UAB
Universitat Autònoma de Barcelona

Dipòsit digital de documents de la UAB

This is the **published version** of the master thesis:

Felip Fernández, Arnau; Crespo-Yepes, Albert , dir. Using Matlab NeuralNetworkStart to classify RTN traces for security applications. 2022. 76 pag. (1170 Màster Universitari en Enginyeria de Telecomunicació / Telecommunication Engineering)

This version is available at https://ddd.uab.cat/record/259451

under the terms of the license

Universitat Autònoma de Barcelona

A Thesis for the

Master in Telecommunication Engineering

Using Matlab NeuralNetworkStart to classify RTN traces for security applications

by
Arnau Felip Fernández

Supervisor: Albert Crespo Yepes

Electronic engineering department

Escola d'Enginyeria
Universitat Autònoma de Barcelona (UAB)

January 2022

El sotasignant, *Albert Crespo Yepes*, Professor de l'Escola d'Enginyeria de la Universitat Autònoma de Barcelona (UAB),

Fa constar:

Que el projecte presentat en aquesta memòria de Treball Final de Master ha estat realitzat sota la seva direcció per l'alumne *Arnau Felip Fernández*.

I, perquè consti a tots els efectes, signa el present document.

Bellaterra, *data_de_sol.licitud_de_lectura*.

Signatura:

Resum:

Aquest treball de fi de Màster s'ha centrat en l'estudi de les traces Random Telegraph Noise (RTN) per a l'entrenament de les xarxes neuronals. Concretament, s'ha estudiat l'aleatorietat de les traces RTN, mitjançant una primera conversió a bits i una posterior validació basada en la incertesa en la repetició dels símbols binaris consecutius. Aquest estudi s'ha realitzat a nivell de simulació, utilitzant el model de les xarxes neuronals basat en les traces RTN, mitjançant l'aplicació de NNStart de MATLAB.

Resumen:

Este Trabajo de final de Máster se ha centro en el estudio de las trazas Random Telegraph Noise (RTN) para el entrenamiento de las redes neuronales. Concretamente, se ha estudiado la aleatoriedad de las trazas RTN mediante una primera conversón a bits y una posterior validación, basada en la incertidumbre en la repetición de símbolos binarios consecutivos. Este estudio se ha realizado a nivel de simulación, utilizando el modelo de redes neuronales basado en las trazas RTN, mediante la aplicación de NNStart de MATLAB.

Summary:

This Master's thesis has focused on the study of Random Telegraph Noise (RTN) traces for the training of neural networks. Specifically, the randomness of RTN traces has been studied by means of a first conversion to bits and a subsequent validation, based on the uncertainty in the repetition of consecutive binary symbols. This study has been carried out at the simulation level, using the neural network model based on the RTN traces, by means of MATLAB's NNStart application.

Presentation

The main objective of this work is to use a virtual neural network capable of discriminating which Random Telegraph Noise (RTN) traces are valid and which are not. To do so, the traces generated will be measured and classified according to the uncertainty in the repeatability of consecutive binary symbols.

MATLAB, a programming software, and, in particular, the NNStart application will be used to train the neural network. This application is capable, on its own, of verifying and validating the traces.

It has been decided to use the characteristics of RTN traces because of their ability to not follow decipherable patterns. This randomness makes it an interesting phenomenon for encryption applications, cybersecurity and security applications such as random one-time codes, PUDFs, etc...

In order to fulfil the established objective, the following tasks have been executed:

-Generate different RTN traces by modifying the statistical parameters of the random variables that generate the levels, the emission and capture times, etc....

-Obtain codes of different lengths from the RTN traces and make validations using two different methods: average current value and capture and emission events.

-Validate the binary codes separately, associating a 01 if the code is correct and a 10 if the code is not correct.

-Make the WTLP of each generated binary code to be used as input to the neural network.

-Train the neural network and analyse the training results.

Index

1. Introduction 11
  1.1 MOS structure and its operation regions 11
  1.2 MOSFET structure.. 13
    1.2.1 Operation regions of MOSFET structure 14
    1.2.2 Scaling MOSFET devices 16
      1.2.2.1 Threshold voltage effect 17
      1.2.2.2 Quantum tunnelling limit 18
      1.2.2.3 Gate oxide tunnelling 18
      1.2.2.4 Short Channel Effect (SCE) 18
      1.2.2.5 Drain-Induced Barrier Lowering (DIBL) 19
      1.2.2.6 Channel Length Modulation.. 19
    1.2.3 FD-SOI technology 20
    1.2.4 Variability and Aging mechanisms 21
      1.2.4.1 Channel Hot Carries (CHC) 21
      1.2.4.2 Bias Temperature Instability (BTI) 22
      1.2.4.3 Random Telegraph Noise (RTN) 24
  1.3 Neural Networks 28
  1.4 MATLAB and NNstart 30
2. RTN trace generation..33
  2.1 Characterisation of experimental RTN 33
    2.1.1 Gaussian distribution 33
    2.1.2 Signal noise 34
    2.1.3 Number of defects 35
    2.1.4 Jumps 36
    2.1.5 Offsets... 36
  2.2 Definition of the criterial for the generation of RTN traces 36
    2.2.1 Number of traces to be generated and number of samples per RTN traces 37
    2.2.2 Transmission and capture time 37
    2.2.3 Transmission and capture mean 38
    2.2.4 Standard deviation and variance... 38
  2.3 Table of RTN traces parameters 39

2.4 Results of generated RTN traces 39
2.4.1 Case 1 graphs 40
2.4.2 Case 2 graphs 41
2.4.3 Case 3 graphs 43
2.4.4 Cases 4 graphics 44
2.4.5 Case 5 graphics 47
2.4.6 Cases 6 graphics 48
3. Conversion of RTN traces to binary codes 51
3.1 Method based on the average current value 51
3.1.1 Procedure 51
3.2 Catch-emission method 52
3.2.1 Procedure 52
4. Binary code validation 53
4.1 Method based on the probability of followed bits 53
5. Neural network training 57
5.1 Code groups used 57
5.2 Network input data preparation 58
5.3 Network output data preparation 59
5.4 Results of training 59
5.4.1 Training specifications 60
5.4.2 Training specifications 62
6. Conclusions 69
X. Bibliography 71

List of figures

Figure 1: MOS structure. 11
Figure 2: NMOS (a) and PMOS (b) structures. 12
Figure 3: Capacitances vs voltage of MOS-C devices for n-type. 13
Figure 4: MOSFET cross sections NMOS transistor (a) and PMOs transistor (b). 14
Figure 5: MOSFET regions of operations. 16
Figure 6: Schematic diagram of device scaling. 17
Figure 7: Potential barrier between two transistors. 18
Figure 8: Effect by reducing the length of the transient gate affects the tension barrier by reducing it [2]. 19
Figure 9: NMOS transistor cutting (left) vs FD-SOI technology cross section (right). . 20
Figure 10: Schematic diagram of channel-hot-carrier injection. 22
Figure 11: MOSFET $Id - Vg$ (IV) curves show (a) $|VTH|$ and (b) $|IDOFF|$ increases under NBTI stress. $Id - Vg$ measurements show gate leakage (IG) also increase 23
Figure 12: Example of BTI lifetime projection form accelerated test to operating condition. (a) Test data under high $VG - VTH$ (accelerate stress) condition is used to extract model parameters and predict device lifetime. (b) The accuracy of the prediction is usually verified by the comparison of test data under use-bias and model prediction. 23
Figure 13: (a) NBTI to PMOS transistors degradation. (b) PBTI to NMOS transistors degradation [5]. 24
Figure 14: Two-level RTN waveform along with an illustration of the underlying carrier trapping process [7]. 24
Figure 15: (a) It is a waveform with two levels, the first level is at 3.80 µA and the second level at 3.60 µA. (b) In the figure give a vision of the TLP that shows two states (2 levels described in (a)), plus points that it has detected in the middle of a transition [8]. 26
Figure 16: Explanation of Time Lag Plot (TLP) [8]. 26
Figure 17: RTN hidden by background noise [9]. 26
Figure 18: - (a) Typical multilevel RTN signal measured with a semiconductor parameter analyser. VAPP=1.25V, step time ~6ms and number of measured points 8000. (b) Trap levels obtained by using the W-TL method [10]. 27
Figure 19: Oscilloscope traces captured in different time window, obtaining the interval time of a deffect [10]. 28
Figure 20: Diagram of a neural network with two input and output layers and three hidden layers. 29
Figure 21: NNStart Interface. 31
Figure 22: The red curve is the standard normal distribution 34
Figure 23: RTN signals case 1. 40
Figure 24: Behaviour of drian current case 1. 41
Figure 25: RTN traces case 2. 42
Figure 26: Behaviour of drian current case 2. 42
Figure 27: RTN signals case 3. 43

Figure 28: Behaviour of drian current case 3. 44
Figure 29: RTN signals case 4 (section 1)... 45
Figure 30: RTN signals case 4 (section 2)... 45
Figure 31: Behaviour of drian current case 4 (setion 1). 46
Figure 32: Behaviour of drian current case 4 (section 2). 46
Figure 33: RTN signals case 5... 47
Figure 34: Behaviour of drian current case 5. 48
Figure 35: RTN signals case 6 (section 1)... 49
Figure 36: RTN signals case 6 (section 2)... 49
Figure 37: Behaviour of drian current case 6 (section 1). 50
Figure 38: Behaviour of drian current case 6 (section 2). 50
Figure 39: Example of balanced RTN. 51
Figure 40: Example of unbalanced RTN. 52
Figure 41: Function $fn = 12n$. 54
Figure 42: MATLAB function $fn = 12n$. 55
Figure 43: Binary WTLP of case 2.. 58
Figure 44: Binary WTLP of case 3.. 58
Figure 45:Example of patter recognition app performance. 60
Figure 46: Example of patter recognition app training state. 61
Figure 47: Example of patter recognition app error histrogram. 61
Figure 48: Basic training results (10 neurons in the hidden layer)............................... 63
Figure 49: Confusion matrix results (10 neurons in the hidden layer). 63
Figure 50: ROC graphs results (10 neurons in the hidden layer). 64
Figure 51: Basic training results (100 neurons in the hidden layer).............................. 65
Figure 52: Confusion matrix results (100 neurons in the hidden layer). 65
Figure 53: ROC graphs results (100 neurons in the hidden layer). 66
Figure 54: Basic training results (1000 neurons in the hidden layer)............................ 66
Figure 55: Confusion matrix results (1000 neurons in the hidden layer). 67
Figure 56: ROC graphs results (1000 neurons in the hidden layer). 67

List of tables

Table 1: Regions of operations to NMOS and PMOS transistors, where 'V' is the voltage and the sub-indexes mark whether it is the gate (G), the drain (D), the source (S) or voltage threshold (TH). 15
Table 2: NMOS and PMOS transistor currents for minimum output voltage. 15
Table 3: Values of $nmean$ and $nsigma$ in the different cases 35
Table 4 Values of N and n in the different cases. 37

Table 5: Tc and Te values in the different cases. 38
Table 6: $memean$ and $memean$ values in the different cases. 38
Table 7: Table of RTN signal parameters. 39
Table 8: Table of probabilities as function of n. 54

1. Introduction

This section of the work introduces the basic concepts on which this master's thesis is based. First, the MOSFET transistor and its main characteristics (operating regions, I-V curves...) will be presented, including variability effects such as RTN that occurs during the device operation. Then, it will be explained how RTNs (Random Telegraph Noise) traces can be used as random phenomena for security applications, and how the Matlab's NNstart can be used to classify the RTN traces depending on their own characteristics. To conclude this section, we will describe the different characteristics and/or objectives of these concepts and how we will work to achieve the objective of the work.

1.1 MOS structure and its operation regions

A MOSFET transistor is based on the MOS (Metal-Oxide Semiconductor) structure. It consists of a metal (usually polysilicon), silicon oxide (insulator, SiO_2) and an N-type or P-type silicon semiconductor. In the case of an N-type structure (NMOS), it consists of a silicon substrate doped with holes. If it is a P-type structure (PMOS), it consists of an electron-doped silicon substrate.

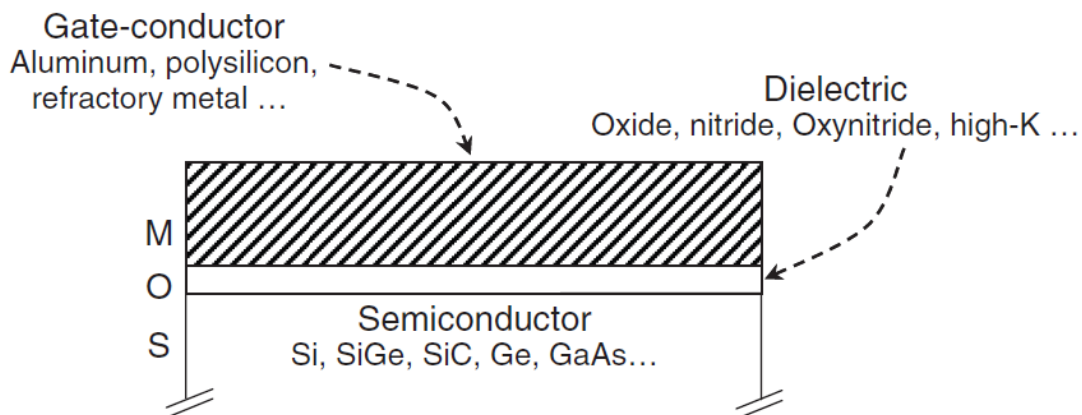

Figure 1: MOS structure.

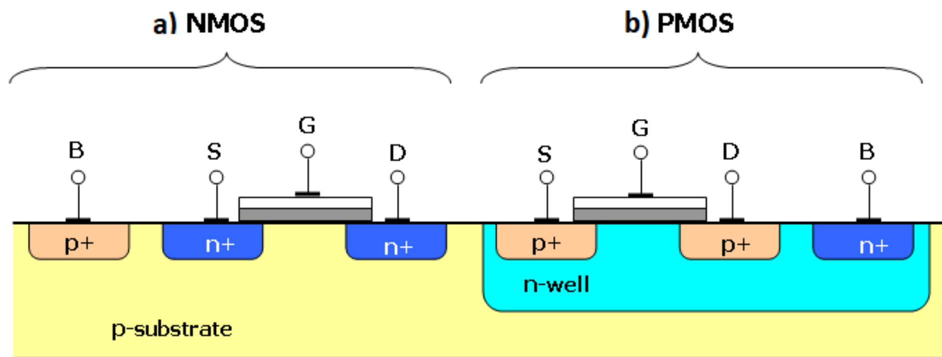

Figure 2: NMOS (a) and PMOS (b) structures.

In both structures the device behaves like an electronic capacitor due to the accumulation of electrical charges on the oxide and the semiconductor. Depending on the level of applied voltage, three regions of operation are distinguished: accumulation region, depletion region and reversal region.

-Accumulation region: in this stage the charges are stored in the oxide. The dielectric is polarised proportional to the applied electric field. In an NMOS, by applying a negative gate potential, electrons are induced, attracting holes to the interface and creating an electric field. If the structure is a PMOS, the applied gate potential will be positive, inducing holes and also creating an electric field.

-Depletion region: In this region, the gate potential is increased, causing electrons and holes to begin to recombine in the semiconductor. In NMOS, a positive potential is applied to the gate, accumulating positive charge on the metal, attracting electrons to the interface and pushing the holes away. This generates an electric field from the metal to the semiconductor (opposite direction to the build-up case). In PMOS, the potential applied to the gate is negative, attracting the holes to the interface and creating an electric field. In this case, the holes are attracted to the interface and the electrons are repelled, producing an electric field in the build-up direction.

-Reversal region: by further increasing the gate voltage, the reversal region is reached. When working with an NMOS, the applied voltage is so negative that the material is filled with electrons. In this way, the semiconductor holes are attracted to the interface, creating

an electric field from the semiconductor to the metal. When working with a PMOS, a very positive voltage is applied, inducing many holes in the metal and creating a large electric field from the metal to the semiconductor (high intensity).

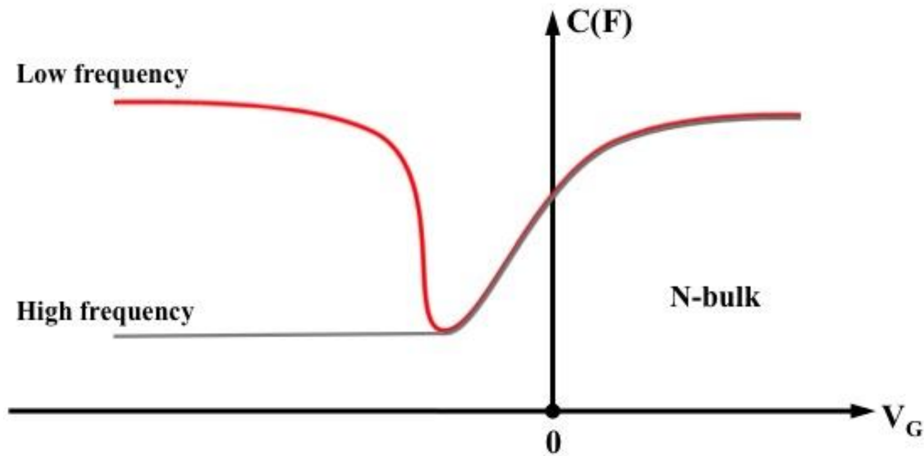

Figure 3: Capacitances vs voltage of MOS-C devices for n-type.

1.2 MOSFET structure

MOSFETs (Metal Oxide Semiconductor Field Effect Transistors) are semiconductor switching devices that have three terminals: gate (S), source (G) and drain (D). One of the fundamental characteristics of the MOSFET is that it is a device in which there is no electrical connection between the port and the substrate, making the gate isolated. This type of transistor is used to amplify or switch electronic signals. There are two types of MOSFETs: enrichment ones and deplexing ones.

-Enrichment MOSFETs: these are based on the creation of a channel between the drain and source by applying a gate voltage. This voltage attracts minority carriers into the channel, creating an inversion region (the opposite region to the original substrate). When there is an increase in electron concentration in the channel, an nMOSFET or NMOS is obtained. When the concentration is of holes, it has pMOSFET or PMOS. Therefore, an NMOS is built with a p-type substrate and a PMOS with an n-type substrate.

-Duplexing MOSFET: an electrical voltage is applied to the gate to make the channel, which is in a quiescent state, disappear. This voltage generates a decrease in the number of charge carriers and conductivity.

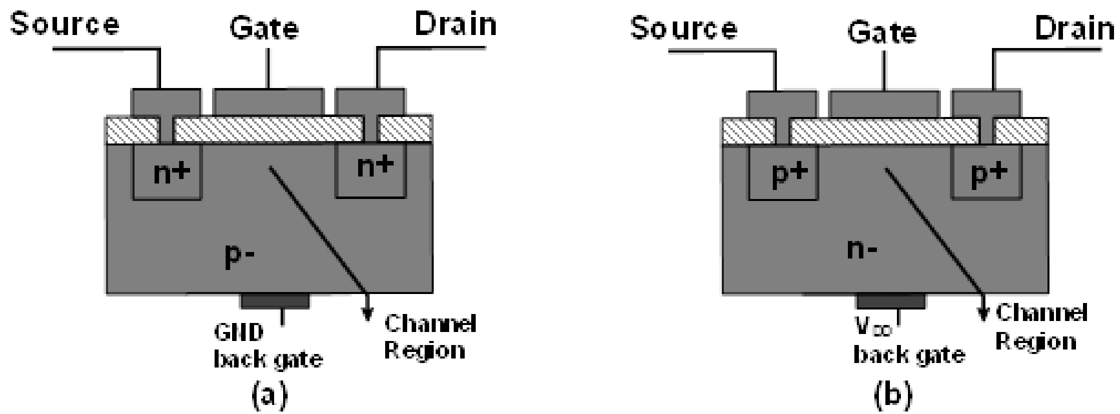

Figure 4: MOSFET cross sections NMOS transistor (a) and PMOs transistor (b).

1.2.1 Operation regions of MOSFET structure

There are different regions of operation for NMOS and PMOS transistors. These depend on the voltage between drain and source (V_{DS}), the voltage between gate and source (V_{GS}) and the threshold voltage of the transistor (V_{TH}). The different operating regions will be explained for the NMOS case, and the table 1 will also show them for the PMOS case.

The first of the regions is called cut-off region which occurs when $V_{GS} < V_{TH}$. At this point, the drain current (I_D) is practically zero. Therefore, it is said that in this region the transistor is turned off.

In the linear region the applied voltage is higher than the threshold creating a depletion region in the region separating source and drain. If the applied voltage is further increased, in an NMOS transistor the minority carriers will be electrons and in a PMOS transistor the minority carriers will be holes. In this region the transistor behaves like a resistor depending on the gate voltage.

The last of the regions is the saturation region. In this region, the drain and source voltages exceed a certain limit causing the channel to disappear. The I_D current is invariant to changes in V_{DS} and depends only on the applied V_{GS} voltage.

Regions of operations	NMOS	PMOS
Cut-off region	$V_{GS} \leq V_{TH}$	$V_{GS} \geq V_{TH}$
Linear region	$V_{GS} \geq V_{TH}$ $V_{DS} < V_{GS} - V_{TH}$	$V_{GS} \leq V_{TH}$ $V_{DS} > V_{GS} - V_{TH}$
Saturation region	$V_{GS} \geq V_{TH}$ $V_{DS} \geq V_{GS} - V_{TH}$	$V_{GS} \leq V_{TH}$ $V_{SD} \leq V_{GS} - V_{TH}$

Table 1: Regions of operations to NMOS and PMOS transistors, where 'V' is the voltage and the sub-indexes mark whether it is the gate (G), the drain (D), the source (S) or voltage threshold (TH).

The drain current must also be considered. Depending on the area and the type of MOSFET, PMOS or NMOS, it has different values (see table 2).

Regions of operations	NMOS currents	PMOS currents
Cut-off region	$I_D = 0$	$I_D = 0$
Linear region	$I_D = \mu_n C_{ox} \frac{W}{L}\left[(V_{GS} - V_{TH})V_{DS} - \frac{{V_{DS}}^2}{2}\right] - \frac{V_{DS}^2}{2}$	$I_D = \mu_p C_{ox} \frac{W}{2L}[(V_{SG} - \lvert V_{TH}\rvert)V_{SD} - \frac{{V_{SD}}^2}{2}]$
Saturation region	$I_D = \mu_n C_{ox} \frac{W}{L}(V_{GS} - V_{TH})^2$	$I_D = \mu_p C_{ox} \frac{W}{2L}(\lvert V_{GS}\rvert - \lvert V_{TH}\rvert)^2$

Table 2: NMOS and PMOS transistor currents for minimum output voltage.

All these concepts can be seen in the figure below (Figure 5). The different operating regions of the transistor are shown as a function of the drain current (I_D) and the voltage between drain and source (V_{DS}).

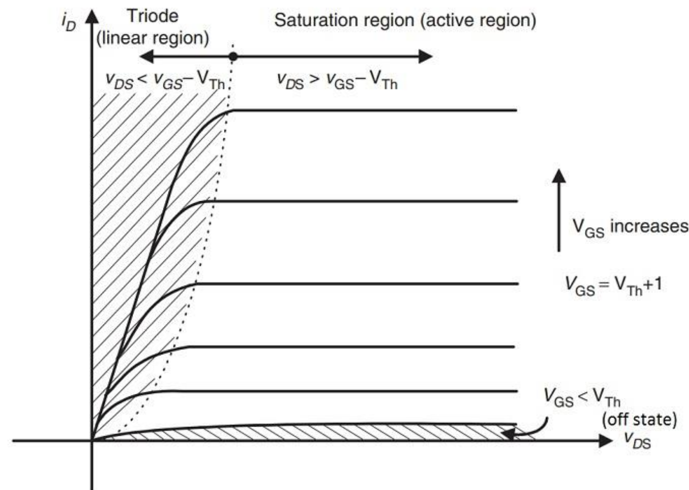

Figure 5: MOSFET regions of operations.

1.2.2 Scaling MOSFET devices

One of the main objectives of the semiconductor industry is scaling. This technique consists of reducing the size of devices while maintaining the same performance, reducing the cost of production, and offering a larger number of devices per area.

Moore's laws [1] have been followed for many years, but the constant evolution of technology and the need for constant performance improvement means that this technique has evolved and is now in all semiconductor parameters: length, thickness, channel, supply voltages, etc. The development of the MOSFET scaling technology is shown in Figure 6.

The conventional CMOS device is approaching its scaling limits. When the MOSFET device was introduced, the gate length was measured in micrometres. Today, the channel length has reached the nanoscale. Although the fabrication of reduced-size transistors presents some advantages such as area efficiency, increased speed or improved performance, decreasing the gate length introduces several problems. These problems arise when the size of the MOSFET enters the nanoscale, as its performance degrades.

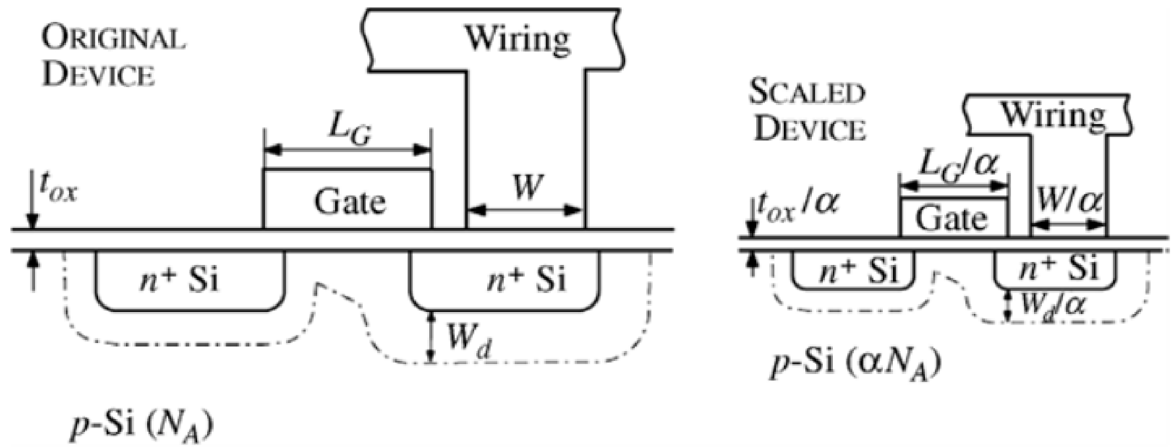

Figure 6: Schematic diagram of device scaling.

Two scaling techniques are generally used: constant voltage scaling and constant field scaling. Constant voltage scaling consists of decreasing the MOSFET dimensions by a factor α except for the supply and terminal voltages. This type of mechanism can lead to reliability problems by generating oxide breakdown, electrical overload and electromigration. In contrast, in constant field scaling, the dimensions and voltages are reduced by the factor α and the doping and charge densities are increased by an equal factor α', causing the electric field to be unaffected. Thus, the circuit velocity increases by a factor α and the circuit density increases by a factor 2α. Due to the problems discussed with constant voltage scaling, constant field scaling is the most used.

1.2.2.1 Threshold voltage effect

When scaling techniques are applied, the reduction of the MOSFET channel is also proportional to the reduction of the supply voltage and active power. In contrast, the threshold voltage cannot be reduced in the same way because the higher power consumption is due to the leakage current of the device.

Therefore, the V_{TH} scaling has been slowed down to avoid a drastic increase in I_{OFF}. To achieve to achieve a large drive current, the gate overload ($V_{DD} - V_{TH}$) must be significant and therefore the V_{DD} scaling also has to be slower, which results in an increase of the active power, which results in an increase of the active power density.

1.2.2.2 Quantum tunnelling limit

As explained above, the supply voltage cannot be reduced in proportion to the channel length. Scaling techniques cause the electric field strength across the gate oxide to be increased. This increased electric field in turn causes the mobility of the carriers to be impaired and cause disturbance to the devices. The possibility of carrier movement through a scaled MOSFET can be seen in Figure 7.

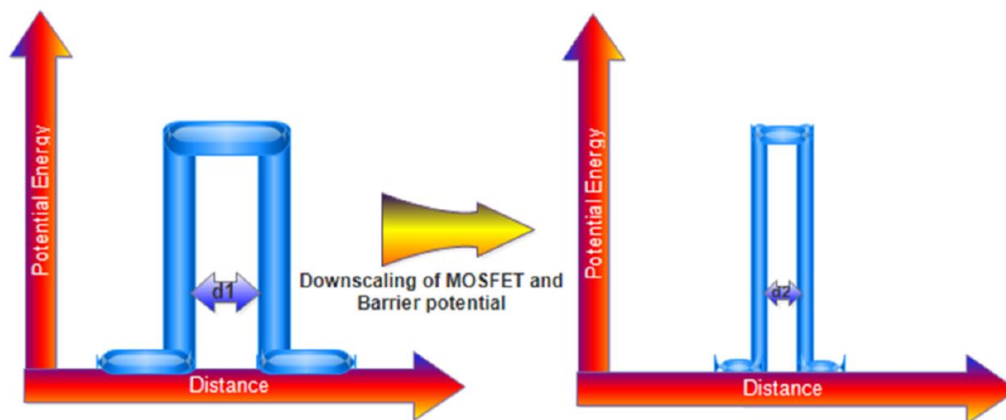

Figure 7: Potential barrier between two transistors.

1.2.2.3 Gate oxide tunnelling

Since the thermal electron voltage, kT/q, is a constant at room temperature, the ratio of operating voltage to thermal voltage decreases as the MOSFET size is reduced. This can lead to higher leakage currents due to thermal diffusion of electrons. When applying scaling techniques that lead to reductions in channel lengths, they require that the reduction in oxide thickness be taken into account. These reductions are subject to quantum tunnelling as the gate leakage current increases exponentially as the gate thickness is reduced.

1.2.2.4 Short Channel Effect (SCE)

A MOSFET is considered short-channel when the channel length is comparable to the depletion layer widths of the source and drain junctions. It is when the voltage applied to the device is significantly reduced that short-channel effects appear. These effects include

drain-induced barrier lowering, velocity saturation, quantum confinement and hot carrier degradation.

1.2.2.5 Drain-Induced Barrier Lowering (DIBL)

This effect is due to the drain bias and intensifies the short-channel effects. In contrast to a long channel, an increase of the drain-source bias causes a reduction of the threshold voltage and an increase of the sub-threshold current. Therefore, the short-channel effect is intensified due to the DIBL that increases with high drain voltages and shorter channel lengths. This type of effect can cause permanent damage to the transistors due to localised melting of the material.

Figure 8 shows how the short-channel effect is intensified by the polarisation of the drain, due to the drain induced barrier lowering [2].

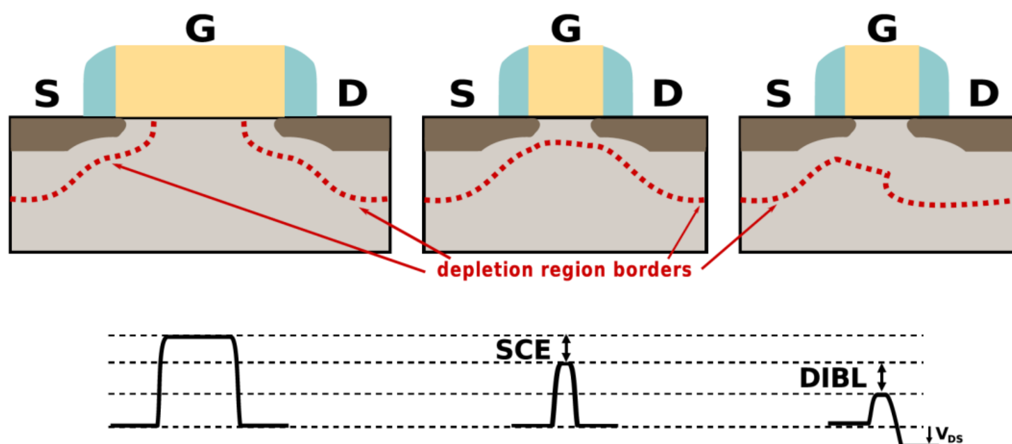

Figure 8: Effect by reducing the length of the transient gate affects the tension barrier by reducing it [2].

1.2.2.6 Channel Length Modulation

Short channel effects (SCE) occur when the channel length is reduced as the potential differential between gate and drain increases. Therefore, as discussed above, the channel length will now depend on the V_{DS} voltage. To correct for this effect, a term $(1+\lambda V_{DS})$ is added where λ is the modulation coefficient and depends inversely on the channel length. That is, the smaller this coefficient is, the longer the channel length will be. The drain current would be defined by the following equation:

$$I_D = \mu C_{ox} \frac{W}{2L} (V_{GS} - V_{TH})^2 (1 + \lambda V_{DS}) \quad (1)$$

Adding the term (1+λV_{DS}) results in a slope in the $I_D - V_{DS}$ characteristic in the saturation region. The drain current saturates at the V_{DS} value causing a channel pinch at the drain end.

1.2.3 FD-SOI technology

In order to solve scaling problems, fully depleted silicon-on-insulator (FD-SOI) technology is a process that takes advantage of existing manufacturing methods and offers reduced silicon geometries with increased performance and low power consumption. In this way, it is possible to extend Moore's Law [1] without the need for such complex manufacturing processes. This process is made possible by combining the use of an ultra-thin oxide insulator on top of the base silicon and the use of a very thin layer of silicon that creates the transistor channel. The thinness of this channel allows the transistor to be depleted, i.e. the channel does not need to be doped. The FD-SOI structure is therefore the result of the evolution of the bulk CMOS process.

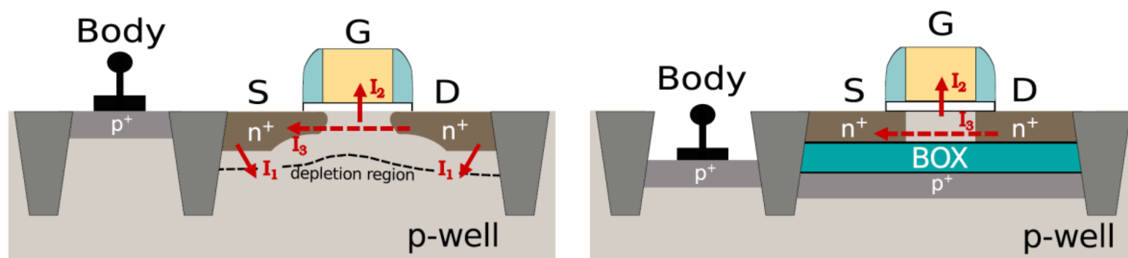

Figure 9: NMOS transistor cutting (left) vs FD-SOI technology cross section (right).

As can be seen in Figure 9, the parasitic capacitance between the source and drain of the Bulk-CMOS structure is reduced due to the buried FD-SOI oxide layer. Thus, the electron flow between the source and drain is also reduced. This effect also reduces the leakage current which degrades performance and power. The channel, which is fully depleted, also reduces potential leakage [3].

With this technology, it is possible to control the behaviour of the transistors through the gate or by applying a voltage (biasing) to the substrate underneath the device. This technique is made possible by low stray current leakage because the dielectric insulation created by the buried oxide layer is much more effective. The properties of the insulation allow higher bias voltages to be used, allowing dynamic control of the transistor to select between speed and energy efficiency. These advantages allow for lower transistor manufacturing costs.

There are two types of biasing: Forward Body Biasing (FBB) and Reverse Body Biasing (RBB). The first requires less gate voltage to switch the transistor, resulting in faster transistor switching with less power consumption (power minimisation). The second bias, RBB, applies gate voltage to the transistor to switch the transistor. This type of biasing allows designers to choose between faster or more efficient operation when high speed is required or lower leakage power when performance is not as critical.

1.2.4 Variability and Aging mechanisms

As explained in the previous sections, scaling techniques can cause different physical phenomena in scaling devices affecting their reliability. In order to measure the reliability of these devices, variability and degradation techniques are used. The best known in MOSFET transistors are the Channel Hot Carries (CHC), the Random Telegraph Noise (RTN) and the Bias Temperature Instability (BTI).

1.2.4.1 Channel Hot Carries (CHC)

This phenomenon is also known as Hot Carrier Injection (HCI) in which charges, electrons or holes, gain enough kinetic energy to break an interface state. The term "hot" refers to the effective temperature used to model the carrier density, not the overall temperature of the device. These charges can be injected into the gate oxide and become trapped in the gate of a MOS transistor, the switching of the transistor can be modified. New interface states can also be generated and produce effects on the gate current.

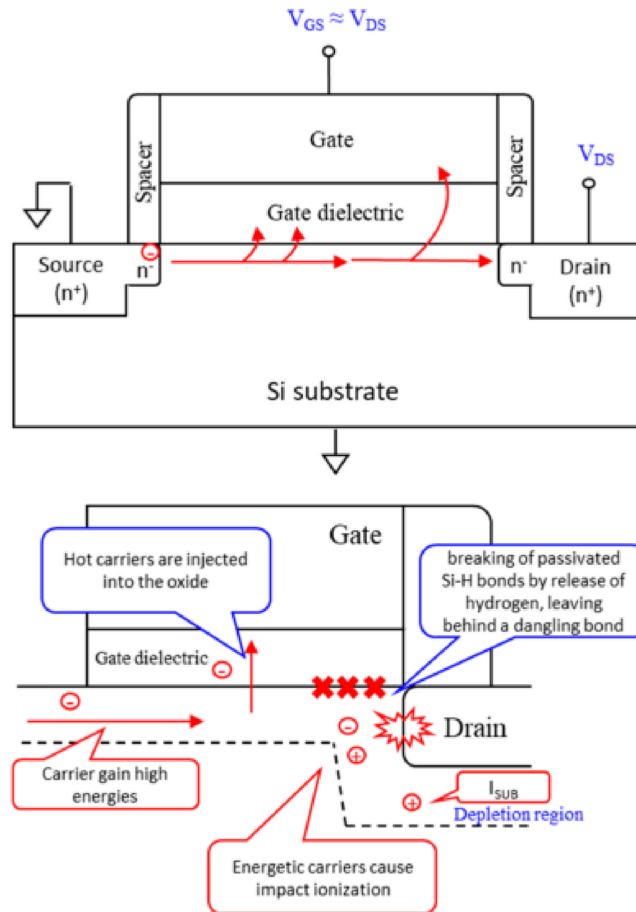

Figure 10: Schematic diagram of channel-hot-carrier injection.

The accumulation of such damage can degrade the device over extended periods of time by affecting parameters such as the threshold voltage, which would be shifted by such damage. The accumulation of damage resulting in device degradation due to hot carrier injection is referred to as "hot carrier degradation".

1.2.4.2 Bias Temperature Instability (BTI)

The Bias Temperature Instability (BTI) describes a phenomenon that degrades the performance of a device when a bias is applied to the gate of the MOSFET, and that channel is turned on. This phenomenon will increase the device umbra voltage $|V_{TH}|$ reducing the device conduction current $|I_D|$ (Figure 11 a) and the operating frequency of the circuit. It will also increase the absolute "off" current I_{DOFF} (Figure 11 b) and gate leakage I_G (Figure 11 c) increasing the circuit power consumption [4].

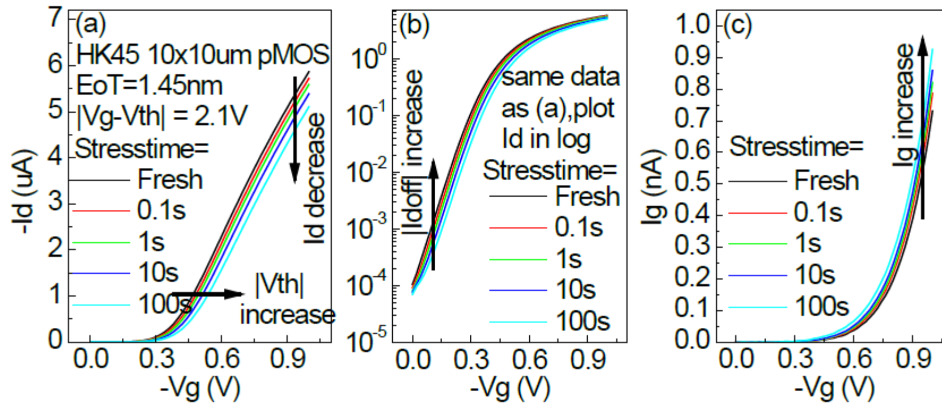

Figure 11: MOSFET $Id - Vg$ (IV) curves show (a) $|V_{TH}|$ and (b) $|I_{DOFF}|$ increases under NBTI stress. $Id - Vg$ measurements show gate leakage (I_G) also increase

The standard BTI lifetime criterion is that $|\Delta V_{TH}|$ does not exceed a certain level, approximately 100 mV, after the device reaches its lifetime. The lifetime of a device is 10 years, which is too long to reach in a laboratory. For this reason, accelerated BTI testing is done by applying a much more severe voltage than the operating condition. The device lifetime at operating conditions is projected from the accelerated tests within an acceptable test time (<10^6 seconds) using a time evolution model. The predictive capability of the model can be checked by comparing the predicted value with test data under operating conditions, as shown in Fig. 12.

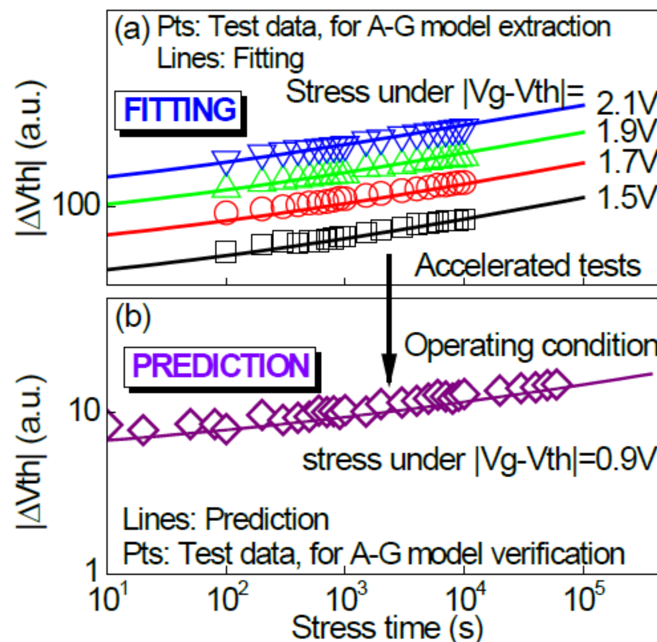

Figure 12: Example of BTI lifetime projection form accelerated test to operating condition. (a) Test data under high $|V_G - V_{TH}|$ (accelerate stress) condition is used to extract model parameters and predict device lifetime. (b) The accuracy of the prediction is usually verified by the comparison of test data under use-bias and model prediction.

As for the applied gate voltage, it can be positive (Positive Bias Temperature Instability, PBTI) or negative (Negative Bias Temperature Instability, NBTI). When a positive gate potential, PBTI, is applied, the effects only occur on the NMOS transistors by shifting the threshold voltage $V_{TH}$to higher values. In contrast, when a negative gate voltage, NBTI, is applied, the effects occur on both NMOS and PMOS. NBTI manifests itself as an increase in umbra voltage, a degradation of mobility, drain current and transconductance. These two effects can cause the device to degrade as shown in the figure below (Figure 13).

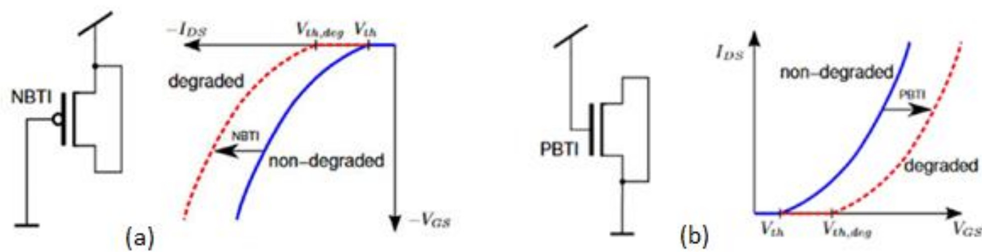

Figure 13: (a) NBTI to PMOS transistors degradation. (b) PBTI to NMOS transistors degradation [5].

1.2.4.3 Random Telegraph Noise (RTN)

This phenomenon is a type of electrical noise that occurs in ultra-thin gate semiconductors [6]. It is a low-frequency noise that increases with decreasing device size. RTN consists of unexpected step shapes between two or more discrete voltage or current levels that occur randomly and unpredictably. These shapes appear in the drain current between discrete current levels when the voltage is constant at both the gate and the drain. Figure 14 shows an RTN signal obtained in a 45 nm MOSFET.

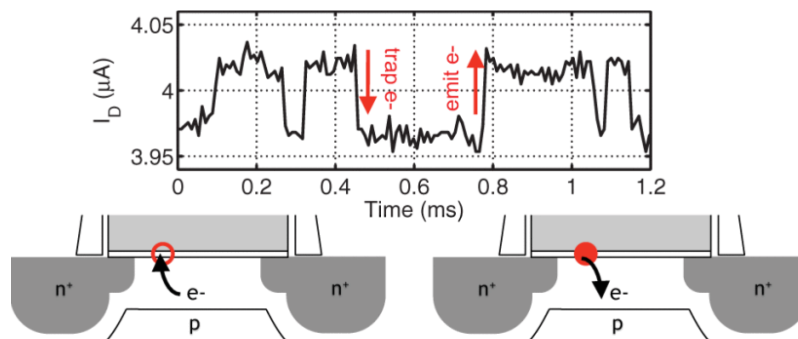

Figure 14: Two-level RTN waveform along with an illustration of the underlying carrier trapping process [7].

This noise allows to know the reliability of the devices by obtaining data analysis, verifications and simulations. To compare the behaviour and analyse the defects appearing in the devices, the measurements will be made with the "new" transistor and after applying stress techniques.

The RTN phenomenon consists of three essential parameters: amplitude, capture time and emission time. The amplitude reflects the impact of the RTN trap, the averaged capture time (τ_c) and emission time (τ_e) can be used to extract information of the trap energy according to equation (1).

$$\frac{\tau_c}{\tau_e} = \exp\left(\frac{E_T - E_F}{kT}\right) \tag{2}$$

Where:
E_T is the RTN responsible trap energy,
E_F is the fermi level,
k is the Boltzmann constant,
T is the temperature in Kelvins.

In some cases, the RTN cannot be easily obtained as noise can mask current changes making the detection of possible defects in the reliability of devices unlikely. It is for this reason that there are two possible analyses of the RTN: when it is not masked by noise and when it is masked by noise.

In the first case, the Time Lag Plot (TLP) is used to detect the RTN. This technique is based on the study of the correlation of two different data series. To do so, it plots an 'i' data series plus the next 'i+1', so that the traps can be visualised [8][9].

This process can be seen in Figure 15, where the TLP plots the samples obtained depending on the level of the RTN and the path of the transitions. The final process obtained is a mapping of the transitions and the level at which the defect is found, Figure 16.

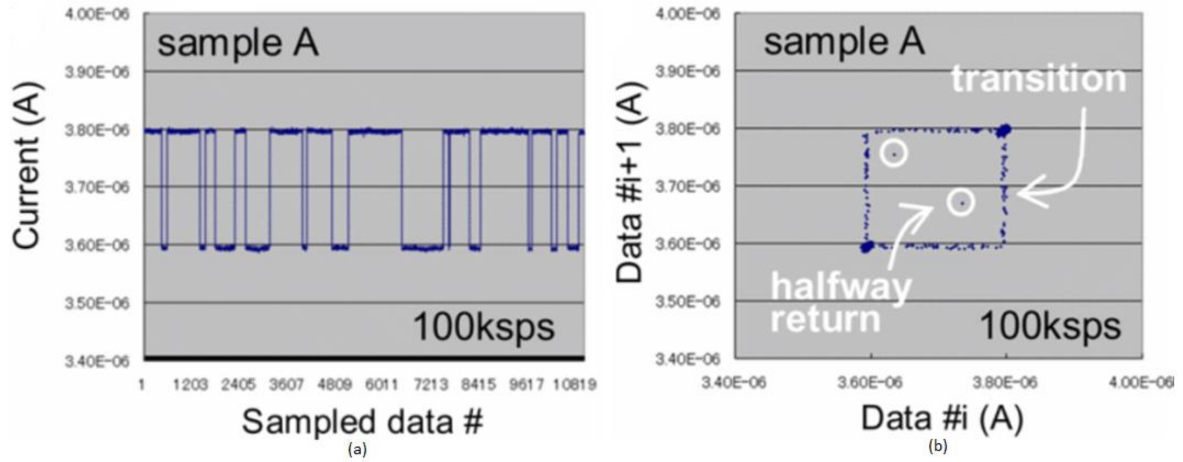

Figure 15: (a) It is a waveform with two levels, the first level is at 3.80 μA and the second level at 3.60 μA. (b) In the figure give a vision of the TLP that shows two states (2 levels described in (a)), plus points that it has detected in the middle of a transition [8].

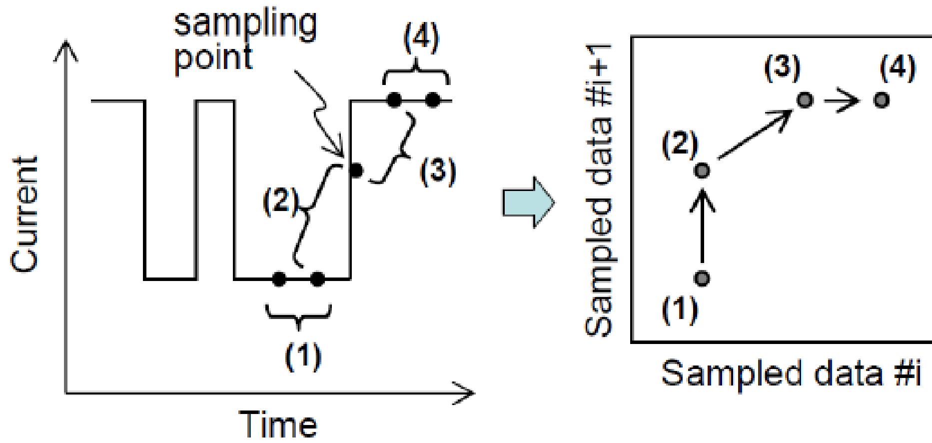

Figure 16: Explanation of Time Lag Plot (TLP) [8].

In the second case, when the RTN is hidden by background noise (Figure 17), the Weighted Time-Lag Plot (WTLP) is used. This method extends the TLP by minimising the effect of noise on the RTN and allows a more accurate extraction of the parameter.

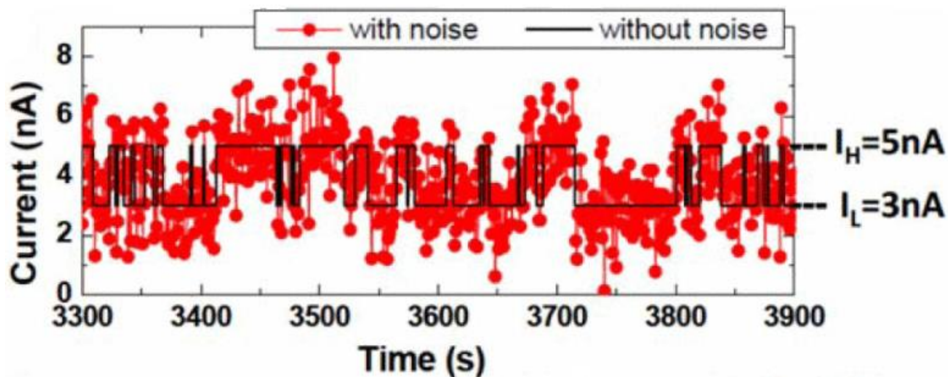

Figure 17: RTN hidden by background noise [9].

The process consists of marking the coordinates (i, i+1) and defining the equation$\phi_i(x, y)$. Equation 3 is a bivariate normal distribution with an 'alpha' deviation and a correlation coefficient of zero, representing the probability of corresponding to a level or transition [9].

$$\phi_i(x, y) = \frac{1}{2\pi\alpha^2} \exp\left(\frac{(I_i - x)^2 + (I_{i+1} - y)^2}{2\alpha^2}\right) \tag{3}$$

The weighted time lag equation is defined as equation 4, where the value of 'K' which is constant is to normalize the maximum value to '1' and N is the number of points in the Random Telegraph Signal (RTS). Giving the TLP histogram.

$$\Psi(x, y) = K \sum_{i-1}^{N-1} \phi_i \tag{4}$$

Other interesting parameters of the RTN are often analysed. The first of these is the multiple levels that the signal can have. Another important data that is also analysed is the time in which the defect of an RTN prevails in its level. These parameters can be seen in Figures 18 and 19 respectively.

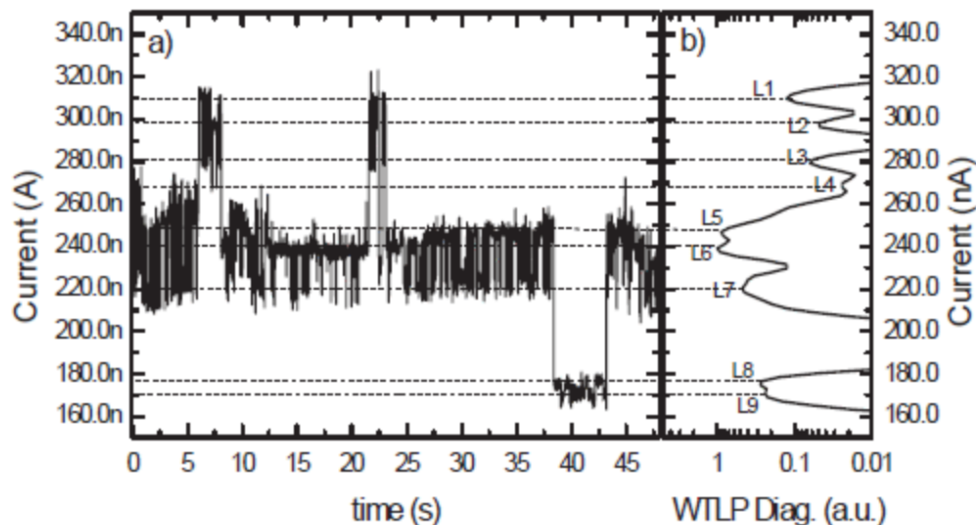

Figure 18: - (a) Typical multilevel RTN signal measured with a semiconductor parameter analyser. VAPP=1.25V, step time ~6ms and number of measured points 8000. (b) Trap levels obtained by using the W-TL method [10].

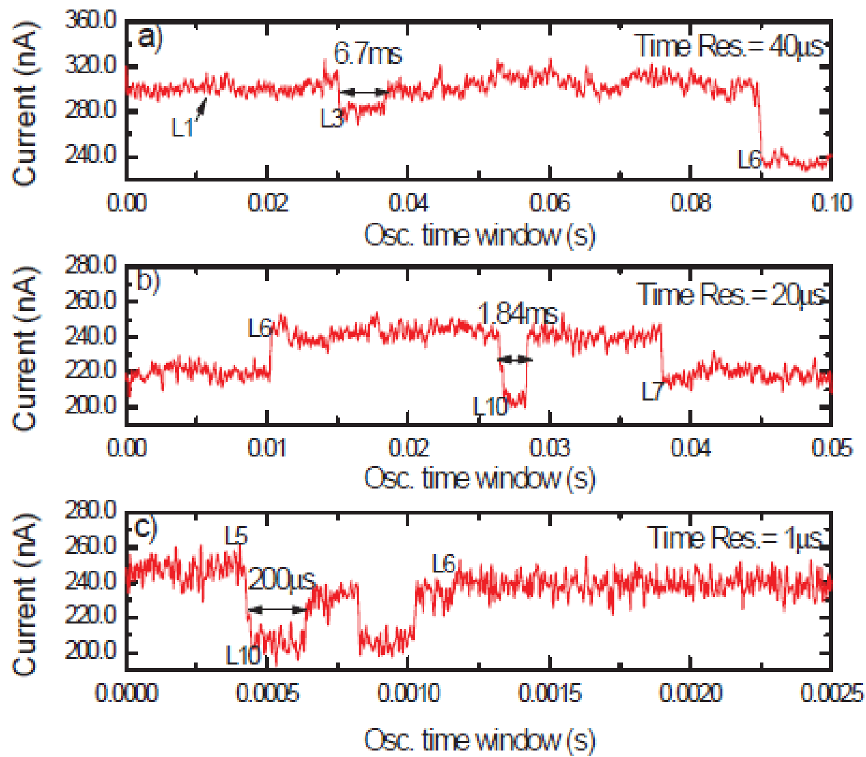

Figure 19: Oscilloscope traces captured in different time window, obtaining the interval time of a deffect [10].

1.3 Neural Networks

Neural networks are a mathematical computational model that structurally details the biological behaviour of a neural system. These networks are made up of a set of artificial neurons that are connected to each other and work together, without each one having a specific function. The key to such networks is to emulate all the possible connections of a neuron, of approximately 10^4. There are two modes of operation of an artificial synapse: the analogue mode and the digital mode (also known as binary mode).

In the former mode, the synaptic weight can be set from a range. This results in continuous control over the conductivity of the system. To increase the performance of the system, it is necessary to control the linearity and symmetry of the conductivity change, either in increasing or decreasing conductivity.

In the second mode, the binary mode, a comparison is made between the synaptic weight: high (HCS) or low (LCS). This simplicity makes the system more tolerant to conductivity variation. The consequence of this mode is that the system decreases its performance.

This type of network consists of identical nodes (neurons) that communicate with each other and are grouped in different layers: input and output layer and hidden layer. All nodes in a layer are connected between nodes in other layers by connections called synapses. Each of these connections has a different weight. The greater the relationship between connections, the more related the neurons are. The output value of each neuron is multiplied by this weight.

In Figure 20, an example of a neural network with n input and output layers and three hidden layers can be seen. The circles correspond to neurons (perceptrons). The lines correspond to synapses.

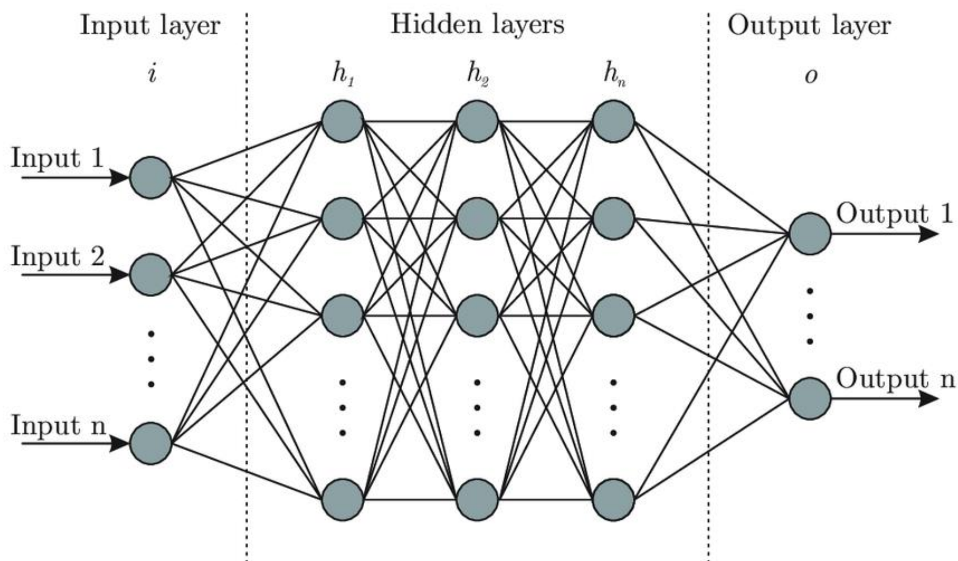

Figure 20: Diagram of a neural network with two input and output layers and three hidden layers.

The input layer collects the signals fed into the neural network and sends them to the hidden layer. In this last layer, all the mathematical alterations are made to the input signals due to the interaction of the perceptrons and their weights with the received signal. Finally, the resulting signal from the hidden layer reaches the output layer where linear

perceptrons are responsible for transmitting the information to the output of the neural network.

The goal of a network is to learn from previous experiences and to be able to solve any problem in real time, behaving as a biological brain would do. It is for this reason that this type of network is based on learning algorithms where new patterns are constantly being created so that these networks are capable of self-managing themselves.

1.4 MATLAB and NNstart

MATLAB is a programming language and numerical computing environment developed by MathWorks. This software allows the manipulation of matrices, the plotting of functions and data, the implementation of algorithms, the creation of user interfaces and the interconnection with programs written in other languages.

MATLAB is initially intended for numerical computation, but also uses an optional toolbox using a MuPAD symbolic engine that allows access to symbolic computation capabilities. There is also an additional package, Simulink, which adds multi-domain graphical simulation and model-based design for dynamic and embedded systems. MATLAB users come from various fields of engineering, science and economics.

Simulink is a MATLAB-based graphical programming environment for modelling, simulating and analysing multidomain dynamic systems. Its main interface is a graphical block diagramming tool and a customisable set of block libraries. It offers tight integration with the rest of the MATLAB environment and can drive MATLAB or be programmed from MATLAB. Simulink is widely used in automatic control and digital signal processing for multidomain simulation and model-based design [11][12].

NNstart opens a window with start buttons for the Neural Net Fitting, Neural Net Pattern Recognition, Neural Net Clustering and Neural Net Time Series applications. It also provides links to lists of datasets, examples, etc. All these applications allow to:

- Import data from a file, the MATLAB® workspace, or use one of the example datasets.
- Define and train a neural network.
- Evaluate network performance using mean squared error and regression analysis.
- Analyse the results using visualisation plots, such as autocorrelation plots or an error histogram.
- Generate MATLAB scripts to reproduce the results and customise the training process
- Generate functions suitable for deployment with MATLAB Compiler™ and MATLAB Coder™ tools, and export to Simulink® for use with Simulink Coder.

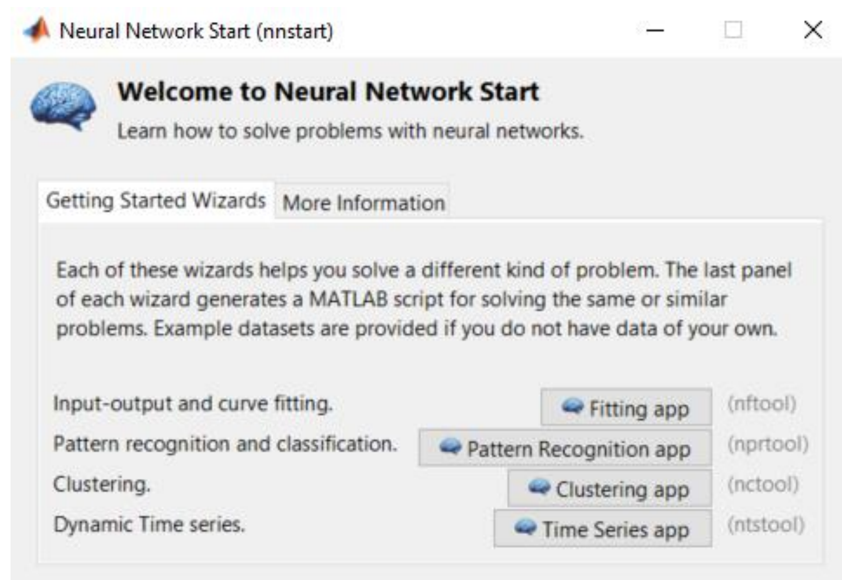

Figure 21: NNStart Interface.

However, there are some of the functions of these applications that are specific depending on the desired requirements or objectives:

- Neural Net Fitting and Neural Net Pattern Recognition: these applications make it possible to create, visualise and train a two-layer feed-forward network to solve data fitting problems. It also makes it possible to divide data into training, validation and test sets.
- Neural Net Clustering: this application provides the possibility to create, visualise and train networks of self-organising maps to solve clustering problems. It also

allows to analyse the results through visualisation graphs, such as distance between neighbours, weight planes, sample impacts and weight position.

- Neural Net Time Series: The Neural Net Time Series application allows you to create, visualise and train dynamic neural networks to solve three different types of non-linear time series problems. It also makes it possible to divide data into training, validation and test sets.

2. RTN trace generation

This chapter of the paper focuses on the creation of RTN traces. The most relevant characteristics and the parameters used to parameterise them will be described. Once these parameters have been explained, the changes that have been made will be detailed. To conclude the chapter, the different RTN traces obtained will be visualised and the effects caused by the changes will be analysed.

2.1 Characterisation of experimental RTN

In MOSFETs, RTN is associated with charge trapping in device defects. These defects can be inherent to the manufacturing process or created by ageing mechanisms triggered during device operation.

As explained above, the management and understanding of RTN signals has become of great interest as they allow the reliability of electronic devices to be determined. Some of the most characteristic parameters of RTN traces are explained below.

2.1.1 Gaussian distribution

In order to represent the levels of the RTN traces, Gaussian distributions have been used (Figure 22). These distributions are described by the following equation 5:

$$f_X(x) = \mathcal{N}\,(x;\ \mu, \sigma^2) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{1}{2}\left(\frac{x-\mu}{\sigma}\right)^2} \tag{5}$$

Where:

x random variable,
μ is the mean,
σ standard deviation,
and σ^2 vaciance.

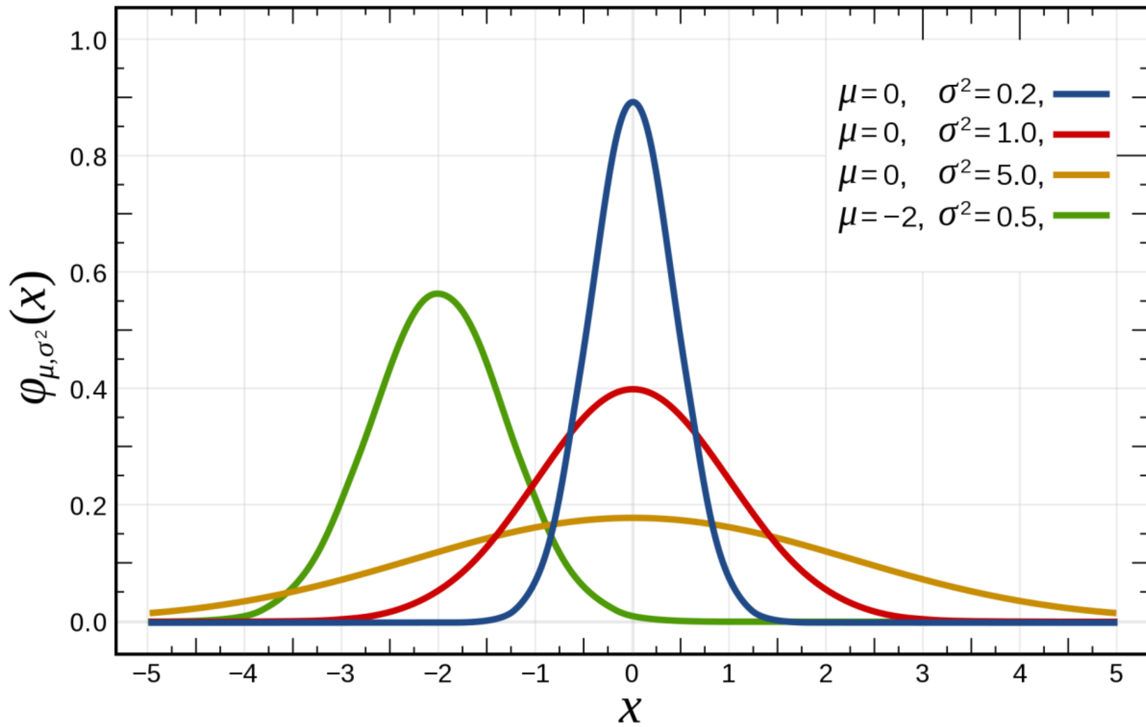

Figure 22: The red curve is the standard normal distribution.

Two Gaussian distributions are needed; one for the higher (upstream) values and one for the lower (downstream) values. For each of the generated RTN current levels, a Gaussian distribution is constructed. For this reason, two means and two standard deviations have to be defined, respectively.

The upstream parameters will be called emission parameters, and the downstream parameters will be called capture parameters. In the following section, these parameters will be detailed.

2.1.2 Signal noise

Noise is the result of various types of disturbances that tends to mask information when it is presented in its bandwidth. It is impossible to eliminate noise, as electronic components are not perfect, but it is possible to limit its value so that the quality of communication is acceptable.

In testing and modelling communication channels, Gaussian noise is used. A Gaussian process is a stochastic process, a collection of random variables indexed by time or space, such that each finite collection of these random variables has a multivariate normal distribution. Each finite linear combination of them is distributed following a normal distribution.

When $U = \left(\frac{x-\mu}{\sigma}\right)$ is normal with mean is 0 and variance 1:

$$f_U(u) = \mathcal{N}\ (u;\ 0{,}1) = \frac{1}{\sqrt{2\pi}} e^{-\frac{1}{2}u^2} \tag{6}$$

A special case is white Gaussian noise, where the values are identically distributed and statistically independent and therefore uncorrelated.

For this reason, additive white Gaussian noise (AWGN) has been added in the generation of RTN traces, as it allows the modelling of numerous natural, social and psychological phenomena. The noise value used in all cases is a mean equal to 0 ($n_{mean} = 0$) and a standard deviation $n_{sigma} = 10^{-8}$. These values can be seen in Table 3.

	Case1	**Case2**	**Case3**	**Case4_1**	**Case4_2**	**Case5**	**Case6_1**	**Case6_2**
$\boldsymbol{n_{mean}}$	0	0	0	0	0	0	0	0
$\boldsymbol{n_{sigma}}$	10^{-8}	10^{-8}	10^{-8}	10^{-8}	10^{-8}	10^{-8}	10^{-8}	10^{-8}

Table 3: Values of n_{mean} and n_{sigma} in the different cases

2.1.3 Number of defects

This parameter measures the number of defects or traps that a channel of a device may contain. Depending on the number of defects the signal will have a deterministic set of values.

With the following equation, equation 7, the possible states in which a trace can be found can be obtained:

$$NEP = 2^N \tag{7}$$

Where:
NEP is the number of possible states,
N is the number of defects.

Since the value of the defects is binary, 0 or 1, its behaviour can be parameterised through a Poisson distribution.

2.1.4 Jumps

A jump marks the numerical contribution of a defect in a signal. That is, each defect in a transistor will have a jump value associated with it. It is for this reason that RTN traces have the same number of jumps as defects. This parameter can be identified graphically as a change of state.

2.1.5 Offsets

The offset of a signal is the continuous base value on which all other values are found. In the case of RTN traces, this parameter is the value obtained when the carrier flux is constant.

This parameter also follows the Gaussian distribution as the noise does. Therefore, it is also necessary to determine a mean value, a standard deviation and a variance. To find the mean value and the variance of the Offset, it is necessary to find the value closest to zero for each trace.

2.2 Definition of the criterial for the generation of RTN traces

The main objective of this point is to create different RTN traces with different characteristics. Considering that RTN traces can be parameterised according to certain intrinsic specifications, it has only been necessary to modify them. In the following, the

different parameters that have been considered in the generation of these traces will be defined

2.2.1 Number of traces to be generated and number of samples per RTN traces

As is well known, both the experimental and mathematical worlds are subject to limitations because physical and computational resources are not infinite. This is where it becomes important to correctly size the amounts of data to be worked with.

For this reason, different tests have been carried out by changing the number of traces (N). On the other hand, the number of samples (nt) of each of the traces has remained the same for all cases. In the following table, Table 4, the different values of the traces and the value of the number of samples per trace chosen can be seen:

	Case1	**Case2**	**Case3**	**Case4_1**	**Case4_2**	**Case5**	**Case6_1**	**Case6_2**
$\boldsymbol{N}$	10000	3750	3750	2500	2500	3750	2500	1250
$\boldsymbol{nt}$	10000	10000	10000	10000	10000	10000	10000	10000

Table 4 Values of N and n in the different cases.

2.2.2 Transmission and capture time

The period is the time elapsed between two equivalent points in the waveform. The sampling rate is defined as the number of samples per unit time taken from a continuous signal to produce a discrete signal, during the process necessary to convert it from analogue to digital.

For the generation of RTN traces, two periods of the signal have been used: the emission time ($T_{e_{mean}}$) and the capture time ($T_{c_{mean}}$). This concept considers the geometry of each defect and the capacity to retain or release the carriers circulating in the channel.

The different time values are listed in the following table (Table 5):

	Case1	Case2	Case3	Case4_1	Case4_2	Case5	Case6_1	Case6_2
$T_{c_{mean}}$	0.5	0.5	0.5	0.3	0.3	0.5	0.15	0.05
$T_{e_{mean}}$	0.5	0.5	0.5	0.1	0.003	0.5	0.3	0.3

Table 5: T_c and T_e values in the different cases.

2.2.3 Transmission and capture mean

Theoretically, the mean is obtained from the sum of all its values divided by the total number of addends. The mathematical expression of this concept is given in the equation 8:

$$\mu = \frac{1}{N}\sum_{i=0}^{N-1} x_i \tag{8}$$

In the practical case, this average has been used to generate normal distributions of the emission (me_{mean}) and capture (mc_{mean}) processes. The different values of these parameters are shown in Table 6:

	Case1	Case2	Case3	Case4_1	Case4_2	Case5	Case6_1	Case6_2
mc_{mean}	10^{-5}	10^{-5}	10^{-5}	10^{-5}	10^{-5}	$3 \cdot 10^{-6}$	10^{-5}	10^{-5}
me_{mean}	$1.8 \cdot 10^{-5}$	10^{-5}	$1.8 \cdot 10^{-5}$	$1.8 \cdot 10^{-5}$	$1.8 \cdot 10^{-5}$	$1.8 \cdot 10^{-5}$	$1.8 \cdot 10^{-5}$	$1.8 \cdot 10^{-5}$

Table 6: me_{mean} and me_{mean} values in the different cases.

2.2.4 Standard deviation and variance

The standard deviation is a measure of the amount of variation or dispersion of a set of values. A low standard deviation indicates that the values tend to be close to the mean, while a high standard deviation indicates that the values are spread over a wider range.

This parameter is most often represented by the lower case Greek letter sigma σ. The standard deviation of a random variable is the square root of its variance. Equation 9 shows the formula for its discrete calculation:

$$\sigma = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(x_i - \mu)^2} \tag{9}$$

In contrast, the variance is a measure of dispersion defined as the expectation of the square of the deviation of that variable from its mean. It is represented by letter sigma squared σ^2. Its mathematical expression is the following (equation 10)

$$\sigma^2 = \frac{1}{n-1}\sum_{i=1}^{n}(x_i - \bar{x})^2 \tag{10}$$

2.3 Table of RTN traces parameters

The following table lists the different parameter values used for the generation of the RTN traces. In this way, it is easier to visualise the applied changes.

	Case1	**Case2**	**Case3**	**Case4_1**	**Case4_2**	**Case5**	**Case6_1**	**Case6_2**
$\boldsymbol{N}$	10000	3750	3750	2500	2500	3750	2500	1250
$\boldsymbol{nt}$	10000	10000	10000	10000	10000	10000	10000	10000
$\boldsymbol{T_{c_{mean}}}$	0.5	0.5	0.5	0.3	0.3	0.5	0.15	0.05
$\boldsymbol{T_{e_{mean}}}$	0.5	0.5	0.5	0.1	0.003	0.5	0.3	0.3
$\boldsymbol{mc_{mean}}$	10^{-5}	10^{-5}	10^{-5}	10^{-5}	10^{-5}	$3 \cdot 10^{-6}$	10^{-5}	10^{-5}
$\boldsymbol{me_{mean}}$	$1.8 \cdot 10^{-5}$	10^{-5}	$1.8 \cdot 10^{-5}$	$1.8 \cdot 10^{-5}$	$1.8 \cdot 10^{-5}$	$1.8 \cdot 10^{-5}$	$1.8 \cdot 10^{-5}$	$1.8 \cdot 10^{-5}$
$\boldsymbol{n_{mean}}$	0	0	0	0	0	0	0	0
$\boldsymbol{n_{sigma}}$	10^{-8}	10^{-8}	10^{-8}	10^{-8}	10^{-8}	10^{-8}	10^{-8}	10^{-8}

Table 7: Table of RTN signal parameters.

2.4 Results of generated RTN traces

In this section of the chapter, the generated RTN signals are plotted. As mentioned above, different cases have been generated to test the effects that can be caused by changes in the values that parameterise the RTN signals. Different plots have been collected from each of the generated traces.

The first graph corresponds to the number of RTNs generated. As explained above, the parameter used for this purpose is N. Therefore, as many traces as indicated will be obtained

The second, always identified as (a) of the combination of 4 graphs, shows the behaviour of the current during the time that voltage is applied together with the values of the drain current.

The third one, always identified as (b) of the combination of 4 graphs, shows the different levels of the RTNs.

The fourth, identified as (c) of the combination of 4 plots, shows a Gaussian that reports the number of points at the different levels of maximum and minimum current that gives rise to the RTN.

The last of the plots, identified as (d) of the combination of 4 plots, details the Weighted Time-Lag Plot (WTLP) of the RTN signals. This technique allows the RTN levels along with the sub-levels, as well as the steps between these levels, to be easily plotted.

2.4.1 Case 1 graphs

It can be deduced that this is an ideal case. As can be seen in Table 7, this is the model that generates the most traces (Figure 23).

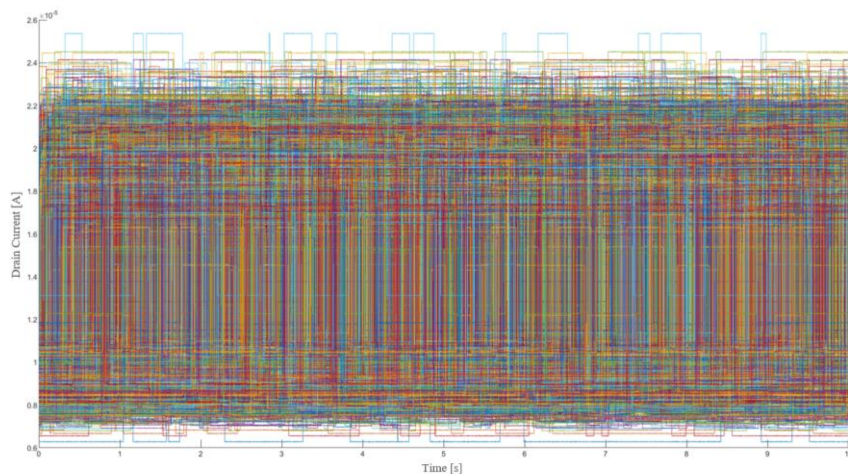

Figure 23: RTN signals case 1.

On the one hand, If the group of graphs is analysed, the noise does not noticeably affect the samples as the RTN signal can be identified in Figure 24 (b).

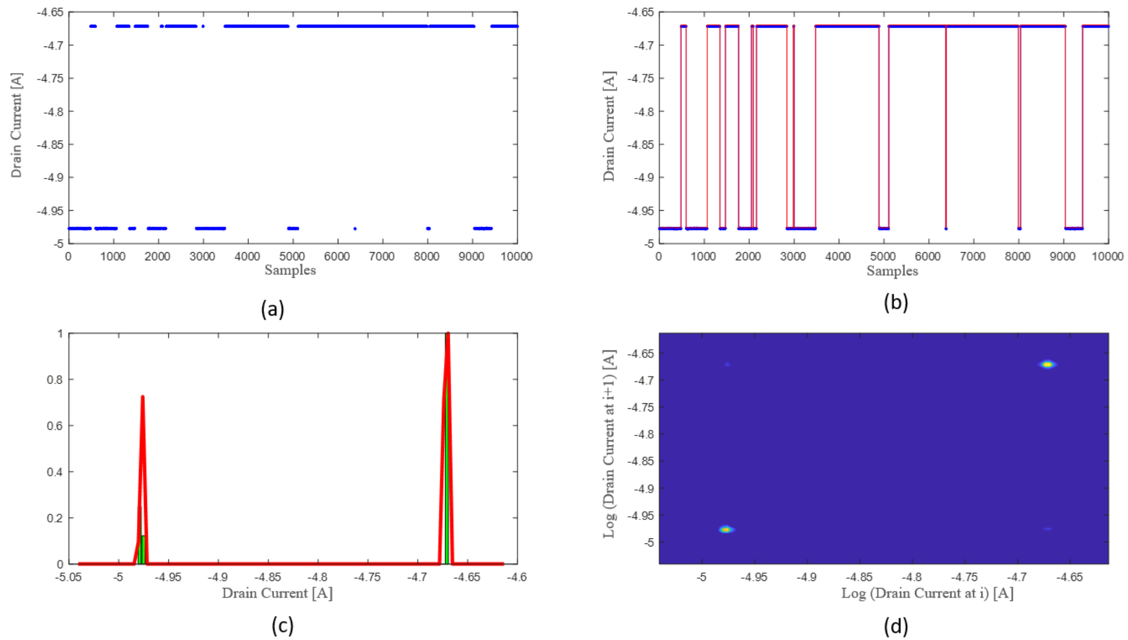

Figure 24: Behaviour of drian current case 1.

On the other hand, it is easy to identify the two Gaussians generated in Figure X (c) which clearly show that the maximum current occurs at approximately $I_D = -4.75\ A$ and the minimum current at approximately $I_D = -5.5\ A$.

The WTLP also identifies these two current levels and details them in yellow. The transitions between levels are reflected in the verticals of the diagonal peaks, marked in a much lower yellow colour according to the number of transitions present in the $I - t$ trace.

2.4.2 Case 2 graphs

The changes introduced have been applied to the number of traces generated and the average emission. Both values have been reduced. With regard to the number of traces, the number of traces generated has been reduced from 10000 traces ($N_1 = 10000$) to 3750 ($N_2 = 3750$). In the case of the emission mean, it has been forced to have the same

value as the capture mean, changing its initial value from $me_{mean_1} = 1.8 \cdot 10^{-5}$ to $me_{mean} = mc_{mean} = 10^{-5}$.

The graphs for case 2 are shown below.

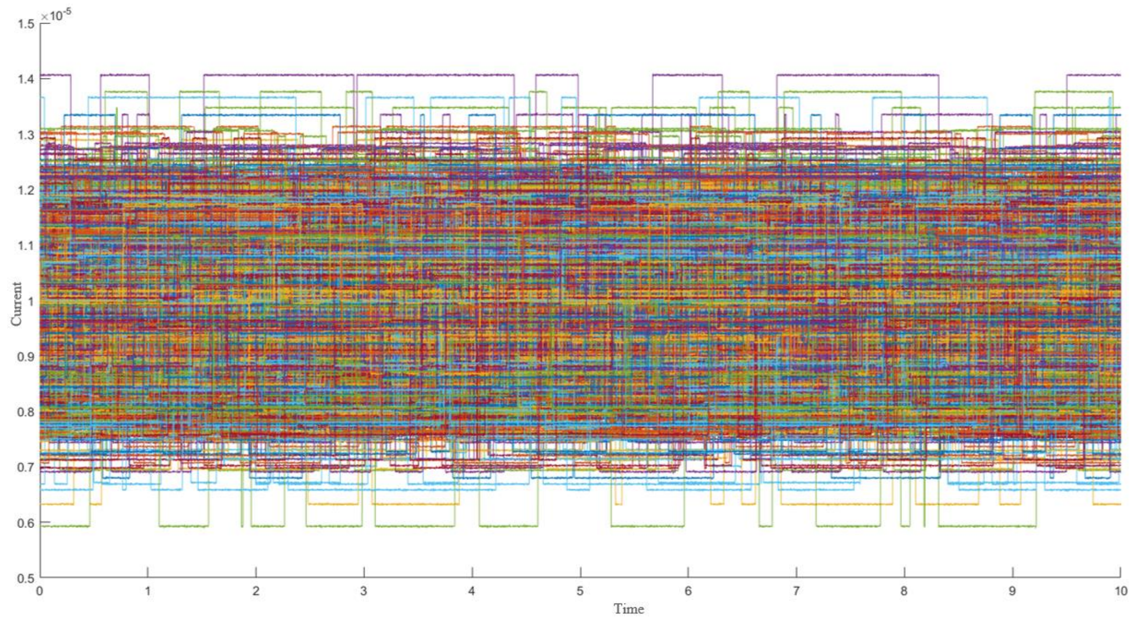

Figure 25: RTN traces case 2.

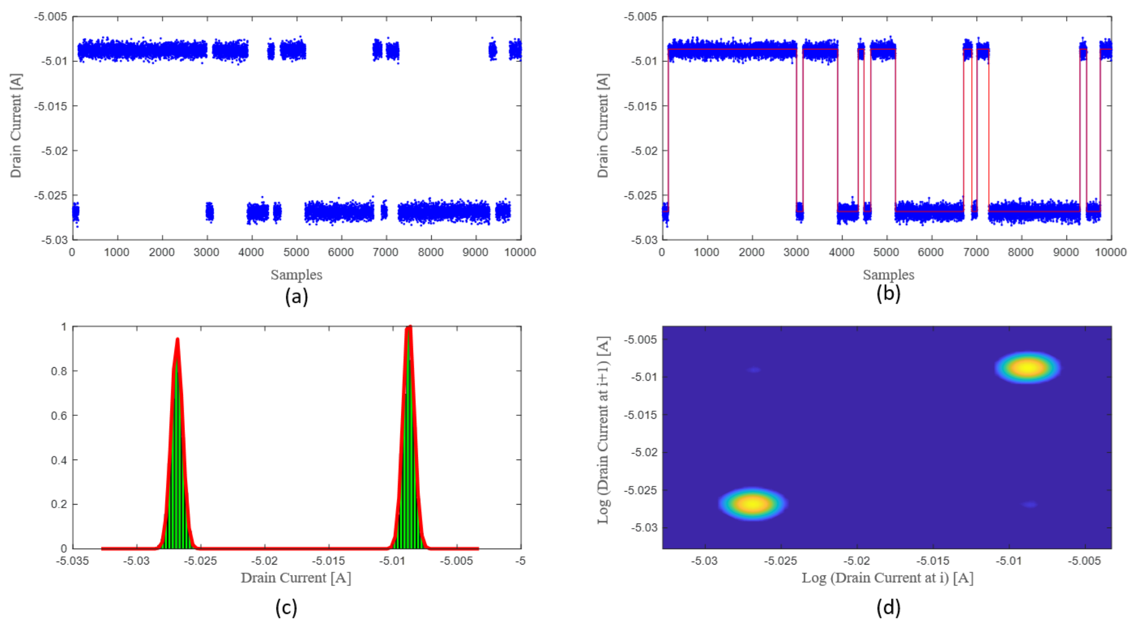

Figure 26: Behaviour of drian current case 2.

At first glance, quite a few changes can be perceived with respect to case 1.As expected, in Figure 25, much less RTN signals can be observed due to the reduction of the N value.

In the following figure, Figure 26, it can be seen that the noise now has more weights in the samples obtained. It is also visible that the current levels are now stronger and more similar. This causes that in the WTLP the intensities are plotted with a more intense yellow colour, while the transitions have the same value as in the previous case.

2.4.3 Case 3 graphs

In this case, the average capture and emission used to generate the noise have been forced to have the same value. For this reason, that the effect of the noise is very noticeable in the graphs obtained.

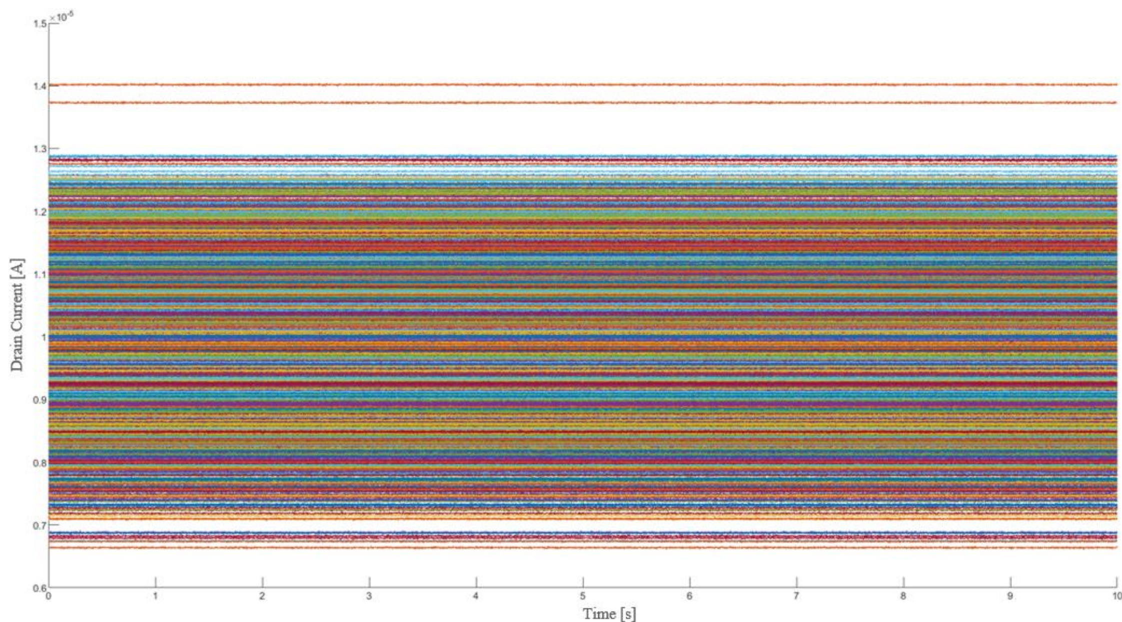

Figure 27: RTN signals case 3.

In contrast to the previous cases, in Figure 27, only noise is observed. This effect is due to the fact that the noise masks the RTN traces and overlaps their values.

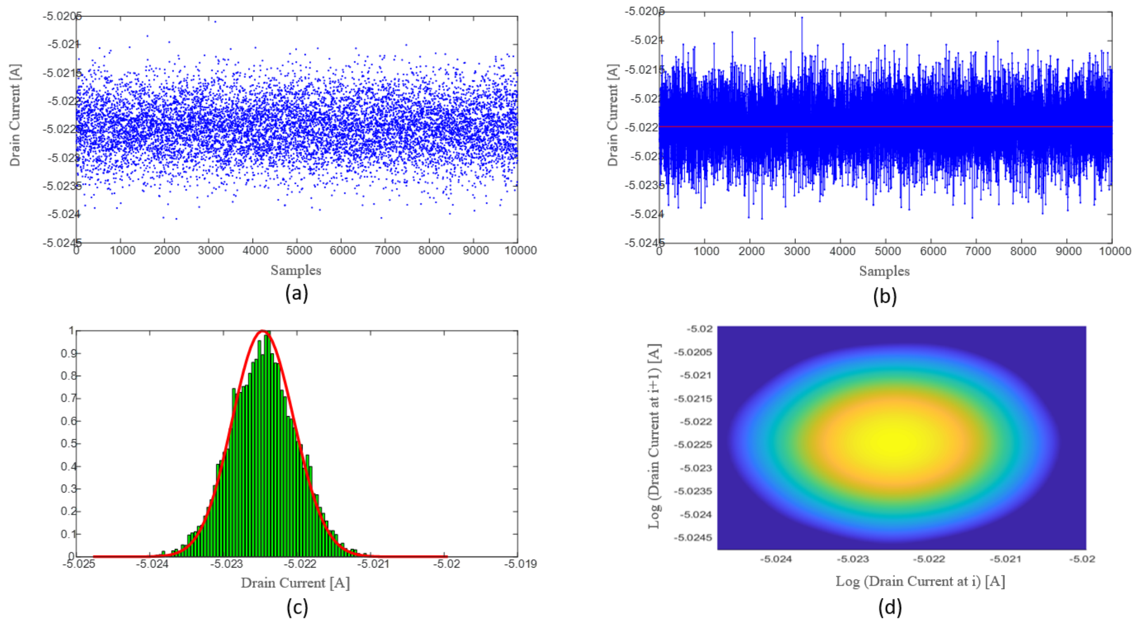

Figure 28: Behaviour of drian current case 3.

Analysing the following representations of Figure 28; it is intuitive that the effect produced by the noise is also noticeable. In representations (a) and (b) it is impossible to identify any shape reminiscent of the RTN traces.

In (c) only 1 Gaussian is visible, since, as mentioned above, both averages for the generation of the emission and capture noise are identical. This effect also causes the WTLP to only be able to.

2.4.4 Cases 4 graphics

Unlike the previous ones, in this section of the RTN trace generation, two sections have been generated, where the emission time has been forced to be always lower than the capture time. Another of the changes has been to reduce, once again, the number of traces generated, setting it at $N = 2500$ traces.

In the following, the figures obtained above will be plotted and the results will be compared between the two sections.

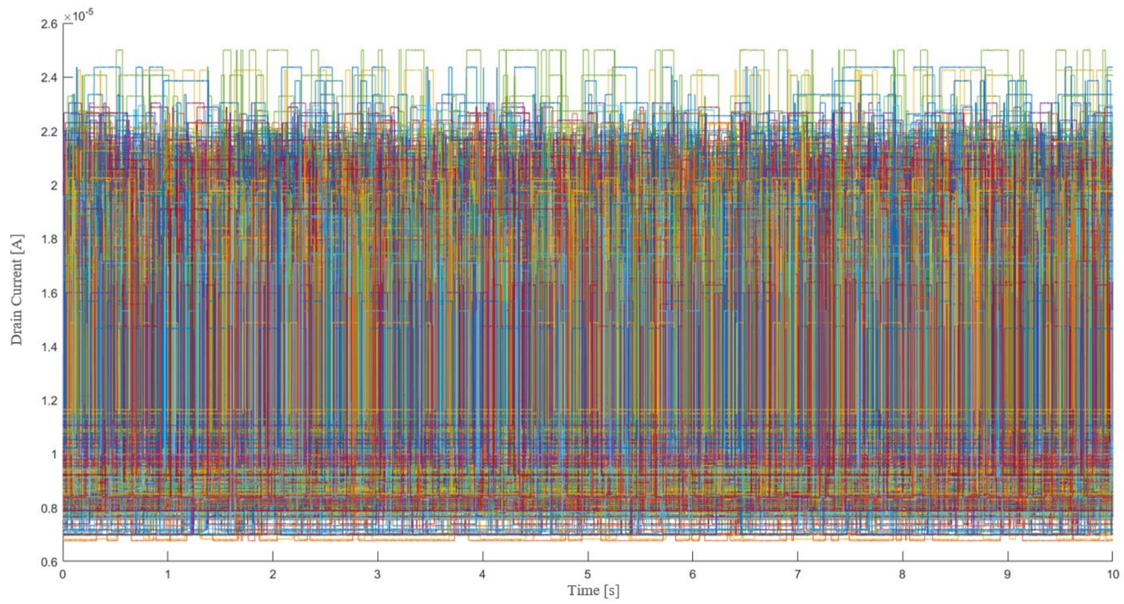

Figure 29: RTN signals case 4 (section 1).

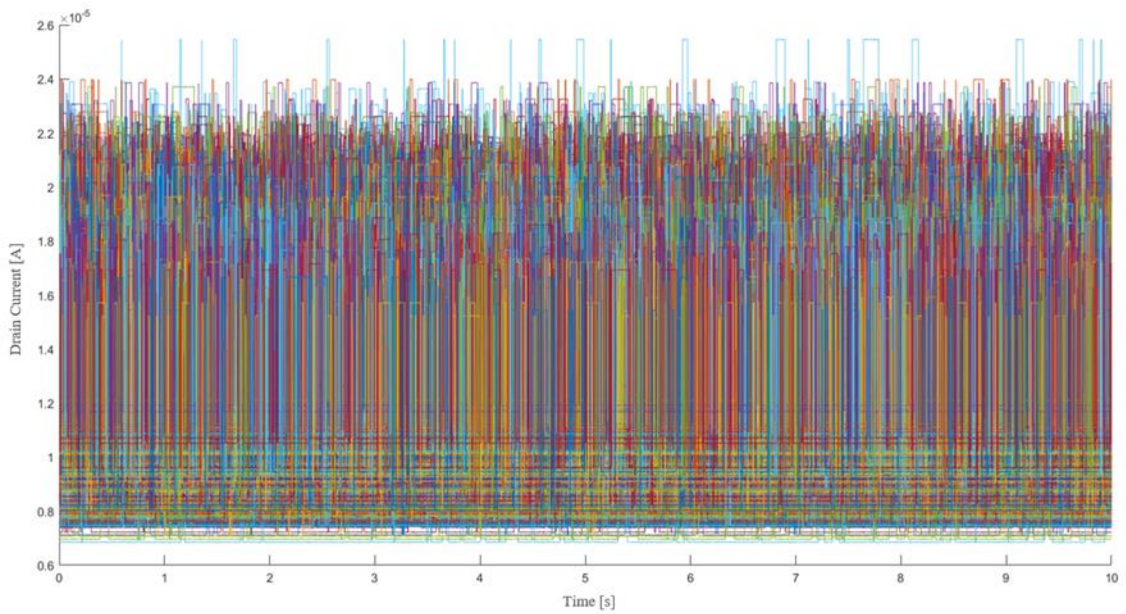

Figure 30: RTN signals case 4 (section 2).

If you compare the two figures above with each other, Figure29 and Figure 30, it is difficult to see any difference. On the other hand, if it is compare them with the cases mentioned above, it can be perceived a decrease in the traces generated.

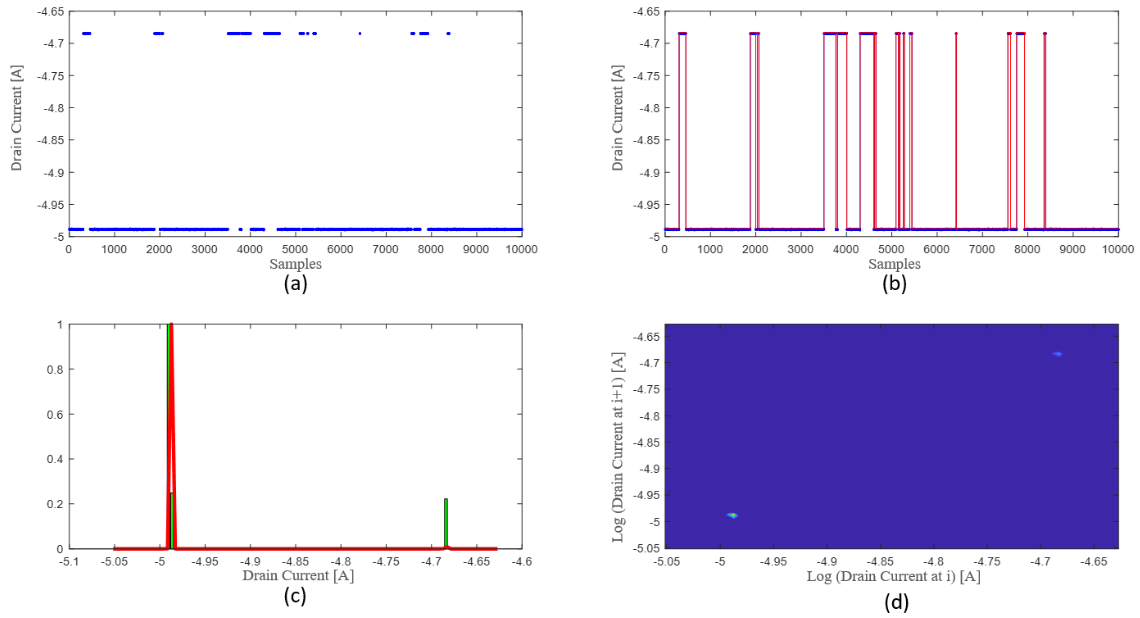

Figure 31: Behaviour of drian current case 4 (setion 1).

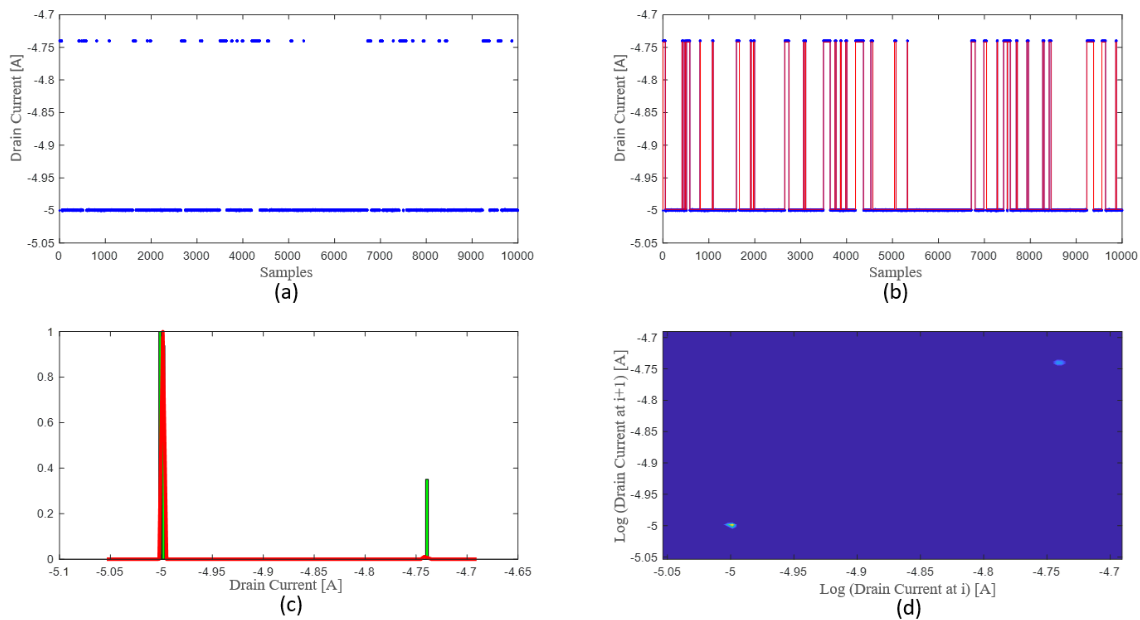

Figure 32: Behaviour of drian current case 4 (section 2).

When comparing the 4 graphical representations in Figures 31 and 32, the applied changes in the values of emission time and capture time are more evident.

f figures (a) and (b) of both figures are compared, it can be seen that in Figure 32 there are more changes of state. This effect is due to the reduction of T_e, since it goes from T_e=0.1, case Figure 31, to T_e=0.03 ,case Figure 32.

Finally, if we analyse graph (c) it can see that the current levels when $T_e = 0.03$ are higher. Consequently, the WTLP (d) in Figure 32, the yellow colour representing the current level is intensified.

2.4.5 Case 5 graphics

This model is particularised by the change in standard deviation. In all other models, this value is $m_{c_{mean}} = 10^{-5}$. In the current model, the value of the standard deviation is $m_{c_{mean}} = 3 \cdot 10^{-6}$.

In addition, the number of traces to be generated is reduced from $N = 10000$ traces to $N = 3750$, thus recovering the number of traces of cases 2 and 3.

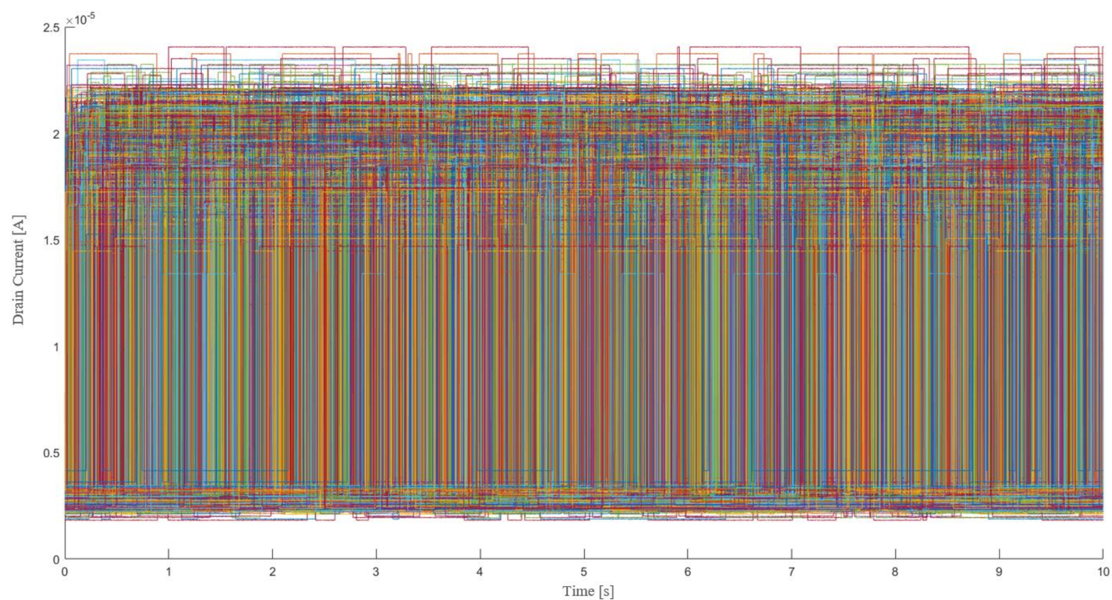

Figure 33: RTN signals case 5.

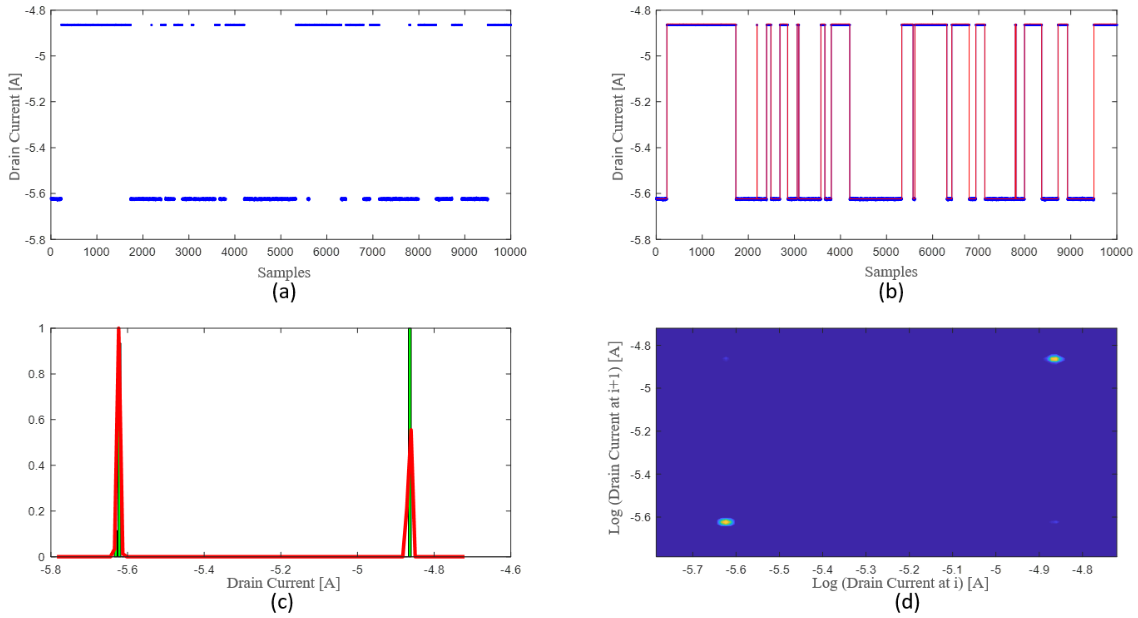

Figure 34: Behaviour of drian current case 5.

In the different representations of Figure 34, it can be seen that the noise is not very noticeable in the samples obtained.

2.4.6 Cases 6 graphics

As in case 4, two possible scenarios have also been generated in this case. Although in this case, the value that is reduced is that of T_c and not T_e. The values of N, which marks the number of traces to be generated, have also been modified.

In the first scenario, a total of $N = 2500$ traces were generated with a capture time $T_c = 0.15$ and an emission time of $T_e = 0.3$. In the second scenario, $N = 1250$ traces were created with a capture time $T_c = 0.05$ and an emission time of $T_e = 0.3$.

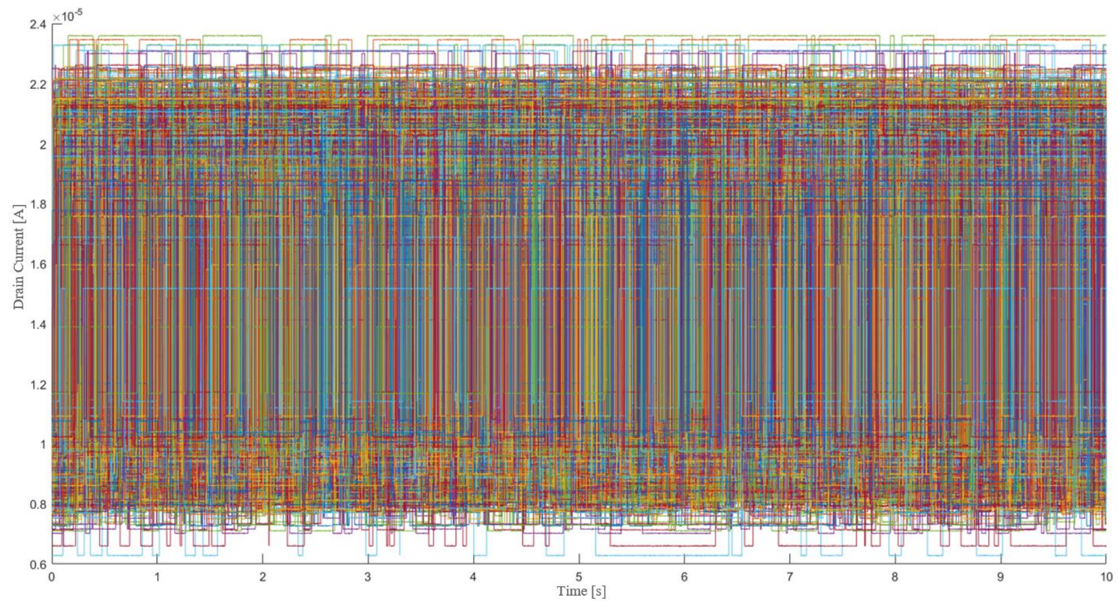

Figure 35: RTN signals case 6 (section 1).

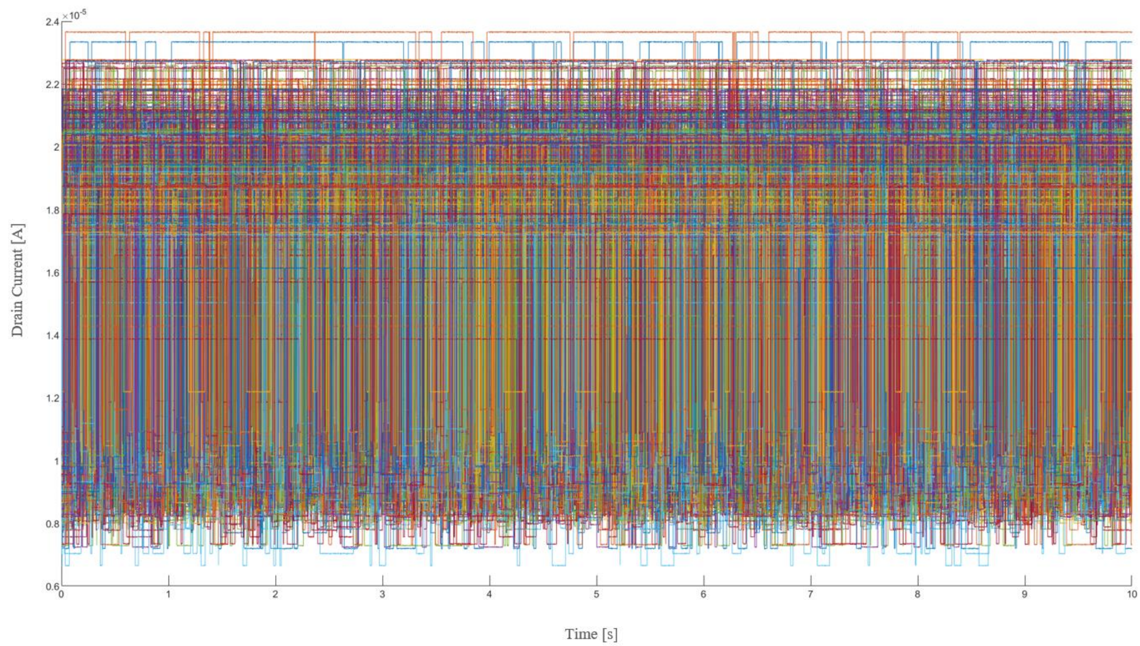

Figure 36: RTN signals case 6 (section 2).

As with case 4, it is also difficult to identify differences in this case. Even so, as explained above, it can be seen that in Figure 36 the number of traces is much smaller than in Figure 35.

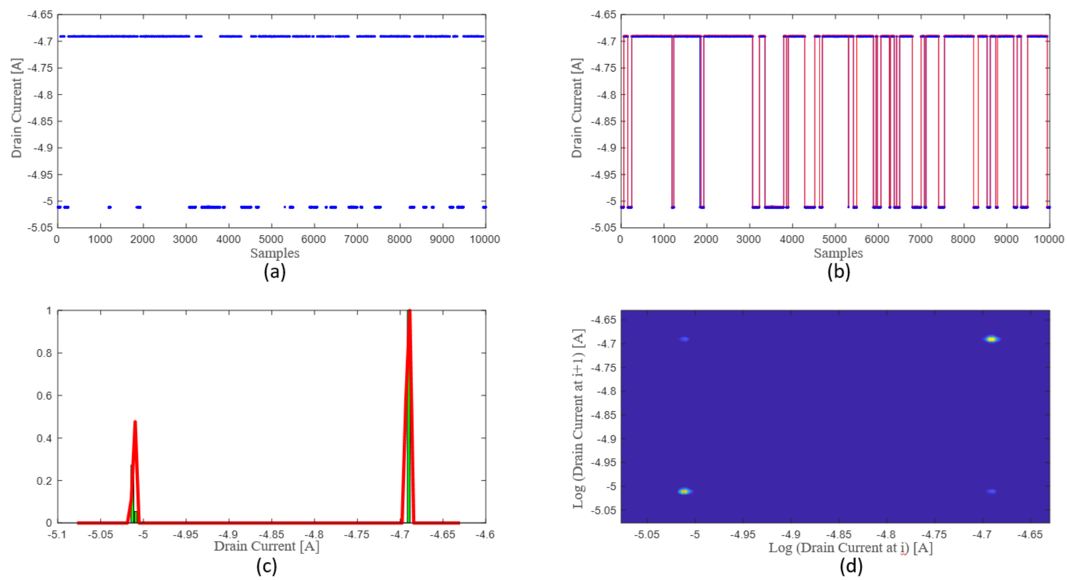

Figure 37: Behaviour of drian current case 6 (section 1).

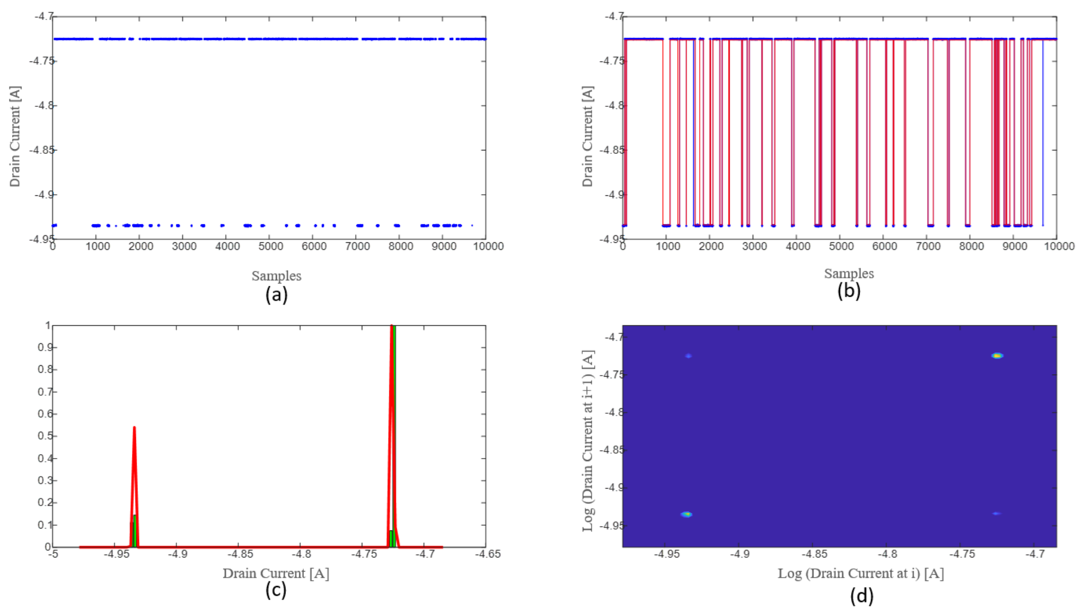

Figure 38: Behaviour of drian current case 6 (section 2).

When comparing Figures 37 and 38, some differences can be quickly detected. The first one is detected by looking at the graph (b) of both figures: reducing the value of T_c has caused the minimum current value to appear in smaller intervals. This event is the opposite of what occurred in case 4, as it was the T_e that was reduced and the maximum current value that appeared in smaller intervals. The second is detected when analysing the graphs (c), as the levels in Figures 38 are higher.

3. Conversion of RTN traces to binary codes

Once the different RTN traces have been obtained, the next step is to convert them into binary codes. To do this, two methods have been developed: a method based on the average current value and a method based on captures and emissions.

The main objective of these conversion methods is data compression. The reduction of data allows storage capacity to be gained. On the other hand, it is obvious that some accuracy is lost. However, this loss of accuracy is acceptable for the purpose of this work.

3.1 Method based on the average current value

3.1.1 Procedure

The objective of this point is to establish a mean value, marked in red in Figure X, for each of the traces generated. Once this average value has been obtained, it will be discretised by assigning a 1 to values above this average value and a 0 to values below it. Therefore, the bit allocation is determined by the value of the level.

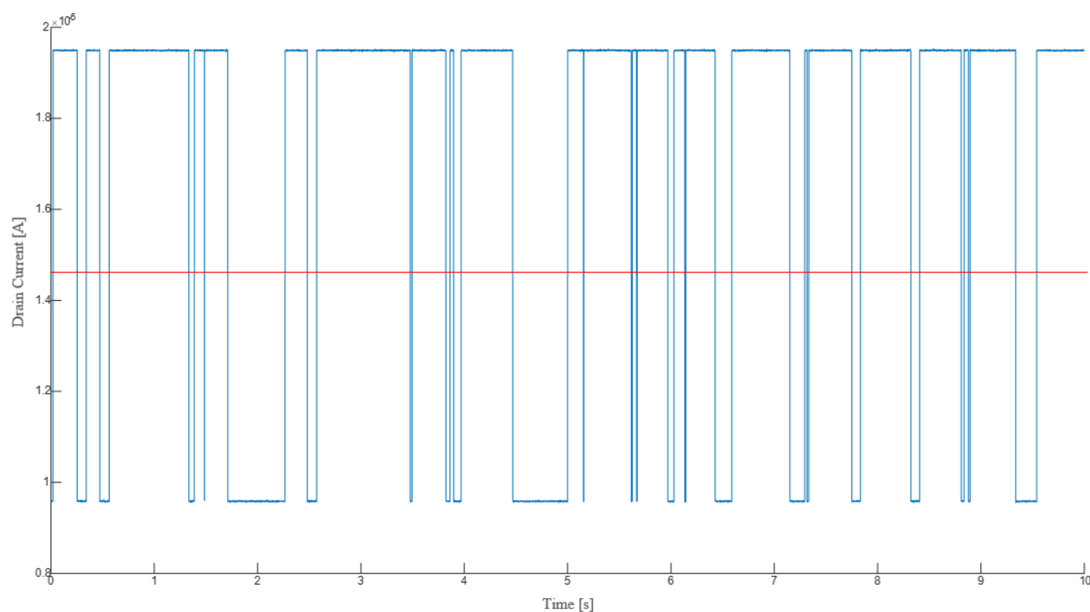

Figure 39: Example of balanced RTN.

In summary, it can be deduced that the more samples the files working with this method have, the more accurate they will be as the mean value will be more accurate and the easier it will be.

3.2 Catch-emission method

3.2.1 Procedure

When an RTN trace does not have approximately the same number of points at the high level as at the low level, it is unbalanced. An example of such a trace is shown in Figure 39.

Applying the average current value method would result in many more 0's than 1's, since the time the signal is at the low level is much longer than the time it is at the high level.

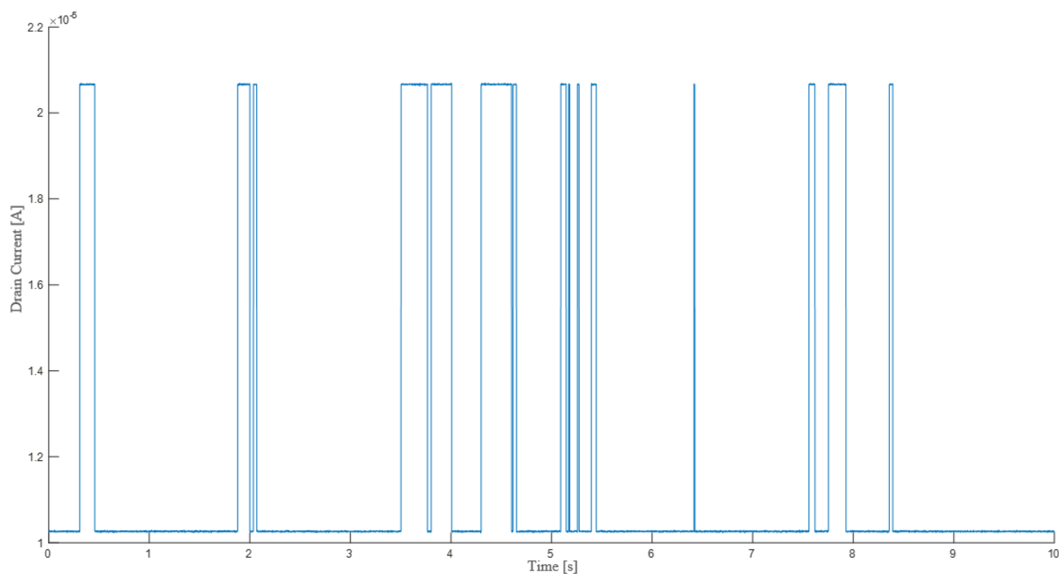

Figure 40: Example of unbalanced RTN.

For this reason, an alternative method must be used. Bit 0 or 1 will now be set only when there is a transition, i.e. a change of state. The value of this bit will be maintained until the next state change. Therefore, no matter how much more frequent one of the signal levels is than the other, a balancing between the bits will be obtained. Unlike the mean value method, this can be a good method when there are only a few samples.

4. Binary code validation

The last step before training a neural network is to develop a method that is able to determine whether a trace is correct or not.

By having the generated RTN traces converted into bits, it has been easier to develop a method for the validation of these signals. This method has to be able to certify both bit traces based on the average current value and those based on the capture and emission method.

This method and the results obtained once it is applied to the generated traces will be explained in more detail below.

4.1 Method based on the probability of followed bits

Once the generated traces have been transformed into bits, it only remains to develop a validation method for these traces. Considering that two procedures have been used in the conversion of RTN signals to bits, the validator will have to be able to perform its function in both procedures.

As the name indicates, this method focuses its resources on finding the probability between equal binary symbols. RTN traces have been assigned two bits: 1 and 0.

Therefore, the probability that once a 1 or a 0 appears, the same bit will appear again is to be calculated. This concept can be defined mathematically by the following equation (Equation 11):

$$f(n) = \frac{1}{2^n} \tag{11}$$

Where:

n is the number of consecutive times the same bit can appear.

Thus, each time n is increased, the probability of the same bit appearing decreases. A perfect case of this mathematical function can be seen in Figure 41.

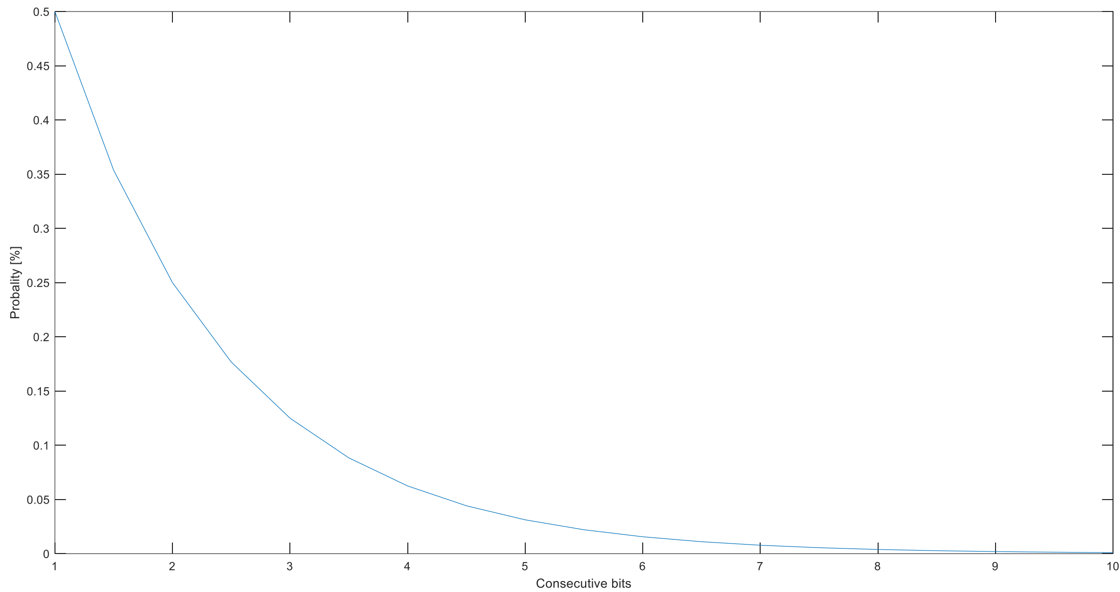

Figure 41: Function $f(n) = \frac{1}{2^n}$.

Considering RTN traces are governed by random processes, the graph obtained will not be the same as the one depicted in Figure 41, but it should resemble it.

For this reason it is necessary to provide a margin of error to the measurements obtained. Four degrees of probability ($n = 4$) have been established, assuming that they are already sufficient to validate that the trace follows the Equation 11. The margin of error for each n-value can be seen in the following table (Table 8):

n	**Probabilities (P_{nb})**	**Range of probabilities**
1	$P_{1b} = 50$	$[42.5 < P_{1b} < 57.5]$
2	$P_{2b} = 25$	$[21 < P_{2b} < 29]$
3	$P_{3b} = 12.5$	$[11 < P_{3b} < 14]$
4	$P_{4b} = 6.25$	$[5 < P_{4b} < 7.5]$

Table 8: Table of probabilities as function of n.

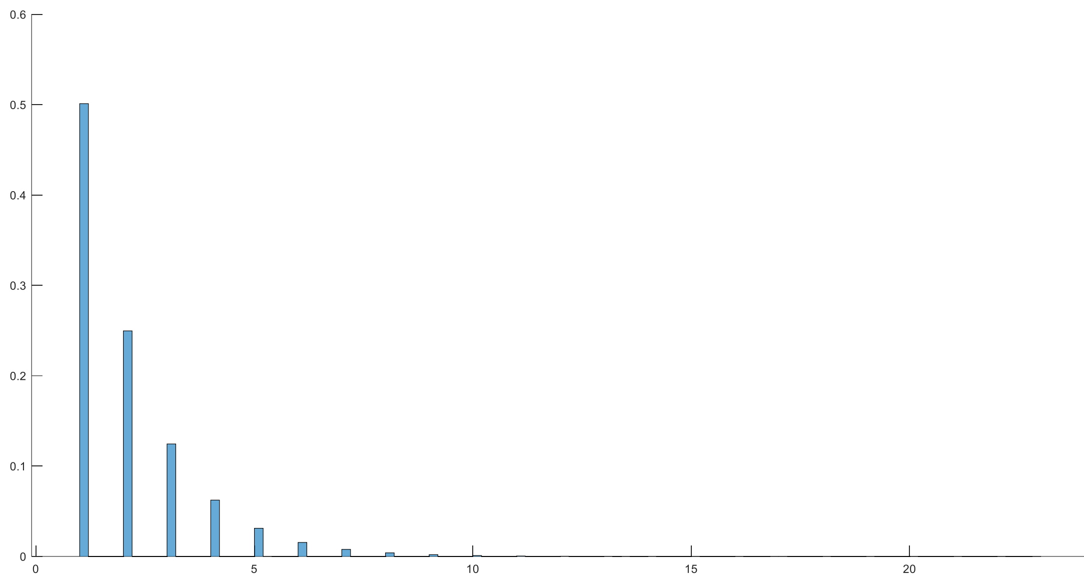

Figure 42: MATLAB function $f(n) = \frac{1}{2^n}$.

The validation process for case 3 is shown in Figure 42 below. As can be seen, the shape of the function follows the same pattern as the one depicted in Figure 41.

5. Neural network training

The last procedure of this work is to train the neural network. To do it, RTN traces had to be generated, converted to binary codes and validated. Once the necessary validations have been made, all the information has to be prepared for training the neural network. To achieve it, the inputs and outputs of the neural network must be organised and a criterion must be established to assess whether the training of the network has been correct or not.

As mentioned above, the MATLAB application NNStart will be used to train the network. This application uses, by default, 70% of the data to train the network, 15% to measure the generalisation of the network and to stop training when the generalisation stops improving, and 15% to provide an independent measure of the network performance during and after training. The application allows the last two percentages to be changed according to the requirements of each user.

It is also possible to save the weights of the network, its architecture and then continue to use the weights resulting from the network training.

5.1 Code groups used

The initial idea of this work was to train the neural network with different traces of different lengths. Due to limited time and resources, it was decided to choose a more practical and simplified case.

The chosen case is case 3. As it has been seen in section 2.4, among all the cases, it is the most realistic case due to the fact that noise is more present in the samples obtained. In some of the traces generated with the specifications of case 3, it could be observed that the noise was capable of masking the samples. This effect results in the generation of invalid traces. For this reason, by choosing the traces generated in case 3, the training of the neural network will be more realistic.

5.2 Network input data preparation

To do the neural network training well, it is necessary to have a set of valid inputs. To achieve it, it has been decided to use the Weighted Time-Lag Plot (WTLP) of the binary code and to divide the obtained maps into columns.

As explained above, the WTLP is able to obtain a three-dimensional graphical map that takes into account the position of each data item and its weight (times it is repeated). Once WTLP has been applied, the data are synthesised in a matrix that represents each of the RTN traces generated. The problem now arises because the data are synthesised in the form of a matrix and it is of interest that they are synthesised in the form of a vector. Two three-dimensional WTLPs are shown in the following figures (Figure 43 and 44).

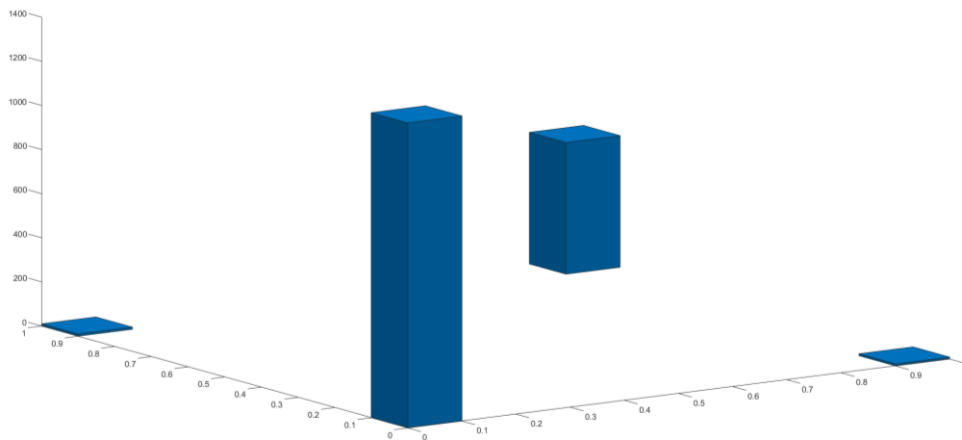

Figure 43: Binary WTLP of case 2.

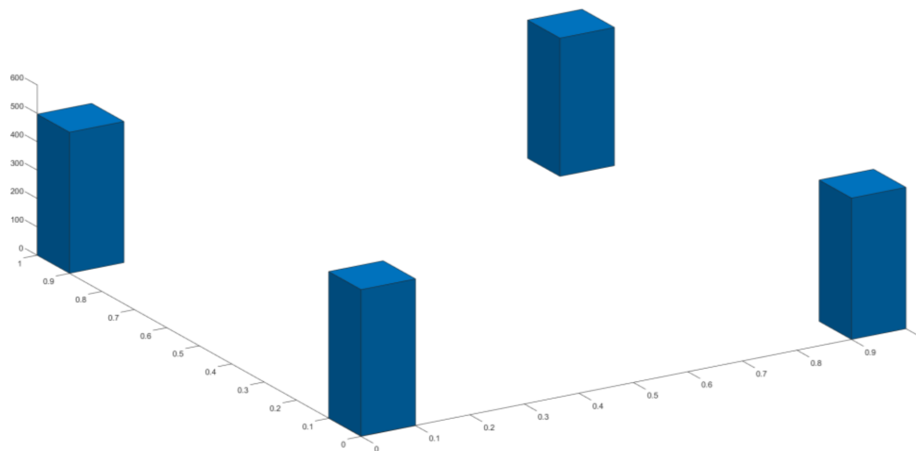

Figure 44: Binary WTLP of case 3.

To achieve it, is neceesary to rotate the data columns of the matrix and concatenating them into a vector. This methodology is easier to understand with a simple example: 3x3 matrix.

In the first iteration, the first column of the matrix is copied and moved to the first three positions of the output vector. In the second iteration, the second column of the matrix is copied and moved to positions 4, 5 and 6 of the vector. Finally, the third column of the matrix shall be copied and moved to positions 7, 8 and 9 of the output vector.

5.3 Network output data preparation

Once the inputs to the neural network have been prepared, it is important to do the same for the outputs. In order to be able to validate which traces are correct and which are not, the neural neutwork must be trained with codes that have valid and invalid traces.

As mentioned above, the method used to validate the traces will be the probability of consecutive bits. If a trace does not follow the shape of Figure 41, it means that the trace is incorrect.

Once the output vector has been configured, as explained in the previous section, validation must be carried out. For each of the columns of the input vector, a 01 will be assigned if the trace is adequate, and a 10 if the trace is not.

Although this way the number of cases obtained is greatly reduced and information is lost, this discretisation also allows the neural network training to be faster.

5.4 Results of training

As mentioned above, the MATLAB application NNStart has been used to train the neural network. This application allows us to obtain different graphs that describe the different behaviours of the trained traces.

The NNStart application, specifically the Pattern Recognition app, allows different graphs describing the behaviour of the trained traces to be obtained. It also allows the performance of the network to be evaluated by means of the cross-entropy and confusion matrices.

In the trainings carried out, the default percentage has been used. That is, 70% of the data will be used for training, 15% for validation and 15% for testing.

5.4.1 Training specifications

Once the data has been organised, the network training can be executed. But first, the different information that can be obtained with the Pattern Recognition app will be described. The graphs obtained with this application are the following ones:

- Performance: This graph plots the error values of the training log versus the number of training epochs (Figure 45). Generally, the error reduces after more training epochs, but may start to increase in the validation dataset when the network starts to overfit the training data. By default, training stops after six consecutive increases in validation error, and the best performance is taken from the epoch with the lowest error.

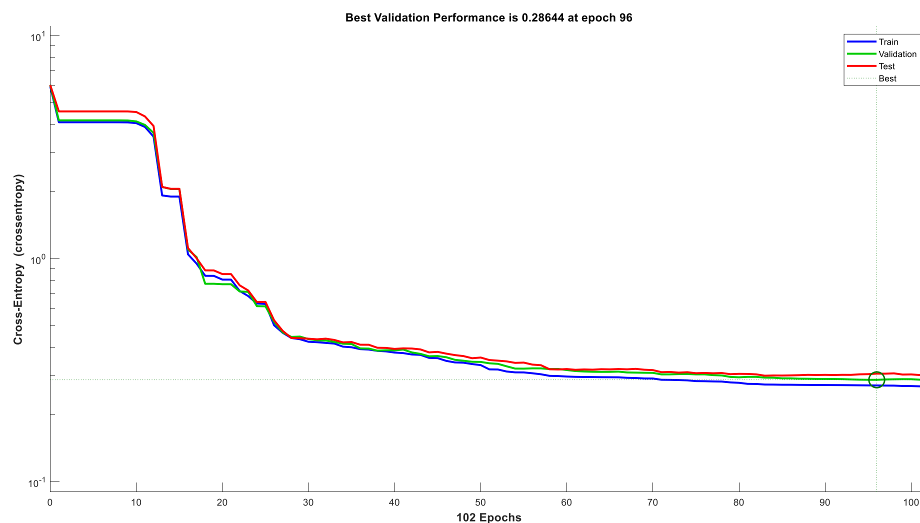

Figure 45:Example of patter recognition app performance.

- Training state: This function plots the values of the training state. It is observed that 2 plots are obtained: one for the gradient and one for the validation checks.

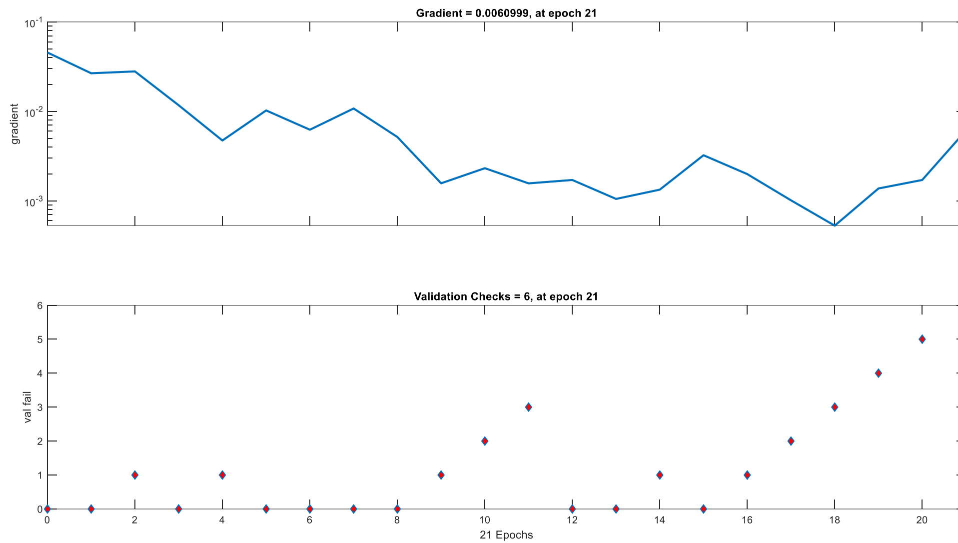

Figure 46: Example of patter recognition app training state.

- Error Histrogram: The error histogram plots a confusion matrix for the true labels (targets) and the predicted labels (outputs).

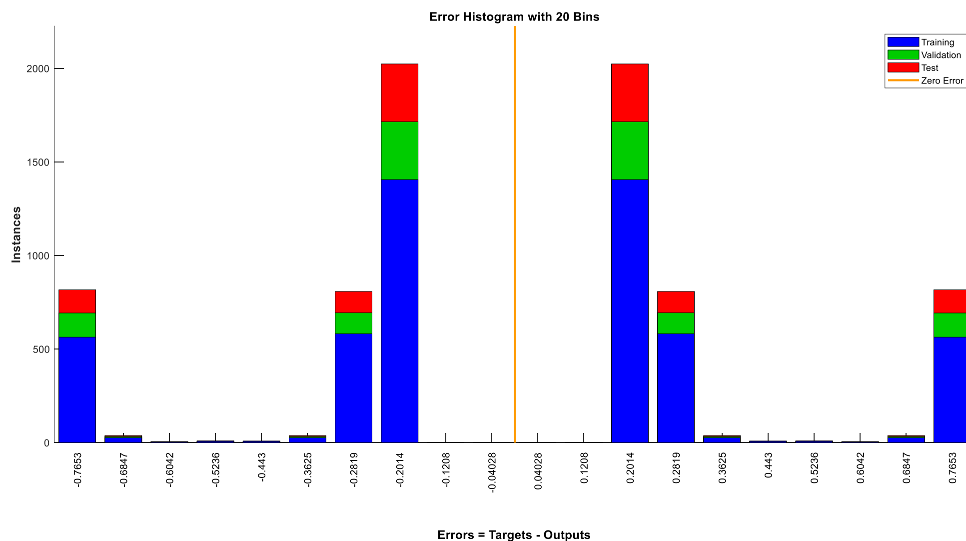

Figure 47: Example of patter recognition app error histrogram.

- Confusion: In this graph, the rows correspond to the output class and the columns to the destination class. The cells on the diagonal correspond to the correctly made observations. The cells outside the diagonal correspond to the incorrectly classified observations. In each cell you can see the number of observations and the percentage of the total number of observations. The following chapter will clarify this concept.

- Receriver operating Characteristics: This function of the Pattern Recognition app plots the ROC where the receiver operating characteristic is plotted for each output class. In this particular case, it plots the ROC of the training part, the validation part and the test part. The following chapter will clarify this concept.

It should also be noted that this application allows to know the cross-entropy (quality of classification, CE) and the error rate (fraction of samples that are misclassified, %E).

5.4.2 Training specifications

As mentioned at the beginning of this chapter, the case selected for training the neural network was case 3. One of the advantages of the "Pattern Recognition app" tool is that it allows you to choose the number of neurons in the hidden layer that you want to work with. The 3 cases carried out are shown below: for 10, 100 and 1000 neurons in the hidden layer.

- 10 neurons in the hidden layer

  The graphs obtained by fixing 10 neurons for the hidden layer can be seen below. In figure 48, you can see the cross-entropy minimisation and the percentage error. It can be seen that the entropy value is close to 0, so it can be considered that a good classification has been made. As far as the error is concerned, it can be seen that the percentage is high.

Results

	Samples	CE	%E
Training:	2624	4.48576e-1	22.90396e-0
Validation:	563	7.78592e-1	23.97868e-0
Testing:	563	7.78896e-1	23.80106e-0

Figure 48: Basic training results (10 neurons in the hidden layer).

Figure 49 shows the confusion plot for training. In order not to be repetitive, only the training case will be explained as the analysis would be the same for validation and testing.

The first two diagonal cells show the number and percentage of correct classifications of the network. 0.1% of the generated traces are correctly classified as valid. 77% are correctly classified as invalid traces.

In this case, none of the incorrect traces that are classified as correct have been detected (0%). In contrast, 22.9% of the valid traces analysed are incorrectly classified as invalid.

Figure 49: Confusion matrix results (10 neurons in the hidden layer).

Figure 50 shows the ROC. Two values are calculated in these plots: the true positive ratio and the false positive ratio. The closer the plotted functions are to the left and top edges, the better the classification. In this particular case, it can be seen that the procedure with the best classification is the validation procedure.

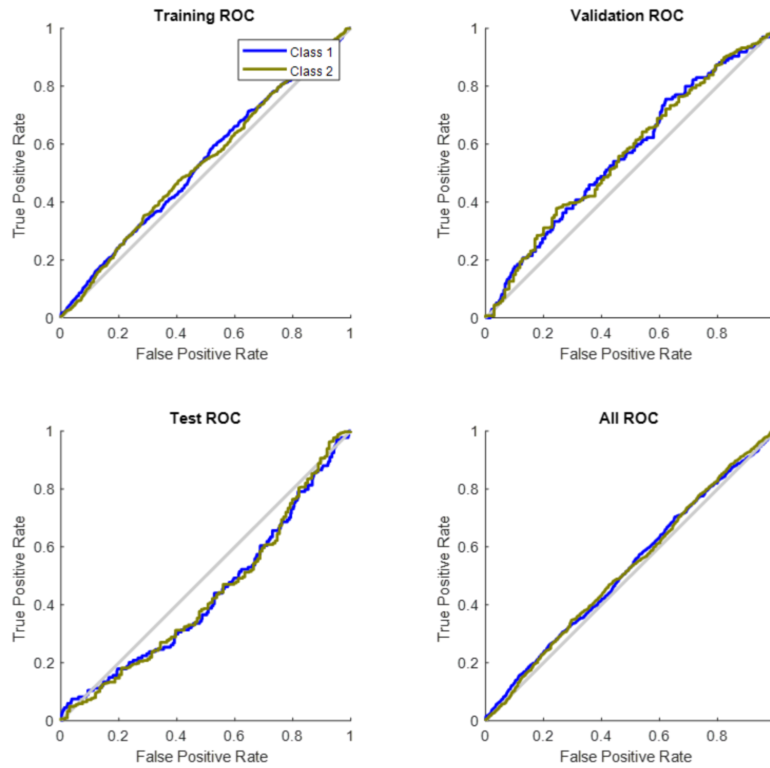

Figure 50: ROC graphs results (10 neurons in the hidden layer).

- 100 neurons in the hidden layer

The analysis of this case will follow the methodology used in the previous case but now with 100 neurons in the hidden layer.

Figure 51 again illustrates the minimisation of the cross-entropy and the error rate. Comparing it with the previous case, it can be seen that the increase of neurons in the hidden layer does not imply a great advantage, since the values of the CE and %E parameters are very similar.

Results	Samples	CE	%E
Training:	2624	4.51684e-1	22.75152e-0
Validation:	563	7.87883e-1	23.26820e-0
Testing:	563	7.90143e-1	25.93250e-0

Figure 51: Basic training results (100 neurons in the hidden layer).

In the following figure (Figure 52), the confusion plot is shown. A small improvement can be seen in both the validation and test matrices. Even so, it is still very similar to the previous case.

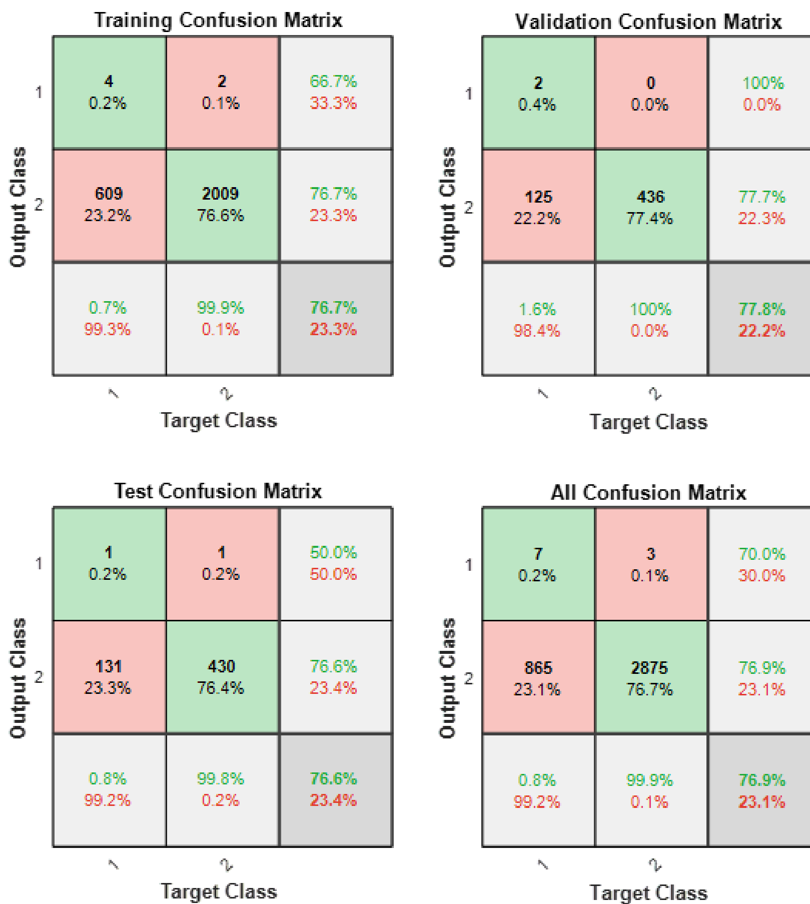

Figure 52: Confusion matrix results (100 neurons in the hidden layer).

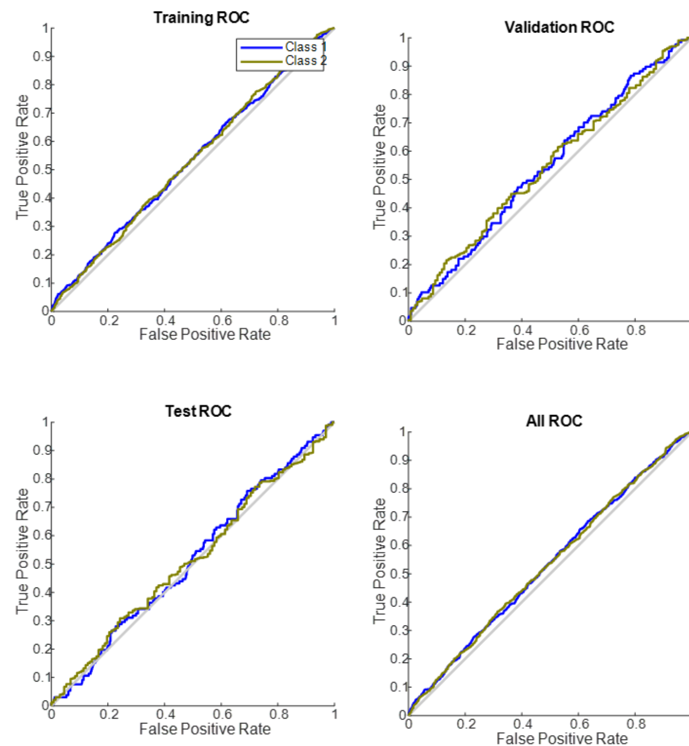

Figure 53: ROC graphs results (100 neurons in the hidden layer).

Figure 53 shows the ROC of the processes. A substantial improvement can be seen in all processes. Especially in the testing process.

- 1000 neurons in the hidden layer

In the last of the training runs, 1000 neurons have been used in the hidden layer to see if more neurons result in optimal training for the neural network.

The first parameters to be analysed are the cross-entropy minimisation and the error rate (Figure 54). As with the first increase of neurons in the hidden layer, this increase does not translate into an improvement of the CE and %E parameters either.

Results

	Samples	CE	%E
Training:	2624	4.66995e-1	22.94207e-0
Validation:	563	8.34237e-1	23.80106e-0
Testing:	563	8.34899e-1	26.11012e-0

Figure 54: Basic training results (1000 neurons in the hidden layer).

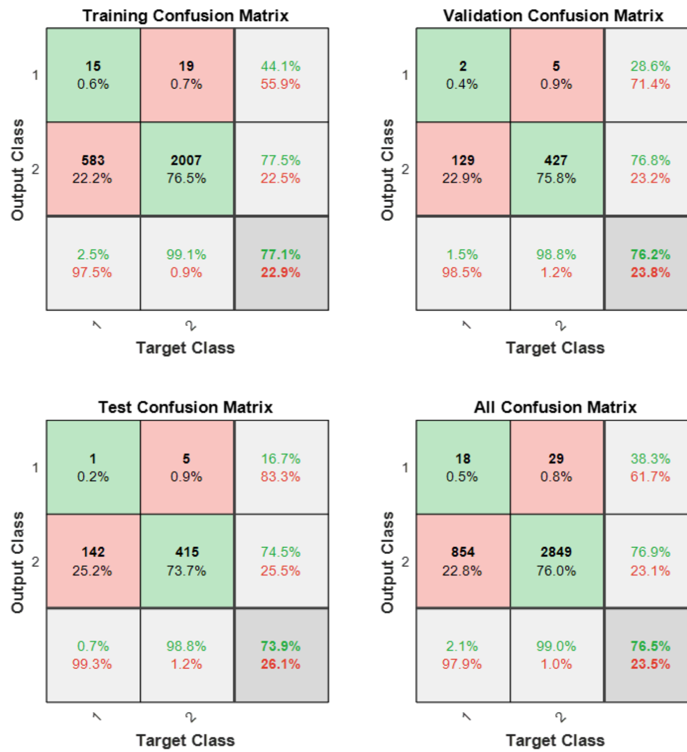

Figure 55: Confusion matrix results (1000 neurons in the hidden layer).

Figure 56 shows the confusion plot. In this case, although the values are very similar to the values of the previous cases, it can be seen that there is an increase in the errors in the training of the neural network.

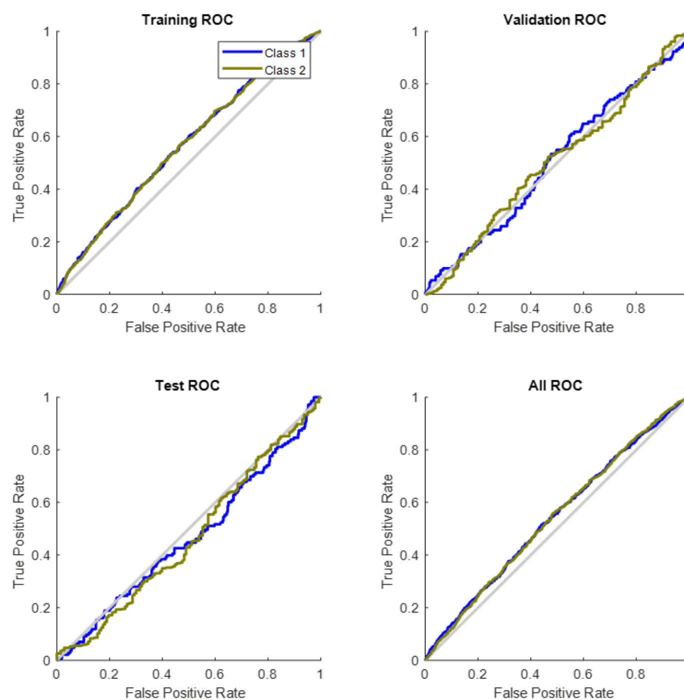

Figure 56: ROC graphs results (1000 neurons in the hidden layer).

To conclude the training study, Figure 56 shows the ROC of the training, validation and test procedures. In the training method, there is a very noticeable improvement.

It should be noted that, although the other two processes do not have improvements to highlight, it is observed that the ROC of all the processes together is the best of the three cases carried out.

6. Conclusions

This work has studied the behaviour of Random Telegraph Noise (RTN) for the training of a neural network. The initial objective of the project has been achieved, which was to create and apply a methodology capable of training a neural network that, in the end, would be able to discern which of the traces introduced are correct and which are not.

In order to achieve the established objective, the behaviour and the most relevant parameters of the RTN signals had to be parameterised first. Once the nature of these signals was understood, they were converted into binary codes.

Two validation methods have been developed: a method based on the mean current value and a method based on catches and emissions. It has been shown that, depending on the behaviour of the generated traces, one method or the other should be used. It should be emphasised that the mean current value method is more suitable for training neural networks because the more traces generated (more data), the more accurate it is.

Before the neural network could be trained, it was necessary to set an acceptance criterion. The purpose of this criterion was to identify whether the generated traces are valid or not. To achieve this, a method has been developed that focuses its resources on searching for the probability between equal binary symbols with a specific degree of acceptance.

Finally, the neural network was trained. MATLAB's NNStart application was used, specifically the Pattern Recognition app. As mentioned in the previous chapter, due to a lack of computational resources and limitations in the delivery date of the work, the training of the neural network had to be limited. For this reason, a specific case has been chosen (case 3).

The results obtained after training indicate that no matter how many more neurons are used in the hidden layer, it does not mean that the training will improve. It has been observed that a balance must be found between the amount of data to be worked on and the number of neurons required. In other words, if more neurons are used than strictly necessary, the probability of errors increases.

It should be noted that modifications could be made to extend the development, training and study of the neural network. With a higher computational level, the neural network could be supplied with much more data.

This increase in data would also lead to an increase in the training of the neural network. In this way, the criteria used for the validation of the generated RTN traces could be readjusted. In summary, an increase in software resources would allow the development of a neural network with higher accuracy, higher performance and higher efficiency.

X. Bibliography

[1] "Moore's Law: The Life of Gordon Moore, Silicon Valley's Quiet Revolutionary Arnold Thackray, David C. Brock, and Rachel Jones," *MRS Bulletin*, vol. 41, no. 5, pp. 412–413, May 2016, doi: 10.1557/mrs.2016.107.

[2] T. Ferreira and P. Leite, "FD-SOI technology opportunities for more energy efficient asynchronous circuits." [Online]. Available: https://tel.archives-ouvertes.fr/tel-02295530

[3] J. G. Simmons, "Electric Tunnel Effect between Dissimilar Electrodes Separated by a Thin Insulating Film," *Journal of Applied Physics*, vol. 34, no. 9, pp. 2581–2590, 1963, doi: 10.1063/1.1729774.

[4] S. Mahapatra, *Fundamentals of Bias Temperature Instability in MOS Transistors: Characterization Methods, Process and Materials Impact, DC and AC Modeling*, vol. 52. 2016. doi: 10.1007/978-81-322-2508-9.

[5] R. M. Valls, A. Crespo, Y. Directora, and M. Nafría, "Characterization of FD-SOI transistor," 2020.

[6] A. Ranjan *et al.*, *CAFM based spectroscopy of stress-induced defects in HfO2 with experimental evidence of the clustering model and metastable vacancy defect state*. 2016. doi: 10.1109/IRPS.2016.7574576.

[7] S. Realov and K. Shepard, "On-chip combined CV/IV transistor characterization system in 45-nm CMOS," Jan. 2011.

[8] C. Y. P. Chao *et al.*, "Statistical analysis of the random telegraph noise in a 1.1 µm pixel, 8.3 MP CMOS image sensor using on-chip time constant extraction method," *Sensors (Switzerland)*, vol. 17, no. 12, Dec. 2017, doi: 10.3390/s17122704.

[9] J. Martín-Martínez, J. Diaz, R. Rodríguez, M. Nafría, and X. Aymerich, "New Weighted Time Lag Method for the Analysis of Random Telegraph Signals," *IEEE Electron Device Letters*, vol. 35, pp. 479–481, 2014.

[10] Institute of Electrical and Electronics Engineers, IEEE Electron Devices Society, and I. International Conference on Ultimate Integration on Silicon (2015 : Bologna, *EUROSOI-ULIS 2015 : 2015 Joint International EUROSOI Workshop and International Conference on Ultimate Integration on Silicon : January 26-28, 2015, Bologna, Italy.*

[11] C. D. Bodemann, "THE SUCCESSFUL DEVELOPMENT PROCESS WITH MATLAB SIMULINK IN THE FRAMEWORK OF ESA'S ATV PROJECT."

[12] J. Reedy, S. Lunzman, and B. Mekari, "Model Based Design Accelerates the Development of Mechanical Locomotive Controls," Oct. 2010, doi: 10.4271/2010-01-1999.

[13] P. Mingola and B. Rajput, "A Study of Poisson and Related Processes with Applications," 2013. [Online]. Available: https://trace.tennessee.edu/utk_chanhonoproj/1613

[14] M. M. Plaza and J. Martín Martínez, "Red neuronal clasificadora de trazas RTN para la generación de números aleatorios."