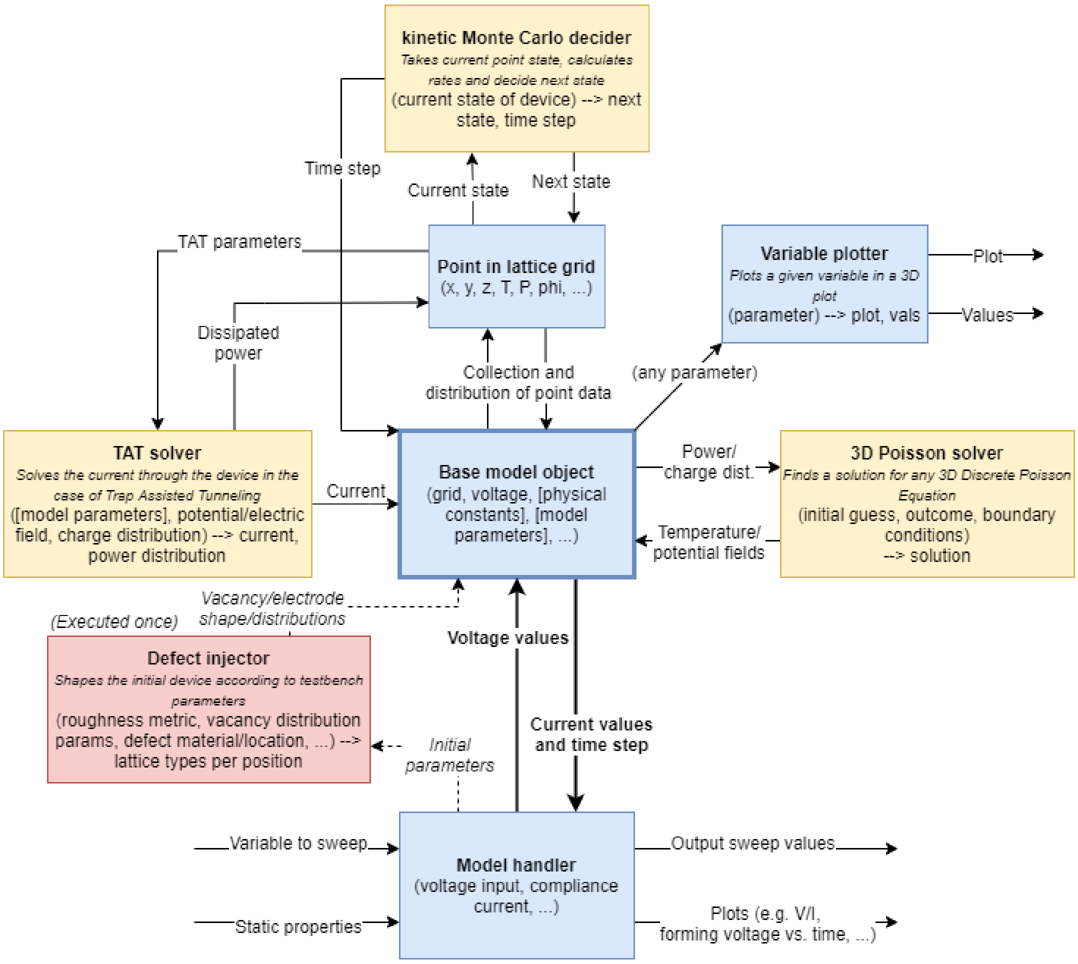

kinetic Monte Carlo decider
Takes current point state, calculates rates and decide next state
(current state of device) --> next state, time step
Time step
Current state
Next state
TAT parameters
Point in lattice grid
(x, y, z, T, P, phi, ...)
Variable plotter
Plots a given variable in a 3D plot
(parameter) --> plot, vals
Plot
Values
Dissipated power
Collection and distribution of point data
(any parameter)
TAT solver
Solves the current through the device in the case of Trap Assisted Tunneling
([model parameters], potential/electric field, charge distribution) --> current, power distribution
Current
Base model object
(grid, voltage, [physical constants], [model parameters], ...)
Power/ charge dist.
Temperature/ potential fields
3D Poisson solver
Finds a solution for any 3D Discrete Poisson Equation
(initial guess, outcome, boundary conditions) --> solution
(Executed once)
Vacancy/electrode shape/distributions
Voltage values
Defect injector
Shapes the initial device according to testbench parameters
(roughness metric, vacancy distribution params, defect material/location, ...) --> lattice types per position
Initial parameters
Current values and time step
Variable to sweep
Static properties
Model handler
(voltage input, compliance current, ...)
Output sweep values
Plots (e.g. V/I, forming voltage vs. time, ...)

Figure 4.2: The model structure of this work.

step, it returns the current and the time step (and any other information the handler requests), and accepts a new voltage value from the handler, reiterating the same loop.

- **Lattice element object:** (Section 4.2) represents a point in the lattice in the model space. Its type defines its function, such as a normal oxide, vacancy or electrode. Every lattice element also stores fields such as its local potential or temperature. The properties of every element are updated according to calculations from the physics modules.

- **Variable plotter:** plots a 3D block plot (e.g. Figure 5.2) of any property of all lattice elements.

- **3D Poisson solver:** (Section 4.3) solves the Poisson equation in 3D to obtain temperature or potential fields from respectively the power or charge distribution.

- **TAT solver:** (Section 4.4) calculates the current through the device based on the current state of the model and returns it together with a dissipated power profile for all lattice elements.

- **Kinetic Monte Carlo engine:** (Section 4.5) picks a state changing event, i.e., diffusion, generation, or recombination, based on the current state of the model and returns it with a corresponding time step.

- **Defect injector:** (Section 4.6) modifies the layout of the initial lattice grid according to (potentially) include a defect, described by parameters supplied by the handler, before the model starts running.

Together, these modules are able to model the operation of an RRAM device on a macroscopic scale that contains a manufacturing defect. In the next sections, the modules discussed in the previous list will be described in detail.

4.2. General modules

This section describes the general modules of the MATLAB model. They handle communication, input and output, and serve as the glue between the physics modules. The core general modules of the model structure, as displayed in Figure 4.2, are the model handler (source code function name: `modelHandler`), the base model object (`model`) and a point in the lattice grid (`latticeElement`). The variable plotter was mostly used for debugging and imaging purposes and has no use in the model itself, so it will be omitted in this chapter. Examples of it in use can, however, be seen in Chapter 5 and the source code is available in the complete repository at [47].

Model handler (`modelHandler`)

The `modelHandler` function serves as a wrapper for the base model object. Calling the function without arguments automatically sets up the model object with the default settings, simulating a perfectly manufactured RRAM device with default dimensions (see also: [47]). The function then handles three tasks:

- **Determine the max time step.** The applied voltage $V(t)$ is given as a MATLAB symbolic function (`syms`) dependent on the time `t`. The time step is determined by the kMC engine (see also: Section 4.5) and in turn depends again on the applied voltage, since a higher electric field in the oxide increases the event rates. This cyclical dependency is problematic: when the applied voltage is very small, event rates - which depend on the externally applied field, ergo voltage - also become very small, which produces an event with a very large time step. This would make sense if the applied voltage did not change over time; however, after some time the voltage might have increased, reducing the timestep. Therefore, the time step of an event must be verified before executing it.

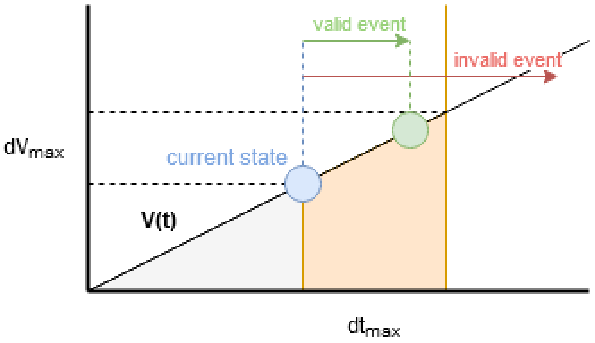


Figure 4.3: The process of picking a time step limit (`dt_max`). The symbolic function $V(t)$ is evaluated at the current state in time (blue) and, deducing from a max voltage step supplied by an argument (`dV_max`), a time step limit is calculated (orange). The kMC engine (Section 4.5) produces an event with a corresponding time step; if this step is within the limit, the event will be executed (green). Otherwise, it is discarded (red).

  Figure 4.3 shows the process of deriving a time step limit. The handler arguments supply a maximum voltage step, which defines the resolution of the simulation. MATLAB's symbolic package is then used to solve for the maximum time step to stay within the provided limits of voltage change. This time step limit is provided to the kMC engine, which then decides whether or not to execute an event, depending on the limit provided by the handler. If an event crosses the provided time limit, it is discarded.

- **Execute a model step.** With the derived time step limit, a step is executed using the `doStep` method of the base model object. This method completes one cycle of the flowchart displayed in Figure 4.1, using the components displayed in Figure 4.2. The step can result in a change of the lattice state, if the kMC event rates are high enough, or otherwise continue without events. Eitherway, a time step is returned, which is added to the simulation time.

- **Plot the model progress.** When enabled (mostly used for testing and debugging purposes), the handler function can display the state of the model and graph of several relevant variables, such as the current over time or the temperature profile, using the `plotGrid` method of the base model object.

The simulation ends either when a compliance current is reached, or when the simulation time exceeds a maximum value. The handler function then automatically saves the full progress of the simulation and returns the model object in its final state, which can then be reused, e.g. to simulate a full RS cycle after forming.

Base model object (model)

The base model object is the central controller that stores information on the model state and has several methods to handle the physics modules. It is defined in a class, using MATLAB Object Oriented Programming (OOP). The reason for this choice of implementation is that model now represents a physical object, i.e. the RRAM device. Its properties represent the state of the device, and its methods represent the actions performed on it, changing its state.

The fact that the base model object is a MATLAB OOP object also means that simulation time should be managed elsewhere, to keep the implementation intuitive. The model object receives instructions from an external handler (modelHandler) which manages the time externally. Thus, a division between the device and its testing setup is made: model represents the device itself at a fixed moment in time, while modelHandler represents the testing setup attached to its electrodes.

The most important methods of the model class are the following:

- model: the constructor method. It accepts a structure holding lattice properties, the applied voltage at the start of the simulation and the ambient temperature. It then builds the initial lattice, potentially including defects (see also Section 4.6) and returns a model object.
- doStep: executes a single step, traversing the flowchart in Figure 4.1 in the following order:
    1. Update the potential and electric field profile according to the new model state and applied voltage, using the Poisson equation solver (Section 4.3).
    2. Calculate the current through the device and the dissipated power profile, using the TAT solver (Section 4.4).
    3. Update the temperature profile according to the new power profile, using the Poisson equation solver.
    4. Calculate the time step and pick an event using the kMC engine (Section 4.5). If the event's time step is low enough, the event is executed.
    5. Return the calculated current and the time step.

  For debugging reasons, control over the model and other surveillance during model operation, the model object can by itself only execute one step at a time. An external handler (modelHandler) is used to perform the full simulation, while also keeping logs of the model's state after every step.
- plotGrid: this method represents the variable plotter displayed in Figure 4.2. It accepts an optional parameter, indicating a field of the objects in the lattice grid array (e.g. potential (phi), temperature (T), electric field (F), etc.) and plots the values of the parameter in a 3D block plot. A parameter is optional: without a parameter, the method plots the physical state of the model grid, showing the position of e.g. vacancies, ions, and the electrodes (examples of plotGrid in use can be found in the results in Chapter 5).

When the handler finishes the simulation, the model object is returned. It retains its final state, frozen in time, and can be used as the starting point for another simulation, or to investigate the properties of the simulated device in detail, by using the methods and properties of the model class.

Lattice grid element (latticeElement)

The core property of the model object is the grid. The grid consists of an array of latticeElement objects that each represent a physical point in the model lattice. They are arranged in a regularly spaced 3D grid and have a number of fields and methods to store and modify their local state. The fields of a latticeElement are:

- x,y,z: the 3D position of the element in the model space.
- type: the type of the element, saved as a string which can be any one of the following:

  - Regular oxide (`'oxide'`)
  - Oxide with an oxygen vacancy (`'V+2'`)
  - Oxide with an interstitial oxygen ion (`'O-2'`)
  - Top electrode (`'upper'`)
  - Bottom electrode (`'lower'`)

- `phi`: the local electric potential.
- `F_vec`: the local electric field as a 3D vector.
- `T`: the local temperature.
- `Q`: the electrical charge of this element.
- `P`: the power dissipated in this element.
- `E`: the trap energy of this element (see also Section 4.4).

Note that, in the source code [47], a `latticeElement` has more fields, however, these fields are only relevant to a specific part of the model (e.g. `A_row` for the Poisson solver) and will be mentioned at that part of the model to prevent confusion. The full list of fields is available in the source code [47].

The information stored in the fields of the `latticeElement` array represents the physical state of the model and is therefore used to calculate the current and next event in the TAT solver and kMC engine. This is displayed in Figure 4.2 by the connections between the `latticeElement` block and the physical modules.

The next three sections will describe the implementation of the physical modules; the Poisson equation solver, the TAT solver and the kMC engine.

4.3. Solving the discrete Poisson equation in 3D

The 3D Poisson equation solver, abbreviated as the Poisson solver, is crucial in determining the temperature and electric potential distribution inside the model lattice grid. This section will explain how it works: first, the Poisson equation itself and its applications in the model are explained. Then, the Finite Difference Method (FDM) is introduced and explained, which is crucial to applying the Poisson equation in a discrete space. Finally, the Conjugate Gradient (CG) method is explained, which is used to solve the discrete Poisson equation.

The Poisson equation and its applications

The Poisson equation is a differential equation, named after the French mathematician and physicist Siméon Denis Poisson. It appears in several forms in theoretical physics, but it always has the same general structure:

$$\nabla^2\phi = f \tag{4.1}$$

In this general form of the Poisson equation, f is given, and ϕ is sought. ∇ (pronounced “nabla”) is the divergence operator, which in 3D Cartesian coordinates takes the following form:

$$\nabla = \left[\frac{\partial}{\partial x}, \frac{\partial}{\partial y}, \frac{\partial}{\partial z}\right]^{\mathrm{T}} \tag{4.2}$$

∇ is a vector operator, which is used on a vector field to retrieve a scalar field that represents the volume density of the outward flux of the field. If multiplied with itself, however, as in the Poisson equation in Equation 4.1, it becomes a scalar operator:

$$\nabla^2 = \left(\frac{\partial^2}{\partial x} + \frac{\partial^2}{\partial y} + \frac{\partial^2}{\partial z}\right) \tag{4.3}$$

This operator is also known as the Laplace operator, written as Δ. Replacing ∇^2 with its Cartesian form in Equation 4.1 and expanding it reveals that the Poisson equation is indeed of differential nature and that solving for ϕ is non-trivial:

$$\frac{\partial^2}{\partial x}\phi(x,y,z) + \frac{\partial^2}{\partial y}\phi(x,y,z) + \frac{\partial^2}{\partial z}\phi(x,y,z) = f(x,y,z) \tag{4.4}$$

Like with any differential equation, the Poisson equation can only be solved if boundary conditions are given. This implies that within a defined boundary domain, $\phi(x,y,z)$ must have pre-determined boundary values. In the model, this domain is usually defined by the shape of the electrodes, since the operation of the model is controlled by the voltage applied on the electrodes.

In the RRAM model, the Poisson equation is used for two calculations: the electric potential field and the temperature distribution. The electric potential $\phi(x,y,z)$ is related to the charge density $\rho(x,y,z)$ and dielectric constant ϵ through Gauss's Law in differential form:

$$\nabla^2\phi(x,y,z) = -\frac{\rho(x,y,z)}{\epsilon} \tag{4.5}$$

Similarly, the temperature $T(x,y,z)$ is related to the power density $P(x,y,z)$ and the thermal conductivity k_{th} through the steady-state heat equation:

$$\nabla^2 T(x,y,z) = \frac{P(x,y,z)}{k_{th}} \tag{4.6}$$

Both Equation 4.5 and Equation 4.6 are simply Poisson equations and can therefore be solved in the exact same manner, using the same module. Therefore, from now on, the right side of the equation is defined as the given function $f(x,y,z)$, and the sought function is defined as $\phi(x,y,z)$, producing the general Poisson equation as given in Equation 4.1.

The problem is, however, that the given function $f(x,y,z)$ is determined by $\rho(x,y,z)$ or $P(x,y,z)$, neither of which can be described by a simple function. Furthermore, they are not described as an analytical function, but rather as an arbitrary collection of data points. Therefore a discrete solver of the Poisson equation must be implemented. To achieve this, the Finite Difference Method is used.

Finite Difference Method

The Finite Difference Method (FDM) is a technique to find the derivative of equations that are defined not as an analytical function, but discretely. Discrete functions are a series of data points of a finite length. A discrete version of ϕ is, for example, defined as N indexed points ϕ_i, for index $i \in [1,2,3,\ldots,N]$. Every discrete data point ϕ_i represents a sample from a continuous space. If these samples are equally spaced, they can be connected to a continuous space by defining a step size. A 1D example, with a step size Δx, is:

$$\phi_i = \phi(i\Delta x) \tag{4.7}$$

The value of $\phi(x)$ *between* the data points ϕ_i (for $x \neq i\Delta x$), however, is not known and must be assumed by interpolation. The FDM uses linear interpolation to assume the shape of the continuous function between the known samples. Interpolation means defining the values between discrete points as an analytic function, using the values of the points. Linear interpolation accomplishes this by using first-order linear functions: simple lines. An example of linear interpolation between data points is displayed in Figure 4.4a.

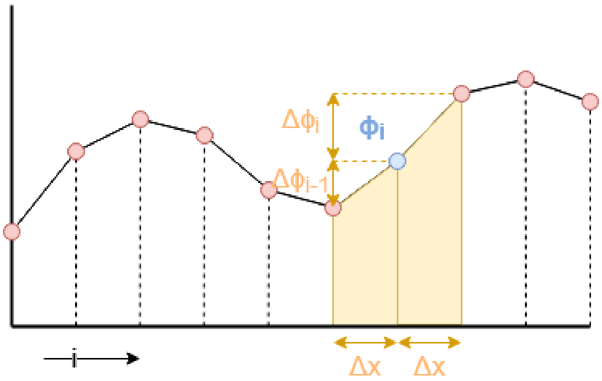
Linear interpolation of the discrete points ϕ_i allows the calculation of the local derivative: $\frac{\mathrm{d}\phi}{\mathrm{d}x}$. Since the interpolated piece is a simple line, its derivative is a constant gradient, defined by:

$$\frac{\mathrm{d}\phi}{\mathrm{d}x} \approx \frac{\Delta\phi_i}{\Delta x} \tag{4.8}$$

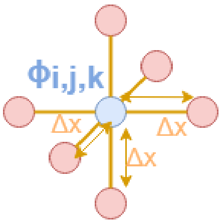
where $\Delta\phi_i = \phi_{i+1} - \phi_i$, or the difference between two data points.

The Poisson equation requires a *second* derivative (see Equation 4.4). That implies the derivative of the derivative, or the change in change. The FDM must thus be applied again, but this time calculating the difference between the gradients of two line pieces. This is also illustrated in Figure 4.4a. The second derivative at ϕ_i is thus calculated as:

$$\frac{\mathrm{d}^2}{\mathrm{d}x^2}\phi = \frac{\mathrm{d}}{\mathrm{d}x}\left(\frac{\mathrm{d}\phi}{\mathrm{d}x}\right) \approx \frac{\left(\frac{\Delta\phi_i}{\Delta x}\right) - \left(\frac{\Delta\phi_{i-1}}{\Delta x}\right)}{\Delta x} \tag{4.9}$$



(a) Linear interpolation and the Finite Difference Method. The samples ϕ_i (red) are interpolated with first-order linear functions between every two points. The second derivative at the blue point is determined with the FDM by calculating the difference between the gradients $\Delta\phi_i$ and $\Delta\phi_{i-1}$.



(b) Expanding the FDM to a data point $\phi_{i,j,k}$ in 3D space.

Expanding $\Delta\phi_i = \phi_{i+1} - \phi_i$, as displayed in Figure 4.4a, Equation 4.9 collapses to:

$$\frac{\mathrm{d}^2}{\mathrm{d}x^2}\phi \approx \frac{1}{\Delta x^2}\left(\phi_{i-1} + \phi_{i+1} - 2\phi_i\right) \tag{4.10}$$

This same process can be applied to the full left side of the Poisson equation in Equation 4.4, expanding the FDM to three dimensions, with in the x, y and z directions respectively N_x, N_y and N_z data points. The data points indices are now i, j and k: $\phi_{i,j,k}$ for $i \in [1,2,3,\dots,N_x]$, $j \in [1,2,3,\dots,N_y]$ and $k \in [1,2,3,\dots,N_z]$. As can also be seen on the left side of Equation 4.4, the Laplace operator is the sum of three second derivatives over the three dimensions x (i), y (j) and z (k). All points are still equally spaced with a distance of Δx, as is displayed in Figure 4.4b, but a single data point $\phi_{i,j,k}$ now has not 2, but 6 neighbors. Applying Equation 4.9 for a 3D space thus produces:

$$\nabla^2\phi \approx \frac{1}{\Delta x^2}\left(\phi_{i-1,j,k} + \phi_{i,j-1,k} + \phi_{i,j,k-1} + \phi_{i+1,j,k} + \phi_{i,j+1,k} + \phi_{i,j,k+1} - 6\phi_{i,j,k}\right) \tag{4.11}$$

In this way, the FDM can be used to define the left side of the Poisson equation, if the given function f (and therefore the sought function ϕ) consists of discrete points. Substituting Equation 4.11 into Equation 4.1 and shifting some variables gives:

$$6\phi_{i,j,k} - \phi_{i-1,j,k} - \phi_{i,j-1,k} - \phi_{i,j,k-1} - \phi_{i+1,j,k} - \phi_{i,j+1,k} - \phi_{i,j,k+1} = -\Delta x f_{i,j,k} \tag{4.12}$$

This equation is known as the discrete 3D Poisson equation for uniform spatialization (all points are distanced equally). When f and Δx are given, the values of ϕ can be found by solving all $N_x \times N_y \times N_z$ linear equations. Solving these equations is the basis for the Poisson solver, which is done by applying the Conjugate Gradient method, a method used to find the factor vector in a matrix-vector product.

Conjugate Gradient method

Before applying the Conjugate Gradient (CG) method, the discrete Poisson equation and all its factors must first be rewritten. The system of $N_x \times N_y \times N_z = N_e$ linear equations can be interpreted as a Matrix-Vector Product (MVP) by "flattening" the 3D samples space into column vectors, $\boldsymbol{\phi}$ and $\boldsymbol{b}$:

$$\boldsymbol{\phi} = \left[\phi_{1,1,1}, \phi_{2,1,1}, \cdots, \phi_{N_x,1,1}, \phi_{1,2,1}, \cdots, \phi_{N_x,N_y,1}, \phi_{1,1,2}, \cdots, \phi_{N_x,N_y,N_z}\right]^{\mathrm{T}} \tag{4.13}$$

$$\boldsymbol{b} = -\Delta x\left[f_{1,1,1}, f_{2,1,1}, \cdots, f_{N_x,1,1}, f_{1,2,1}, \cdots, f_{N_x,N_y,1}, f_{1,1,2}, \cdots, f_{N_x,N_y,N_z}\right]^{\mathrm{T}} \tag{4.14}$$

(Note that the 3D samples have been flattened such that x-neighbors are 1 index apart, y-neighbors are N_x indices apart, and z-neighbors are $N_x \times N_y$ indices apart. This is how MATLAB stores 3D arrays. Any other consistent ordering of the 3D elements is also possible.)

The MVP associated with the discrete Poisson equation then becomes:

$$A\boldsymbol{\phi} = \boldsymbol{b} \tag{4.15}$$

The matrix A holds the factors of the elements of ϕ on the left side of the discrete Poisson equation. It is called the Laplacian matrix, because it represents the Laplace operator ∇^2. The Laplacian matrix defines the connections in a 3D graph and thus describes which points in the list neighbor each other. Almost all of the entries in the Laplacian matrix are zero, because just a small amount of elements of the total N_e play a role in a single linear equation, as is evident from Equation 4.12. This means that A is considered a sparse matrix. Figure 4.5 shows how a 3D graph translates to a Laplacian matrix.

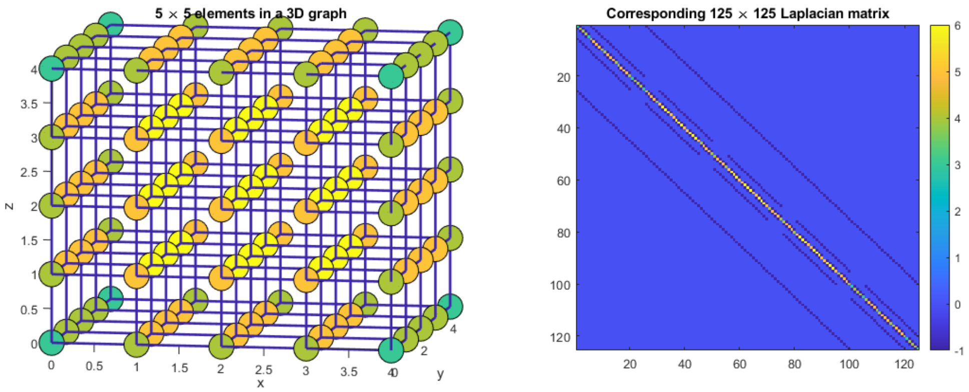


Figure 4.5: An example of a Laplacian matrix for a 3D space of 5 × 5 elements with $\Delta x = 1$. The left shows the elements as they are connected in a 3D graph, with their nodes colored according to their degrees. The right figure shows an image of the corresponding Laplacian matrix, with colors matching the 3D graph.

For non-uniform grids, the Laplacian matrix is not trivial and requires special methods to be constructed. An example is the more general form of FDM, called the Finite Element Method [2]. In any case however, the Laplacian matrix is a sparse matrix that describes how data points are geometrically connected in a 3D graph. For this work a uniform grid suffices, Equation 4.12 applies, and the values in A become trivial.

As can be seen in the right figure of Figure 4.5, the diagonal of A holds the degree of every element. The degree of a node in a graph represents how many other nodes are connected to it. For most of the elements, this number is therefore 6, but on edges of the model space, elements have less neighbors, and the degree can be as small as 3 (for the 8 corners of the 3D space). Off-diagonal, every column that is connected to the element index on its row, has the value -1. A consequence is that the matrix is by definition symmetrical ($A^{\mathrm{T}} = A$) and positive-definite ($\boldsymbol{x}^{\mathrm{T}} A\boldsymbol{x} > 0$ for *any* vector $\boldsymbol{x}$).

Now that the system of linear equations is defined as an MVP, it can be solved for $\boldsymbol{\phi}$. An obvious solution - especially within MATLAB, software tailored to matrix manipulation - would be to invert the Laplacian matrix A, and then multiply it with the right side of the equation, $\boldsymbol{b}$, so that $\boldsymbol{\phi} = A^{-1}\boldsymbol{b}$. Unfortunately, the matrix A is extremely large (for a 20 × 20 × 20 system, it has *64 million* entries) so simply inverting the matrix is, even for MATLAB, expensive and unnecessarily cumbersome.

Instead, iterative methods can be used to attempt to fit a vector $\boldsymbol{\phi}'$ such that $A\boldsymbol{\phi}' \approx \boldsymbol{b}$. Iterative methods accomplish this by repeated trial and error to try to minimize the residual vector $\boldsymbol{r} = \left|A\boldsymbol{\phi}' - \boldsymbol{b}\right|$ until its magnitude $|\boldsymbol{r}|$ is acceptably low. One of these methods is the Conjugate Gradient method.

The CG method finds a set of conjugate vectors, and adds them together to form the solution $\boldsymbol{\phi}$. Every iteration, a coefficient is calculated depending on the residual vector $\boldsymbol{r}$, which determines the next conjugate vector. The vectors are then added until the residual vector is sufficiently small (or a max number of iterations was reached). The specific algorithm associated with CG will be omitted here for brevity and can be found in full in the model source code [47].

Implementation and optimization

The Laplacian matrix A is a sparse matrix, which means that almost all of its entries are 0. Therefore it is unnecessary, and resource-intensive, to save the entirety of the matrix for the lattice grid. Instead, only the non-zero values of A are stored, accompanied by coordinates to show their row and column numbers. The matrix is divided into rows, because the rows correspond with the elements in the lattice grid and the indices of the lattice elements can serve as row coordinates. Every `latticeElement` object therefore has a field `A_row`, which consists of a two-column list of coordinates and their values. This reduces the size of a row of A to 14 entries at most (1 node plus 6 edges, with coordinates) compared to 8000 entries for e.g. a $20 \times 20 \times 20$ system. Multiplication of a sparse matrix with a vector is still possible and relatively easy to implement: every row of the solution vector is equal to the sum of the entries in the sparse matrix, multiplied by the entries in the row vector as indicated by the column coordinates in the sparse matrix.

The operator ∇ (Equation 4.2) is also stored per row in every `latticeElement`. This operator is defined as either the divergence operator, operating on a vector field to produce a scalar field, or the *gradient* operator, operating on a scalar field to produce a vector field. The latter is necessary to calculate the electric field vector for every lattice element using:

$$\nabla\phi = F \tag{4.16}$$

Since ∇ produces a vector from a scalar, it is discretely represented by three matrices, using the coefficients of FDM (Equation 4.8). This makes it necessary to sparsify it, similar to A: instead of one entry per coordinate however, `Nabla_row` holds three.

This concludes the description of the implementation of the Poisson solver.

4.4. Calculating the current: the charge transport model

The charge transport model is reproduced from the multiphonon-assisted Trap Assisted Tunneling (TAT) solver as presented in [59, 91]. This section will explain the specifics of both the physics and the software implementation of the TAT solver.

First, some of the basics of semiconductor physics concerning TAT are explained. Then, electron capture and emission times of traps and their relevance are explained. After that, the phonon particle is explained and its role in TAT. Then the density of states ($N(E)$) and the Fermi-Dirac occupation probability ($f(E)$) [71], both factors in the capture and emission times, are explained.

Calculating the TAT current also requires calculating a tunneling probability, for which the Wentzel-Kramer-Brillouin (WKB) approximation [71] is used. The implementation of the WKB calculator is also explained in a subsection. The subsection after that concerns the calculation of phonon capture and emission rates.

The last two subsections explain how the electron capture and emission times of traps are applied to calculate the current through the separate percolation paths. Then finally, the MATLAB implementation of the full TAT solver and its optimizations are described.

Basics of TAT semiconductor physics

The theoretical physics of TAT require some basic understanding of semiconductor physics, specifically on:

- Crystal lattices and vacancies;
- Conduction and valence bands;
- The effect of vacancies on the conduction band.

On a molecular scale, every material has a certain structure. The chemical formula that describes the material, e.g. HfO_2, only shows how much of each atom exists in the material, but its properties also rely on how these atoms are structured. The degree of structure in a solid material is defined by three crystalline phases as displayed in Figure 4.6.

- **Amorphous:** there is no or very little structure in the lattice, all the atoms are randomly connected through the material.
- **Poly-crystalline:** separate grains of structured material, misaligned with each other.
- **Crystalline:** the entire material is structured.

The crystal structure is, however, never perfect and there are always some lattice defects. These are not necessarily manufacturing defects, as they play a crucial role in the normal operation of RRAM, but they are parts of the lattice that deviate from the overall structure. Two types of lattice defects are relevant for RRAM operation: vacancies and interstitial ions, displayed in Figure 4.7.

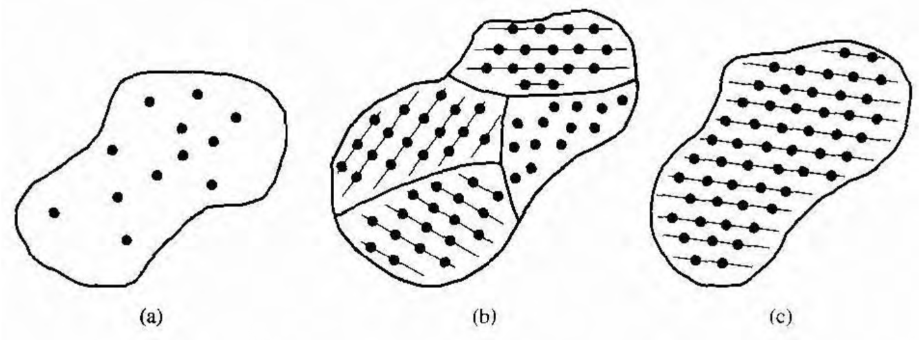


Figure 4.6: The three crystalline phases: (a) Amorphous, (b) Poly-crystalline and (c) Crystalline. Copied from [71].

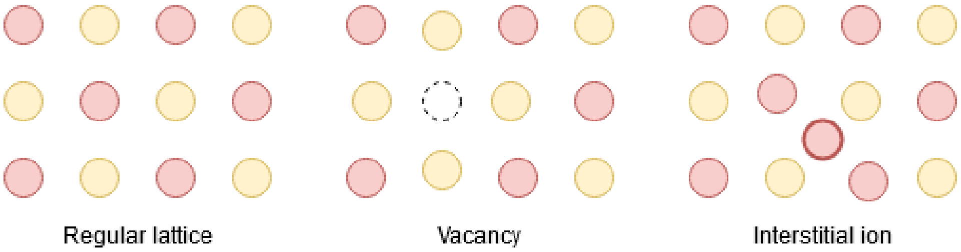


Figure 4.7: On the left: a regular lattice, without lattice defects. In the middle: a missing atom leaves a positively charged vacancy. On the right: a negatively charged interstitial ion between the structured atoms.

- **Vacancies** are left by atoms that are missing in the lattice structure. If the vacancy in question is left by an oxygen ion - which is the case for metal oxides in RRAM - the vacancy is *positively* charged.
- **Interstitial ions** are extra ions that exist between the lattice structure. If the interstitial ion is an oxygen ion, it is *negatively* charged.

Quantum physics dictates that an electron can only exist on certain discrete energy levels. This can be proven using the Pauli exclusion principle and Schrödinger's wave equation [71], but this proof will be omitted for the sake of conciseness. However, it only applies to atoms which are spaced far apart. If the atoms are closer together, the probability density functions of electrons, that exist around the atoms, start to interact. This splits the allowed energy levels into two. Adding another atom splits the energy levels again, and in the case of many closely spaced atoms in a crystal lattice, the split energy levels are so numerous and so close together that they form a quasi-continuous band. Figure 4.8 shows how an energy band is created by splitting a single energy level into a range by moving atoms closer together.

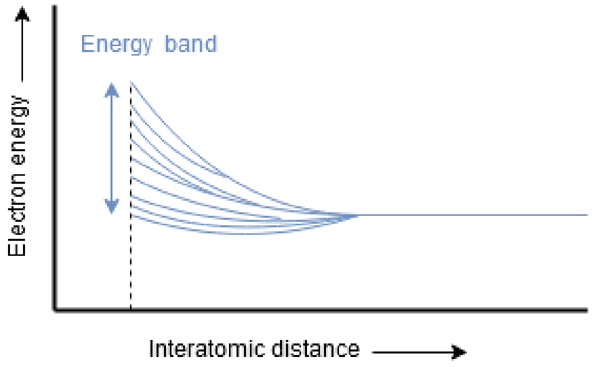


Figure 4.8: The creation of an energy band of allowed electron energies, by reducing the mutual distance between atoms in a lattice. Adapted from [71].

In this way, the allowed energy levels for electrons in a material span several bands. Two of these bands are of particular interest for semiconductors: the valence band and the conduction band. These are the bands that are closest to the Fermi level of the material, an energy level that is a unique property of a material. In the valence band, electrons do not have enough energy to conduct current, but in the conduction band, electrons can move freely and thus act as charge carriers for a current.

There can be a gap between the conduction band and the valence band, called the band gap. The length of this gap, called the band gap energy, is also a unique property of a material and defines its conductivity. Electrons can not exist in this gap - however, they can jump over it if they have enough energy. The three categories of conduction are distinguished and displayed in Figure 4.9.

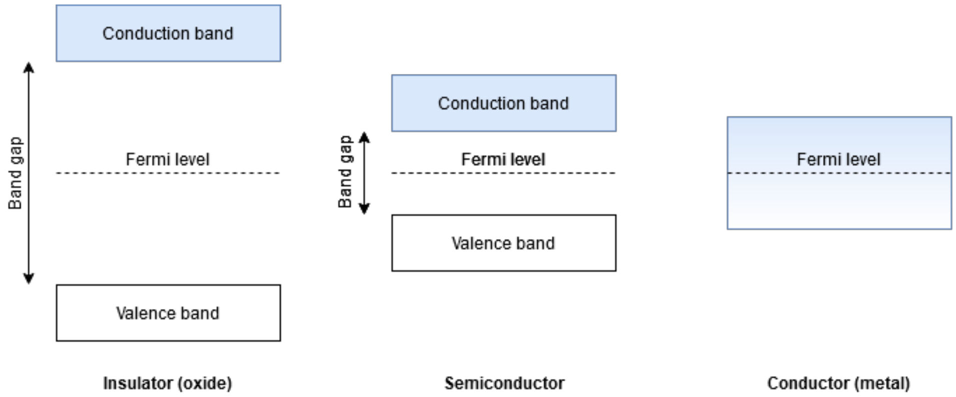


Figure 4.9: Three different kinds of materials: on the left, an insulator (e.g. a metal oxide). On the right, a conductor (e.g. a metal), in the middle, a semiconductor.

If the band gap is large, it takes more effort for an electron to jump to the conduction band. Materials with a large band gap are therefore *insulators* and are used when minimal conductivity is desired. If a material has no band gap, it takes hardly any effort for electrons to move around. This type of material is a *conductor* and is used when maximal conductivity is desired. The middle ground, a moderately large band gap, is the *semiconductor*. Semiconductors can be used to control the conductivity of a material by applying a voltage, thus creating a “transfer resistor” - or as we know it, a transistor.

A metal oxide normally acts as an insulator and therefore used in semiconductor technology to separate different conducting parts from each other. Figure 4.10 shows the energy bands of a metal-insulator-metal stack, like that of an RRAM device, when a voltage bias is applied to it.

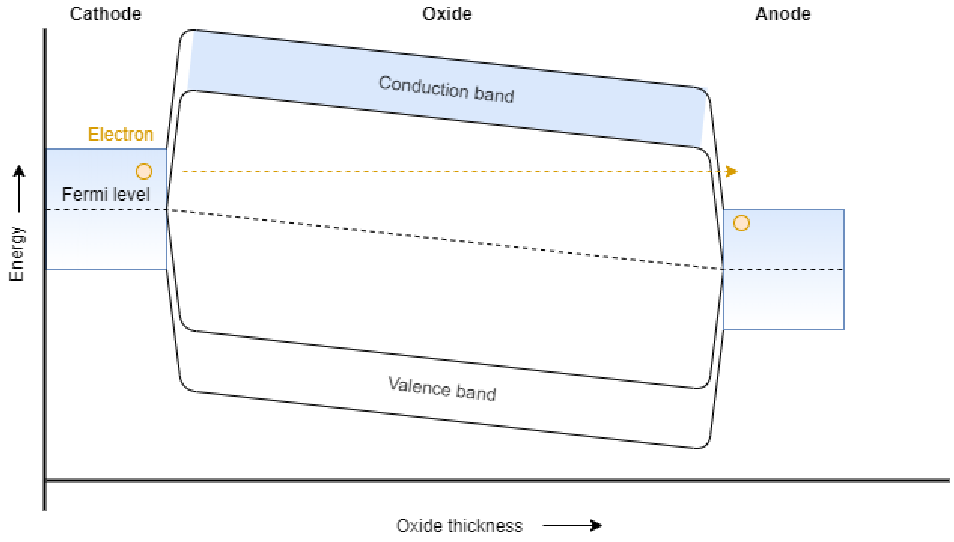


Figure 4.10: Energy bands of an MIM stack with an applied voltage bias. The electron (orange) in the left electrode has to tunnel through the entire thickness of the oxide to reach the right electrode.

To get from the negatively charged electrode (the *cathode*) to the positively charged electrode (the *anode*) the electron must go through a potential wall, as the conduction energy of the oxide is too high. This process is called *tunneling*. In a perfect oxide, an electron has to tunnel from the anode directly to the cathode, a process called *direct tunneling* or DT. For sufficiently thick oxides, as is the case in RRAM devices, this probability is negligibly small, resulting in a very large resistance, as is expected of an insulator. However, as stated before, no crystal structure is perfect and some lattice defects will exist in the oxide. These defects aid the electron in its tunneling process.

Vacancies, specifically oxygen vacancies, act like positively charged points in the oxide. The positive charge attracts the negatively charged electrons and thus an electron can be captured or "trapped" by the vacancy, as displayed in Figure 4.11. This trapping process is why oxygen vacancies in a metal oxide are also referred to as *traps* - hence, trap assisted tunneling.

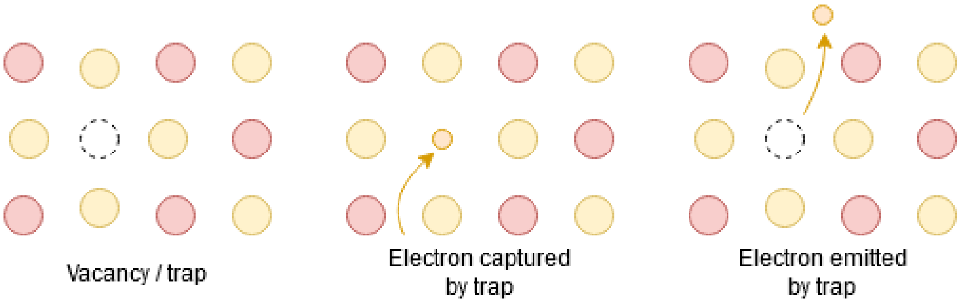


Figure 4.11: The process of electron trapping in an oxygen vacancy. The electron tunnels towards the trap and gets captured by it. The electron then emits from the trap to tunnel further.

In terms of energy bands, a trap is a local dip in the conduction energy of the oxide. This dip pulls the conduction band of the oxide downwards and thus creates an intermediate stepping stone for the electron to tunnel towards. This is visualized in Figure 4.12.

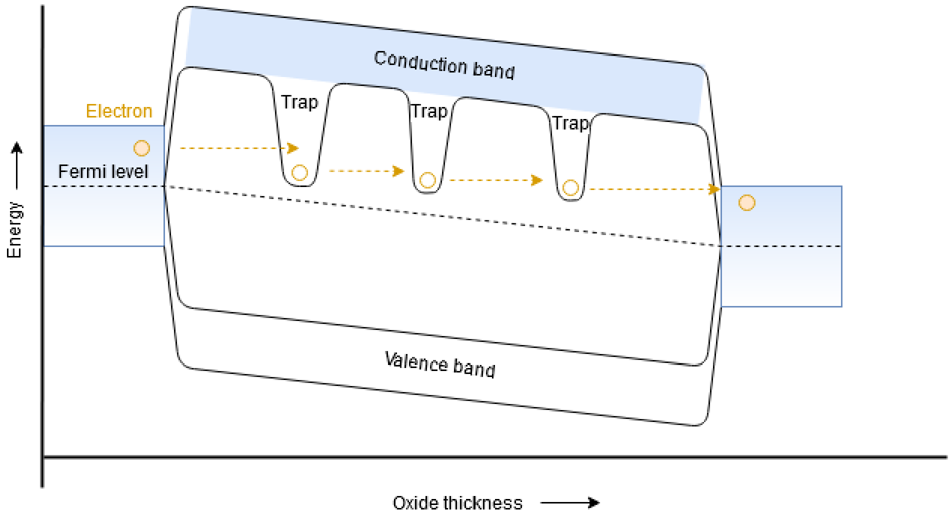


Figure 4.12: Energy bands of an MIM stack with an applied voltage bias and traps distributed over the oxide. The electron (orange) now only has to tunnel through thin parts of the oxide to reach the anode.

Because the electrons now have smaller barriers to tunnel through, the traps increase the flow rate of electrons through the oxide and therefore the current. It is the dominant conduction method through an RRAM device when it is in HRS.

In the reference model by Padovani et al., the charge transport model is based on the TAT solver presented by Vandelli et al. and Larcher ([59, 91]). This solver does not only account for the energy dip caused by positively charged traps, but also for the effect of temperature on the energy of the electrons tunneling across the traps. The temperature is related to discrete energy increases caused by multiple quasiparticles called *phonons*. Thus these works simulate *multiphonon-assisted* TAT.

Phonons and their role in TAT

Phonons are a quasiparticle, which means that they act like particles, but do not physically manifest as a particle. Their name comes from the Greek word $\phi\omega\nu\eta$ (phonè), which means "sound". Phonons represent a wave travelling through a material, like a sound wave or any acoustic vibration. For example, hitting a table surface will make vibrations travel through the table, a process that can be represented by phonon quasiparticles bouncing around inside the table.

These are *acoustic* phonons, but there are also *optical* phonons. This type of phonon does not represent acoustic vibrations but rather vibrations caused by heat. Phonons are described by the frequency of the wave they represent. The denomination "optical" comes from the fact that radiated heat is, in fact, light. When a heat source heats up a material, the energy from the source causes the surrounding atoms to vibrate in their lattice positions. A heat source is therefore actually emitting phonons.

Phonons carry some energy with them, represented by the vibrating of the atoms that accompanies phonons. The vibrating frequency of a phonon translates directly to a discrete energy level (for optical phonons) through Planck's constant: $E_{ph} = \hbar\omega_{ph}$. Traps are able to absorb this energy, giving an electron that is trapped inside the trap a "boost" to tunnel through the potential wall to the next the trap or electrode. Figure 4.13 shows the effect this has on the energy bands of the TAT process.

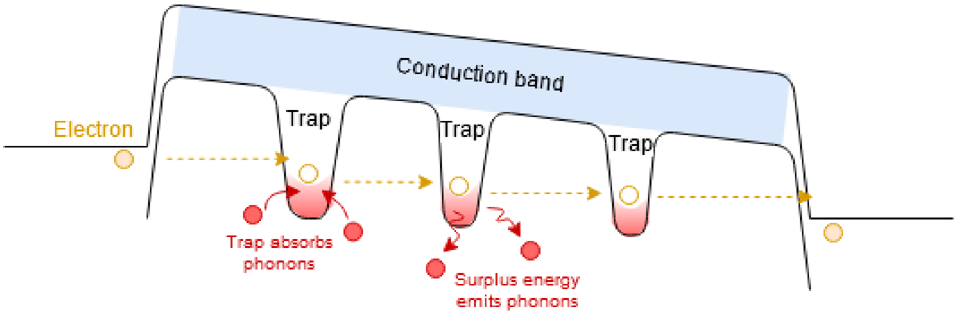


Figure 4.13: Energy bands of the process of multiphonon-assisted TAT: electrons (orange), captured by a trap, receive a boost in their energy level attributed to the absorption of phonons (red) by the trap. The energy surplus of electrons captured by a trap is turned into phonons, which in turn increase the temperature around a trap (red gradient).

When an electron is captured by a trap, it was emitted before from a higher energy level. This means that is has a surplus energy that is lost by the electron "falling" into the trap. The surplus energy is dissipated as heat by the emission of phonons into the surrounding lattice. Traps are therefore heat sources; and the temperature increase around the trap is determined by Fourier's heat equation (Equation 4.6) which is solved by the Poisson solver (Section 4.3).

Capture and emission times

The flow rate of electrons in the TAT process is determined by the time it takes for a trap to capture and emit an electron. The shorter these times, the more electrons can pass through a trap over time, which is directly translated to an increase in current. The capture and emission times relate to the current as such: an electron flow rate R_i can be defined for every trap i that exists in the oxide, by looking at how many electrons the trap can both capture and emit at a time. This is represented by the capture and emission rates of electrons for a trap i, $(R_{c,i}, R_{e,i})$ which in turn rely on the capture and emission times $(\tau_{c,i}, \tau_{e,i})$, and the probability of an electron occupying a trap $(f_{t,i})$ [59]:

$$R_{c,i} = \tau_{c,i}^{-1} \cdot (1 - f_{t,i}) \tag{4.17}$$

$$R_{e,i} = \tau_{e,i}^{-1} \cdot f_{t,i} \tag{4.18}$$

Equation 4.17 implies that the capture rate relies on the time it takes to *capture* an electron, and the probability that the trap is *unoccupied* $(1 - f_{t,i})$. Similarly, Equation 4.18 implies that the emission rate relies on the time it takes to *emit* an electron, and the probability that the trap is *occupied* $(f_{t,i})$.

The TAT solver [59] assumes that no charge buildup can occur in the traps. This means that the in-flow of electrons (capture rate) must be equal to the out-flow (emission rate). Since this means that $R_{c,i}$ and $R_{e,i}$ are equal, the electron flow rate through a trap i can be defined as $R_i = R_{c,i} = R_{e,i}$. Setting Equation 4.17 equal to

Equation 4.18 then defines the electron occupation probability of a trap by the capture and emission times as:

$$f_{t,i} = \tau_{e,i} \cdot \left(\tau_{c,i} + \tau_{e,i}\right)^{-1} \tag{4.19}$$

Inputting this equation in either one of the capture and emission rate equations, the electron flow rate for a trap i, R_i, becomes defined by only the capture and emission times [59]:

$$R_i = \frac{1}{\tau_{c,i} + \tau_{e,i}} \tag{4.20}$$

Equation 4.20 defines an electron flow rate for every trap in the lattice, which can then be used to find how electrons percolate the the oxide through multiple percolation paths, as will be explained later in this section. First, the calculation of the capture and emission times is explained.

The assistance of phonons is taken into account by summing over the (inverted) electron capture and emission times for all discretized energy levels caused by phonons [59, 91]. It is assumed the phonons involved are optical phonons of a single frequency, ω_0. This means that a single phonon carries an energy of $E = \hbar\omega_0$. Given m phonons are involved in the capture an electron, and n phonons are involved in the emission of an electron, the capture and emission times are defined by [59, 91]:

$$\tau_{c,i}^{-1} = \sum_m \left(\tau_{c,i,m}\right)^{-1} \tag{4.21}$$

$$\tau_{e,i}^{-1} = \sum_n \left(\tau_{e,i,n}\right)^{-1} \tag{4.22}$$

Where $\tau_{c,i,m}$ is the time required for the capture of an electron in trap i, associated with the release of m phonons (energy $m\hbar\omega_0$) into the surrounding lattice, and $\tau_{e,i,n}$ is the time required for the emission of an electron from trap i, associated with the absorption of n phonons (energy $n\hbar\omega_0$) from the surrounding lattice. $\tau_{c,i,m}$ and $\tau_{e,i,n}$ are defined as:

$$\left(\tau_{c,i,m}\right)^{-1} = N_{i-1}\left(E_{i,m}\right) f_{i-1}\left(E_{i,m}\right) Ca_{i,m} P_T\left(E_{i-1}, E_{i,m}\right) \tag{4.23}$$

$$\left(\tau_{e,i,m}\right)^{-1} = N_{i+1}\left(E_{i,n}\right)\left(1 - f_{i+1}\left(E_{i,n}\right)\right) Em_{i,n} P_T\left(E_{i,n}, E_{i+1}\right) \tag{4.24}$$

The factors of these equations represent:

- $\boldsymbol{E_i}$: the energy level of an electron at trap i, determined by the local electric potential ϕ_i, the electron charge $-q$ and the trap energy $E_{t,i}$, as: $E_i = -q\phi_i - E_{t,i}$
- $\boldsymbol{E_{i,k}}$: the energy level of trap i, with the addition of k phonons: $E_{i,k} = E_i + k\hbar\omega_0$.
- $\boldsymbol{N_i(E)}$: the density of states, in trap i, at energy level E.
- $\boldsymbol{f_i(E)}$: the Fermi-Dirac occupation probability, in trap i, at energy level E.
- $\boldsymbol{Ca_{i,m}}$ and $\boldsymbol{Em_{i,n}}$: the capture and emission rates that account for carrier-phonon interaction.
- $\boldsymbol{P_T\left(E_{i,n}, E_{i+1}\right)}$: the tunneling probability, for an electron at energy $E_{i,n}$ towards a trap with energy E_{i+1}.

The following three sections will explain the calculation of the density of states ($N_i(E)$) and the Fermi-Dirac occupation probability ($f_i(E)$), the calculation of the tunneling probability ($P_T\left(E_{i,n}, E_{i+1}\right)$) through the WKB approximation and the formulae for the phonon capture ($Ca_{i,m}$) and emission ($Em_{i,n}$) rates.

Density of states and Fermi-Dirac occupation probability

From the first subsection of this section, it became clear that energy bands are not completely continuous, but rather consist of a very large amount of discrete energy states. The density of these states depends on the difference from the conduction energy and reflects how many spaces there are for an electron to possibly exist [71]. In [59], the density of states for an energy E is given as:

$$N_i(E) = \frac{1}{2\pi^2}\left(\frac{2m_e}{\hbar^2}\right)^{3/2} \sqrt{E - E_i} \cdot u(E - E_i) \tag{4.25}$$

Where m_e is the effective electron mass (a constant dependent on material properties), $\hbar$ is the modified Planck constant, E_i is the conduction energy at trap i and u is the unit step function:

$$u(E - E_i) = \begin{cases} 0 & (E < E_i) \\ 1 & (E \geq E_i) \end{cases} \tag{4.26}$$

which reflects that in the forbidden region under the conduction band there are no allowed states.

While the density of states describes the amount of spaces that electrons can occupy, the Fermi-Dirac occupation probability describes the probability that these spaces are occupied. It depends on the difference from the Fermi level and is defined as [71]:

$$f_i(E) = 1/\left(1 + \exp\left[\frac{E - E_{F,i}}{kT_i}\right]\right) \tag{4.27}$$

Where $E_{F,i}$ is the Fermi level at trap i, k is the Boltzmann constant and T_i is the temperature at trap i.

The Fermi level of HfO_2 used in the TAT solver is not given in any of the reference works [58, 59, 76, 89, 91, 92] so it had to be deducted for this work, from the band gap energy E_g of HfO_2. The band gap energy has been measured [14] and found to be $E_g = 5.8\,\text{eV}$, but an assumption still has to be made for where the Fermi level is within the band gap.

It is known that materials with a surplus of positive charge carriers have a Fermi level close to the conduction band [71]. If the Fermi level is therefore assumed to be exactly in the middle of the band gap, with respect to the conduction band *without* energy dips caused by the traps, then locally on trap sites the Fermi level will be much closer to the conduction band, as was already visible in Figure 4.10 and Figure 4.12. This indicates that the oxide locally has a surplus of positive charge, which further motivates the assumption. Therefore the Fermi level is assumed to be constant relative to the *trap-less* conduction band of the oxide.

With this assumption in mind, the Fermi level at trap i becomes easy to calculate, knowing the local electric potential ϕ_i and the width of the band gap E_g:

$$E_{F,i} = -q\phi_i - E_g/2 \tag{4.28}$$

The product of the density of states and the Fermi-Dirac occupation probability reflects the amount of electrons that are present on a certain energy level, or the *electron supply*. $N_i(E)$, $f_i(E)$ and their product are plotted in Figure 4.14, using the equations presented in this subsection and the parameters used in the TAT solver, all of which can be found in Appendix A. The figure shows that, near the conduction energy, the supply of electrons is largest. Above that it exponentially decreases to 0. Under the conduction band, there are no electrons, because levels below the conduction energy exists inside the forbidden band gap.

Phonon capture and emission rates

Charge carriers in the TAT process are assisted by multiple phonons at once, which can be captured and emitted from the trap at respectively a capture and emission rate. The factors $Ca_{i,m}$ and $Em_{i,n}$ are calculated in [59, 91] by the following given formulas:

$$Ca_{i,m} = c_0 L(m) \tag{4.29}$$

$$Em_{i,n} = c_0 L(n) \exp\left[\frac{-n\hbar\omega_0}{kT}\right] \tag{4.30}$$

Where c_0 is a variable dependent on the electric field magnitude F and the trap capture radius r_t [59]:

$$c_0 = \frac{4\pi^2 r_t^2}{E_g} \frac{q^2 F^2 \hbar}{2m_e} \tag{4.31}$$

Here q is the elementary charge, m_e is the effective electron mass and E_g is the oxide band gap energy. $L(m)$ - in Equation 4.29 and Equation 4.30 - is the *multiphonon transition probability* for m phonons. It is calculated by summing over the overlap of multiple phonon vibrational states between the trap with and without a trapped charge [91]. A visualization of this sum is displayed in Figure 4.15. This produces the formula:

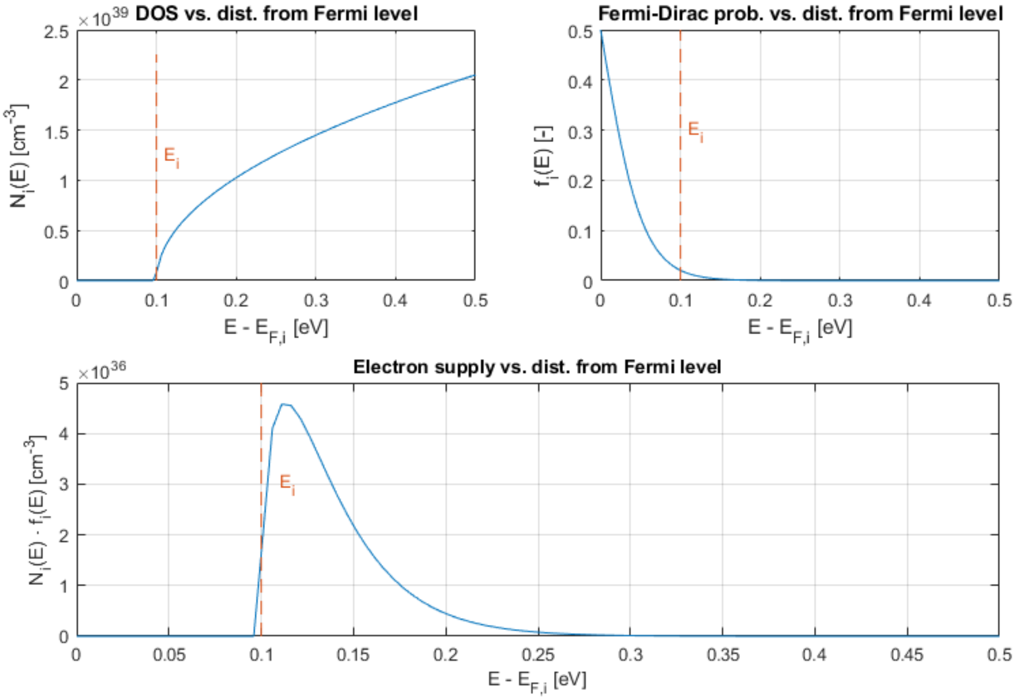


Figure 4.14: Plots of the density of states (DOS) (top left), the Fermi-Dirac occupation probability (top right) and the electron supply function (bottom). The energy on the x-axes is the energy relative to the Fermi level. The conduction energy at the trap site (E_i) is displayed in orange.

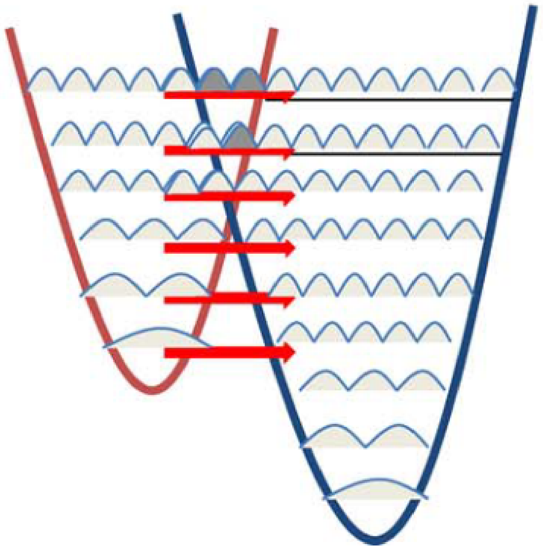


Figure 4.15: Summing the overlap of vibrational states between a trap without charge (left) and a trap with a captured charge (right). Copied from [91].

$$L(m) = \left(\frac{f_B + 1}{f_B}\right)^{m/2} \exp - S\left(2 f_B + 1\right) I_m \left(2S\sqrt{f_B\left(f_B + 1\right)}\right) \tag{4.32}$$

where f_B is the Bose function:

$$f_B = \frac{1}{\exp\left[\frac{\hbar\omega_0}{kT} - 1\right]} \tag{4.33}$$

and I_m is the modified Bessel function of the order m. This function is already implemented in MATLAB by the function `besseli`. S is the Huang-Rhys factor, which represents the number of phonons required for the rearrangement of the lattice around a trap to accommodate the trapped charge [91]. The value of S determines the dependability on temperature of the TAT current. The values for these physical properties are provided in a complete list in Appendix A.

WKB tunneling probability

Finally, the calculation of the tunneling probability $P_T\left(E_{i,n}, E_{i+1}\right)$ by using the Wentzel-Kramers-Brillouin approximation requires a dedicated implementation for this model. A separate function within the TAT module was constructed, that automatically and analytically calculates the tunneling probability P_T between two `latticeElement`s, using the elements' properties as parameters. To understand the implementation of this function, a deeper understanding of quantum physics is first required, specifically on probability density functions and Schrödinger's time-independent wave equation [71]:

$$\frac{\partial^2\psi(x)}{\partial x^2} + \frac{2m_e}{\hbar^2}\left[E - U(x)\right]\psi(x) = 0. \tag{4.34}$$

Schrödinger's time-independent wave equation is a second-order differential equation that is used to find the wave function $\psi(x)$, which describes the time-independent behaviour of a particle in a space. The wave function is mathematically complex and is based on some physical constants (m_e, $\hbar$), the energy of the particle involved (E) and the local potential energy ($U(x)$). The local potential energy, in this case, represents the conduction energy in the energy bands diagram. The magnitude squared of the wave function produces the *probability density function*: $\left|\psi(x)\right|^2$. This function gives the probability that a particle exists on certain point x in space.

Based on the difference between E and $U(x)$, two regions can be distinguished: the classical region and the quantum tunneling region. In the classical region, where $E > U(x)$, the wave function looks like a sinusoidal function that bounces off potential walls, as is expected according to classical theory. However, in the quantum tunneling region, where $E < U(x)$, the wave function describes the particle existing *inside* the potential wall. This is impossible within the limitations of classical physics, but quantum physics has shown that tunneling is definitely possible and that it is a major contributor to charge transport models, such as TAT.

Figure 4.16 shows the solution to Equation 4.34 for a particle of $E = 1\,\text{eV}$, incident on a potential wall of $U_{wall} = 4\,\text{eV}$ of $t_{wall} = 3\text{Å}$ thick, starting at position $x = 0$. The probability density function inside of the potential wall takes the shape of an exponential function. To the right of the wall, the probability density is still > 0, implying that it is indeed possible that the particle tunnels through the wall.

The WKB approximation is a method of predicting the shape of $\psi(x)$ and approximating it by neglecting relatively insignificant factors. This saves effort calculating the tunneling probability, which is a necessity: because many traps may exist in the oxide, the calculation of the flow rate between two traps may be called millions of times per simulated time step. The WKB approximation method assumes that the solution of Schrödinger's wave equation is a complex exponential function shaped like:

$$\psi(x) = \exp\left[\Phi(x)\right] \tag{4.35}$$

where the first derivative of the exponent, $\Phi'(x)$, takes the shape of

$$\Phi'(x) = A(x) + jB(x) \tag{4.36}$$

The approximation concerns the values of $A(x)$ and $B(x)$ in $\Phi'(x)$. $A(x)$ is the real part of $\Phi'(x)$ and therefore translates to the *change in magnitude* of the wave function: $|\psi(x)|'$. $B(x)$ is the imaginary part of $\Phi'(x)$ and therefore translates to the *change in phase* of the wave function: $\angle\psi(x)'$.

The magnitude of the wave function is the most interesting, because it is linked directly to the probability density function, which is used for the calculation of tunneling probability P_T. Because only tunneling is considered, $E < U(x)$ is assumed. This leads to an approximation possibility: as can be seen in Figure 4.16,

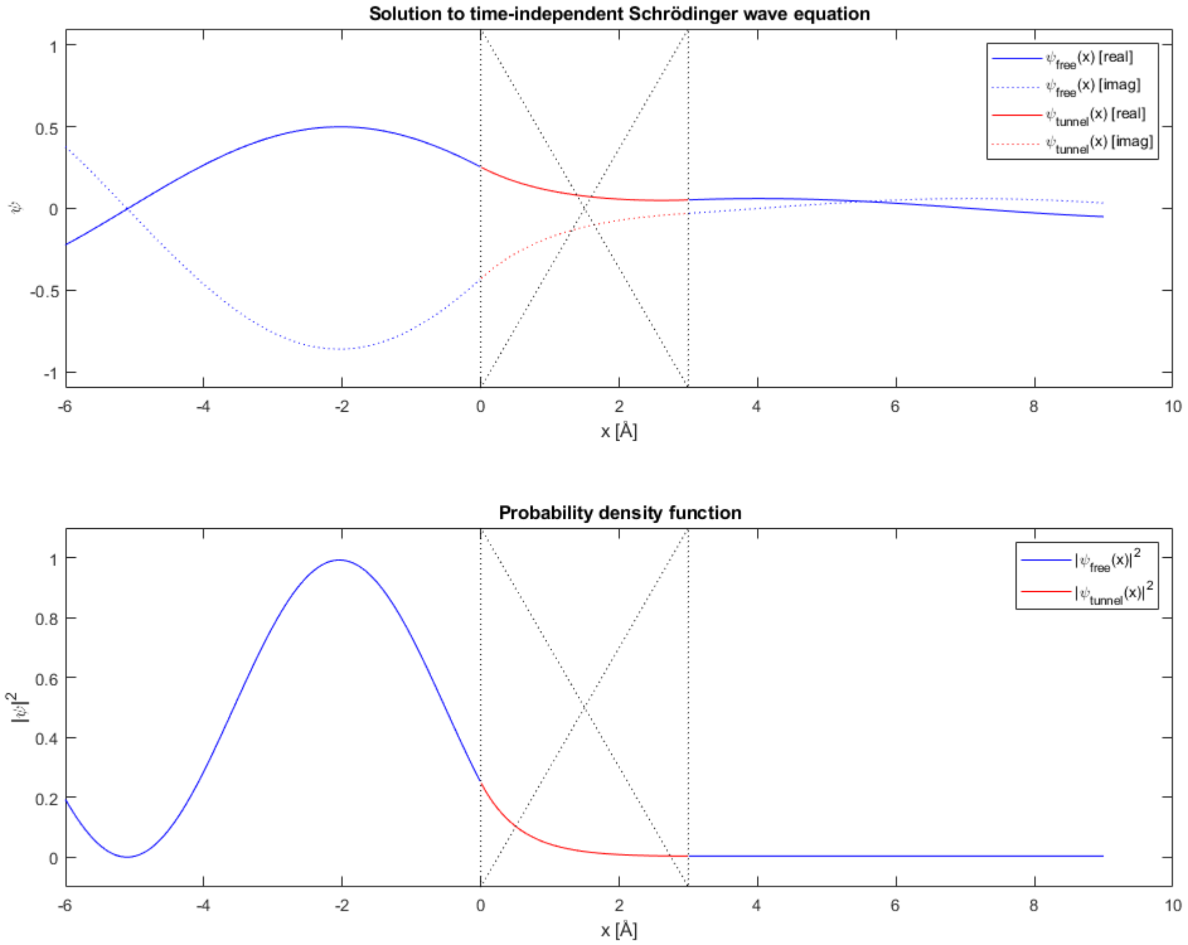


Figure 4.16: The solution to Schrödinger's time independent wave equation for a particle incident on a 3Å potential wall, solved with help of MATLAB's Symbolic toolbox. The crossed box indicates the potential wall. The blue parts represent the classical region ($\psi_{free}(x)$) and the red parts the quantum tunneling region $\psi_{tunnel}(x)$. The bottom plot shows the probability density function, $|\psi(x)|^2$.

in the quantum tunneling region, the phase of the wave function changes very little. This can also be proved mathematically [71], but is omitted here for the sake of conciseness.

If the phase change is insignificant, as is the case in the quantum tunneling region, it can be assumed that $B(x) \approx 0$. This sets $\Phi'(x) = A(x)$ and the probability density function becomes:

$$|\psi(x)|^2 = \exp\left[2\int_{x_0}^{x} A(x')\mathrm{d}x'\right] \tag{4.37}$$

where x_0 is the turning point between the classical and non-classical regions. Using the 0th-order WKB approximation for quantum tunneling [71]:

$$A(x) = A_0(x) = -\sqrt{\frac{2m_e}{\hbar^2}(U(x) - E)} \tag{4.38}$$

The tunneling probability is then retrieved by calculating the quotient of the probability density function at the end and at the start of the potential barrier [71]. A particle tunneling from trap i to trap j, meeting a potential wall between x_i and x_j, therefore has a tunneling probability of:

$$P_T = \frac{\left|\psi(x_j)\right|^2}{\left|\psi(x_i)\right|^2} = \frac{\exp\left[2\int_{x_0}^{x_j} A(x')\mathrm{d}x'\right]}{\exp\left[2\int_{x_0}^{x_i} A(x')\mathrm{d}x'\right]} = \exp\left[-2\frac{\sqrt{2m_e}}{\hbar}\int_{x_i}^{x_j}\sqrt{U(x')-E}\mathrm{d}x'\right] \tag{4.39}$$

Now, if $U(x)$ is given as an integrable function, it can be directly substituted in Equation 4.39 to analytically derive the tunneling probability. This is faster than an iterative function, because after analytical derivation, it takes just a single computation to calculate P_T. Fortunately, the potential at the start (U_i) and the end (U_j) of the potential barrier is already known from the solution of the potential field provided by the Poisson solver. It is therefore possible to define the potential $U(x)$ between two traps distanced by $D_{i,j}$ as a simple first-order linear function, as displayed on the left of Figure 4.17:

$$U(x) = U_i - F_{i,j}x \tag{4.40}$$

where $F_{i,j} = \left(U_i - U_j\right)/D_{i,j}$.

However, a single line does not properly reflect the valley-shaped potential dip in the conduction energy, as previously displayed in Figure 4.12. Larcher [59] proposes to shape the potential barrier instead as a three-piece linear function. The traps have a *capture radius* ($r_{t,i}$) that reduces the potential barrier around the two traps. The function $U(x)$ now consists of three regions:

$$U(x) = \begin{cases} U_1(x) = \frac{U_{r_{t,i}}}{r_{t,i}}x & \left(x < r_{t,i}\right) \\ U_2(x) = U_i - F_{i,j}x & \left(x \geq r_{t,i}\right) \cup \left(x < D_{i,j} - r_{t,j}\right) \\ U_3(x) = U_j - E_{t,j} + \frac{E_{t,j}+F_{i,j}r_{t,j}}{r_{t,j}}\left(D_{i,j}-x\right) & \left(x \geq D_{i,j} - r_{t,j}\right) \end{cases} \tag{4.41}$$

where $U_{r_{t,i}} = U(r_{t,i})$, $U_{r_{t,j}} = U(r_{t,j})$ and $E_{t,j}$ is the trap energy of trap j. The function is displayed on the right side of Figure 4.17. Note: the zero potential point ($U(x) = 0$) was chosen to be on the *bottom* of trap i ($U_i = E_{t,i}$) (see also the black dashed line in Figure 4.17). The zero could be placed anywhere, as Equation 4.39 depends only on the *difference* between the particle energy and the potential barrier ($U(x) - E$). This choice was made so that E can be defined as the difference between the particle energy and the bottom of trap i.

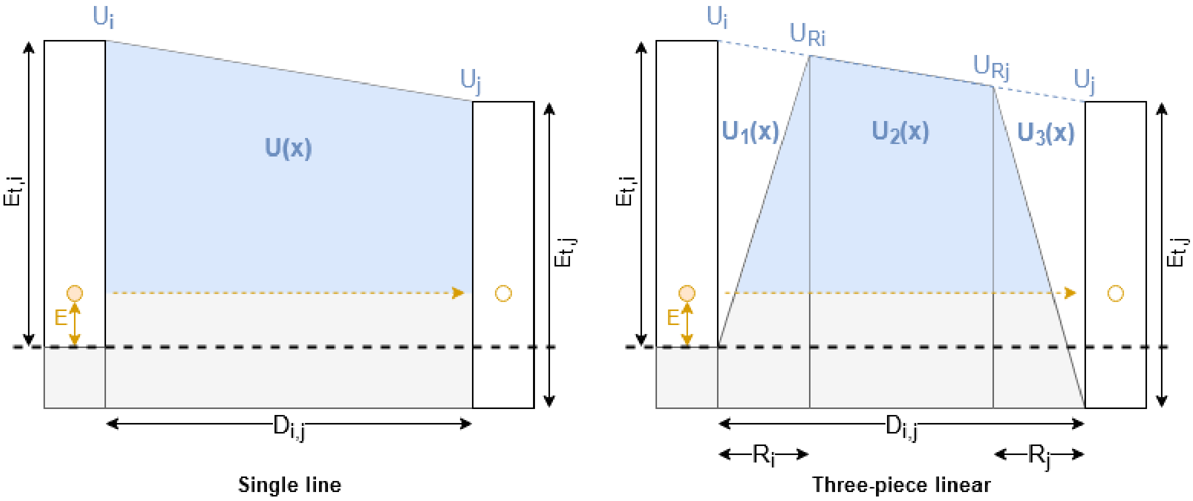
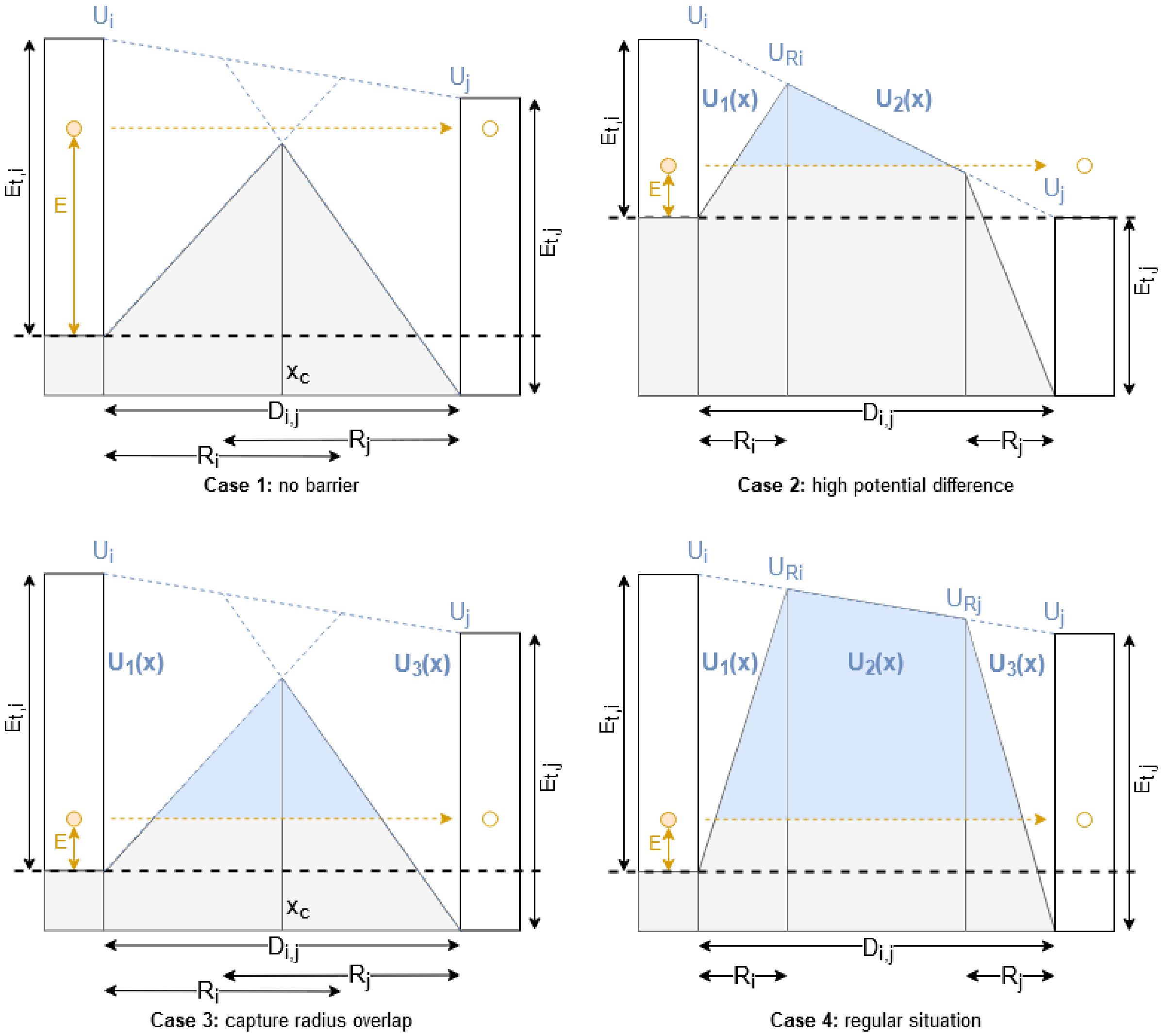


Figure 4.17: On the left: the potential function $U(x)$ between trap i and trap j, using a single linear function over the mutual distance $D_{i,j}$. On the right: the potential function $U(x)$ in three linear functions, their regions determined by the trap capture radii, $r_{t,i}$ and $r_{t,j}$.

An effect of the trap capture radii is that once the mutual distance between traps decreases, the capture radii start to overlap [59, 91]. This greatly reduces the potential barrier and thus increases the tunneling probability. At a certain critical mutual distance, the potential barrier vanishes completely, and the tunneling probability $P_T = 1$ [76]. This means that now a *defect subband* is formed: electrons no longer have to tunnel to cross the oxide and drift current overtakes TAT as the dominant conduction mechanism. In this way, the reduction of the potential barrier is able to smoothly model the transition from TAT to drift current.

The MATLAB function that calculates the WKB tunneling probability, `getTunnelingProbability`, determines P_T by picking a precalculated formula, analytically derived from Equation 4.39. Since only the potential barrier ($U(x) > E$) is considered, the integration limits must be adapted according to the level of the particle energy E and the shape of $U(x)$. The following list lists the four possible cases and their corresponding precalculated formulas. The cases are visualized in Figure 4.18.



Case 1: no barrier

Case 2: high potential difference

Case 3: capture radius overlap

Case 4: regular situation

Figure 4.18: The four possible cases of integration for the WKB tunneling probability. The blue shaded part of the potential function, where $U(x) > E$, represents the result of the integration, which is used in the exponent of Equation 4.39.

- **Case 1:** the particle energy E is above the potential barrier over the entire considered distance $D_{i,j}$. Regardless of the shape of $U(x)$,

$$P_T = 1 \tag{4.42}$$

- **Case 2:** the potential difference between the traps is so large, that the right-most part of the potential function is completely below the particle energy E. $U(x)$ is now a two-piece linear function of $U_1(x)$ and $U_2(x)$:

$$P_T = \exp\left[\frac{4}{3}\frac{\sqrt{2m_e}}{\hbar}\left(-\frac{r_{t,i}}{U_{r_{t,i}}}(U_{r_{t,i}} - E)^{3/2} - \frac{1}{F_{i,j}}(U_{r_{t,i}} - E)^{3/2}\right)\right] \tag{4.43}$$

- **Case 3:** the mutual distance between the traps is so short, that the trap capture radii overlap. $U(x)$ is now a two-piece linear function of $U_1(x)$ and $U_3(x)$. The crossing point x_c, where $U_1(x_c) = U_3(x_c)$, defines the boundary between the two regions:

$$x_c = \frac{U_j - E_{t,j} + \left(\frac{E_{t,j}}{r_{t,j}} + F_{i,j}\right) D_{i,j}}{\frac{U_i}{r_{t,i}} + \frac{E_{t,j}}{r_{t,j}}} \tag{4.44}$$

The tunneling probability is then:

$$P_T = \exp\left[\frac{4}{3}\frac{\sqrt{2m_e}}{\hbar}\left(-\frac{r_{t,i}}{U_{r_{t,i}}}\left(\frac{U_{r_{t,i}}}{r_{t,i}}x_c - E\right)^{3/2} - \frac{1}{\frac{E_{t,j}}{r_{t,j}} + F_{i,j}}\left(U_j - E_{t,j} + \left(\frac{E_{t,j}}{r_{t,j}} + F_{i,j}\right)\cdot\left(D_{i,j} - x_c\right) - E\right)^{3/2}\right)\right] \tag{4.45}$$

- **Case 4:** if none of the above three cases apply, then the potential function is equal to Equation 4.41. Integrating over the three regions of the potential barrier then gives:

$$P_T = \exp\left[\frac{4}{3}\frac{\sqrt{2m_e}}{\hbar}\left(-\frac{r_{t,i}}{U_{r_{t,i}}}(U_{r_{t,i}} - E)^{3/2} + \frac{1}{F_{i,j}}\left((U_{r_{t,j}} - E)^{3/2} - (U_{r_{t,i}} - E)^{3/2}\right) - \frac{1}{\frac{E_{t,j}}{r_{t,j}} + F_{i,j}}(U_{r_{t,j}} - E)^{3/2}\right)\right] \tag{4.46}$$

Note that these options have multiple edge cases where the formulas that were listed have to be modified slightly, e.g. when $F_{i,j} = 0$, or when one side is an electrode and has no capture radius ($r_{t,i/j} = 0$). For conciseness, these edge cases have been left out of this section. The full implementation is documented in the MATLAB code on the TU Delft repository and does take into account *all* possible edge cases.

Percolation paths, power and charge

The methods for calculating the electron capture and emission times $\tau_{c,i}$ and $\tau_{e,i}$, together with all the factors in Equation 4.23 and Equation 4.24, have now been explained. Now the electron flow rate R_i can be calculated for every trap i in a percolation path. Note, however, that Equations 4.23 and 4.24 concern not one, but *three* traps: the current trap in the path (trap i), the trap before this trap (trap $i-1$) and the trap after this trap ($i+1$). This implies that the electron flow rate through a trap depends both on the trap the electrons are flowing from, and the trap the electrons are flowing towards.

The process of finding percolation paths is based on the assumption that charge carriers will always choose the path of least resistance [59]. Electrons start at the cathode (the negatively charged electrode) and jump from trap to trap, across the oxide, to the anode (the positively charged electrode). The path it takes in between is determined as such:

1. The electron starts at the cathode, and picks from all the traps the pair of traps with the highest rate from the cathode (cathode → trap → trap). If a single trap path has the highest rate (cathode → trap → anode) this path is picked instead. Any traps that were picked are marked as being in a path.

2. The rate from the current trap to the anode is calculated, together with the rates from the current trap to all other unmarked traps.

3. If the rate to the anode is the largest, or if no unmarked traps remain, the path of the electron ends as it has reached the anode and the loop exits.

4. If the rate to another trap is largest, this trap is selected as the next one in the path and marked. The loop returns to Step 2 and continues.

Every time the loop exits, the minimum rate in the calculated path is selected. This “bottleneck” rate R_{min} determines the flow rate for the entire path, because electrons in the path can not flow faster than its weakest link. The current through this percolation path is then calculated as such:

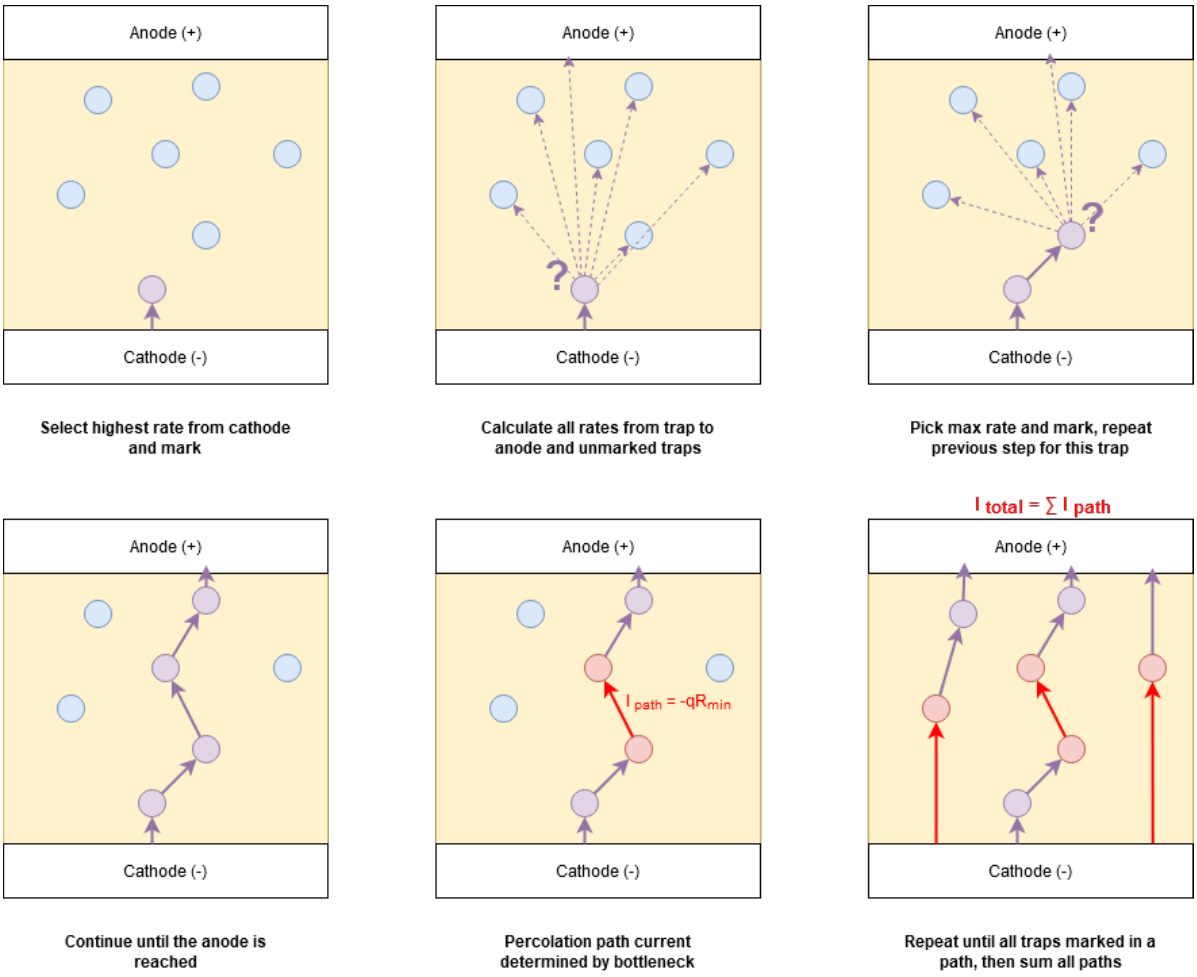
$$I_{path} = -qR_{min} \tag{4.47}$$

Note that the charge of an electron is negative, resulting in a current direction from the anode to the cathode, as is expected.

The path finding process is repeated, until no more unmarked traps are in the oxide. The total current through the device is then calculated by summing all individual path contributions:

$$I_{total} = \sum I_{path} \tag{4.48}$$

The full process of finding percolation paths is illustrated in Figure 4.19.



Select highest rate from cathode and mark

Calculate all rates from trap to anode and unmarked traps

Pick max rate and mark, repeat previous step for this trap

Continue until the anode is reached

Percolation path current determined by bottleneck

Repeat until all traps marked in a path, then sum all paths

Figure 4.19: The full process of exhaustively finding percolation paths to include contributions of all traps (blue) in the oxide (yellow). Marked traps are purple, bottleneck traps are red.

The electron flow rate of the full path is also used to calculate the power dissipation in each of the traps. If an electron jumps between a trap i and a trap j, it loses an amount of energy equal to $\Delta E = E_i - E_j$. This lost energy is translated into dissipated power by using the flow rate of the path it is in:

$$P_i = \Delta E \cdot R_{min} \tag{4.49}$$

Meanwhile, the capture and emission times define the probability of an electron being trapped in the trap $(f_{t,i})$, according to Equation 4.19. Since the capture and emission times of every trap are already calculated, the electron occupation probability can be easily calculated to determine the net charge of the trap:

$$Q_i = q \cdot (1 - f_{t,i}) \tag{4.50}$$

Note that the change in trap charge due to trapped electrons is not mentioned clearly in the reference model [76], nor does it clarify the charge of a trap without an electron occupying it. The microscopic model by Duncan et al. ([27]) indicates however that, while different trap charges can exist, traps charged by +1e have the most tendency to cluster together. Since this model concerns the formation of a filament caused by trap clustering, the unoccupied trap charge was therefore set as +1e.

It was found that this strategy of finding percolation paths can sometimes cause a sudden decrease in current, even though the voltage is increasing. On further investigation it was discovered that the strategy has a tendency of picking local maxima: a locally higher electron flow rate can cause the algorithm to pick a detour path which eventually ends up reducing the total current flow. This was fixed by storing the calculated paths of the previous run of the TAT solver and applying them whenever the calculated current is lower than expected.

Implementation and optimizations

In the MATLAB implementation of the above described TAT solver, some optimizations were made to improve the speed of the module. This subsection will address these implementation details and optimizations. A flowchart of the TAT solver is displayed in Figure 4.20. The full implementation consists of the following steps:

1. From the provided lattice grid, extract the traps (`latticeElement`s of type `'V+2'`) and sort them according to their potential ϕ_i. The reason for this will become clear below.
2. Decide which of the two electrodes is the cathode and which the anode, depending on the voltage at the electrodes (types `'upper'` and `'lower'`). Then, using the function `getRate`, calculate all electron flow rates:

   - From the cathode to a trap i to the anode;
   - From the cathode to a trap i to a trap j;
   - From a trap i to a trap j to the anode.

   Two optimizations were made for this step.

   Firstly, *every* `latticeElement` has the index of the closest `'upper'` and `'lower'` electrode element pre-stored. These indices are precalculated during the building of the lattice grid and, since the electrodes are considered immutable, they are valid throughout the entire simulation. Therefore, the computationally intensive process of finding several thousands of closest distances only has to be executed once per a simulation.

   Secondly, the rates between the traps and electrodes are similarly calculated only once outside the path finding loop and then stored, so that they can be checked for every step of the pathfinding loop without having to calculate them again (see also Figure 4.20). Since the `getRate` function is computationally intensive, minimizing calls to it is important to optimize the TAT solver.
3. A loop is started that exhaustively finds paths across traps between the two electrodes that works as follows.

   (a) Pick the largest trap rate from the cathode to an unmarked trap. A second loop is started that finds traps in the percolation path:

      i. Mark this trap.

      ii. Calculate all rates from this trap to unmarked traps. This is by far the most computationally intensive process of the TAT solver: this action exists in a loop within a loop and concerns all the traps in the lattice, possibly resulting in millions of calls to the `getRate` function per step. The computation time is improved greatly by assuming that electrons can only tunnel towards a trap of a lower electric potential. Since the traps were already sorted by potential in the first step of the TAT solver, all traps with an index smaller than i are thus considered marked. This limits the `getRate` calls to only unmarked traps with indices above i.

      iii. If the (precalculated) rate to the anode is larger than the rates to all other traps, exit this loop. Otherwise, repeat the trap finding loop.

   (b) With the found path, calculate the path current, the power dissipated in every trap and the trap charge according to the electron occupation probability.

   (c) If there are no more unmarked traps left, exit this loop. Otherwise, repeat the path finding loop.
4. Finally, sum all the individual path contributions to the total device current, collect the dissipated power into the power profile $P(x, y, z)$ and collect the trap charges into the charge profile $Q(x, y, z)$. Return these values to the base model object.

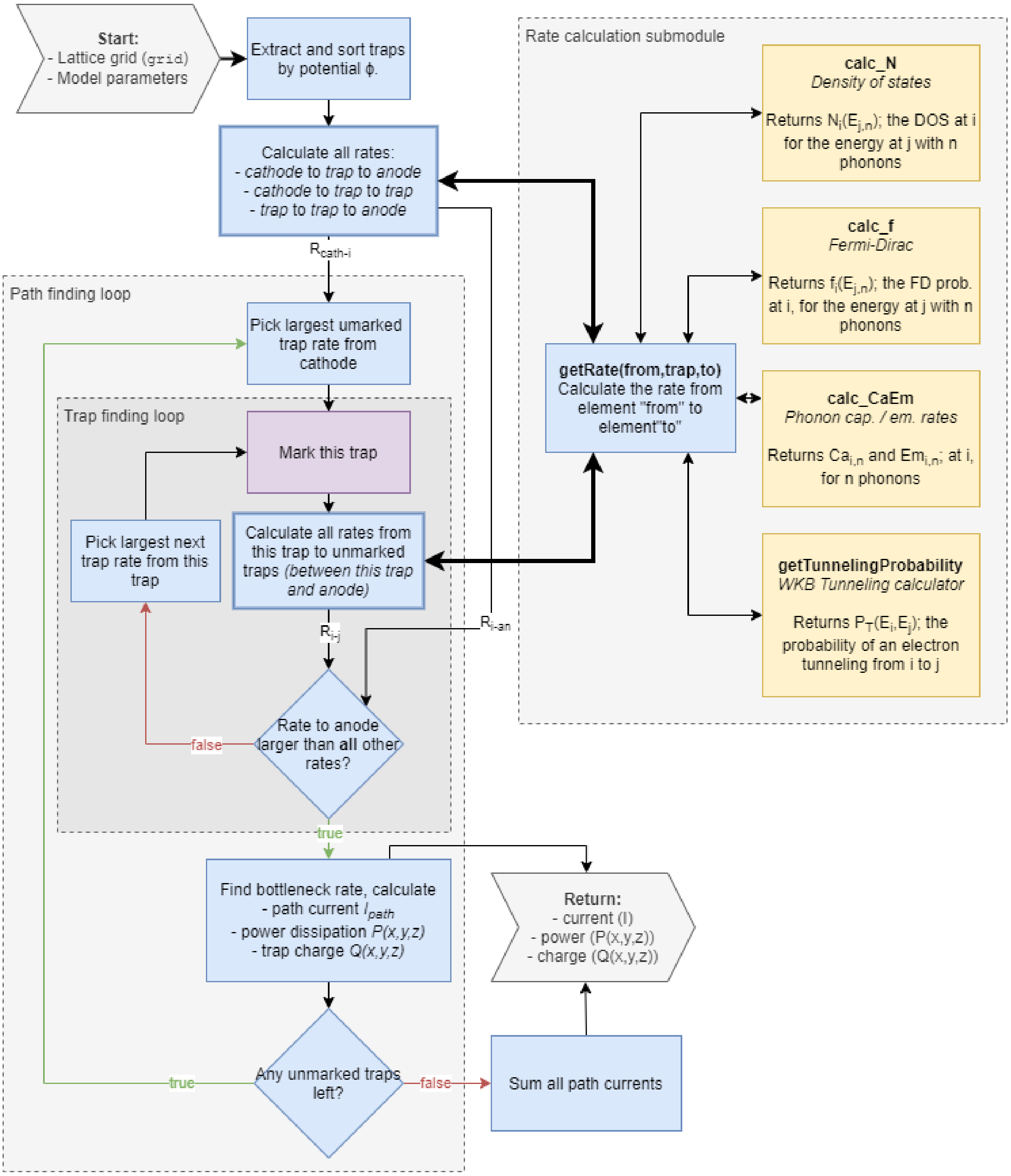

Start:
- Lattice grid (grid)
- Model parameters
Extract and sort traps by potential ϕ.
Calculate all rates:
- cathode to trap to anode
- cathode to trap to trap
- trap to trap to anode
R_{cath-i}
Path finding loop
Pick largest umarked trap rate from cathode
Trap finding loop
Mark this trap
Pick largest next trap rate from this trap
Calculate all rates from this trap to unmarked traps (between this trap and anode)
R_{i-j}
R_{i-an}
Rate to anode larger than all other rates?
false
true
Find bottleneck rate, calculate
- path current I_{path}
- power dissipation P(x,y,z)
- trap charge Q(x,y,z)
Any unmarked traps left?
true
false
Sum all path currents
Return:
- current (I)
- power (P(x,y,z))
- charge (Q(x,y,z))
Rate calculation submodule
getRate(from,trap,to)
Calculate the rate from element "from" to element"to"
calc_N
Density of states
Returns $N_i(E_{j,n})$; the DOS at i for the energy at j with n phonons
calc_f
Fermi-Dirac
Returns $f_i(E_{j,n})$; the FD prob. at i, for the energy at j with n phonons
calc_CaEm
Phonon cap. / em. rates
Returns $Ca_{i,n}$ and $Em_{i,n}$; at i, for n phonons
getTunnelingProbability
WKB Tunneling calculator
Returns $P_T(E_i,E_j)$; the probability of an electron tunneling from i to j

Figure 4.20: The MATLAB implementation of the TAT solver as a flow chart. The submodules calculating the physics explained in the previous subsections are yellow. The rate calculation submodule, the path finding loop and the trap finding loop are indicated by a shading with a dashed line. Calls to the rate calculation submodule have a thick outline, indicating their increased computational intensity.

This concludes the description of the TAT solver. However, TAT is not the only charge transport model that can cause a flow of current through an RRAM device. For thin oxides (< 4 nm [91]), the contribution of electrons tunneling directly from electrode to electrode also becomes relevant.

Direct tunneling

For the calculation of Direct Tunneling (DT), the Tsu-Esaki model was used [35, 88]. This model calculates the net current density by an integration:

$$J = \frac{4\pi m_e q}{h^3} \int_{E_{min}}^{E_{max}} P_T\left(E_x\right) N\left(E_x\right) \mathrm{d}E_x \tag{4.51}$$

where m_e is the effective mass of an electron, q is the elementary charge, h is the Planck constant, and $P_T\left(E_x\right)$ and $N_s\left(E_x\right)$ are respectively the tunneling probability and the supply function at energy level E_x:

- $P_T\left(E_x\right)$ is calculated in the same way as in the TAT solver, by using the WKB approximation. The approximation used is a simplified version of the previously explained module, since DT always concerns tunneling from an electrode to an electrode, and thus involves no barrier reduction due to positively charged traps.

- $N_s\left(E_x\right)$ describes the difference in the supply of carriers at the interfaces of the dielectric:

$$N_s\left(E_x\right) = \int_0^{\infty} \left(f_1\left(E\right) - f_2\left(E\right)\right) \mathrm{d}E_p \tag{4.52}$$

  where E_p is the longitudinal part of the energy level, parallel to the electrode. Filling in the Fermi-Dirac occupation probability for f (Equation 4.27) and integrating, gives:

$$N_s\left(E_x\right) = kT \log\left[\frac{1+\exp\left[-\frac{E_x - E_{f,1}}{kT}\right]}{1+\exp\left[-\frac{E_x - E_{f,2}}{kT}\right]}\right] \tag{4.53}$$

  where k is the Boltzmann constant, T is the temperature, and $E_f, 1/2$ is the Fermi level at either electrode 1 or 2.

The integration of Equation 4.51 is not trivial and therefore has to be performed by using a *middle Riemann sum*: for a number of discrete values within the limits $E_x \in [E_{min}, E_{max}]$, the value of the to be integrated formula in Equation 4.51 is calculated. Then the area between two discrete energy points $E_{x,i}$ and $E_{x,i+1}$, connected by a line, is calculated by averaging between the two points:

$$J = \frac{4\pi m_e q}{h^3} \sum_{E_{x,i}=E_{min}}^{E_{max}} \frac{1}{2}\left(P_T\left(E_{x,i}\right) N\left(E_{x,i}\right) E_{x,i} + P_T\left(E_{x,i+1}\right) N\left(E_{x,i+1}\right) E_{x,i+1}\right) \tag{4.54}$$

The limits of integration E_{min} and E_{max} are based on the three types of conduction that contribute to DT [35]:

- Electrons tunneling from the conduction band (ECB):
  - E_{min} is the highest conduction band edge of the two electrodes;
  - E_{max} is the highest conduction band edge of the dielectric.

- Holes tunneling from the valence band (HVB):
  - E_{min} is the lowest valence band edge of the two electrodes;
  - E_{max} is the lowest valence band edge of the dielectric.

  The sign of the integration limits must be flipped since the valence band is on the opposite side of the Fermi level.

- Electrons tunneling from the valence band (EVB):

- E_{min} is the lowest conduction band edge of the two electrodes;
- E_{max} is the highest valence band edge of the two electrodes.

At a sufficiently high potential difference, the valence band of one electrode can be partly aligned with the conduction band of the other electrode. If this is the case, this conduction type should be taken into account. For this work this will never happen, because the band gap energy of HfO_2 is 5.8 eV, and in normal operation, there will never be 5.8V applied to the device. For completeness, however, it was included.

The complete Tsu-Esaki process is performed separately for every electrode point at the *cathode-oxide interface*, tunneling towards the closest pre-calculated point at the *anode*. This is necessary because in a defective device, the interface of an electrode may not be perfectly planar, which means the oxide thickness is not constant, and thus the DT current density can differ over the area of the device.

After calculating Equation 4.54 for all three conduction types and for every point of the cathode-oxide interface, the separate contributions of every electrode lattice element point are summed and the total DT current is returned.

Note that DT charge transport is *negligible for thick oxides* [91]. In Padovani et al., the oxide thickness used is 10 nm, which is sufficiently thick to ignore DT in favor of the more dominant TAT and drift current. For defect analysis, however, DT current becomes a significant contributor. Therefore, it was also included in the this work's charge transport model.

This concludes the description of the complete charge transport model. The full MATLAB implementation can be found on the repository of the TU Delft [47].

4.5. Determining state change: the kMC engine

This section will explain the kinetic Monte Carlo (kMC) engine, which picks from a list of events to be executed and so regulates the evolution of the model state over time.

First, the basics of the kMC method is explained. Then the three types of events that can occur in the model, as described in [76], will be listed and elaborated. Finally, the implementation and optimization details of the kMC engine will be addressed.

Basics of the kinetic Monte Carlo method

The kMC method, also called the dynamic Monte Carlo method, is a method of describing the change of the state of a stochastic process in steps. Each step is associated with a different time step, hence, “kinetic” or “dynamic” MC.

Every step in a kMC process represents a change from the current state, to any of the possible next states. Every transition associated with these next states, called an *event*, has a corresponding rate, for example R_x for an event x. Events that occur more often have higher rates and vice versa.

The rates of all N possible events in the system are calculated and then summed up to a total rate:

$$R_{tot} = \sum_{x=1}^{N} R_x \tag{4.55}$$

The position of every event in this sum is remembered. Then, a random value is picked from a uniform distribution $u \in [0,1]$. The position in the total rate, uR_{tot}, is then used to find the corresponding event x by checking:

$$R_x < uR_{tot} < R_{x+1} \tag{4.56}$$

The total rate is used to calculate the time step associated with the execution of this event x:

$$\Delta t = R_{tot}^{-1} \tag{4.57}$$

The kMC process is illustrated in Figure 4.21.

Whether or not the chosen event x is executed is, in this model, based on a precalculated time step limit provided by the model handler. If the total timestep Δt is too large, the event will be discarded instead.

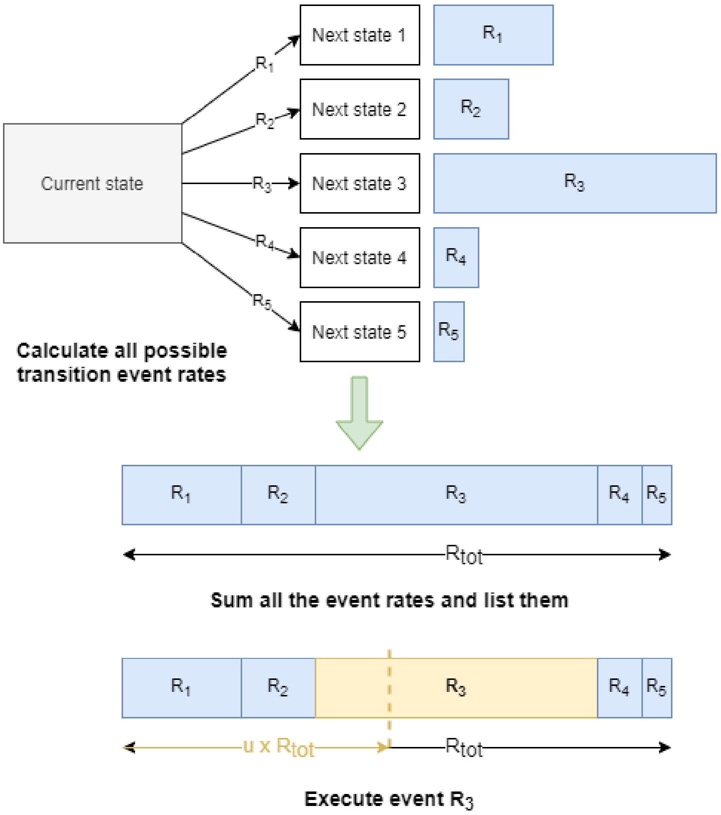


Figure 4.21: The basics of the kMC method, given five possible next states. The rates are summed up and an event is chosen by the random value u.

Events in the model

The model by Padovani et al. [76] uses three different kMC events to summarize the physical processes of RRAM operation:

- **Generation of a Frenkel pair:** a Frenkel pair is a vacancy with an interstitial ion next to it. It therefore represents the generation of vacancies that form the filament, together with the ions that drift away towards the anode. Generation rates are calculated as:

$$G_F(x, y, z) = \nu \cdot \exp\left[-\frac{E_a - bF(x, y, z)}{kT(x, y, z)}\right] \tag{4.58}$$

  where $\nu = 7 \times 10^{13}$ Hz is the effective vibration frequency of O-Hf bonds, E_a is the zero-field activation energy to break O-Hf bonds, $b = 40$Å is the bond polarization factor, $F(x, y, z)$ is the local electric field, k is the Boltzmann constant and $T(x, y, z)$ is the local temperature.

  The generation event rate $G_F(x, y, z)$ is calculated for every `latticeElement` that is an `'oxide'` type, because these elements can turn into vacancies. However, neither the original work [76] nor the similar works it references [58, 90] seem to address the directional nature of vacancy generation: if both a vacancy and an ion are generated, where should the new ion be created? This model fixes this issue by turning the single event rate $G_F(x, y, z)$ into 6 seperate events, $G_{F,-x}(x, y, z)$, $G_{F,-y}(x, y, z)$, $G_{F,-z}(x, y, z)$, $G_{F,+x}(x, y, z)$, $G_{F,+y}(x, y, z)$ and $G_{F,+z}(x, y, z)$. Every one of these six events represents the corresponding oxide element turning into a vacancy type, with the oxide element at the indicated direction turning into an interstitial ion type.

  The rates are determined by projecting the local electric field $F(x, y, z)$ onto the direction vectors pointing towards where the ions will be pushed.

- **Oxygen ion drift:** an ion drifts from one interstitial position to another, pushed by the local electric field. The reference model considers vacancies to be stationary [76]. Ion drift rates are calculated as:

$$R_D(x,y,z) = \nu \cdot \exp\left[\frac{E_{a,d} - k_D F(x,y,z)}{kT(x,y,z)}\right] \tag{4.59}$$

where ν, $F(x,y,z)$, k and $T(x,y,z)$ are the same values as in Equation 4.58, but $E_{a,d}$ is the activation energy for an ion to drift to another position and k_D is a field-induced barrier reduction factor.

The drift event rate $R_D(x,y,z)$ is calculated for every `latticeElement` that is an `'O-2'` type, because these are the ions that can drift to a next position. Just like the vacancy generation rate, drift is of a directional nature and the drift event rates are split into six events separated by the direction in which the ion will drift.

- **Vacancy-ion recombination:** if an ion exists next to a vacancy, it is possible that they recombine into an oxide. Recombination rates are calculated as:

$$R_R(x,y,z) = \nu \cdot \exp\left[\frac{E_{a,r}}{kT(x,y,z)}\right] \tag{4.60}$$

where ν, k and $T(x,y,z)$ are again the same values as in Equation 4.58, but $E_{a,r}$ is the activation energy for an ion to recombine with a vacancy back into an oxide. This event is independent of the local electric field.

The recombination event rate $R_R(x,y,z)$ is calculated for every `latticeElement` that is an `'O-2'` type and exists next to a `'V+2'` type.

Implementation and optimizations

The implementation of the kMC engine is, at its base, relatively simple. The calculation of the event rates however, is not. Even though the variables in the equations given in [76] are provided with the proper values, the resulting rates appear to be unrealistic. This will be further addressed in Section 5.3. For this section, it will be assumed that the factors used in the calculation of the event rates are proper.

Before calculating the event rates, the provided grid of `latticeElement`s is partitioned into two seperate grids: `gridOxide`, which holds all the `'oxide'` elements, and `gridIons`, which holds all the `'O-2'` elements. The other elements types can not be the source of events, as generation is calculated only for oxides, and drift and recombination are calculated only for ions.

First, the Frenkel pair generation rates are calculated using Equation 4.58. The six rates per oxide element over the six possible generation directions are calculated, and put in an $N_{oxide} \times 7$ matrix, where N_{oxide} is the amount of `'oxide'` elements. The 7 columns store the index of the element and the six corresponding rates. If an event for a certain direction is impossible, for example when generation is blocked by a non-oxide element, or the element exists on the edge of the model, the corresponding rate is set to 0.

After the oxide elements, the ion elements are considered. For every oxygen ion, the drift rates are calculated according to Equation 4.59 and also the recombination rates according to Equation 4.60. Similar to the oxide rates, the ion rates are stored in a $N_{ion} \times 13$ matrix. If it is impossible for an ion to drift in a direction, the rate is set to 0, and if there is no `'V+2'` element neighboring on a certain direction, the rate is also set to 0.

All the non-index entries of the rate matrices are then summed. Their position is still remembered, because they are saved in two matrices. A random part of the total rate is determined using the built-in MATLAB function `rand`, and finally the corresponding event is found by searching through the two rates matrices.

The event is returned from the kMC engine to be executed within the base model environment as the MATLAB structure `event`:

```
event = struct(...
    'type',   'G_F', ... The type of event, G_F, R_D or R_R
    'index', 3452, ...  The index of the element where the event occurs
    'to', 6 ...         The direction in which the event occurs (1 to 6 is -x,-y,-z,+x,+y,+z)
    );
```

This event is then interpreted by the base model object and executed according to the descriptions given in the previous subsection.

This concludes the description of the implementation of the kMC engine, and thus the description of the reference model. The following section will focus on the new addition to the model, the defect injector.

4.6. Injection of defects in the model device

In this final section on the model implementation, the new addition to the model will be introduced, the defect injector. The function is used to interpret arguments from the model handler and then build a grid of `latticeElement`s accordingly, as can be seen in Figure 4.2. This means that the defect injector function entails both the building of the defect-free validation grid, and eventually the defective device grid for the defect analysis.

First, the implementation of the defect-free grid builder will be described. Then, the implementation of four different kinds of defects will be explained.

Defect-free grid

The reference model [76] uses a MIM stack with a thickness of 10 nm. The forming occurs at a grain boundary (GB) site [9] which is characterized by an increased density of pre-existing oxygen vacancies, due to the misalignment of the crystal grains as pictured in Figure 4.6b. Based on measurements of the leakage current through an HfO_2 stack [78], the vacancy density on the GB was set to $2.1 \times 10^{21}\,\mathrm{cm}^{-3}$, and the vacancy density around the GB, i.e. in the grain, was set to $1.9 \times 10^{19}\,\mathrm{cm}^{-3}$.

The reference model also uses top and bottom electrodes with a planar oxide interface. Because this work aims to include defects such as electrode roughness and otherwise misshaped electrodes, the electrode-oxide interface can not be considered planar. Instead, the electrodes are modeled as a collection of grid points, which are in terms of implementation the same kind of object as oxide points (`latticeElement`s, see also Section 4.2). The advantage of this implementation is that any electrode shape can be modeled, within the resolution of the lattice grid.

The spacing between the lattice elements is a point of discussion (see also Section 5.3). The reference model [76] and all of the earlier model iterations that it references itself [58, 90, 92] do not mention how the lattice elements are spaced inside the model. It is unknown whether the reference model even uses fixed, discrete positions (as in [1, 69, 82]), or rather a continuous space (as in microscopic models). Since the rates calculated by the kMC engine rely on the existence of fixed event sites, that is the oxide and ion elements, it is assumed that the lattice elements must exist on fixed points in space.

The configuration is therefore assumed to be a *regularly spaced grid*, similar to other macroscopic RRAM models in the state of the art [1, 69, 82]. The grid distance between lattice elements was extracted from Figure 4g-i in [76], together with images of the HfO_2 crystal structure from [46], and was determined to be 3 Å.

Because all the parts of the model are now points in space, they are a 3D graph, and can therefore be supplied with information for the Poisson solver (Section 4.3). Every lattice element i is supplied with a field `A_row` and `Nabla_row`, containing the entries of i-th row of respectively the Laplacian matrix A (Equation 4.15) or the divergence operator matrix ∇ (Equation 4.2). Both are coordinate-encoded, to be compatible with the sparse-matrix-vector-product implementation of the Poisson solver.

The following list describes the steps to building the model lattice grid.

1. The electrode elements are set on the top and bottom edges of the grid, by checking limits imposed on the z coordinate of the lattice elements.

2. After setting the electrodes, the remaining oxide is populated with vacancies assuming a uniform distribution. Every `'oxide'` lattice element has a probability of being a `'V+2'` according to:

$$P_V = \left(N_V \cdot 10^6\right)^{-1/3} \tag{4.61}$$

   Where N_V is the vacancy density (in cm^{-3}). N_V can vary over different regions within the lattice, because the vacancy density is higher around a grain boundary. If a vacancy is set, its energy $E_{t,i}$ is chosen from a normal distribution, where the expected value $\mu = 2.8\,\mathrm{eV}$ and the deviation is $\sigma = 0.1\,\mathrm{eV}$ [59].

3. To ease calculation time in the TAT solver, for every element in the grid the shortest distance to the lower and upper electrode is calculated. The index belonging to this electrode point, that exists closest to the element in question, is also saved as a field in `latticeElement`.

4. Any points of the top electrode that directly border the oxide are set as interface layers, representing the oxygen gettering layer.

Now a grid of `latticeElement` objects is created. It was displayed using the base model function `plotGrid` (Section 4.2) and displayed in Figure 4.22.

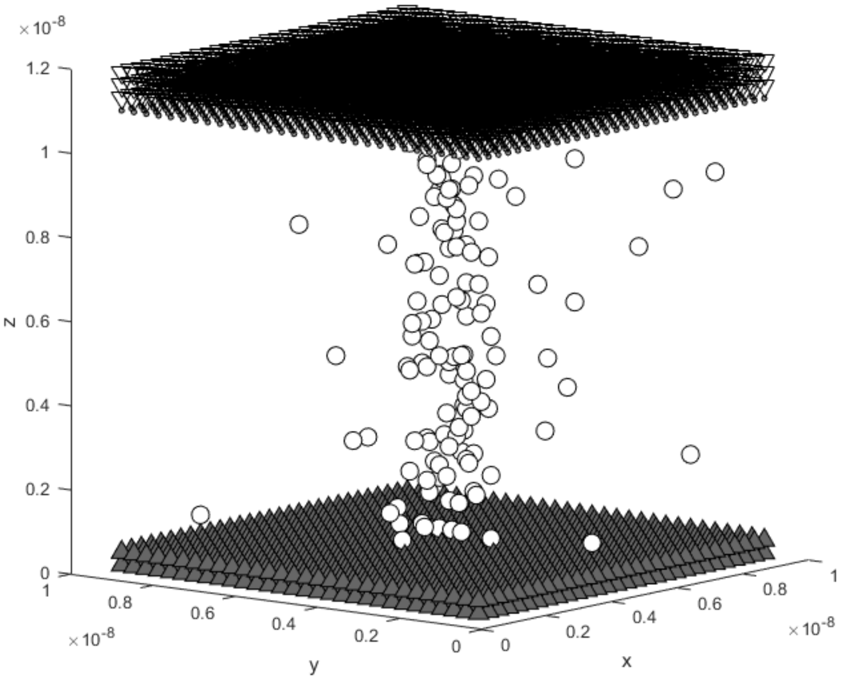


Figure 4.22: The grid of `latticeElement` objects for a defect-free RRAM device. The lower electrode elements are grey triangles (pointing up), the upper electrode elements are transparent triangles (pointing down), the interface layer elements are light grey dots, and the vacancies are white circles.

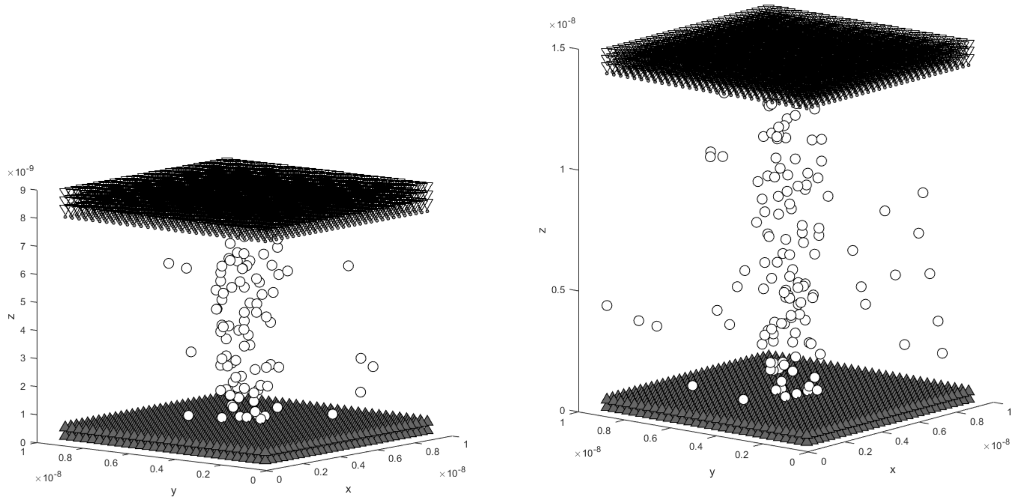
Apart from generating a defect-free grid, the defect injector is also able to add the three different types of defects described in Section 2.4 to the initial grid: oxide thickness variation, electrode roughness and impurities. These will be explained next.

Injecting oxide thickness variation

Varying the oxide thickness is relatively simple: the thickness of the oxide is directly connected to the thickness of the entire lattice grid (including electrodes). Therefore, by varying N_z (the number of lattice elements in the z direction) the oxide thickness of the model can be manipulated, as is shown in the lattice grids generated and plotted in Figure 4.23.

Injecting electrode roughness

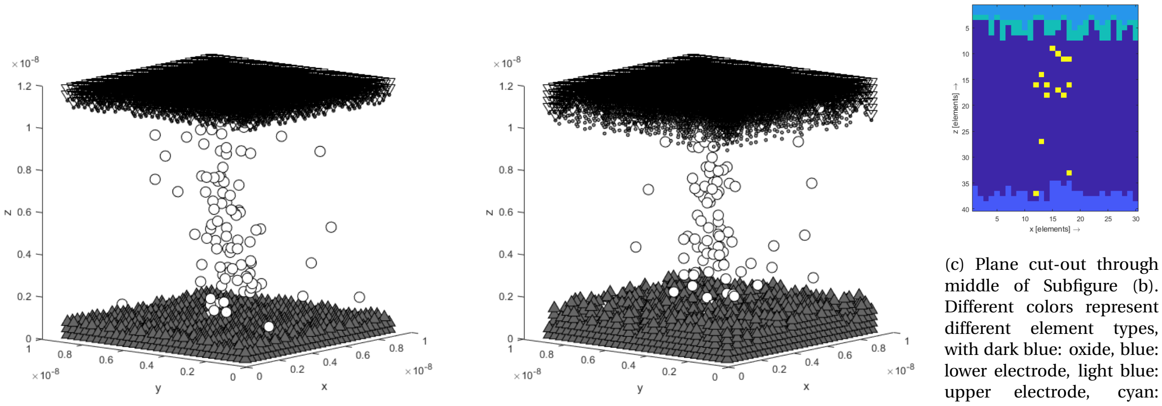
Introducing electrode roughness requires some more implementation. An argument is added in the handler that defines the deviation from the regular, planar electrode surface. This deviation determines how many extra electrode elements are added at the top and bottom electrodes by choosing a uniformly distributed random number between 0 and the provided electrode roughness argument. This fits the implementation of other models on the scale of this work [16, 80]. Grids generated with electrode roughness are displayed in Figure 4.24.



(a) $N_z = 30$ ($t_{ox} = 7.2\,\text{nm}$)

(b) $N_z = 50$ ($t_{ox} = 13.2\,\text{nm}$)

Figure 4.23: Injecting oxide thickness variation in the lattice element grid. The nominal value for N_z is 40 ($t_{ox} = 10.2\,\text{nm}$). The legend is the same as in Figure 4.22



(a) 1 element ($\sigma = 0.3\,\text{nm}$)

(b) 4 elements ($\sigma = 1.2\,\text{nm}$)

(c) Plane cut-out through middle of Subfigure (b). Different colors represent different element types, with dark blue: oxide, blue: lower electrode, light blue: upper electrode, cyan: interface layer, yellow: vacancy.

Figure 4.24: Injecting electrode roughness in the lattice element grid. Nominally, there is no electrode roughness. The legend is the same as in Figure 4.22. Subfigure (c) shows a plane cut-out of Subfigure (b).

Injecting impurities

Another advantage of defining the model lattice grid as a set of discrete points is that certain regions can be assigned different types. In this way, impurities can be injected everywhere into the MIM stack. Impurities are implemented as spheroid regions with a horizontal and vertical radius. An impurity is thus defined by four variables, listed in Table 4.1:

Table 4.1: The four variables that define an impurity, and examples of the values that could be supplied to it.

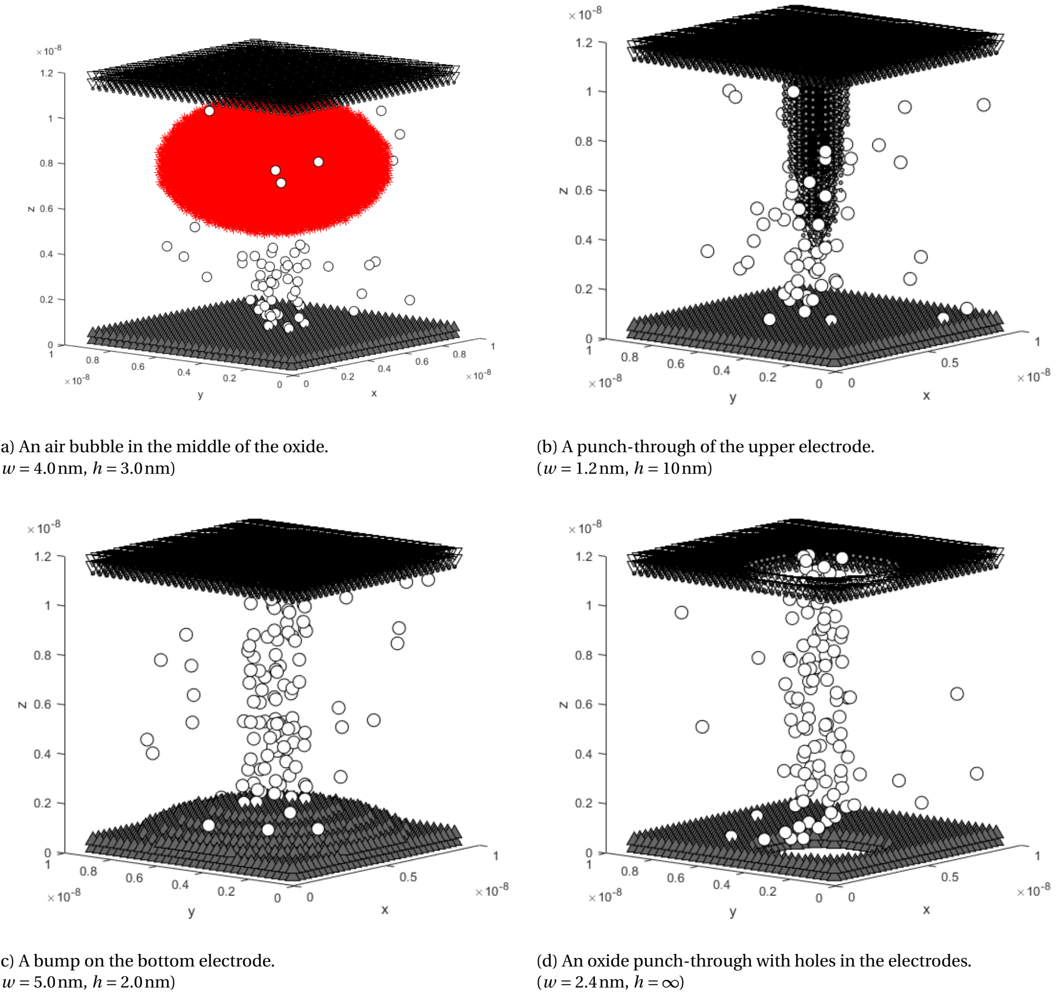
Variable	Description	Example input
Type	The type of the impurity.	`'air'`
Width	The horizontal radius of the impurity.	4 nm
Height	The vertical radius of the impurity.	3 nm
Pos	The x, y, z coordinates of the center of the impurity.	[4.5 nm, 4.5 nm, 8 nm]

Note that for the purpose of defect analysis, the new `latticeElement` type `'air'` was introduced. This element can not change type and has the sole purpose of blocking conduction paths. Therefore it could also be interpreted as any other foreign insulator contaminating the MIM stack.

A `latticeElement` exists within the defined (spheroid) impurity region if the following equation is true:

$$\frac{(x-x_{im})^2+\left(y-y_{im}\right)^2}{w^2}\frac{(z-z_{im})^2}{h^2}<1 \tag{4.62}$$

where x, y, z are the coordinates of the element, x_{im}, y_{im}, z_{im} are the coordinates of the center of the impurity, w is the horizontal radius and h is the vertical radius. This description of an impurity allows a wide range of unique impurity shapes, as is demonstrated in Figure 4.25.



(a) An air bubble in the middle of the oxide.
($w = 4.0$ nm, $h = 3.0$ nm)

(b) A punch-through of the upper electrode.
($w = 1.2$ nm, $h = 10$ nm)

(c) A bump on the bottom electrode.
($w = 5.0$ nm, $h = 2.0$ nm)

(d) An oxide punch-through with holes in the electrodes.
($w = 2.4$ nm, $h = \infty$)

Figure 4.25: Four unique impurities injected into the MIM stack. The legend is the same as in Figure 4.22, with the addition of air, displayed as red x-es.

With the defect injector tool, a wide range of defects can be injected into the model's MIM stack, to research the direct consequences on the resistive switching operation of the RRAM device. Paired with the reference model [76], this model could provide a valuable insight into the symptoms and limits of defects, based on a direct connection between the physical processes associated with RS and the electrical characteristics at the terminals of a device.

This concludes the chapter describing the implementation of this work's MATLAB model.

5

Validation of model

In this chapter, the results of the implementation discussed in Chapter 4 will be compared with verified data such as experimental measurements and known laws of physics. First, the results of the Poisson solver will be verified in Section 5.1 using known laws of physics that are a direct consequence of the Poisson equation. Then, the TAT-solver will be verified using experimental data extracted from the works it is based on [91] in Section 5.2. Afterwards the kMC engine will be discussed, with as a most important point the important information missing from the reference model [76] in Section 5.3. Finally the full defect-free model will be presented in Section 5.4, which, like the kMC engine, was missing important information and underwent several adjustments in attempts to have it operate reliably. Simulations were performed on the TU Delft QCE cluster.

5.1. Validating the solution of the 3D Poisson equation solver

To validate the solution from the 3D Poisson equation solver, the solution vector $\boldsymbol{\phi}$ must be shaped so that the Poisson equation holds:

$$\nabla^2 \phi = f. \tag{5.1}$$

Or, in its discrete form:

$$A\boldsymbol{\phi} = \boldsymbol{b} \tag{5.2}$$

Verifying that this equation always holds after a run of the Poisson solver validates that it correctly calculates either the electric potential or the temperature.

First however, the Laplacian matrix A should be verified. If A incorrectly represents ∇^2, the discrete Equation 5.2 will hold, but the original, analytic Equation 5.1 will not. To verify that $A \approx \nabla^2$, the outcome function f is set to 0, producing the Laplace equation:

$$\nabla^2 \phi = 0 \tag{5.3}$$

Analytical solutions to the Laplace equation exist given boundary regions for ϕ. One of these analytical solutions is that of a region bounded by two (infinite) parallel planes of different potential. For this model, that translates to the two electrodes when a voltage is applied to them. In this case, the electric field profile is expected to be constant:

$$\boldsymbol{F}_i = -\left(V_U - V_L\right) / t_{ox} \hat{\boldsymbol{k}} \tag{5.4}$$

where V_L (V_U) is the electric potential at the lower (upper) electrode, t_{ox} is the oxide thickness and $\hat{\boldsymbol{k}}$ is the unit vector in the z direction. The corresponding potential profile is obtained by applying the gradient (∇) and is therefore expected to be:

$$\phi_i = \nabla\left(\boldsymbol{F}_i\right) = V_L + F \cdot z_i \tag{5.5}$$

where z_i is the coordinate of element i parallel to the oxide thickness and F is the magnitude of the electric field.

Note that FDM uses only the adjacent neighbors for calculating the derivative in a direction. This means that even though the Poisson solver outcome ϕ_i will have a constant increase, the derived magnitude of F on to the boundary points will be exactly *half* of Equation 5.4. The reason behind it is this: away from the boundaries, an element sees the same slope on both sides. On boundaries, an element sees a slope only on one side. This anomaly will be taken into account when verifying the equation Equation 5.4. It has no effect on the operation of the model, since the boundary points exist outside the simulation space, the oxide.

Verifying the Laplacian matrix with an analytical solution

Figure 5.1 shows the results when comparing the solution of the Poisson solver for $A\boldsymbol{\phi} = \mathbf{0}$ (Equation 5.3).

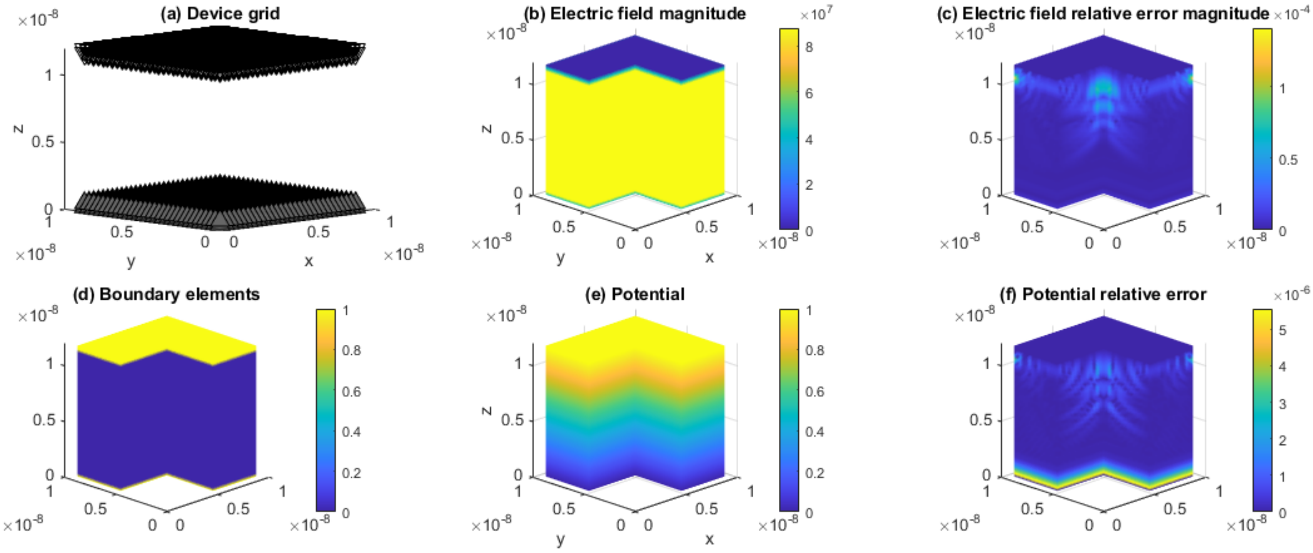


Figure 5.1: Validating the Laplacian matrix of the Discrete Poisson equation in 3D from the solver described in Section 4.3. In reading order: (a) Layout of the grid; (b) electric field magnitude F calculated by the Poisson solver; (c) the relative error compared to Equation 5.4; (d) the boundary points used in the Poisson solver; (e) electric potential ϕ calculated by the Poisson solver; and (f) the relative error compared to Equation 5.5.

It considers an oxide that is 37 elements thick (including 2 interface layers). This means that the distance between the two electrodes is $38 \times 3\text{Å} = 11.4\,\text{nm}$. A voltage of 1V is applied to the upper and lower electrodes. According to Equation 5.4, this should result in a constant field of $F = 8.772 \times 10^7\,\text{V/m}$ and a linear function of z (Equation 5.5) for the Poisson equation solution ϕ.

The solution is indeed valid: given the lattice grid as displayed in Figure 5.1(a), the maximum relative difference between the solution of the Poisson solver $\boldsymbol{\phi}$ Figure 5.1(e) and that of Equation 5.5 is only $\approx 5.6 \times 10^{-6}$ Figure 5.1(f). It can thus be assumed that the Laplace matrix is determined correctly. The maximum difference between the calculated electric field $\boldsymbol{F}$ Figure 5.1(b) and that of Equation 5.4 is $\approx 1.4 \times 10^{-4}$ Figure 5.1(c), therefore the calculation of the gradient of $\boldsymbol{\phi}$ can also be considered accurate.

Validating the full potential of the Poisson solver

Now that the Laplacian matrix A has been verified, the Poisson solver can be put to the test. A random grid of `latticeElement`s was generated, with a high density of point charges of +1e existing inside the grid. The electrodes are no longer planar but rather irregular, with a bump added to the lower electrode. A voltage of 2V is applied to the electrodes. Figure 5.2 shows the results of applying the Poisson solver to this grid to calculate the potential profile.

It can be seen that the Conjugate Gradient method reduces the relative error $\left|A\boldsymbol{\phi} - \boldsymbol{b}\right| / \boldsymbol{\phi}$ down to $\approx 7.2 \times 10^{-7}$. Several observations can be made from the found solution:

- The large amount of charge inside the oxide greatly increases the electric potential inside the oxide (Figure 5.2(b)).

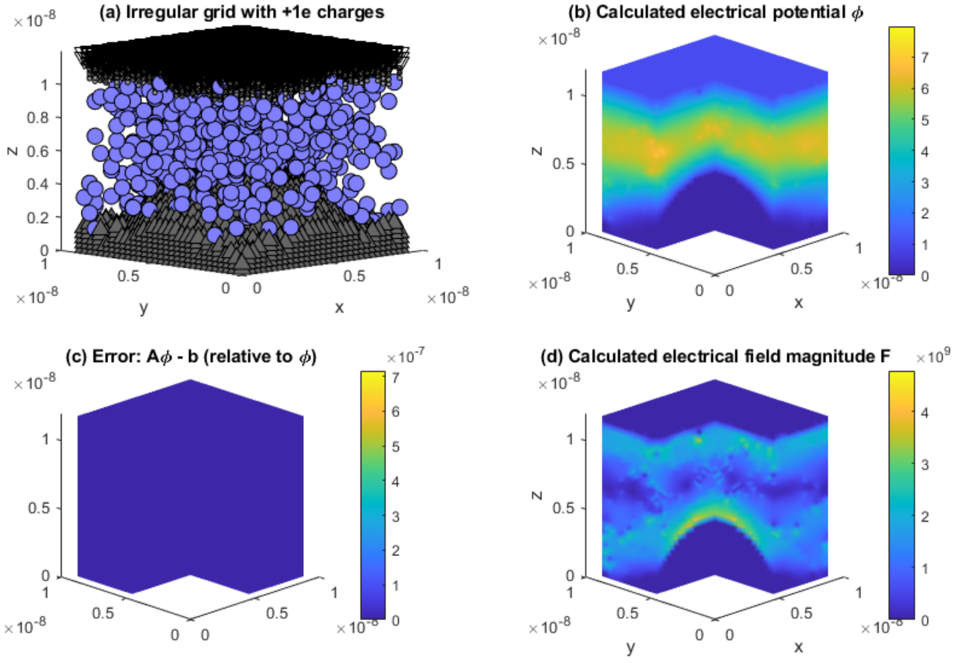


Figure 5.2: Validating the solution to the Discrete Poisson equation in 3D from the solver described in Section 4.3. In reading order: Layout of the grid; electric potential $\boldsymbol{\phi}$ calculated by the Poisson solver; the error from Equation 5.2; and the magnitude of the electrical field.

- The magnitude of the electric field is large (Figure 5.2(d)) and points outwards (not visible in Figure 5.2(d), but evident from the visible gradient in (Figure 5.2(b)).

- Near the bump on the lower electrode, the electric field is of a higher magnitude, as the oxide is thinner.

- The pairs of lighter and darker spots in Figure 5.2(d) represent the superposition of a point-charge electric field on an existing field: if a field points in a direction, and a point charge field is superposed on it, it will amplify the field on one side and dampen it on the other side. Hence, the lighter and darker spots.

These observations align with the expected behaviour of electrostatics, and are therefore enough validation that the Poisson solver is indeed able to find a solution ϕ to the Poisson equation in 3D.

5.2. Validating the calculated device current

For the validation of the charge transport model, measurements of the leakage current through the hafnium oxide gate of a NMOS transistor from Vandelli et al. [91] were used. The TAT model from Larcher and Vandelli et al., as described in Section 4.4, was applied to a similar setup. The device was not set up using the lattice builder from the main model (Section 4.6), but, since only the traps are considered in this case, it was set up using a reduced version of the lattice builder. The source code for this reduced lattice builder can be found in the TU Delft repository [47].

This reduced version only generates vacancy elements and assumes constant, planar electrodes, as is the case in the reference. The vacancies are generated randomly and uniformly throughout the grid according to a given density. They are not connected by a Laplacian matrix, because in this reduced lattice grid with planar electrodes the electric field F is constant over space and the potential is a simple function of the position of a trap parallel to the oxide thickness, $\phi(z)$. The setup therefore only concerns the TAT solver, and leaves out the Poisson solver and the kMC engine.

The measurements in [91] were performed on a 5 nm thick HfO_2 oxide, layered onto a 1.1 nm thick SiO_2 interface layer and a TiN electrode. The device area is very large ($30 \times 30 \mu m^2$), to average out statistical fluctuations. Current density measurements were then performed on 4 ambient temperatures: 50 K, 200 K, 300 K and 400 K, over constant values from 0 to 2V.

Within the HfO_2 layer, there is a uniform vacancy density of $4.5 \times 10^{19}\,\mathrm{cm}^{-3}$. The vacancy density within the interface layer, as well as its thickness without the TiN electrode, was not clear from the referenced work [91] and therefore was left out. Since the oxide is thinner than in the reference measurements, the simulated current density is therefore expected to be higher than the measured current density, but the exact difference will be investigated.

Instead of simulating a single $30 \times 30\,\mu\mathrm{m}$ device, a large number ($N_{sim} = 1000$) of smaller devices ($20 \times 20\,\mathrm{nm}$) was simulated in parallel. The statistical fluctuations of the current density are in this way equally averaged out, with the additional benefit of faster simulation by parallelization. A complete overview of all the properties used in the verification of the TAT solver is shown in Table 5.1.

Table 5.1: Overview of all the variables used in the verification of the TAT solver, as obtained from [91]. A complete list of all properties used in simulations can be found in Appendix A.

Variable	Description	Value	Note
t_{ox}	Oxide thickness	5 nm	Thicker in reference, but *unclear value*
$d_x \times d_y$	Device area	$20 \times 20\,\mathrm{nm}$ ($\times 1000$)	Larger in reference, but statistical fluctuations still mitigated
E_T	Trap energy in HfO_2	$1.4 - 2.4\,\mathrm{eV}$	
S	Huang-Rhys factor in HfO_2	17	
r_t	Trap capture radius	5.64 Å	Obtained from trap cross section $\sigma_t = 1 \times 10^{-14}\,\mathrm{cm}^2$ by $r_t = \sqrt{\sigma_t/\pi}$
N_V	Trap density in HfO_2	$4.5 \times 10^{19}\,\mathrm{cm}^{-3}$	

Figure 5.3 shows the setup of the TAT solver verification. The results of the simulation are shown and compared to the measurement data in Figure 5.4. The deviation between the simulated current density and the measured current density is shown in Figure 5.5.

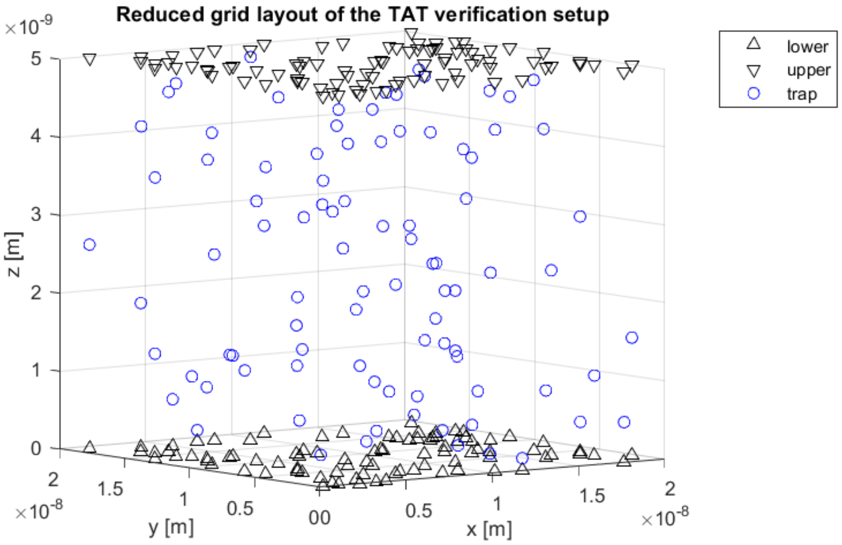


Figure 5.3: The grid setup of the TAT verification simulation.

To validate the percolation path finder described in Section 4.4, the percolation paths were visualized by lines between the electrodes in Figure 4.19. The paths in the figure show that the electrons indeed prefer a path of least resistance, with some fluctuations due to the randomized trap energies (see also Table 5.1).

Some traps are not part of a percolation path. These traps exist too far away from the cathode and other traps, which means that any possible path towards the trap was either already marked by another, faster path, or the path has a rate of 0.

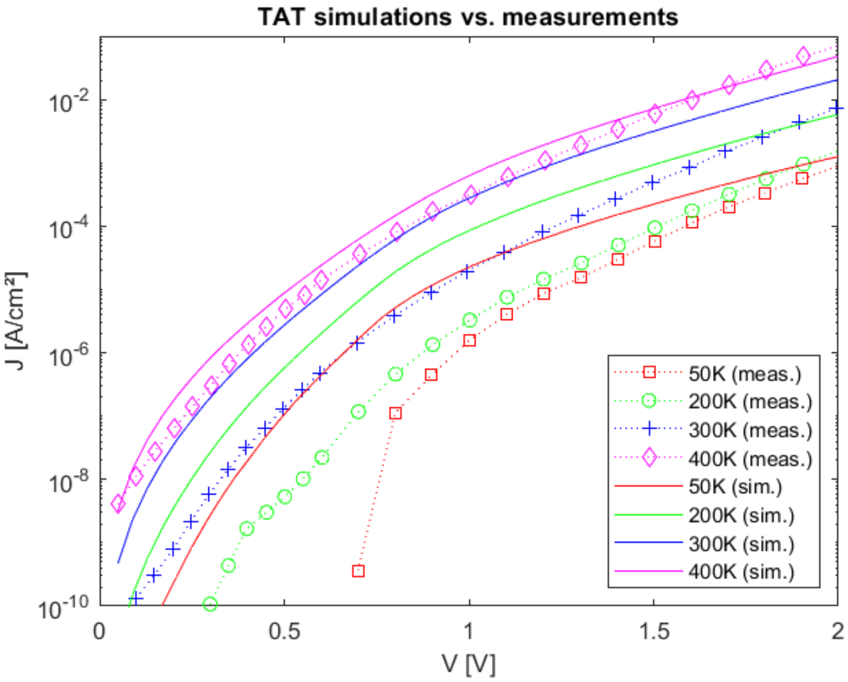


Figure 5.4: The results of TAT verification simulation. The measurement data are displayed by symbols connected by dashed lines. The simulation results are displayed by unbroken lines.

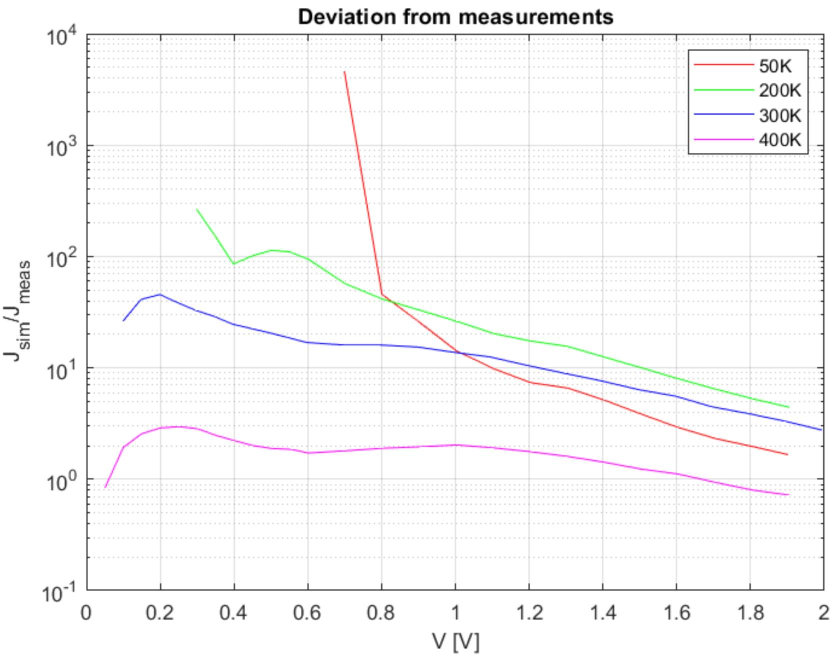


Figure 5.5: The relative deviation of the results from the measurement data.

It is evident from Figure 5.4 that the simulation results do not completely represent the measurement data: the simulation current is higher. The reason for this is that the SiO_2 interface layer described in [91] was *not included* in this setup (Figure 5.3), because the vacancy density and thickness of this layer were, unfortunately, not sufficiently clear. This causes the oxide to be slightly thinner, which is expected to increase the electron flow rates: the paths that the electrons have to take are shorter, the electric field is higher and the potential differences larger.

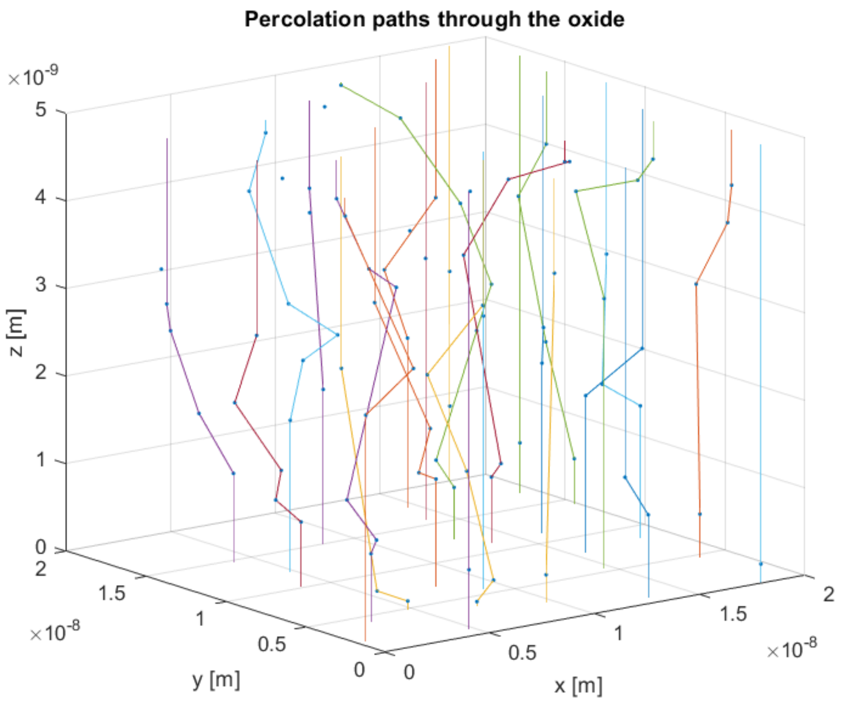


Figure 5.6: An example of percolation paths as determined through the oxide, for $V = 1\,\mathrm{V}$.

The relative deviation from the measurement data is displayed in Figure 5.5. The deviation from the measured data appears to decrease for higher temperature and higher voltages. The largest deviation is observed for 50 K, which is shown to decrease rapidly for low voltages in the measurements, but not in the simulations (Figure 5.4).

Even though the results deviate from the measured data, they do appear realistic. The deviation decreases as the temperature increases, which is in line with the information given in Vandelli et al.: the conductivity of HfO_2 traps is more temperature-dependent than the missing interface layer, due to the larger Huang-Rhys factor ($S = 6$ for SiO_2, $S = 17$ for HfO_2). The Huang-Rhys factor represents the ability of charge carriers to couple with phonons [91] (see also: Section 4.4). Because of this higher temperature dependency, HfO_2 is expected to dominate conduction for higher temperatures.

For the purpose of this model, it was therefore decided that the charge transport model is valid. In the future, more research might be necessary to completely validate the model, but with the available information in [91], this was unfortunately not possible.

5.3. Validating the events that change the model state

The kMC engine should pick an event from a large list of events calculated using the rates that were elaborated in Section 4.5. This section provides a critical analysis of the event rates that drive the kMC engine, using the values for the relevant variables that were provided by Padovani et al. [76] and its references. It is found that these values *can not produce* realistic rates in this kMC implementation and that, evidently, additional information is missing in the original sources [58, 59, 76, 90, 91].

Recall (Section 4.5) that the three events that occur during the simulation of the model are the stress-induced generation of a vacancy-ion pair (G_F), the drift of an oxygen ion from one interstitial position to the next (R_D) and the recombination of a vacancy-ion pair (R_R). The corresponding equations for these rates are:

$$G_F(x, y, z) = \nu \cdot \exp\left[-\frac{E_a - bF(x, y, z)}{kT(x, y, z)}\right] \tag{5.6}$$

$$R_D(x, y, z) = \nu \cdot \exp\left[\frac{E_{a,d} - k_D F(x, y, z)}{kT(x, y, z)}\right] \tag{5.7}$$

$$R_R(x, y, z) = \nu \cdot \exp\left[\frac{E_{a,r}}{kT(x, y, z)}\right] \tag{5.8}$$

where $F(x, y, z)$ is the magnitude of the local electric field, and $T(x, y, z)$ is the local temperature. The values for the remaining physical constants in these equations were obtained either from Padovani et al. or any of its references, and are displayed in Table 5.2.

Property	Description	Value
ν	Effective O-Hf bond vibration frequency	7×10^{13} Hz [58, 76]
E_a	Zero-field effective O-Hf bond breakage energy	2.9 eV [75, 76]
b	Bond polarization factor	40 eÅ [75, 76]
$E_{a,d}$	Ion diffusion activation energy	0.7 eV [76]
k_D	Factor depending on material properties	−3 eÅ [58]
$E_{a,r}$	Recombination activation energy	0.2 eV [76]

Table 5.2: The variables in the kMC event rate equations, the description of their meaning, and their value, together with the reference(s) from which the value was obtained.

Even though, in the reference work, these values for the properties appear to produce realistic results, verified with experimental measurements, the rates that are calculated using Equations 5.6 to 5.8 appear unrealistic. In order to clarify this problem, the constant and variable parts of the equation must be identified.

Every property listed in Table 5.2 is a constant, as well as the Boltzmann constant k. This means that the only variable parts of the event rates equations are the local electric field magnitude $F(x, y, z)$ and the local temperature $T(x, y, z)$. These are the parts that change over the course of a simulation and therefore fully control the change in event rates. Several problems with the equations can be pointed out by plotting the outcomes of the event rate equations versus the temperature and electric field magnitude, as is done in Figure 5.7.

From this figure, three problems are apparent:

- **The electric field strength plays a more important role that stated.** Padovani et al. states that the soft dielectric breakdown during the forming of the filament is mostly driven by an increase in *temperature* in the grain boundary. They also state that in the calculation of the electric field magnitude, the charge of vacancies and ions in the lattice is taken into account. From the results of this work's Poisson solver (see also Section 5.1), it appears that a point close to a charged point experiences a significant local field, which can possibly be much more significant than the externally applied field. This has the effect of events firing *independent of the externally applied voltage*, which is unrealistic: the high local field magnitude is only increased with the addition of newly generated vacancies, which further accelerates generation, causing a full dielectric breakdown within a nanosecond of simulation time. However, the referenced model does not sufficiently mention the relevance of the local electric field to justify its importance in the event rates equations. To illustrate this problem, the electric field magnitude at the lattice element neighboring a +2e point charge vacancy is calculated at 3Å, using Coulomb's law:

  $$F = \frac{1}{4\pi\epsilon}\frac{q}{r^2} \tag{5.9}$$

  where $\epsilon = \epsilon_r \cdot \epsilon_0$ is the dielectric permittivity, q is the magnitude of the point charge and r is the distance from the point charge:

  $$F = \frac{1}{4\pi\epsilon}\frac{+2\text{e}}{\left(3\,\text{Å}\right)^2} = 320\,\text{MV/cm} \tag{5.10}$$

  For the electrode-induced field to reach this magnitude in an oxide of 10 nm thick, 320 V needs to be applied to the terminals of the device. Obviously, this magnitude of electric field is too large, which causes a runaway generation of vacancies regardless of the applied voltage, even when no voltage is applied at all. This problem is pictured in Figure 5.8.

  Note that Coulomb's law does not precisely describe the electric field calculated by the Poisson solver, because the Poisson solver also takes into account the boundary conditions that are imposed on the electric potential at the electrodes. The order of magnitude is, however, comparably large.

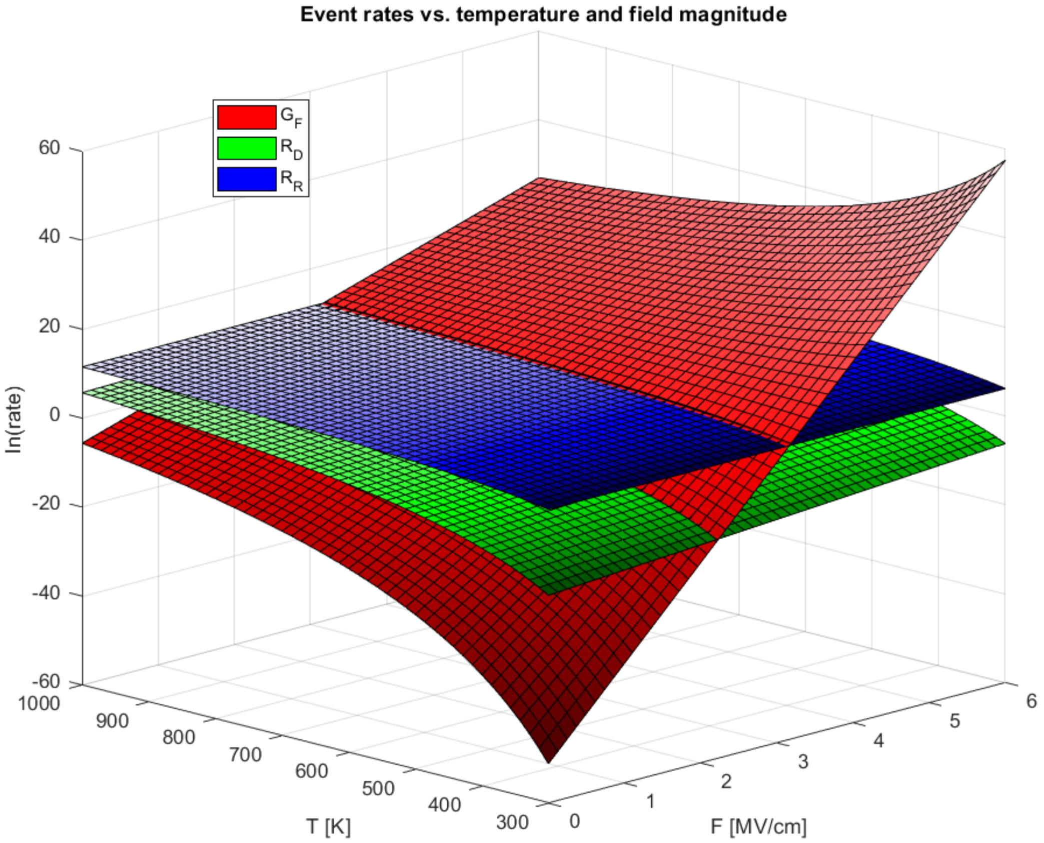


Figure 5.7: A surface plot of the three kMC event rates vs. the local temperature $T(x,y,z)$ and the local electric field $F(x,y,z)$. The z axis shows the natural log of the event rate.

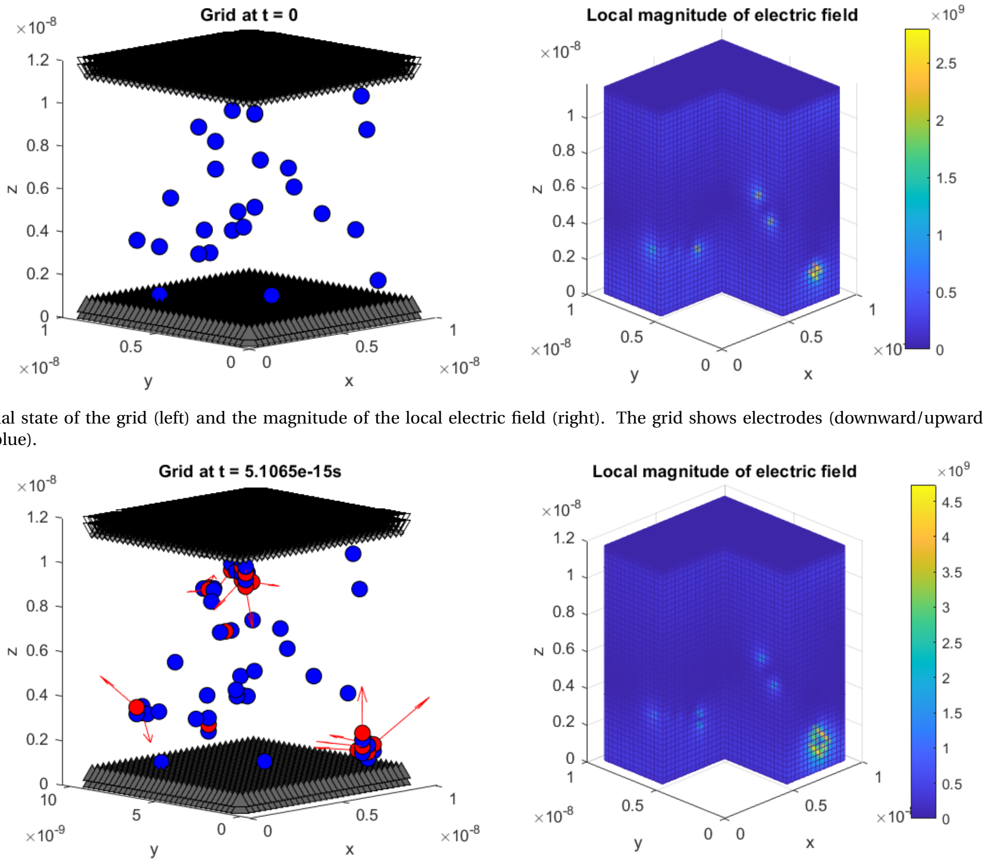
- **The drift rate is never dominant.** The rate that describes the diffusion of oxygen ions (R_D, displayed in green in Figure 5.7) is *never* higher than the two other rates, for *all* values of $T(x,y,z)$ and $F(x,y,z)$. For low values of $F(x,y,z)$, R_R dominates, which implies that an oxygen ion will recombine before it drifts away. For high values of $F(x,y,z)$, G_F dominates, which implies that a new vacancy-ion pair will form before the ion drifts away. In other words, the event describing the diffusion of an oxygen ion is oddly rare. This is unrealistic because the drift of oxygen ions is the main mechanism behind resistive switching in OxRAM devices [97]: thus, the rate is expected to be more dominant.

- **For high values of the electric field, the temperature dependency flips.** As can be seen in Figure 5.7, the rates that are influenced by the local electric field (G_F and R_D) mirror their dependency on temperature as soon as they cross a certain threshold. This threshold represents the crossing of the barrier set by the activation energy, e.g. for the generation rate:

$$E_a - bF(x,y,z) = 0 \tag{5.11}$$

This implies a local electric field magnitude of 3.59 MV/cm for generation, and 23.3 MV/cm for drift rates. For an oxide of 10 nm thick (as in the reference model), this translates to the same applied voltage values. This means that if the applied voltage crosses 3.56 V, generation rates no longer favor the highest temperature regions, but rather the *lowest temperature* regions - which is unrealistic. The references concerning this model all mention temperature being the main driving force behind the formation of a filament [10, 58, 76]. In other words, the high temperature caused by the increased conductivity of a forming filament should not *slow down* generation, but *accelerate* it.

An explanation for the temperature dependency flipping for high electric field magnitudes could be



(a) The initial state of the grid (left) and the magnitude of the local electric field (right). The grid shows electrodes (downward/upward triangles) and vacancies (blue).

(b) The state of the grid after ≈ 5×10^{-15} s (left) and the magnitude of the local electric field (right). The grid shows electrodes (downward/upward triangles), vacancies (blue) and interstitial oxygen ions (red). The red arrows indicate the direction of the electric field at the position of the ions.

Figure 5.8: An example of the runaway generation of vacancies, when no voltage is applied to the electrodes (V = 0V). In the grid, new vacancies can be seen generating randomly around existing vacancies, because the local electric field magnitude at these points is too high.

that the local electric field magnitude should never cross 3.59 MV/cm in the normal operation of the model. But, as was apparent from the first point of this list, the local field magnitude is able to far cross that value independent of the applied voltage.

The above three problems are directly evident from Figure 5.7. However, there is one more problem that may not be immediately visible.

- **The generation activation energy (E_a) is too high for the referenced timestep.** Using the values from Table 5.2, the generation rate can be calculated for a case where, according to the reference work, the self-accelerated soft breakdown should be started. According to Figure 5.9, a result copied from the reference model, this is when ≈ 2.5 V is applied to the device, corresponding with an electric field magnitude of 2.5 MV/cm for the referenced oxide thickness of 10 nm. If the temperature is already relatively high (500 K), the generation rate (Equation 5.6) for an element becomes:

$$G_F(x,y,z) = 7\times10^{13}\,\mathrm{Hz}\cdot\exp\left[-\frac{2.9\,\mathrm{eV}-40\,\mathrm{e\AA}\cdot2.5\,\mathrm{MV/cm}}{1.381\times10^{-23}\,\mathrm{J/K}\cdot500\,\mathrm{K}}\right] = 5.023\times10^{-6}\,\mathrm{Hz} \tag{5.12}$$

Considering that the total timestep of a single model cycle corresponds to the sum of all rates (see also Section 4.5), one could assume that every single lattice grid point in the 30 × 30 × 40 grid is at the referenced critical condition for a vacancy to form. This means that the total timestep of the model becomes:

$$\Delta t = \frac{1}{R_{tot}} = \frac{1}{36000\cdot5.023\times10^{-6}\,\mathrm{Hz}} = 5.530\,\mathrm{s} \tag{5.13}$$

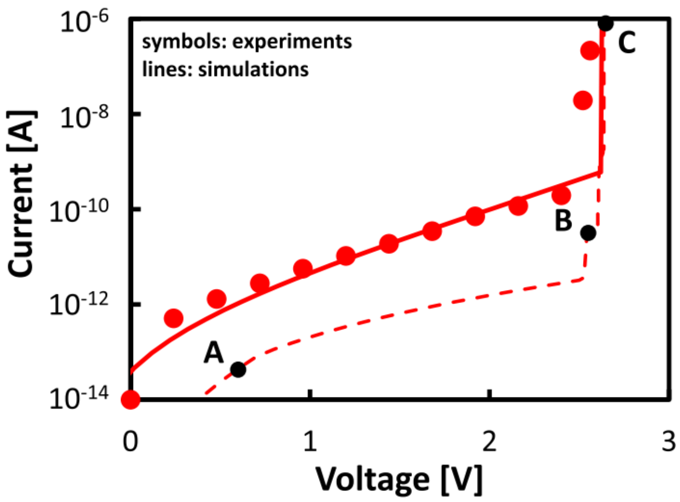


Figure 5.9: I-V curve of the forming process as simulated in [76]. The forming voltage lies around 2.5V, where generation rates should peak. Copied from Padovani et al. [76]

Which implies that, under the (very generous) assumption that *every single* lattice element exists at the critical vacancy forming conditions, it still takes *several seconds* for a single vacancy to form. This makes no sense considering that realistic forming times, both referenced in Padovani et al. [76] and in literature (see also Chapter 2) are in the scale of a few microseconds.

A reason for this oddly low generation rate could be that the local electric field, and by that the charge of non-oxide lattice elements, plays a quite significant role in the formation of new vacancies. The calculation and effect of charge trapping in vacancies is however hardly mentioned by the reference works [58, 59, 76, 91]. The only information that is given is that effects such as charge trapping are taken into account when calculating the electric field strength - the value of this charge, and the strategies of modeling these effects, are not mentioned. Without the charge of vacancies and ions, the local electric field can not be calculated, and as stated above, the generation rates are too low.

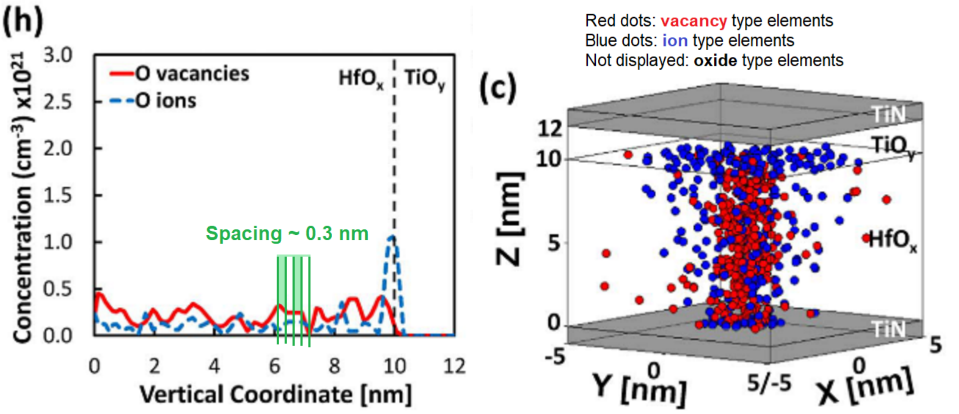
In conclusion, there are too many unexplained oddities in the referenced implementation of the kMC engine. These oddities should be further investigated, or perhaps redefined entirely, in future works. For this work, however, the kMC engine was not able to realistically simulate the change of the device state over time, for the reasons shown above. It was therefore decided that state change had to be excluded from this work's final model.

5.4. Validating the full model

Apart from validating the separate modules of the model, the complete model also needs to be validated by comparing it to I-V plots and lattice layout plots from the reference work [76]. Unfortunately, it became clear in Section 5.3 that the kMC engine is unable to operate as presented in the reference model, because too much information is either unrealistic or missing. Outside of the kMC engine as well, many contextual but crucial details remain unmentioned in the reference work. Modules such as the Poisson solver used, the spacing of the lattice elements, charge of the lattice elements, details of the kMC engine are glossed over and unreferenced. It was therefore impossible to build a fully functional model with just the information provided in [76] and its references.

Several attempts were made, however, to fill in the gaps of missing information, using results from other works, literature research, and simple scaling of parameters to fit the reference. This section will thus provide a list of the attempts made to fill two major gaps of information:

- **Spacing and type of the lattice elements:** how are the lattice elements placed and connected in the model space, and what do they physically represent?
- **Charge of vacancies and ions:** how is the charge of vacancies and ions determined?



(a) Extracting a spacing of ≈ 3Å from the horizontal resolution of Figure 4h in [76].

(b) Extracting the lattice element types of vacancies, ions and oxide from Figure 4c in [76].

Figure 5.10: Extracting lattice element spacing and type from figures in the reference work. Copied and adapted from [76].

Spacing and type of the lattice elements: attempted solutions

Two aspects of the model lattice are important for understanding the operation of the model:

- **Spacing:** the position of lattice elements within the model space. This could be a regular grid-like structure, but also a uniform distribution, or a crystal structure. The spacing also influences the connections between the lattice elements, which in turn may drastically change the implementation of the Poisson solver.
- **Type:** what a lattice element represents. Element types can be high-level (oxide, vacancy, interstitial ion) or of a lower level (Hf/O atom, Hf$_+$/O$_-$ ion).

Neither are clearly defined in the reference model. Its figures, however, do appear to show an indication of grid element space and types, specifically Figure 4 [76]. Figure 5.10a shows how the spacing was assumed from the resolution of one of the figures of the reference model. Figure 5.10b shows how the lattice element types were assumed from the visualization of the model grid. From these figures, the following assumptions were made:

- **Assumed spacing:** the assumed spacing is a regular grid with a mutual distance of 3Å.
- **Assumed type:** the assumed element types is a high-level list of possible states (see also `latticeElement` in Section 4.2).

Since this did evidently not produce usable results, two attempts were made to fit this work's model to the reference model: the addition of an interstitial grid, and the generalizing of the lattice coordinates to a monoclinic grid.

- **Additional interstitial grid.** In the first attempt, oxygen ions would replace oxide elements as the result of generation and drift events. This seemed unrealistic, as the neighboring oxide element - that was replaced by the ion element - would no longer be able to generate a new vacancy-ion pair. In realistic operation, oxygen ions on generation drift not to another point in the lattice, but rather to a position *between* the lattice points, an interstitial position (see also Section 4.4) [10, 33, 71]. Therefore a new implementation was made, which can be found in the branch `interstitial` on the complete repository [47].
  - The interstitial elements are stored at the end of the list of `latticeElement` objects. Before implementing the interstitial grid, the `latticeElement` list was stored in MATLAB as a *3D array*. After implementing the grid is turned into a regular *vector*, with all geometric information such as position and connections stored inside the `latticeElement`s.
  - Generation event rate calculation no longer uses the Cartesian components of the local electric field (F_x, F_y, F_z) but rather the components that point to the 8 surrounding interstitial positions.

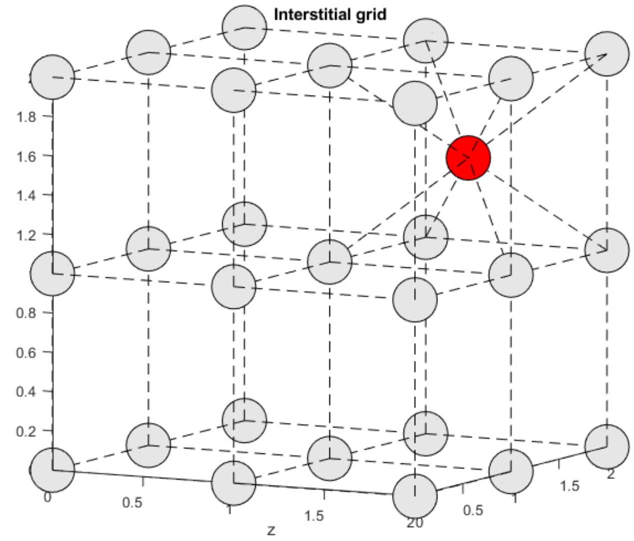


Figure 5.11: Example of a layout of the regular grid, with an additional interstitial regular grid. A single interstitial element exists between the elements of the regular grid.

  To achieve this, the 8 unit vectors pointing along the diagonals towards the interstitial positions defined by the 8 × 3 vector set:

$$\hat{U} = [\hat{\boldsymbol{u}}_1, \hat{\boldsymbol{u}}_{\boldsymbol{2}}, \cdots, \hat{\boldsymbol{u}}_{\boldsymbol{8}}]^{\mathrm{T}} \tag{5.14}$$

  Multiplying this vector set with the local electric field vector $\boldsymbol{F}(x, y, z)$ then produces the 8 × 1 vector of the local electric field projected onto the 8 diagonals:

$$\hat{U} \times \boldsymbol{F}(x, y, z) = \boldsymbol{F}_{proj} \tag{5.15}$$

  The values in $\boldsymbol{F}_{proj}$ are then used to calculate the event rates of 8 different events that represent the 8 different interstitial spots where an oxygen ion could be generated.

  - The electric potential and temperature profiles within the interstitial grid can still be calculated using the regular FDM Poisson solver, but interaction between the regular grid and the interstitial grid requires unique entries in the Laplacian matrix (see also Section 4.3) since the diagonal edges connecting the interstitial elements and the oxide elements are irregular.

    Therefore, the entries of the Laplacian matrix at interstitial elements and the directly neighboring oxide elements are modified according to the connections in the 3D graph: the diagonal at an interstitial element is set to its degree (8) and the diagonals at neighboring elements are increased by 1 for every neighboring interstitial, as the degree increases. All off-diagonal entries of the Laplacian that represent the edges between interstitials and regular grids are set to -1.

    In this way, the influence of charge and power in the regular grid is communicated to the interstitial grid, and vice versa.

  Results: even though the interstitial grid inadvertently both generalized the model and increased its complexity, this implementation still did not exhibit the behavior as reported in the reference model. The problem of high local electric fields (see also Section 5.3) was made worse by the relatively smaller distance between the ions and the oxide, causing rampant generation of vacancies around the ions, which further prevented them from drifting away. It seems that this more physical interpretation of lattice element spacing is either too complex for its purpose, or not realistic enough.

- **Adapting the regular grid to a monoclinic structure.** After an inquiry session with two of the original co-authors of the model (Gennadi Bersuker and Luca Larcher) it was clarified that the material structure of the grid plays an important role in the calculation of the state change. Therefore the regular grid was adapted to a more general structure that represents the monoclinic crystal structure of substochiometric hafnium oxide [46] (HfO_x), the result of which is displayed in Figure 5.12. This again required an overhaul of the model implementation:

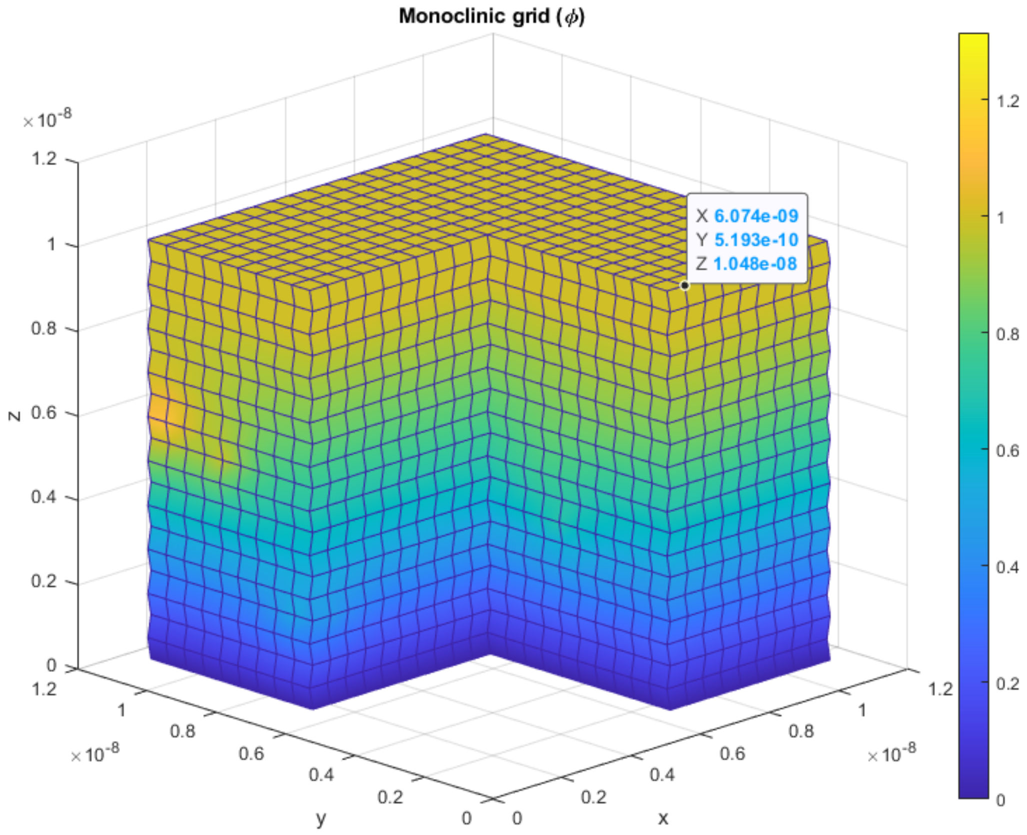


Figure 5.12: Example of a layout of the monoclinic grid, showing the values for the electric potential ϕ in a 3D block plot. The grid is distorted from the regular grid to better represent the crystal structure of hafnium oxide.

- The x,y,z coordinates of the `latticeElement`s are no longer trivial and instead are calculated based on four values that describe a parallellepiped unit of its most stable monoclinic structure: $a = 5.136$Å, $a = 5.136$Å, $a = 5.193$Å, $c = 5.371$Å and $\gamma = 99.63°$ [12].
- The calculation of the Laplacian matrix for the Poisson solver requires a complete overhaul: since none of the points are regularly spaced now, FDM no longer suffices. Instead, the Finite Element Method (FEM) is implemented [2] which can produce a Laplacian matrix for any arbitrary spaced set of points, if they are partitioned in tetrahedral elements (Note: "elements" here refers to the elements used in FEM, not of the lattice) defined by 4 points in the lattice. Figure 5.13 shows the difference between FDM and FEM.

  The subdivision of the monoclinic parallellepiped units in tetrahedral elements is trivial: any 3D region with 8 vertices can be filled with exactly 5 tetrahedra. In this way, the irregular lattice was partitioned into several tens of thousands of tetrahedral elements consisting of 4 `latticeElement` indices. With help of this subdivision, a Laplacian matrix can be constructed.

  Both for brevity, and because this implementation did not cause the model to function any better, the full explanation for implementing the FEM Poisson solver will be omitted. A description of an existing 3D implementation can be found in [2].

Results: just like the previous attempt, this implementation mostly provided further generalization of the model (by allowing arbitrary coordinates for the lattice elements). Unfortunately, adapting the crystal structure to a monoclinic structure did not significantly change the functioning of the model, as the same problems with the kMC engine still persisted.

The two above implementations - as well as the combination of both - were found not to significantly

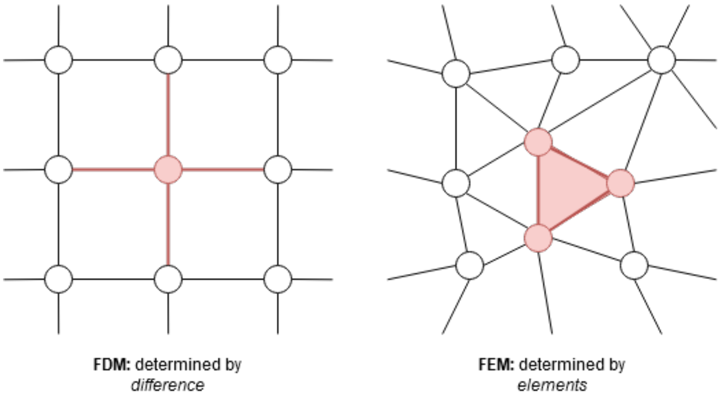

FDM: determined by *difference*
FEM: determined by *elements*

Figure 5.13: The difference between FDM and FEM, in a 2D case: FDM (left) uses the difference between two data points to determine the Laplacian matrix. FEM (right) uses triangular (tetrahedral in 3D) elements to determine the Laplacian matrix.

improve the functioning of this work's RRAM model. For the sake of simplicity, these implementations were reverted to reinstate the initial, more simple implementation, which was explained in Chapter 4.

Spacing and type of lattice elements: potential solutions

There is, however, still a possibility that the reported simple implementation is not representative of the reference model, which might be the cause of the problems listed in this chapter. The following list provides proposals concerning the spacing and type of lattice elements which can potentially cause the model to function properly, but were not yet attempted due to a lack of time. These are: using completely continuous coordinates, applying an atomistic structure, and neglecting oxygen ions.

- **Continuous coordinates.** Instead of initializing the full lattice grid as a structure of elements, only the elements that are relevant to the operation of the device (vacancies, ions, electrode-oxide interface) could be generated. This sparsified grid is no longer be bound to a defined structure, as is realistically the case within the amorphous region of an oxide grain boundary [90] (see also: Section 4.4). This implementation faces a couple of challenges:
  - For the existing elements, their arbitrary coordinates require the FEM implementation of the Poisson solver, raising the question how to partition the lattice into tetrahedral elements for any general distribution.
  - For the newly created elements, the electric field and temperature profiles should be defined *continuously*, as new elements are now allowed to generate anywhere in the model space. This poses another challenge for the implementation of the Poisson solver.
  - The calculation of event rates within the existing kMC engine becomes problematic: Monte Carlo methods can only work if there is a finite amount of state changes. If the electric field and temperature profiles are defined continuously, there are by definition an *infinite* amount of events.
- **Atomistic structure.** Define the lattice grid as a discrete set of points, but modify the types of the lattice elements to represent the low-level atomistic structure of hafnium oxide. The coordinates of the structure are still randomized within a continuous space, but oxygen vacancies can now only generate on pre-defined oxygen atom positions. A form of this implementation is seen in [90]. This implementation is also challenging:
  - The initial positions and connections of the atoms, in both the amorphous region in the GB and the crystalline zone in the grain, can not be randomized but must be based on knowledge about the nano-scale structure of hafnium oxide.
  - Interstitial positions can possibly become vague within the amorphous region: because the atoms are unorganized, the coordinates of the position between the elements can not be trivially deduced.
  - Similarly, the drift directions of an oxygen ion to other interstitial positions becomes hard to define, and the variable distance of the jump also starts to influence the event rates.

- **Neglect interstitial oxygen ions.** Simpler versions of OxRAM models do not consider the drifting of oxygen ions but instead the drifting of vacancies itself [1, 43, 82]. The generation and recombination events do not respectively create or destroy an vacancy-ion pair, but only a vacancy. This implementation would require relatively little adaption:

  - The drift rate calculation for vacancies needs to be adapted, since the reference model assumed that vacancies never drift [76]. The drift rate calculation for ions - and the `'O-2'` type of element - must be removed.

  While this implementation is simpler and might cause the model to function correctly, it eliminates the advantage of physical accuracy that was accounted for in the reference model.

The above proposals are neither proven nor disproven and may serve as pointers for future work. The spacing and type of lattice elements is not the only information missing though: the reference also does not clearly explain the charge of vacancies and ions.

Charge of vacancies and ions: attempted solutions

The charge of vacancies and ions plays a core role in the determining of the electric field. Padovani et al. [76] mention that, by taking the charge of these lattice elements into account while calculating the electric field profile, Coulomb repulsion and attraction are automatically accounted for. Therefore, using the proper values for the charge of vacancies and ions is crucial for proper operation of the kMC engine. Unfortunately, the strategy of calculating these charges is not explained in the reference work.

Once again, several attempts were made to fill in this gap of information. In short, these attempts are:

- Constant charges for both vacancies and ions.
- Reducing vacancy and ion charges by a constant.
- Reducing vacancy charge according to the electron flow rate of the paths.
- Reducing vacancy charge according to neighboring vacancies.
- Reducing vacancy charge according to their electron occupation probability.

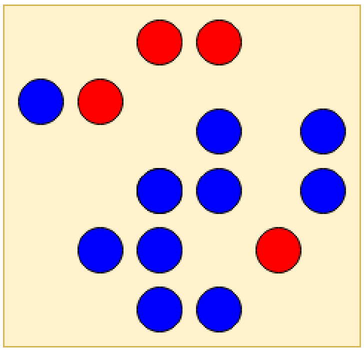


Figure 5.14: Constant charge lattice elements placed inside the oxide: positively charged V_{+2} vacancies (blue) and negatively charged O_{-2} ions (red).

- **Constant charges for both vacancies and ions.** The first attempt for determining the charge of vacancies and ions was a simple assumption based on the symbols used for vacancies (V_{+2}) and oxygen ions (O_{-2}) in the reference work. A vacancy was thus assumed to have a charge of +2e, and an ion was assumed to have a charge of -2e. A visualization of this implementation is displayed in Figure 5.14.

  Results: the main problem that arises from this assumption is that the distance between lattice elements is very small (3 Å, see also Section 4.6). Which causes the magnitude of the electric field to be very large, which causes a runaway generation of vacancies regardless of the applied voltage, as was explained earlier in Section 5.3. It can therefore be concluded that this assumption is incorrect.

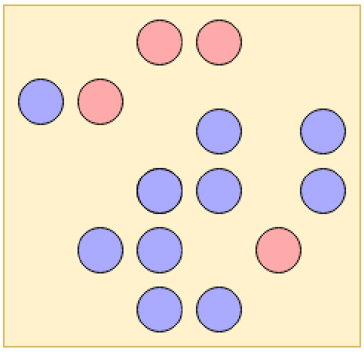


Figure 5.15: Reduced charge by a constant: positively charged V_{+2} vacancies (blue) and negatively charged O_{-2} ions (red). Less color saturation implies less charge.

- **Reducing vacancy and ion charges by a constant.** The simplest way of reducing the extremely large local field that exists next to the point charges, is to reduce the charge by a constant, as visualized in Figure 5.15. The charge density ρ was thus multiplied by a constant value $k_\rho < 1$, which was considered a fitting parameter. This effectively reduces the non-homogeneous electric potential field caused by the charges, while not touching the homogeneous field caused by the applied voltage on the electrodes.

  Results: Reducing the charges by a constant proved a successful solution to stop the runaway generation of vacancies next to other vacancies or ions. The value of the constant is, however, debatable, for two reasons:

  1. A constant reduction of charge is an assumption with no physical basis and is therefore difficult to obtain from literature.
  2. For low vacancy densities, e.g. before forming, the non-homogeneous electric field caused by charges is orders of magnitude smaller than for high vacancy densities, e.g. in LRS. Reducing the constant to account for the large non-homogeneous field for high vacancy densities, means that for low vacancy densities the electric field magnitude is too small to cross the generation activation energy barrier. Vice versa, a larger constant allows generation to occur in low vacancy densities, but does not reduce the large charge-induced electric field for high vacancy densities, which still causes the problem of runaway vacancy generation.

  It was therefore observed that, whereas reducing the vacancy and ion charge does solve the voltage-independent runaway vacancy generation problem, it can not be determined by a constant: a distinction must be made between high and low vacancy densities.

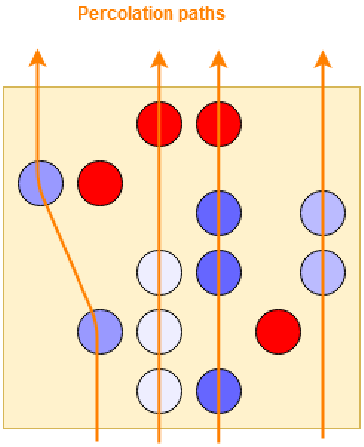


Figure 5.16: Percolation path based charge: positively charged V_{+2} vacancies (blue) and negatively charged O_{-2} ions (red). Less color saturation implies less charge. The charge of the vacancies is determined by the electron flow rate of the percolation path it is in.

- **Reducing vacancy charge according to the electron flow rate of the paths.** To find a physically-based source of charge reduction, the charge transport model was investigated (see also Section 4.4). The physical process behind the charge transport involves the capturing and emitting of an electron into a trap (Trap Assisted Tunneling). A logical conclusion from this would be that the more electrons flow through a vacancy, the more it receives negative charge. Research on the formation of vacancies [10, 27] confirms this, and even shows that the charge of a vacancy can become negative, due to many electrons being captured at once. This also explains how negatively charged oxygen ions can be repelled by vacancies.

  The TAT solver framework [59, 91], however, is based on the assumption that *no charge build-up* occurs inside a trap [59] and therefore does not mention any strategy of calculating the charge of a trap that has captured charge carriers. Determining the vacancy charge is therefore another gap of information that needs to be filled in.

  This was attempted by relating the charge of a trap, or vacancy, to the electron flow rate of the percolation path it exists in, as displayed in Figure 5.16. The electron flow rate of the percolation path R_{path} represents the amount of electrons flowing through the trap per second and is already calculated within the TAT solver to determine the current contribution of a single path. The charge of a trap in this path could therefore be determined by a linear equation:

$$q_i = +2\text{e} - k_q \cdot R_{path} \tag{5.16}$$

  where k_q will be used as a fitting parameter, and q_i is limited to -2e.

  Results: while this implementation does *partly* solve the problem of the previous implementation regarding the magnitude of the local electric field, it only does so when enough current flows through the device. Furthermore, even though a negative charge pushes ions away from the vacancy, the electric field magnitude is still very large, but only flipped. The consequence of this is that vacancies still generate rapidly around existing vacancies regardless of the applied voltage. This assumption is therefore assumed incorrect.

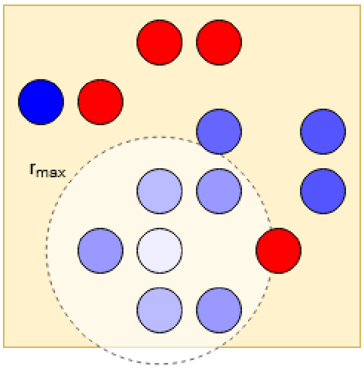


Figure 5.17: Metallization charge: positively charged V_{+2} vacancies (blue) and negatively charged O_{-2} ions (red). Less color saturation implies less charge. The charge of vacancies is determined by the amount of vacancies in its vicinity.

- **Reducing vacancy charge according to neighboring vacancies.** Another physically-based source of vacancy charge reduction is metallization. Microscopic models have shown that on a structural level, oxygen vacancies between hafnium atoms lose their charge once they cluster together [27, 109]. This represents the formation of a metallic, substochiometric region HfO_x, meaning that there are less than 2 oxygen atoms per 1 hafnium atom. In a structural sense, this means that for large vacancy densities, hafnium atoms rather connect to each other than leave holes in the lattice. This causes a reduction in charge as there are less disconnected hafnium atoms.

  The effect of metallization may be modelled by a simple linear function of the distance from a vacancy, which reduces the charge of the vacancies around it. The effect is displayed in Figure 5.17. For every vacancy element i, all surrounding vacancy elements j (within a distance limit $r_{ij} < r_{max}$) multiply their charge as:

$$q_j = q_j \cdot \left(1 - k_m \left(1 - \frac{r_{ij}}{r_{max}}\right)\right) \tag{5.17}$$

where q_j is the charge of element j, and $k_m \in [0,1]$ is a factor representing the strength of metallization. The higher k_m, the more vacancy density influences charge. Therefore, it is a fitting parameter.

Results: for higher vacancy densities, the non-homogeneous electric field caused by the large cluster of charges is succesfully reduced by applying Equation 5.17. For lower vacancy densities however, metallization can not reduce charge and the electric field magnitude is still too large. Combining the metallization implementation with a (relatively low) constant charge reduction implementation however, also keeps the electric field magnitude for low vacancies within acceptable bounds.

There is however still a problem: large clusters of oxygen ions can also form in the model, similarly causing a very high local electric field magnitued. Reducing their charge according to their density to prevent this is not represented by any physical process, as oxygen ions exist separately in interstitial positions and are not inclined to connect. Another solution must therefore be found to reduce the charge of the ions in the model.

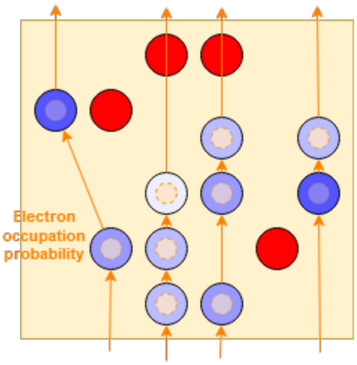


Figure 5.18: Electron occupation probability based charge: positively charged V_{+2} vacancies (blue) and negatively charged O_{-2} ions (red). Less color saturation implies less charge. The charge of vacancies is determined by the probability of an electron existing inside the vacancy.

- **Reducing vacancy charge according to their electron occupation probability.** The final implementation attempt was based on equations from Larcher [59], which form the basis of calculating the electron capture and emission rates per trap for the TAT solver (see also Section 4.4). These equations involve the probability that an electron exists in a trap, the electron occupation probability:

$$f_{t,i} = \tau_{e,i} \cdot \left(\tau_{c,i} + \tau_{e,i}\right)^{-1} \tag{5.18}$$

where $\tau_{c,i}$ and $\tau_{e,i}$ are the electron capture and emission times, respectively. $f_{t,i}$ represents a probability, which means that $f_{t,i} \in [0,1]$.

Note that this probability is different from the Fermi-Dirac occupation probability $f_i(E)$, which shows the probability of an electron existing on a certain energy level E: rather, $f_{t,i}$ shows the probability of an electron existing inside the vacancy.

The capture and emission times are already calculated in the TAT solver framework for every trap (or vacancy) in the lattice and therefore the electron occupation probability for every trap can be deduced using Equation 5.18. The charge of a vacancy may be directly linked to the electron occupation probability by:

$$q_i = q_0 - q \cdot f_{t,i} \tag{5.19}$$

where q_i is the charge of vacancy i, q_0 is the unoccupied charge of vacancy i and $-q$ is the charge of an electron. The effect of this is visualized in Figure 5.18.

The problem of Equation 5.19 is that it is still unknown what the charge is of an *unoccupied* vacancy (q_0). For this, again, the results of microscopic models [27] was used to find that the type of vacancy that is most prone to clustering is that of a single positive charge: V_+. It was therefore assumed that unoccupied vacancies have a charge of $q_0 = +1e$ and the charge of vacancy i was thus determined as:

$$q_i = q \cdot \left(1 - f_{t,i}\right) \tag{5.20}$$

Results: even though this implementation is mostly based on the references, it doesn't always reduce the charge of vacancies enough. The electron occupation probability can be anywhere between 0 and 1, and thus vacancies can have any charge between +0e and +1e, which may still cause a local electric field that is too large. Furthermore, not all oxygen vacancies inside the lattice have a charge of +1e [27]: possible charges include any option from -2e to +2e. The assumption that $q_0 = +1e$ is therefore not completely correct. Furthermore, oxygen ions are not considered by the TAT solver. These still have a debatable charge which can cause too large electric field magnitudes regardless of electrons flowing through the vacancies.

Of the above list, all implementations were attempted, but eventually the final implementation produced the most realistic results. In the future, combinations of these attempts may be considered, or another, new way of determining the charge of vacancies and ions. Vacancy and ion charge appears to be an important aspect of RRAM switching, since many processes are field-driven. Therefore it is peculiar how little research exists on this subject.

In conclusion, there are still too many issues with the full model for it to accurate reproduce the electrical characteristics of an RRAM device. However, pointers were provided in this section to improve the model so that in the future it may be used to more accurately investigate the effect of defects on OxRAM forming and switching.

For this work, only the Poisson solver and the TAT solver were validated. Therefore, even though the time-dependent processes of the reference model can not be reproduced, the charge transport model can still be used to investigate the effects of defects on the resistance of steady-state RRAM. The following chapter will thus use the Poisson solver, the TAT solver and the defect injector to research the effect of RRAM defects in LRS and HRS.

6

Defect analysis

In this chapter, the model described in Chapter 4 and partially validated in Chapter 5 will be applied to investigate the effects of defects on the conductivity of RRAM pre-forming and in LRS. First, the general experimental setup will be presented. Then the four defect types listed in Section 2.4 are injected into the lattice grid: vacancy density variations due to the presence or absence of grain boundaries, oxide thickness variation, electrode roughness, and impurities that contaminate the device.

6.1. Experimental setup

For every defect, a range of values is chosen defining a property of the defect, and a voltage sweep is performed from 0 to 3V, while calculating the current using the charge transport model described in Section 4.4. To account for statistical fluctuation and outliers, $N_{sim} = 100$ simulations will be performed for every experiment. All graphs show the mean of all simulations, together with a shading implying the range of the simulation results. The dimensions and parameters of a nominal device are listed in Table 6.1.

Table 6.1: A list of the parameters describing the nominal, defect-free device.

Parameter	Description	Nominal value
N_x	Total x-dimension of model space.	30 elements (9 nm)
N_y	Total y-dimension of model space.	30 elements (9 nm)
N_x	Total z-dimension of model space.	40 elements (12 nm)
N_{el}	Part of z-dimension representing electrodes.	4 elements (1.2 nm)
N_V	Vacancy density outside grain boundary.	$3.0 \times 10^{19}\,\mathrm{cm}^{-3}$
$N_{V,GB}$	Vacancy density inside grain boundary.	$2.1 \times 10^{21}\,\mathrm{cm}^{-3}$
r_{GB}	Radius of grain boundary.	4 nm
T_{amb}	Ambient temperature	300 K

The rest of all properties, together with physical constants that do not change between simulations, can be found in the full properties list in Appendix A.

For experiments in LRS, a filament had to be inserted into the oxide manually, because the kMC engine governing the state change of the model was unable to function correctly (see also Section 5.3 and Section 5.4). This means that the filament shape must also be pre-determined manually. 3D images of conductive filaments in OxRAM were shown to have an hourglass shape [15], as displayed in Figure 6.1a. Therefore, the shape of the filament was described by an hourglass-shaped region:

$$d_c(i,j) < r(k) \tag{6.1}$$

where $d_c(i,j)$ is the index distance from the center projected on a plane parallel to the electrodes, and $r(k)$ is the index radius of the filament according to:

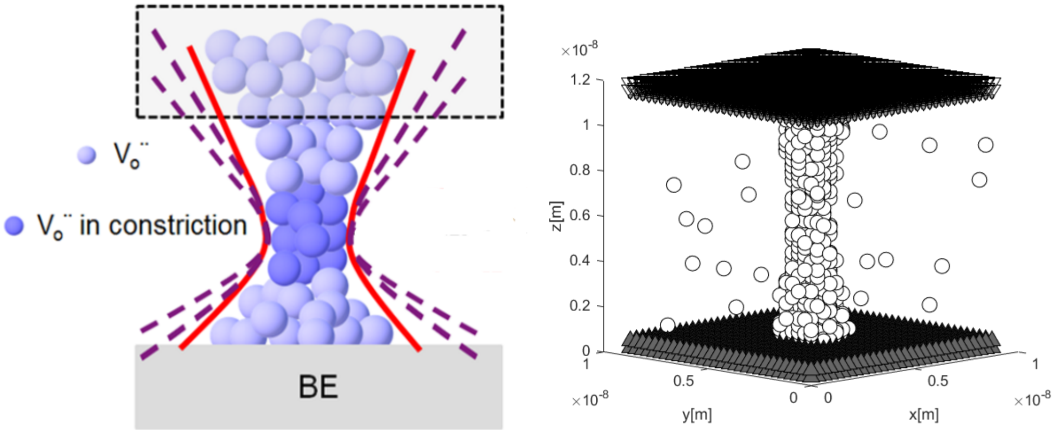
$$d_c(i,j) = \sqrt{(i - N_x/2)^2 + \left(j - N_y/2\right)^2} \tag{6.2}$$

$$r(k) = 0.04 \cdot (k - N_z/2)^2 + 4 \tag{6.3}$$

where i, j and k are the element indices in respectively x, y and z directions, and N_x, N_y and N_z are the amount of elements in those directions. Within this region, the probability of a lattice element being a vacancy ($P_V \in [0,1]$) is determined by:

$$P_V(i,j,k) = 1 - 0.6 \cdot \frac{d_c(i,j)}{r(k)} \tag{6.4}$$

The hourglass-shaped filament resulting from Equation 6.1 and Equation 6.4 is displayed in Figure 6.1b.



(a) Visualization of the shape of a conductive filament in RRAM, obtained by 3D imaging. Adapted from [15].

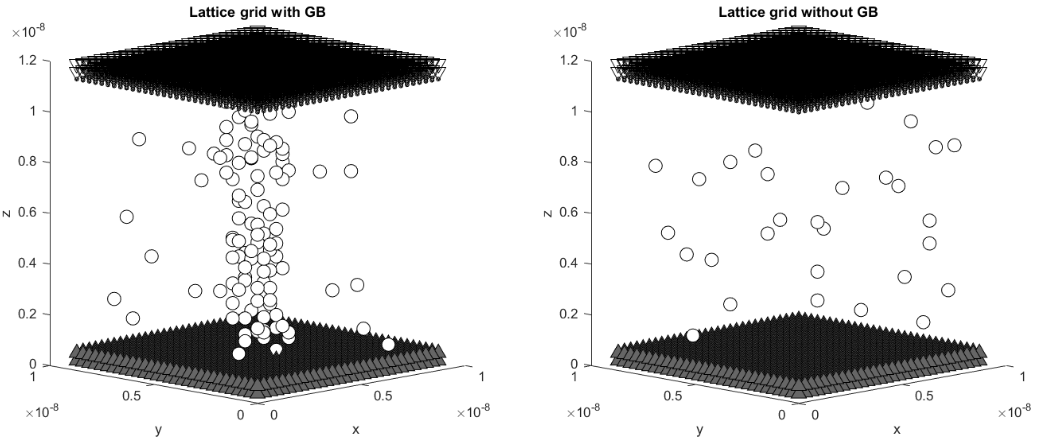
(b) A lattice grid modeling an RRAM device in LRS. with a manually inserted conductive filament shape determined by Equation 6.1 and Equation 6.4.

Figure 6.1: Manually inserting a conductive filament in the oxide of the lattice grid.

The experimental setup for each defect is described below.

Setup for vacancy density fluctuations

The main cause for unexpected vacancy density fluctuations is the presence or absence of grain boundaries (GBs) [9, 78], therefore, to investigate the effect of vacancy density fluctuations, the experimental setup in Section 6.1 was modified to remove the GB in the middle of the oxide. Two examples of the lattice grid with and without the GB are displayed in Figure 6.2.



(a) With a GB.

(b) Without a GB.

Figure 6.2: Examples of the lattice grid, generated by the lattice builder described in Section 4.6, with and without a GB.

Oxide thickness variation

To investigate the effects of oxide thickness variation, the total z dimension of the model space (N_z) was reduced and increased. The general experimental setup from Section 6.1 was used, with N_z taking the values:

- $N_z = 20$ ($t_{ox} = 4.8\,\text{nm}$);
- $N_z = 30$ ($t_{ox} = 7.8\,\text{nm}$);
- $N_z = 40$ ($t_{ox} = 10.8\,\text{nm}$);
- $N_z = 50$ ($t_{ox} = 13.8\,\text{nm}$).

Electrode roughness

The effects of electrode roughness were investigated by appending a uniformly random amount of electrode elements to the electrode-oxide interfaces (see also Section 4.6). Experiments were performed for four different values of roughness, defined by the maximum amount of electrode elements which can randomly be appended:

- roughness = 1 ($\sigma_{el} = 0.3\,\text{nm}$);
- roughness = 2 ($\sigma_{el} = 0.6\,\text{nm}$);
- roughness = 3 ($\sigma_{el} = 0.9\,\text{nm}$);
- roughness = 4 ($\sigma_{el} = 1.2\,\text{nm}$).

Impurities

Impurities are contaminations of the oxide by a foreign material and can have many different shapes and sizes. For the purpose of researching their effect, two types of impurities are injected into the lattice grid: a blocking impurity and a conducting impurity. The blocking impurity is represented by an air bubble in the middle of the oxide, obstructing both the grain boundary and the conductive filament. The conducting impurity is represented by a bump in the lower electrode, locally reducing the oxide thickness. These impurities are described by four parameters: type, width, height, and position (see also Section 4.6). The parameters are listed in Table 6.2. The resulting impurity defects are displayed in Figure 6.3.

Parameter	Blocking impurity	Conducting impurity
Type	`'air'`	`'lower'` (electrode)
Width	4.0 nm	3.0 nm
Height	3.0 nm	8.0 nm
Position	[4.5 nm, 4.5 nm, 8.0 nm]	[3.0 nm, 6.0 nm, 0.0 nm]

Table 6.2: The parameters used in the injection of the blocking and conducting impurities.

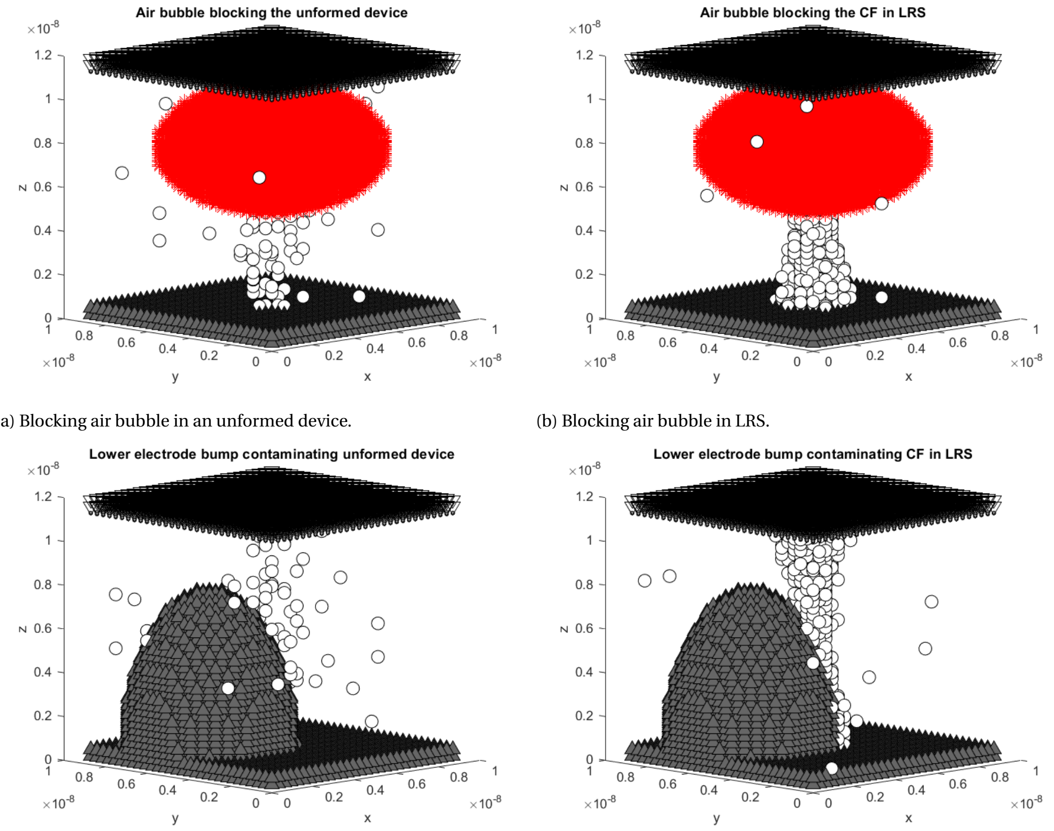
6.2. Results

In this section, the results of the experimental setup in Section 6.1 are presented for the four selected defect types: vacancy density fluctuations, oxide thickness variation, electrode roughness and impurities.

Vacancy density fluctuations

The results of the simulations analysing the effect of GBs affecting the vacancy density are displayed in Figure 6.4.

Figure 6.4a shows that the presence or absence of a GB does have an effect on the RRAM device in pre-forming conditions. Without a GB, the device has a larger resistance, especially for lower voltages. The effect on conductivity does not appear to be significant for higher voltages, apart from reducing the *variation* of conductivity, as displayed by the red and blue shaded areas. This is probably due to the GB acting as a priority conductive path, which guides the percolation paths of the electrons from the cathode to the anode (see also Section 4.4).


Air bubble blocking the unformed device
Air bubble blocking the CF in LRS
Lower electrode bump contaminating unformed device
Lower electrode bump contaminating CF in LRS
×10^-8
z
y
x
0
0.2
0.4
0.6
0.8
1
1.2

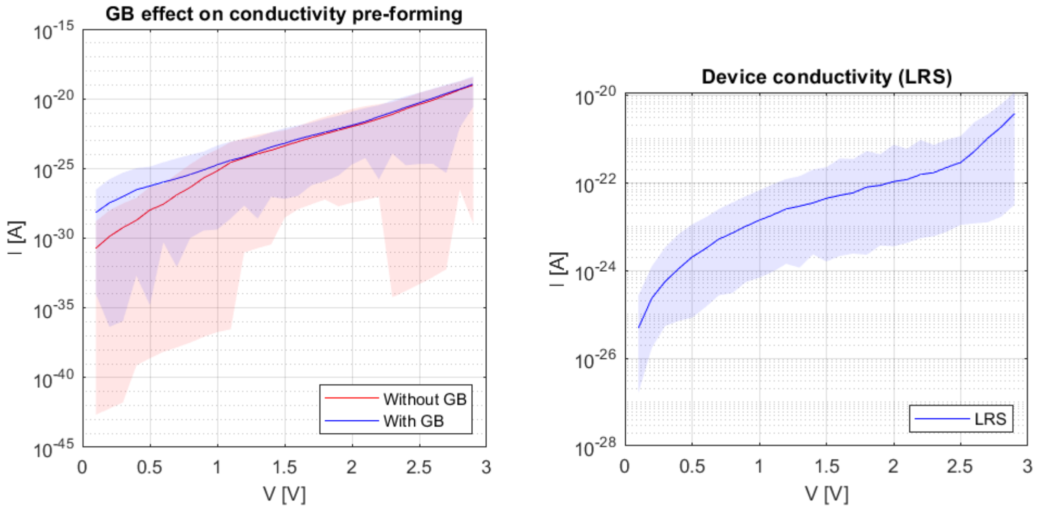
(a) Blocking air bubble in an unformed device.

(b) Blocking air bubble in LRS.

(c) Conducting electrode bump in an unformed device.

(d) Conducting electrode bump in LRS.

Figure 6.3: Four examples of lattice grids with injected impurities, generated by the implementation described in Section 4.6 based on the parameters listed in Table 6.2.



(a) In an unformed device.

(b) In LRS.

Figure 6.4: The current through the device for voltages from 0 to 3V, in the case of a GB existing in the oxide (blue) and no GB existing in the oxide (red). The range of the simulation results are displayed by shadings in the respective colors. Since the effect of GBs on LRS can not be investigated by this model, subfigure (b) shows just the defect-free conductivity in LRS (blue).

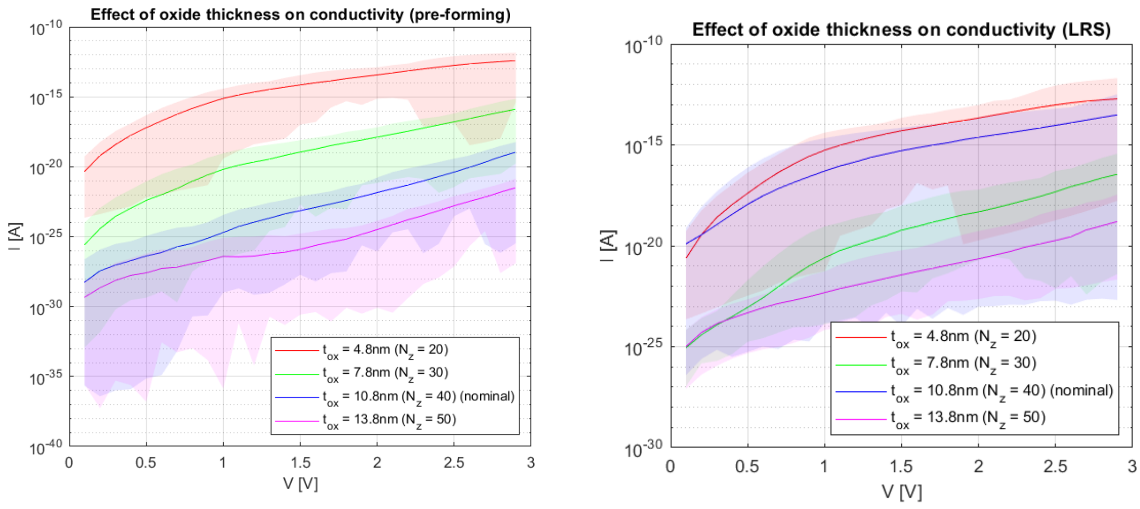
What is not displayed in Figure 6.4a is the effect that a GB has on the generation of new vacancies. Previous models and measurements have shown that conductive filaments favor GBs to form [9, 10, 76, 90]. Because this model's kMC engine is unable to function correctly (see also Section 4.5 and Section 5.3), this can not be verified by this model, but it is clear from previous knowledge the effect of GB outside of its conductive properties is significant

Figure 6.4b shows the conductivity of the device described in Section 6.1 with the manually inserted conductive filament, displayed in Figure 6.1 and described in Equation 6.1 and Equation 6.4. It appears that the conductivity of the filament is only slightly larger than that of the unformed device. This might be due to the oxide being too thick (10.8 nm) to function properly as an RRAM filament. For higher voltages (around 3V), the unformed device appears to conduct better than the formed device. The exact reason for this is unknown, but might be an error in the implementation of the charge transport model (Section 4.4 and Section 5.2).

Another possible reason for the perceived loss of conductivity in LRS is that the charge transport model is not able to realistically model the defect subband that is created for a sufficiently high density of vacancies. The reference model states that this is possible using the same TAT framework as explained in Section 4.4 [59, 76, 91]: as the mutual distance between the traps decreases, their capture radii overlap, reducing the potential barrier and increasing the tunneling probability to 1. From these results, this barrier reducing effect appears to be insufficient to model the drift current. Instead, a different charge transport model may have to be implemented for the device in LRS, as the drift current is not modeled correctly by the TAT solver. Other macroscopic models already do this [1, 43, 82].

Oxide thickness variation

The results of the four simulation sweeps in an unformed device and in LRS are displayed in Figure 6.5.



(a) In an unformed device.

(b) In LRS.

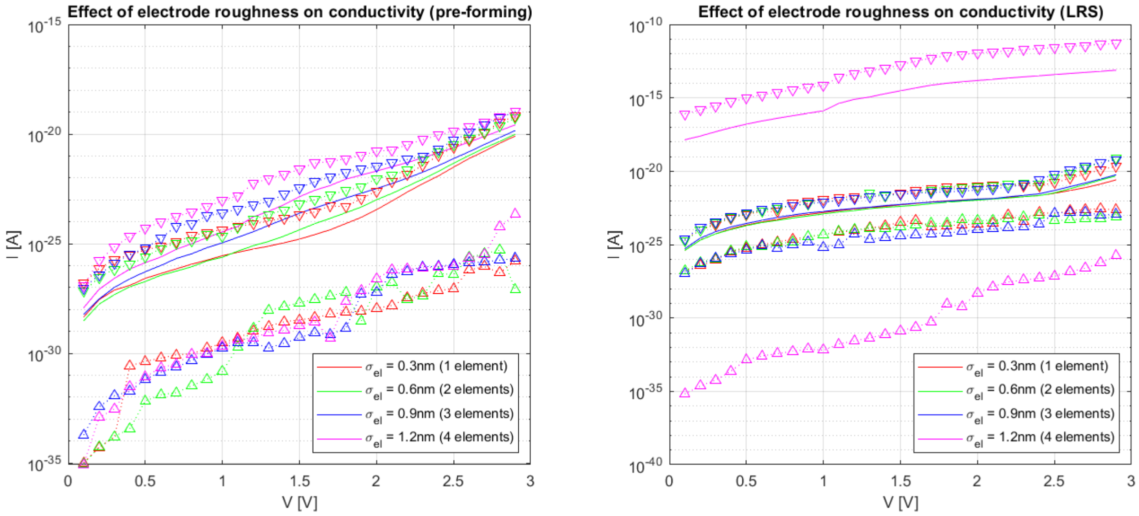
Figure 6.5: The current through the device for voltages from 0 to 3V, for four different oxide thicknesses. The means of each set of 100 simulations are displayed by lines, and the range of all simulation results is displayed by shadings in the corresponding colors.

In an unformed device, the oxide thickness has a clear effect on the conductivity, as shown in Figure 6.5a. For thin oxides, a variation of a few nanometer can already modify the current by five orders of magnitude, as can be seen from the results of t_{ox} = 7.8 nm (green) and t_{ox} = 4.8 nm (red). The oxide thickness has an insignificant effect on the variation of the device conductivity, because the statistical spread of the current does not change as the thickness varies. Furthermore, a thinner oxide also increases the electric field inside the device, subsequently increasing the rate of vacancy generation. This can cause the filament to form earlier than expected.

Figure 6.5b shows the effect of varying oxide thickness on a device in LRS. Once again, it appears that the charge transport model can not successfully reproduce LRS drift current. However, the current does visibly increase for all thicknesses. The effect of oxide thickness variation is less significant than in an unformed device, most likely because the majority of the current is caused by the conductive filament instead of direct tunneling.

Electrode roughness

The results of the four simulation sweeps in an unformed device and in LRS are displayed in Figure 6.6.



(a) In an unformed device.

(b) In LRS.

Figure 6.6: The current through the device for voltages from 0 to 3V, for four different values of electrode roughness. The means of each set of 100 simulations are displayed by lines, and the minimum and maximum values are displayed by triangles in the corresponding colors.

Figure 6.6a shows that in an unformed device, electrode roughness has a relatively small effect on the conductivity. A higher roughness causes a slightly larger current, because the randomly appended electron elements locally decrease the oxide thickness. The ends of the appended electron elements also cause a local peak in the electric field, as is displayed in Figure 6.7. These local peaks will accelerate forming, which can cause an uncontrolled hard dielectric breakdown to occur [32, 79].

Figure 6.6b shows that the current increases only slightly after manually inserting a conductive filament, due to the issues posed earlier. For roughnesses of 1, 2 and 3 elements however, the variation of current visibly reduces, indicating that the local variations of oxide thickness no longer have a dominant effect on the conductivity as the percolation paths prefer the filament. A roughness of 4 elements shows a very wide spread and a very high mean current. The reason for this is unknown, but most likely caused by an error in the charge transport model, which may require further research.

Impurities

Figure 6.8 shows the results of measuring the current through the RRAM device when the two types of impurities are injected, for an unformed device and in LRS.

Figure 6.8a shows that the conducting impurity greatly increases the conductivity of the device, as was expected. The effect is much more significant than that of the blocking impurity, mostly because the resistance of the device is already quite large before the device is formed. Another significant effect of the conducting impurity is not directly related to the device current, but to the electric field it causes, as is displayed in Figure 6.9. The magnitude of the electric field at the tip of the bump is significantly higher than in the rest of the oxide, which causes the forming of the conductive filament to prefer the impurity. The high field magnitude will also speed up generation of vacancies, causing unreliable forming and possibly a hard dielectric breakdown.

Figure 6.8b shows, once again, that the charge transport model is not able to successfully reproduce the drift current of an RRAM device in LRS. It does show, however, that the impurities have an effect on the spread of the different simulations. Nominally (green) the spread of current decreases in LRS, due to percolation paths favoring the conductive filament. In the case of the conducting impurity, the current variation changes little, because the majority of current is caused by the locally thin oxide at the electrode bump. In the case of the blocking impurity, the variation does decrease, because here the majority of current is still caused by the conductive filament.

This concludes the analysis of defects in an RRAM device, using the charge transport model presented in Section 4.4 and the experimental setup Section 6.1. In conclusion, the effects of these defects are clearly visible and unique, but may require further validation.

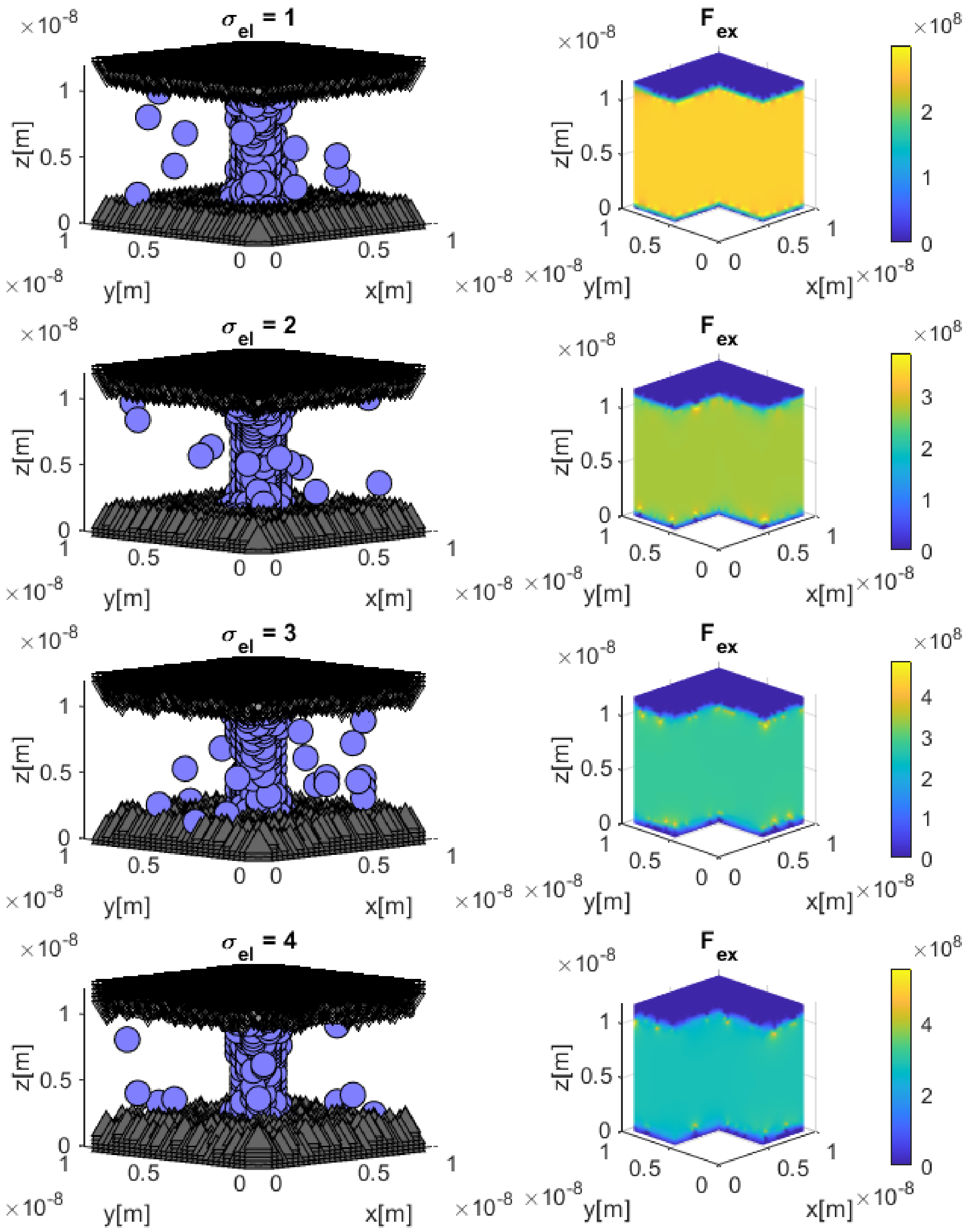
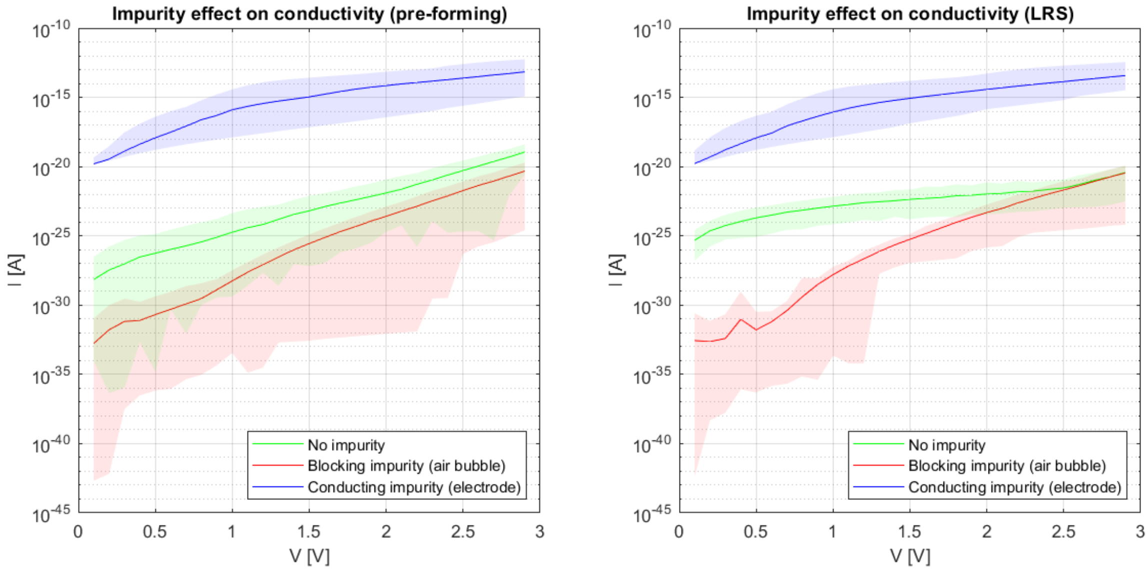


Figure 6.7: Examples of the four electrode roughness values producing the results displayed in Figure 6.6. The left four figures show the layout of the lattice grid for the indicated electrode roughness. Electrodes are displayed as triangles, and vacancies as blue circles. The right four figures show the respective electric fields, excluding the contribution of vacancies.



(a) In an unformed device.

(b) In LRS.

Figure 6.8: The current through the device for voltages from 0 to 3V, for two types of impurities. The means of each set of 100 simulations are displayed by lines, and the range of all simulation results is displayed by shadings in the corresponding colors.

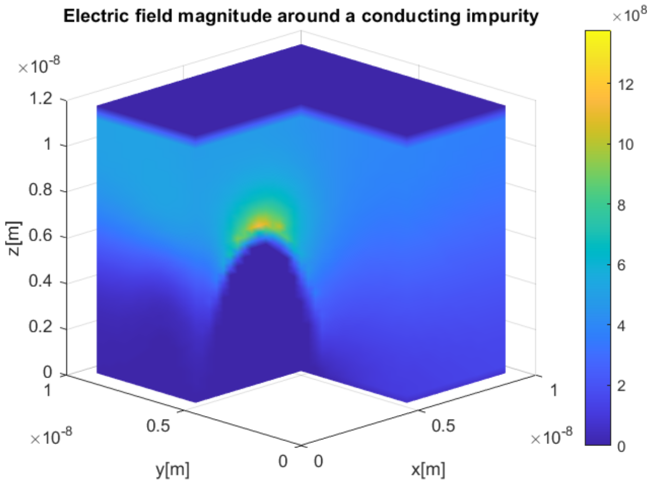


Figure 6.9: The electric field in a lattice grid with a conducting impurity, excluding the electric field caused by vacancies.

7
Discussion

In this chapter, the results of this work are critically observed, but mostly the process of getting these results is discussed. This is because, even though the primary focus of this work was to elaborate an existing physical model with the ability to simulate defects, it eventually became more of an example of the importance of reproducability of scientific publications.

Unexpected work load

Originally this work was planned to focus more on the inclusion of defects instead of the physical operation of defect-free RRAM, since the latter has already been extensively researched, as described in Chapter 3. Thus, a physical, macroscopic model had to be selected which appeared to have been documented extensively enough to rebuild it with relative ease, after which the main focus could turn to the defects. The comparison of existing macroscopic models revealed that the work by Padovani et al. [76] was the best candidate for this defect-free reference model.

However, after several months of attempting to recreate the model, it seemed that it requires an unexpectedly enormous effort to understand all of the concepts and theories involved in the physical processes of RRAM operation, especially for a CE master student and outsider. Even though many of the concepts in the reference work were properly sourced by references and explained, successfully recreating the model still required specific knowledge about quantum physics, solid-state physics, material science, electromagnetics, applied mathematics, and computer science. Obtaining this knowledge and effectively applying it may have been too ambitious for a master thesis.

Reproducability issues

Many of these fields only had to be explored and understood because of missing information, assumptions and unclear definitions in the reference work. The most troublesome are the parameters of the kMC engine (see also Section 5.3), the implementation details of the lattice grid (Section 5.4), and the charge value of the vacancies and ions (see alsoSection 5.4). As was explained in Chapter 5, even with the provided information and knowledge on the relevant subjects, it was still not possible to reproduce the results that were published in the reference work.

Therefore, apart from providing an in-depth analysis of the physical processes used in the reference model, this thesis also raises questions about its reproducability. A scientific research has little value if it is not reproducable. It is difficult to judge the reproducability of any scientific work: although the physical parts of the reference model were clearly documented and sourced, the specific details about the implementation were either assumed or obfuscated. Either way, this work has shown that without these details, the reference model is too complex to reproduce correctly.

Attempts to fill in information

Still, however, several attempts were made to fill in the gaps that were left by this missing information. The entire MATLAB structure of the model, connecting the physical modules described in the reference in an efficient way, had to be constructed from scratch (see also Section 4.2). The WKB tunneling probability calculator (Section 4.4), the Tsu-Esaki direct tunneling calculator (Section 4.4), and the Poisson equation solver (Section 4.3) are examples of core parts of the reference model which were mentioned without a reference and were also constructed from scratch.

Value of this work

The results of this work, presented in Chapter 5 and Chapter 6, have been shown and described to not be completely realistic. The reasons behind these errors has been explained and a solution hypothesized. Therefore, this work may not be a viable reference for the construction of a more compact defect model, as was its motivation (see also Chapter 1). Instead it may provide a basis and source of knowledge for an improvement that may produce more realistic results. To aid this, the information missing in the reference work has been extensively explained in Chapter 4 and its components were both validated and invalidated in Chapter 5.

Future work

Thus, in the future, this model could be adapted and improved to be able to successfully model the full resistive switching process of RRAM, as well as the effects of defects on its operation. The source code is provided publicly [47] and the explanation of its implementation can be found in this work. The model is already able to inject defects into a lattice grid and calculate the current through a RRAM device in HRS. The irregularities of defects are taken into account by the combination of element-based electrodes and custom-made Poisson equation solver. As such, this work will serve a basis for further improvement of the model, so that it can eventually be used for its intended purpose, which is to simulate the full forming and switching process of a single RRAM device, on a physical level, with the inclusion of defects.

The power of this model is its versatility: because of its physical nature, any shape and size of defect can be injected into the lattice grid and its effects observed. If it could be made to work correctly and verified, new types of defects could be added to it next to the types discussed in this model Section 2.4. Eventually, the results of the model can be compacted into a SPICE model, and tests can be developed for any type of defect.

8

Conclusion

This work has laid the basis for a macroscopic, physical-based RRAM defect model, which in the future can be used for advanced, in-depth defect analysis. It has also provided an extensive explanation of all physical processes associated with RRAM operations.

Motivated by the lack of physical basis in RRAM defect modeling, this work aimed to connect the physical processes that drive the generation and recombination of a conductive filament in the RRAM device, to its electrical characteristic of resistive switching. Background information was provided about the function of memory, emerging non-volatile memory technologies and the operating principles of RRAM devices. Also, the manufacturing process of RRAM devices was investigated, and its most common defects were described.

After providing the necessary background information to understand the principles of RRAM defects, the state of the art of RRAM defect modeling was explored in more detail. With the goal of selecting a defect-free model to elaborate with the inclusion of defects, existing defect-free models were compared according to their usability. The macroscopic physics-based model by Padovani et al. was chosen to be used as the reference model.

The structure of the reference model was divided in general modules and three physical modules: the Poisson equation solver, the charge transport model and the Monte Carlo state change engine. The novel addition to the model is the defect injector, which modifies the initial state of the model to include one or more of four defect types: vacancy density fluctuations, oxide thickness variation, electrode roughness and impurities.

The model was then validated, and it was found that missing information and unmentioned assumptions made it impossible to reproduce the results of the reference model correctly, in part due to the confusing description of the referenced Monte Carlo engine. The charge transport model and Poisson equation solver were, however, sufficiently validated and could be used to simulate the effect of defects on the conductivity of a steady-state RRAM device. The results of these steady-state measurements showed the effect of defects on the conductivity of the device and its internal electric field. The effects were used to explain the possibly catastrophic consequences of the defects.

Finally, a collection of all issues concerning the reproducability of the reference model and the validity of this work's results was presented in a discussion. This work was presented as a basis for further improvement of the model in the future. It could eventually be used for the development of defect characterization and test development, based on low-level physical simulations of any arbitrary type of defect.

Bibliography

[1] Elhameh Abbaspour, Stephan Menzel, Alexander Hardtdegen, Susanne Hoffmann-Eifert, and Christoph Jungemann. KMC Simulation of the Electroforming, Set and Reset Processes in Redox-Based Resistive Switching Devices. *IEEE Transactions on Nanotechnology*, 17(6):1181–1188, 2018. ISSN 1536125X. doi: 10.1109/TNANO.2018.2867904.

[2] Bedreddine Ainseba, Mostafa Bendahmane, and L Alejandro. Solving the Laplacian Equation in 3D using Finite Element Method in C # for Structural Analysis. *HAL Archives Ouvertes*, pages 1–6, 2012.

[3] D. T. Wang B. Jacob, S. W. Ng. *Memory systems: Cache, DRAM, Disk.* Morgan Kaufmann, 2008. ISBN 978-0-12-379751-3.

[4] I. G. Baek, M. S. Lee, S. Seo, M. J. Lee, D. H. Seo, D. S. Suh, J. C. Park, S. O. Park, H. S. Kim, I. K. Yoo, U. In Chung, and J. T. Moon. Highly scalable non-volatile resistive memory using simple binary oxide driven by asymmetric unipolar voltage pulses. *Technical Digest - International Electron Devices Meeting, IEDM*, pages 587–590, 2004. ISSN 01631918. doi: 10.1109/iedm.2004.1419228.

[5] Keith Baker, Guido Gronthoud, Maurice Lousberg, Ivo Schanstra, and Charles Hawkins. Defect-based delay testing of resistive vias-contacts a critical evaluation. In *IEEE International Test Conference (TC)*, pages 467–476. IEEE, 1999. doi: 10.1109/test.1999.805769.

[6] A. Beck, J. G. Bednorz, Ch Gerber, C. Rossel, and D. Widmer. Reproducible switching effect in thin oxide films for memory applications. *Applied Physics Letters*, 77(1):139–141, 7 2000. ISSN 00036951. doi: 10.1063/1.126902. URL `http://aip.scitation.org/doi/10.1063/1.126902`.

[7] K Beckmann, J Holt, W Olin-Ammentorp, Z Alamgir, J Van Nostrand, and N C Cady. The effect of reactive ion etch (RIE) process conditions on ReRAM device performance. *Semicond. Sci. Technol.*, 32(9): 95013, 9 2017. doi: 10.1088/1361-6641/aa7eed. URL `http://stacks.iop.org/0268-1242/32/i=9/a=095013?key=crossref.c28efe83f8f4a3e411ae7b040097cbdc`.

[8] G. Bersuker, D. C. Gilmer, D. Veksler, P. Kirsch, L. Vandelli, A. Padovani, L. Larcher, K. McKenna, A. Shluger, V. Iglesias, M. Porti, and M. Nafría. Metal oxide resistive memory switching mechanism based on conductive filament properties. *Journal of Applied Physics*, 110(12):124518, 12 2011. ISSN 00218979. doi: 10.1063/1.3671565. URL `http://aip.scitation.org/doi/10.1063/1.3671565`.

[9] G. Bersuker, J. Yum, L. Vandelli, A. Padovani, L. Larcher, V. Iglesias, M. Porti, M. Nafría, K. McKenna, A. Shluger, P. Kirsch, and R. Jammy. Grain boundary-driven leakage path formation in HfO2 dielectrics. In *Solid-State Electronics*, volume 65-66, pages 146–150. Pergamon, 11 2011. doi: 10.1016/j.sse.2011.06.031.

[10] G. Bersuker, D.C. Gilmer, and D. Veksler. *Metal-oxide resistive random access memory (RRAM) technology: Material and operation details and ramifications.* Elsevier Ltd., 2019. ISBN 9780081025840. doi: 10.1016/b978-0-08-102584-0.00002-4.

[11] Roberto Bez, Emilio Camerlenghi, Alberto Modelli, and Angelo Visconti. Introduction to flash memory, 2003. ISSN 00189219.

[12] Samuel R. Bradley, Alexander L. Shluger, and Gennadi Bersuker. Electron-injection-assisted generation of oxygen vacancies in monoclinic HfO2. *Physical Review Applied*, 4(6):1–7, 2015. ISSN 23317019. doi: 10.1103/PhysRevApplied.4.064008.

[13] Arne Brataas, Andrew D. Kent, and Hideo Ohno. Current-induced torques in magnetic materials, 4 2012. ISSN 14764660. URL `www.nature.com/naturematerials`.

[14] A. Callegari, E. Cartier, M. Gribelyuk, H. F. Okorn-Schmidt, and T. Zabel. Physical and electrical characterization of Hafnium oxide and Hafnium silicate sputtered films. *Journal of Applied Physics*, 90(12): 6466–6475, 12 2001. ISSN 00218979. doi: 10.1063/1.1417991.

[15] Umberto Celano, Ludovic Goux, Robin Degraeve, Andrea Fantini, Olivier Richard, Hugo Bender, Malgorzata Jurczak, and Wilfried Vandervorst. Imaging the three-dimensional conductive channel in filamentary-based oxide resistive switching memory. *Nano Letters*, 15(12):7970–7975, 12 2015. ISSN 15306992. doi: 10.1021/acs.nanolett.5b03078. URL `https://pubs.acs.org/sharingguidelines`.

[16] Arjun Chaudhuri and Krishnendu Chakrabarty. Analysis of Process Variations, Defects, and Design-Induced Coupling in Memristors. In *ITC 2018*, pages 1–10. IEEE, 2018.

[17] An Chen. Utilizing the variability of resistive random access memory to implement reconfigurable physical unclonable functions. *IEEE Electron Device Letters*, 36(2):138–140, 2 2015. ISSN 07413106. doi: 10.1109/LED.2014.2385870.

[18] An Chen and Ming-Ren Lin. Variability of resistive switching memories and its impact on crossbar array performance. In *2011 Int. Reliab. Phys. Symp.*, pages MY.7.1–MY.7.4. IEEE, 4 2011. ISBN 978-1-4244-9113-1. doi: 10.1109/IRPS.2011.5784590. URL `http://ieeexplore.ieee.org/document/5784590/`.

[19] B. Chen, Y. Lu, B. Gao, Y. H. Fu, F. F. Zhang, P. Huang, Y. S. Chen, L. F. Liu, X. Y. Liu, J. F. Kang, Y. Y. Wang, Z. Fang, H. Y. Yu, X. Li, X. P. Wang, N. Singh, G. Q. Lo, and D. L. Kwong. Physical mechanisms of endurance degradation in TMO-RRAM. In *Technical Digest - International Electron Devices Meeting, IEDM*, 2011. ISBN 9781457705052. doi: 10.1109/IEDM.2011.6131539.

[20] Pai Yu Chen and Shimeng Yu. Compact Modeling of RRAM Devices and Its Applications in 1T1R and 1S1R Array Design. *IEEE Transactions on Electron Devices*, 62(12):4022–4028, 12 2015. ISSN 00189383. doi: 10.1109/TED.2015.2492421.

[21] Pang-Shiu Chen, Yu-Sheng Chen, Heng-Yuan Lee, Tai-Yuan Wu, Pei-Yi Gu, Frederick Chen, and Ming-Jinn Tsai. Impact of Flattened TiN Electrode on the Memory Performance of HfO2 Based Resistive Memory. *Electrochem. Solid-State Lett.*, 15(4):H136, 1 2012. ISSN 10990062. doi: 10.1149/2.001205esl. URL `http://esl.ecsdl.org/cgi/doi/10.1149/2.001205esl`.

[22] Yu-Sheng Chen, Heng-Yuan Lee, Pang-Shiu Chen, Wei-Su Chen, Kan-Hsueh Tsai, Pei-Yi Gu, Tai-Yuan Wu, Chen-Han Tsai, S Z Rahaman, Yu-De Lin, Frederick Chen, Ming-Jinn Tsai, and Tzu-Kun Ku. Novel Defects-Trapping ${\backslashrm TaO}_{\backslashrm X}/{\backslashrm HfO}_{\backslashrm X}$ RRAM With Reliable Self-Compliance, High Nonlinearity, and Ultra-Low Current. *IEEE Electron Device Lett.*, 35(2):202–204, 2 2014. ISSN 0741-3106. doi: 10.1109/LED.2013.2294375. URL `http://ieeexplore.ieee.org/document/6689291/`.

[23] Zhengyu Chen, Hai Zhou, and Jie Gu. A RRAM-based coarse grain reconfigurable array for neural network accelerators. In *2018 IEEE SOI-3D-Subthreshold Microelectronics Technology Unified Conference, S3S 2018*. Institute of Electrical and Electronics Engineers Inc., 2 2019. ISBN 9781538676264. doi: 10.1109/S3S.2018.8640182.

[24] Leon O. Chua. Memristor—The Missing Circuit Element. *IEEE Transactions on Circuit Theory*, 18(5): 507–519, 1971. ISSN 00189324. doi: 10.1109/TCT.1971.1083337.

[25] Jason Cong and Bingjun Xiao. FPGA-RPI: A novel fpga architecture with rram-based programmable interconnects. *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, 22(4):864–877, 2014. ISSN 10638210. doi: 10.1109/TVLSI.2013.2259512.

[26] J. L. Hennessy D. A. Patterson. *Computer Organization and Design*. Morgan Kaufmann, 2011.

[27] Dan Duncan, Blanka Magyari-Köpe, and Yoshio Nishi. Filament-induced anisotropic oxygen vacancy diffusion and charge trapping effects in hafnium oxide RRAM. *IEEE Electron Device Letters*, 37(4):400–403, 4 2016. ISSN 07413106. doi: 10.1109/LED.2016.2524450.

[28] Dan Duncan, Blanka Magyari-Köpe, and Yoshio Nishi. Properties of Dopants in HfOx for Improving the Performance of Nonvolatile Memory. *Physical Review Applied*, 7(3):1–10, 2017. ISSN 23317019. doi: 10.1103/PhysRevApplied.7.034020.

[29] A. Fantini, L. Goux, R. Degraeve, D. J. Wouters, N. Raghavan, G. Kar, A. Belmonte, Y. Y. Chen, B. Govoreanu, and M. Jurczak. Intrinsic switching variability in HfO2 RRAM. In *2013 5th IEEE International Memory Workshop, IMW 2013*, pages 30–33, 2013. ISBN 9781467361675. doi: 10.1109/IMW.2013.6582090.

[30] Moritz Fieback, Student Member, Guilherme Cardoso Medeiros, and Student Member. Framework for Defect and Fault Analysis in RRAM. *ACM Journal on Emerging Technologies in Computing Systems*, XX (X):1–7, 2019.

[31] Moritz Fieback, Mottaqiallah Taouil, and Said Hamdioui. Testing Resistive Memories: Where are We and What is Missing? In *Proceedings - International Test Conference*, volume 2018-Octob. Institute of Electrical and Electronics Engineers Inc., 1 2019. ISBN 9781538683828. doi: 10.1109/TEST.2018.8624895.

[32] Moritz Fieback, Lizhou Wu, Guilherme Cardoso Medeiros, Hassen Aziza, Siddharth Rao, Erik Jan Marinissen, Mottaqiallah Taouil, and Said Hamdioui. Device-aware test: A new test approach towards DPPB level. In *Proceedings - International Test Conference*, volume 2019-Novem. Institute of Electrical and Electronics Engineers Inc., 11 2019. ISBN 9781728148236. doi: 10.1109/ITC44170.2019.9000134.

[33] A. S. Foster, F. Lopez Gejo, A. L. Shluger, and R. M. Nieminen. Vacancy and interstitial defects in hafnia. *Physical Review B - Condensed Matter and Materials Physics*, 65(17):1741171–17411713, 5 2002. ISSN 01631829. doi: 10.1103/PhysRevB.65.174117. URL `https://journals.aps.org/prb/abstract/10.1103/PhysRevB.65.174117`.

[34] Daichi Fujiki, Scott Mahlke, and Reetuparna Das. In-memory data parallel processor. In *ACM SIGPLAN Notices*, volume 53, pages 1–14, New York, NY, USA, 3 2018. Association for Computing Machinery. ISBN 9781450349116. doi: 10.1145/3173162.3173171. URL `https://dl.acm.org/doi/10.1145/3173162.3173171`.

[35] Andreas Gehring. *Simulation of Tunneling in Semiconductor Devices.* PhD thesis, Technischen Universität Wien, Messerschmidtgasse 2/2/7 A-1180 Wien, Österreich, November 2003. URL `https://www.iue.tuwien.ac.at/phd/gehring/diss.html`.

[36] D. C. Gilmer, G. Bersuker, H. Y. Park, C. Park, B. Butcher, W. Wang, P. D. Kirsch, and R. Jammy. Effects of RRAM stack configuration on forming voltage and current overshoot. *2011 3rd IEEE International Memory Workshop, IMW 2011*, 2011. doi: 10.1109/IMW.2011.5873225.

[37] Alessandro Grossi, Damian Walczyk, Cristian Zambelli, Enrique Miranda, Piero Olivo, Valeriy Stikanov, Alessandro Feriani, Jordi Sune, Gunter Schoof, Rolf Kraemer, Bernd Tillack, Alexander Fox, Thomas Schroeder, Christian Wenger, and Christian Walczyk. Impact of Intercell and Intracell Variability on Forming and Switching Parameters in RRAM Arrays. *IEEE Transactions on Electron Devices*, 62(8):2502–2509, 8 2015. ISSN 00189383. doi: 10.1109/TED.2015.2442412.

[38] Alessandro Grossi, Cristian Zambelli, Piero Olivo, Enrique Miranda, Valeriy Stikanov, Thomas Schroeder, Christian Walczyk, and Christian Wenger. Relationship among Current Fluctuations during Forming, Cell-To-Cell Variability and Reliability in RRAM Arrays. In *IMW 2015*, pages 1–4. IEEE, 2015. ISBN 978-1-4673-6931-2. doi: 10.1109/IMW.2015.7150303. URL `http://ieeexplore.ieee.org/document/7150303/`.

[39] Alessandro Grossi, Eduardo Perez, Cristian Zambelli, Piero Olivo, and Christian Wenger. Performance and reliability comparison of 1T-1R RRAM arrays with amorphous and polycrystalline HfO2. In *EUROSOI 2016*, pages 80–83. IEEE, 1 2016. ISBN 978-1-4673-8609-8. doi: 10.1109/ULIS.2016.7440057. URL `http://ieeexplore.ieee.org/document/7440057/`.

[40] Alessandro Grossi, Eduardo Perez, Cristian Zambelli, Piero Olivo, and Christian Wenger. Performance and reliability comparison of 1T-1R RRAM arrays with amorphous and polycrystalline HfO2. In *2016 Joint International EUROSOI Workshop and International Conference on Ultimate Integration on Silicon, EUROSOI-ULIS 2016*, pages 80–83. Institute of Electrical and Electronics Engineers Inc., 3 2016. ISBN 9781467386098. doi: 10.1109/ULIS.2016.7440057.

[41] Alessandro Grossi, Cristian Zambelli, Piero Olivo, Enrique Miranda, Valeriy Stikanov, Christian Walczyk, and Christian Wenger. Electrical characterization and modeling of pulse-based forming techniques in RRAM arrays. *Solid. State. Electron.*, 115:17–25, 1 2016. doi: 10.1016/J.SSE.2015.10.003. URL https://www.sciencedirect.com/science/article/pii/S0038110115002828.

[42] Bochen Guan and Jing Li. A compact model for RRAM including random telegraph noise. In *IEEE International Reliability Physics Symposium Proceedings*, volume 2016-Septe, pages MY51–MY54. Institute of Electrical and Electronics Engineers Inc., 9 2016. ISBN 9781467391368. doi: 10.1109/IRPS.2016.7574621.

[43] Ximeng Guan, Shimeng Yu, and H. S.Philip Wong. On the switching parameter variation of metal-oxide RRAM - Part I: Physical modeling and simulation methodology. *IEEE Transactions on Electron Devices*, 59(4):1172–1182, 4 2012. ISSN 00189383. doi: 10.1109/TED.2012.2184545.

[44] Nor Zaidi Haron and Said Hamdioui. On defect oriented testing for hybrid CMOS/memristor memory. In *Proceedings of the Asian Test Symposium*, pages 353–358, 2011. ISBN 9780769545837. doi: 10.1109/ATS.2011.66.

[45] T. W. Hickmott. Low-frequency negative resistance in thin anodic oxide films. *Journal of Applied Physics*, 33(9):2669–2682, 9 1962. ISSN 00218979. doi: 10.1063/1.1702530. URL http://aip.scitation.org/doi/10.1063/1.1702530.

[46] Erwin Hildebrandt, Jose Kurian, Mathis M. Mller, Thomas Schroeder, Hans Joachim Kleebe, and Lambert Alff. Controlled oxygen vacancy induced p-type conductivity in HfO 2-x thin films. *Applied Physics Letters*, 99(11), 9 2011. ISSN 00036951. doi: 10.1063/1.3637603.

[47] Tijs Hol. RRAM defect model Gitlab repository, 2020. URL https://gitlab.tudelft.nl/mcrfieback/rram-defect-model.

[48] Miao Hu, Hai Li, Yiran Chen, Xiaobin Wang, and Robinson E. Pino. Geometry variations analysis of TiO2 thin-film and spintronic memristors. In *Proceedings of the Asia and South Pacific Design Automation Conference, ASP-DAC*, pages 25–30, 2011. ISBN 9781424475155. doi: 10.1109/ASPDAC.2011.5722193.

[49] Jun Woo Jang, Sangsu Park, Yoon Ha Jeong, and Hyunsang Hwang. ReRAM-based synaptic device for neuromorphic computing. In *Proceedings - IEEE International Symposium on Circuits and Systems*, pages 1054–1057. Institute of Electrical and Electronics Engineers Inc., 2014. ISBN 9781479934324. doi: 10.1109/ISCAS.2014.6865320.

[50] Sachhidh Kannan, Jeyavijayan Rajendran, Ramesh Karri, and Ozgur Sinanoglu. Sneak-path testing of crossbar-based nonvolatile random access memories. *IEEE Transactions on Nanotechnology*, 12(3):413–426, 2013. ISSN 1536125X. doi: 10.1109/TNANO.2013.2253329.

[51] P. Karmakar, G. F. Liu, and J. A. Yarmoff. Sputtering-induced vacancy cluster formation on Ti O2 (110). *Physical Review B - Condensed Matter and Materials Physics*, 76(19):1–4, 2007. ISSN 10980121. doi: 10.1103/PhysRevB.76.193410.

[52] A. V. Khvalkovskiy, D. Apalkov, S. Watts, R. Chepulskii, R. S. Beach, A. Ong, X. Tang, A. Driskill-Smith, W. H. Butler, P. B. Visscher, D. Lottis, E. Chen, V. Nikitin, and M. Krounbi. Basic principles of STT-MRAM cell operation in memory arrays. *Journal of Physics D: Applied Physics*, 46(13), 2013. ISSN 00223727. doi: 10.1088/0022-3727/46/13/139601.

[53] Kuk Hwan Kim, Siddharth Gaba, Dana Wheeler, Jose M. Cruz-Albrecht, Tahir Hussain, Narayan Srinivasa, and Wei Lu. A functional hybrid memristor crossbar-array/CMOS system for data storage and neuromorphic applications. *Nano Letters*, 12(1):389–395, 1 2012. ISSN 15306984. doi: 10.1021/nl203687n.

[54] Y S Kim, M Y Sung, Y H Lee, B K Ju, and M H Oh. The Influence of Surface Roughness on the Electric Conduction Process in Amorphous Ta[sub 2]O[sub 5] Thin Films. *J. Electrochem. Soc.*, 146(9):3398, 9 1999. ISSN 00134651. doi: 10.1149/1.1392485. URL http://jes.ecsdl.org/cgi/doi/10.1149/1.1392485.

[55] K Kuhn, C Kenyon, A Kornfeld, M Liu Intel Technology ..., and Undefined 2008. Managing Process Variation in Intel's 45nm CMOS Technology. *search.ebscohost.com*, 12 (2):93–110, 2008. URL `http://search.ebscohost.com/login.aspx?direct=true&profile=ehost&scope=site&authtype=crawler&jrnl=1535864X&asa=Y&AN=32925829&h=M9yr0P3SzLBztoz0aVctkZcZDsNeEY7o1u0Uls9SQXyrrTqETtiHJ4uRNoQxPwwApcGkish5Rdz1eH4W%2FfCw7w%3D%3D&crl=c`.

[56] Kelin J. Kuhn, Martin D. Giles, David Becher, Pramod Kolar, Avner Kornfeld, Roza Kotlyar, Sean T. Ma, Atul Maheshwari, and Sivakumar Mudanai. Process technology variation. *IEEE Transactions on Electron Devices*, 58(8):2197–2208, 8 2011. ISSN 00189383. doi: 10.1109/TED.2011.2121913.

[57] Mario Lanza, H. S.Philip Wong, Eric Pop, Daniele Ielmini, Dimitri Strukov, Brian C. Regan, Luca Larcher, Marco A. Villena, J. Joshua Yang, Ludovic Goux, Attilio Belmonte, Yuchao Yang, Francesco M. Puglisi, Jinfeng Kang, Blanka Magyari-Köpe, Eilam Yalon, Anthony Kenyon, Mark Buckwell, Adnan Mehonic, Alexander Shluger, Haitong Li, Tuo Hung Hou, Boris Hudec, Deji Akinwande, Ruijing Ge, Stefano Ambrogio, Juan B. Roldan, Enrique Miranda, Jordi Suñe, Kin Leong Pey, Xing Wu, Nagarajan Raghavan, Ernest Wu, Wei D. Lu, Gabriele Navarro, Weidong Zhang, Huaqiang Wu, Runwei Li, Alexander Holleitner, Ursula Wurstbauer, Max C. Lemme, Ming Liu, Shibing Long, Qi Liu, Hangbing Lv, Andrea Padovani, Paolo Pavan, Ilia Valov, Xu Jing, Tingting Han, Kaichen Zhu, Shaochuan Chen, Fei Hui, and Yuanyuan Shi. Recommended Methods to Study Resistive Switching Devices. *Advanced Electronic Materials*, 5(1): 1–28, 2019. ISSN 2199160X. doi: 10.1002/aelm.201800143.

[58] L. Larcher, A. Padovani, O. Pirrotta, L. Vandelli, and G. Bersuker. Microscopic understanding and modeling of HfO2 RRAM device physics. In *Technical Digest - International Electron Devices Meeting, IEDM*, 2012. ISBN 9781467348706. doi: 10.1109/IEDM.2012.6479077.

[59] Luca Larcher. Statistical simulation of leakage currents in MOS and flash memory devices with a new multiphonon trap-assisted tunneling model. *IEEE Transactions on Electron Devices*, 50(5):1246–1253, 5 2003. ISSN 00189383. doi: 10.1109/TED.2003.813236.

[60] Muath Abu Lebdeh, Uljana Reinsalu, Hoang Anh Du Nguyen, Stephan Wong, and Said Hamdioui. Memristive device based circuits for computation-in-memory architectures. In *Proceedings - IEEE International Symposium on Circuits and Systems*, volume 2019-May. Institute of Electrical and Electronics Engineers Inc., 2019. ISBN 9781728103976. doi: 10.1109/ISCAS.2019.8702542.

[61] Seok Hee Lee. Technology scaling challenges and opportunities of memory devices. In *Technical Digest - International Electron Devices Meeting, IEDM*, pages 1–1. Institute of Electrical and Electronics Engineers Inc., 1 2017. ISBN 9781509039012. doi: 10.1109/IEDM.2016.7838026.

[62] Haitong Li, Peng Huang, Bin Gao, Bing Chen, Xiaoyan Liu, and Jinfeng Kang. A SPICE model of resistive random access memory for large-scale memory array simulation. *IEEE Electron Device Letters*, 35(2): 211–213, 2 2014. ISSN 07413106. doi: 10.1109/LED.2013.2293354.

[63] Y. T. Li, S. B. Long, H. B. Lv, Q. Liu, M. Wang, H. W. Xie, K. W. Zhang, X. Y. Yang, and M. Liu. Novel self-compliance Bipolar 1D1R memory device for high-density RRAM application. In *2013 5th IEEE International Memory Workshop, IMW 2013*, pages 184–187, 2013. ISBN 9781467361675. doi: 10.1109/IMW.2013.6582130.

[64] Chun Li Lo, Mei Chin Chen, Jiun Jia Huang, and Tuo Hung Hou. On the potential of CRS, 1D1R, and 1S1R crossbar RRAM for storage-class memory. In *2013 International Symposium on VLSI Technology, Systems and Application, VLSI-TSA 2013*, 2013. ISBN 9781467330817. doi: 10.1109/VLSI-TSA.2013.6545588.

[65] Paolo Lorenzi, Rosario Rao, and Fernanda Irrera. Forming kinetics in HfO2-Based RRAM cells. *IEEE Transactions on Electron Devices*, 60(1):438–443, 2013. ISSN 00189383. doi: 10.1109/TED.2012.2227324.

[66] Yang Yang Ma, Ya Li Song, Pei Liu, Yin Yin Lin, Xiao Hui Huang, Qing Tian Zou, and Jin Gang Wu. Endurance enhancement by soft forming algorithm on AlOx/WOyresistive switching memory array. *IEEE Electron Device Letters*, 35(12):1230–1232, 2014. ISSN 07413106. doi: 10.1109/LED.2014.2360511.

[67] Keith McKenna and Alexander Shluger. The interaction of oxygen vacancies with grain boundaries in monoclinic HfO2. *Applied Physics Letters*, 95(22), 2009. ISSN 00036951. doi: 10.1063/1.3271184.

[68] Jagan Singh Meena, Simon Min Sze, Umesh Chand, and Tseung Yuen Tseng. Overview of emerging nonvolatile memory technologies. *Nanoscale Research Letters*, 9(1):1–33, 9 2014. ISSN 1556276X. doi: 10.1186/1556-276X-9-526. URL https://link.springer.com/articles/10.1186/1556-276X-9-526https://link.springer.com/article/10.1186/1556-276X-9-526.

[69] Stephan Menzel, Philip Kaupmann, and Rainer Waser. Understanding filamentary growth in electrochemical metallization memory cells using kinetic Monte Carlo simulations †. *Nanoscale*, 7:12673, 2015. doi: 10.1039/c5nr02258d. URL www.rsc.org/nanoscale.

[70] Vishwani D. Agrawal Micheal L. Bushnell. *Essentials of Electronic Testing*. Kluwer Academic Publishers, 2002.

[71] Donald A. Neamen. *Semiconductor Physics and Devices*. McGraw Hill, 2003. ISBN 0-07-232107-5.

[72] Clement Nguyen, Carlo Cagli, Elisa Vianello, Alain Persico, Gabriel Molas, Gilles Reimbold, Quentin Rafhay, and Gerard Ghibaudo. Advanced 1T1R test vehicle for RRAM nanosecond-range switching-time resolution and reliability assessment. In *IEEE International Integrated Reliability Workshop Final Report*, volume 2016-March, pages 17–20. Institute of Electrical and Electronics Engineers Inc., 3 2016. ISBN 9781467373968. doi: 10.1109/IIRW.2015.7437059.

[73] Takeki Ninomiya, Koji Katayama, Shunsaku Muraoka, Ryutaro Yasuhara, Takumi Mikawa, and Zhiqiang Wei. Conductive filament expansion in TaOx bipolar resistive random access memory during pulse cycling. *Japanese Journal of Applied Physics*, 52(11 PART 1), 11 2013. ISSN 00214922. doi: 10.7567/JJAP.52.114201.

[74] Takeki Ninomiya, Zhigiang Wei, Shusaku Muraoka, Ryutaro Yasuhara, Koji Katayama, and Takeshi Takagi. Conductive filament scaling of TaOx bipolar ReRAM for improving data retention under low operation current. *IEEE Transactions on Electron Devices*, 60(4):1384–1389, 2013. ISSN 00189383. doi: 10.1109/TED.2013.2248157.

[75] Andrea Padovani, Luca Larcher, Gennadi Bersuker, and Paolo Pavan. Charge transport and degradation in HfO2 and HfOx dielectrics. *IEEE Electron Device Letters*, 34(5):680–682, 2013. ISSN 07413106. doi: 10.1109/LED.2013.2251602.

[76] Andrea Padovani, Luca Larcher, Onofrio Pirrotta, Luca Vandelli, and Gennadi Bersuker. Microscopic modeling of HfOx RRAM operations: From forming to switching. *IEEE Transactions on Electron Devices*, 62(6):1998–2006, 6 2015. ISSN 00189383. doi: 10.1109/TED.2015.2418114.

[77] E Perez, L Bondesan, A Grossi, C Zambelli, P Olivo, and Ch. Wenger. Assessing the forming temperature role on amorphous and polycrystalline HfO2-based 4 kbit RRAM arrays performance. *Microelectron. Eng.*, 178:1–4, 6 2017. ISSN 0167-9317. doi: 10.1016/J.MEE.2017.04.003. URL https://www.sciencedirect.com/science/article/pii/S0167931717301314?via%3Dihub.

[78] Onofrio Pirrotta, Luca Larcher, Mario Lanza, Andrea Padovani, Marc Porti, Montserrat Nafría, and Gennadi Bersuker. Leakage current through the poly-crystalline HfO2: Trap densities at grains and grain boundaries. *Journal of Applied Physics*, 114(13):134503, 10 2013. ISSN 00218979. doi: 10.1063/1.4823854. URL http://aip.scitation.org/doi/10.1063/1.4823854.

[79] Peyman Pouyan, Esteve Amat, Said Hamdioui, and Antonio Rubio. RRAM variability and its mitigation schemes. In *Proceedings - 2016 26th International Workshop on Power and Timing Modeling, Optimization and Simulation, PATMOS 2016*, pages 141–146. Institute of Electrical and Electronics Engineers Inc., 1 2017. ISBN 9781509007332. doi: 10.1109/PATMOS.2016.7833679.

[80] Nagarajan Raghavan. Performance and reliability trade-offs for high-\$$\kappa$\$ RRAM. *Microelectron. Reliab.*, 54(9-10):2253–2257, 9 2014. doi: 10.1016/J.MICROREL.2014.07.135. URL https://www.sciencedirect.com/science/article/pii/S0026271414003370.

[81] Nagarajan Raghavan, Michel Bosman, Daniel D Frey, and Kin Leong Pey. Variability model for forming process in oxygen vacancy modulated high-$κ$ based resistive switching memory devices. *Microelectron. Reliab.*, 54(9-10):2266–2271, 9 2014. doi: 10.1016/J.MICROREL.2014.07.118. URL `https://www.sciencedirect.com/science/article/pii/S0026271414003205#bi0005`.

[82] Toufik Sadi, Liping Wang, Louis Gerrer, Vihar Georgiev, and Asen Asenov. Self-consistent physical modeling of SiOx-based RRAM structures. In *18th International Workshop on Computational Electronics, IWCE 2015*. Institute of Electrical and Electronics Engineers Inc., 10 2015. ISBN 9780692515235. doi: 10.1109/IWCE.2015.7301981.

[83] Robert R. Schaller. Moore's law: past, present, and future. *IEEE Spectrum*, 34(6):52–55, 6 1997. ISSN 00189235. doi: 10.1109/6.591665.

[84] Jaehyun Seo, Sangheon Lee, Kwangmin Kim, Sooeun Lee, Hyunsang Hwang, and Byungsub Kim. Automatic ReRAM SPICE model generation from empirical data for fast reram-circuit coevaluation. *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, 25(6):1821–1830, 6 2017. ISSN 10638210. doi: 10.1109/TVLSI.2017.2655730.

[85] Sangho Shin, Kyungmin Kim, and Sung Mo Kang. Memristor applications for programmable analog ICs. *IEEE Transactions on Nanotechnology*, 10(2):266–274, 3 2011. ISSN 1536125X. doi: 10.1109/TNANO.2009.2038610.

[86] Dmitri B. Strukov, Gregory S. Snider, Duncan R. Stewart, and R. Stanley Williams. The missing memristor found. *Nature*, 453(7191):80–83, 2008. ISSN 00280836. doi: 10.1038/nature06932.

[87] M. Stucchi, M. Bamal, and K. Maex. Impact of line-edge roughness on resistance and capacitance of scaled interconnects. *Microelectronic Engineering*, 84(11):2733–2737, 11 2007. ISSN 01679317. doi: 10.1016/j.mee.2007.05.038.

[88] R. Tsu and L. Esaki. Tunneling in a finite superlattice. *Applied Physics Letters*, 22(11):562–564, 6 1973. ISSN 00036951. doi: 10.1063/1.1654509. URL `http://aip.scitation.org/doi/10.1063/1.1654509`.

[89] L. Vandelli, A. Padovani, L. Larcher, R. G. Southwick, W. B. Knowlton, and G. Bersuker. Modeling temperature dependency (6 - 400K) of the leakage current through the SiO2/high-K Stacks. In *2010 Proceedings of the European Solid State Device Research Conference, ESSDERC 2010*, pages 388–391, 2010. ISBN 9781424466610. doi: 10.1109/ESSDERC.2010.5618204.

[90] L. Vandelli, A. Padovani, L. Larcher, G. Broglia, G. Ori, M. Montorsi, G. Bersuker, and P. Pavan. Comprehensive physical modeling of forming and switching operations in HfO 2 RRAM devices. In *Technical Digest - International Electron Devices Meeting, IEDM*, 2011. ISBN 9781457705052. doi: 10.1109/IEDM.2011.6131574.

[91] L. Vandelli, A. Padovani, L. Larcher, R. G. Southwick, W. B. Knowlton, and G. Bersuker. A physical model of the temperature dependence of the current through SiO2/HfO2 stacks. *IEEE Transactions on Electron Devices*, 58(9):2878–2887, 9 2011. ISSN 00189383. doi: 10.1109/TED.2011.2158825.

[92] Luca Vandelli, Andrea Padovani, Luca Larcher, and Gennadi Bersuker. Microscopic modeling of electrical stress-induced breakdown in poly-crystalline hafnium oxide dielectrics. *IEEE Transactions on Electron Devices*, 60(5):1754–1762, 2013. ISSN 00189383. doi: 10.1109/TED.2013.2255104.

[93] Elena Ioana Vatajelu, Peyman Pouyan, and Said Hamdioui. State of the art and challenges for test and reliability of emerging nonvolatile resistive memories. In *International Journal of Circuit Theory and Applications*, volume 46, 2018. doi: 10.1002/cta.2418.

[94] E. Vianello, O. Thomas, G. Molas, O. Turkyilmaz, N. Jovanović, D. Garbin, G. Palma, M. Alayan, C. Nguyen, J. Coignus, B. Giraud, T. Benoist, M. Reyboz, A. Toffoli, C. Charpin, F. Clermidy, and L. Perniola. Resistive Memories for Ultra-Low-Power embedded computing design. In *Technical Digest - International Electron Devices Meeting, IEDM*, volume 2015-Febru, pages 1–6. Institute of Electrical and Electronics Engineers Inc., 2 2015. ISBN 9781479980017. doi: 10.1109/IEDM.2014.7046995.

[95] Ioannis Vourkas and Georgios Ch Sirakoulis. Emerging memristor-based logic circuit design approaches: A review, 7 2016. ISSN 1531636X.

[96] Maurice V Wilkes. The Memory Gap and the Future of High Performance Memories. *AT&T Research Laboratories*, 1990.

[97] H.-S. P. Wong, H. Y. Lee, S. Yu, Y. S. Chen, P.-S. Chen, B. Lee, F. T. Chen, and M.-J. Tsai. Metal–oxide RRAM. *Proceedings of the IEEE*, 100(6):1951–1970, 2012. URL `http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=6193402`.

[98] H. S.Philip Wong, Simone Raoux, Sangbum Kim, Jiale Liang, John P. Reifenberg, Bipin Rajendran, Mehdi Asheghi, and Kenneth E. Goodson. Phase change memory. In *Proceedings of the IEEE*, volume 98, pages 2201–2227. Institute of Electrical and Electronics Engineers Inc., 2010. doi: 10.1109/JPROC.2010.2070050.

[99] Yi Wu, Byoungil Lee, and H. S.Philip Wong. Al2O3-based RRAM using atomic layer deposition (ALD) with 1-μa RESET current. *IEEE Electron Device Letters*, 31(12):1449–1451, 12 2010. ISSN 07413106. doi: 10.1109/LED.2010.2074177.

[100] Wm. A. Wulf and Sally A. McKee. Hitting the memory wall. *ACM SIGARCH Computer Architecture News*, 23(1):20–24, 3 1995. ISSN 0163-5964. doi: 10.1145/216585.216588. URL `https://dl.acm.org/doi/10.1145/216585.216588`.

[101] Yuan Xie. Emerging Memory Technologies - Springer. *Springer*, pages 43–56, 2014. doi: 10.1007/978-1-4419-9551-3{_}1. URL `http://rd.springer.com/book/10.1007%2F978-1-4419-9551-3`.

[102] Cong Xu, Xiangyu Dong, Norman P. Jouppi, and Yuan Xie. Design implications of memristor-based RRAM cross-point structures. In *Proceedings -Design, Automation and Test in Europe, DATE*, pages 734–739, 2011. ISBN 9783981080179. doi: 10.1109/date.2011.5763125.

[103] Xiaoxin Xu, Lu Tai, Tiancheng Gong, Jiahao Yin, Peng Huang, Jie Yu, Da Nian Dong, Qing Luo, Jing Liu, Zhaoan Yu, Xi Zhu, Xiu Long Wu, Qi Liu, Hangbing LV, and Ming Liu. 40× Retention Improvement by Eliminating Resistance Relaxation with High Temperature Forming in 28 nm RRAM Chip. In *2018 IEEE Int. Electron Devices Meet.*, pages 20.1.1–20.1.4. IEEE, 12 2018. ISBN 978-1-7281-1987-8. doi: 10.1109/IEDM.2018.8614593. URL `https://ieeexplore.ieee.org/document/8614593/`.

[104] Jintao Yu, Hoang Anh Du Nguyen, Lei Xie, Mottaqiallah Taouil, and Said Hamdioui. Memristive devices for computation-in-memory. In *Proceedings of the 2018 Design, Automation and Test in Europe Conference and Exhibition, DATE 2018*, volume 2018-January, pages 1646–1651. Institute of Electrical and Electronics Engineers Inc., 4 2018. ISBN 9783981926316. doi: 10.23919/DATE.2018.8342278.

[105] Shimeng Yu, Ximeng Guan, and H. S Philip Wong. On the stochastic nature of resistive switching in metal oxide RRAM: Physical modeling, Monte Carlo simulation, and experimental characterization. *Technical Digest - International Electron Devices Meeting, IEDM*, pages 1–17, 2011. ISSN 01631918. doi: 10.1109/IEDM.2011.6131572.

[106] S. Yuasa and D. D. Djayaprawira. Giant tunnel magnetoresistance in magnetic tunnel junctions with a crystalline MgO(0 0 1) barrier. *Journal of Physics D: Applied Physics*, 40(21), 2007. ISSN 00223727. doi: 10.1088/0022-3727/40/21/R01.

[107] Yue Zha and Jing Li. Reconfigurable in-memory computing with resistive memory crossbar. In *IEEE/ACM International Conference on Computer-Aided Design, Digest of Technical Papers, ICCAD*, volume 07-10-Nove. Institute of Electrical and Electronics Engineers Inc., 11 2016. ISBN 9781450344661. doi: 10.1145/2966986.2967069.

[108] L. Zhang. *Understanding memristors and selectors for future storage and computing applications: Modeling and analysis.* PhD thesis, University of Pittsburgh, 2016.

[109] Liang Zhao, Blanka Magyari-Köpe, and Yoshio Nishi. Polaronic interactions between oxygen vacancies in rutile Ti O2. *Physical Review B*, 95(5), 2 2017. ISSN 24699969. doi: 10.1103/PhysRevB.95.054104.

[110] Lorenzo Zuolo, Cristian Zambelli, Rino Micheloni, and Piero Olivo. Solid-State Drives: Memory Driven Design Methodologies for Optimal Performance. *Proceedings of the IEEE*, 105(9):1589–1608, 9 2017. ISSN 15582256. doi: 10.1109/JPROC.2017.2733621.

A

Model default properties

The following table is a complete overview of all physical properties used in the simulations of the model. Note that variable parameters, such as device dimensions or ambient temperature, are not mentioned here because these variables are directly related to specific simulations and change between them. These variables will therefore be mentioned where they are relevant.

Property	*Value*	*Description*
		General properties
ϵ_r	21	Relative dielectric constant of HfO_2
k_{th}	1.5 K/Wm	Thermal conductivity of HfO_2
		Lattice element properties
Δx	3 Å	Lattice spacing step
$E_{C,el}$	4.45 eV	Electrode conduction band energy
$E_{C,il}$	2.9 eV	Interface layer conduction band energy
$E_{T,\mu}$	1.9 eV	Trap energy mean
ΔE_T	0.5 eV	Trap energy deviation
r_t	5.64 Å	Trap capture radius
		TAT properties
m_e	0.18	
$\hbar\omega_{0,ox}$	0.07 eV	Effective phonon energy in HfO_2
$\hbar\omega_{0,il}$	0.06 eV	Effective phonon energy in interface layer
$\phi_{g0,el}$	0 eV	Band gap energy of electrode
$\phi_{g0,ox}$	5.8 eV	Band gap energy of HfO_2
$\phi_{g0,il}$	8.9 eV	Band gap energy of interface layer
$E_{F,el}$	0	Relative location of Fermi energy in gap from conduction band in electrode
$E_{F,ox}$	1/2	Relative location of Fermi energy in gap from conduction band in oxide
$E_{F,il}$	1/4	Relative location of Fermi energy in gap from conduction band in interface layer
S	17	Huang-Rhys factor in HfO_2
		kMC properties
E_a	2.9 eV	O-Hf bond breakage zero-field effective activation energy
$E_{a,d}$	1.5 eV	Diffusion activation energy
$E_{a,r}$	0.2 eV	Recombination activation energy
p_0	5.2 eÅ	O-Hf dipole moment
ν	7×10^{13} Hz	Effective vibration frequency of O-Hf bonds
k_D	3 eÅ	Factor dependent on molecular properties of HfO_2