UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL
INSTITUTO DE INFORMÁTICA
CURSO DE ENGENHARIA DA COMPUTAÇÃO

FABIANO PEREIRA LIBANO

Reliability Analysis of Neural Networks In Heterogeneous Systems

Work presented in partial fulfillment of the requirements for the degree of Bachelor in Computer Engineering.

Advisor: Prof. Dr. Paolo Rech
Co-advisor: Prof. Dr. Fernanda Kastensmidt

Porto Alegre
2017

“You have to make the rules, not follow them.”
(Isaac Newton)

ACKNOWLEDGEMENTS

I would like to thank my parents Fausto Bastos Libano and Fátima Rosane Pereira Libano as well as my brother Felipe Pereira Libano, for the support not only during the last four and a half years, but throughout my entire life.

To all the professors I had, who put legitimate effort into teaching their classes and contributed to me by exposing their knowledge.

To my classmates and research group colleagues who helped me to succeed academically in a variety of aspects.

To UFRGS, Institute of Informatics and School of Engineering, for helping me to develop the most important trait of an engineer: the ability to solve problems.

And specially, to my advisors Paolo Rech and Fernanda Gusmão de Lima Kastensmidt. They taught me what computing really is from day one, and continue to do so on a daily basis.

To all of you, I sincerely say: thank you.

ABSTRACT

In this work we experimentally and analytically evaluate the reliability of two state-of-the-art types of neural networks for linear regression and pattern recognition (Multi-Layer Perceptron and Single-Layer Perceptron) implemented in a System on Chip composed of an FPGA and a microprocessor. Experiments using a controlled heavy-ions beam show that, for both networks, only a small portion of the observed output errors actually affect the application's correctness. We identify the causes of critical errors through fault injection, and found that faults in hidden layers of the networks are more likely to significantly affect the output.

Keywords: **reliability, SoC, neural networks, radiation, fault injection, FPGA**.

Análise de Confiabilidade de Redes Neurais em Sistemas Heterogêneos

RESUMO

Neste trabalho é realizada uma avaliação experimental e analítica de duas redes neurais do estado-da-arte, para regressão linear e reconhecimento de padrões (Multi-Layer Perceptron e Single-Layer Perceptron) implementadas em um System on Chip composto por uma FPGA e um microprocessador. Experimentos utilizando um raio controlado de íons pesados mostram que, para ambas as redes, apenas uma porção pequena dos erros observados na saída de fato afeta a corretude da aplicação. Foram identificadas as causas dos erros críticos através de injeção de falhas, e descobriu-se que falhas nas camadas ocultas das redes são têm maior probabilidade de afetarem o resultado.

Palavras-chave: confiabilidade, SoC, redes neurais, radiação, injeção de falhas, FPGA.

LIST OF FIGURES

Figure 2.1: Tesla's autopilot 13
Figure 2.2: Google's DeepMind AlphaGo vs World Champion Lee Sedol 14
Figure 2.3: Generic architecture of an APSoC 15
Figure 2.4: Fault, error and failure propagation....16
Figure 2.5: Radiation effects....17
Figure 3.1: The thinned Zynq-7000 device in the irradiation chamber 19
Figure 3.2: Block diagram of the developed setup for testing the DUT....20
Figure 3.3: The fault injection flow 21
Figure 4.1: Zynq-7000's block diagram 23
Figure 4.2: Zynq-7000's PS part....24
Figure 4.3: Relationship between CLBs and Slices in the Xilinx 7-Series FPGAs....25
Figure 4.3: Example of a 2-input function implemented in the LUT 25
Figure 4.4: Sample structure of an ANN 26
Figure 4.5: The original logistic function and the three levels of discretization 27
Figure 4.6: Boston Housing ANN's topology 29
Figure 4.7: Iris Flower ANN's topology....30
Figure 5.1: Heavy-ions cross section of the Boston Housing ANN 33
Figure 5.2: Heavy-ions cross section of the Iris Flower ANN 34
Figure 5.3: AVF of the Boston Housing ANN 35
Figure 5.4: AVF of the Iris Flower ANN 36
Figure 5.5: Total observed errors in fault injection (Boston Housing ANN, TRE= 10%)....37
Figure 5.6: Total observed errors in fault injection (Iris Flower ANN) 37

LIST OF TABLES

Table 4.1: Instance attributes for the Boston Housing dataset.....28
Table 4.2: Zynq's FPGA resource utilization by the linear regression ANN.....29
Table 4.3: Instance attributes for the Iris Flower dataset.....30
Table 4.4: Zynq's FPGA resource utilization by the classification ANN 31
Table 4.5: Radiation effects on ANNs.....32

LIST OF ABBREVIATURES AND ACRONYMS

AI	Artificial Intelligence
ACM	Association for Computing Machinery
ANN	Artificial Neural Network
APSoC	All-Programmable System-On-Chip
AXI	Advanced eXtensible Interface
BRAM	Block Random Access Memory
CLB	Configurable Logic Block
DSP	Digital Signal Processor
ESA	European Space Agency
FPGA	Field-Programmable Gate Array
FPU	Floating Point Unit
HDL	Hardware Description Language
IC	Integrated Circuit
ICAP	Internal Configuration Access Port
IEEE	Institute of Electrical and Electronics Engineers
LAFN	Laboratório Aberto de Física Nuclear
LANSCE	Los Alamos National Science Center
LET	Linear Energy Transfer
LUT	Look-Up Table
ML	Machine Learning
MMU	Memory Management Unit
PL	Programmable Logic
PS	Processing System
PSI	Paul Scherrer Institut
RAL/ISIS	Rutherford Appleton Laboratory
SEE	Single Event Effect
SEU	Single Event Upset
SRAM	Static Random Access Memory
TMR	Triple Modular Redundancy
TRE	Tolerated Relative Error
USP	Universidade de São Paulo

CONTENTS

1 INTRODUCTION 11
2 BACKGROUND 13
2.1 Safety-Critical Applications 13
2.2 Machine Learning 14
2.3 Heterogeneous Systems 15
2.4 Radiation Effects on Electronic Devices 16
2.4.1 Fault, Error and Failure 16
2.4.2 Radiation Sources 16
2.4.3 Radiation Effects 17
3 EXPERIMENTAL METHODOLOGY 18
3.1 Radiation Experiment 19
3.1.1 Setup 19
3.1.2 Cross Section 20
3.2 Fault Injection 21
3.2.1 Framework 21
3.2.2 Architectural Vulnerability Factor (AVF) 22
3.3 Comparison of Experiments 22
4 CASE STUDIES 23
4.1 Zynq-7000 APSoC 23
4.2 Artificial Neural Networks implemented in FPGAs 26
4.2.1 Neuron's Activation Function 27
4.2.2 Boston Housing Dataset 28
4.2.3 Iris Flower Dataset 29
4.3 Error Criticality and Reliability Evaluation 31
5 RESULTS 33
5.1 Radiation Experiment 33
5.2 Fault Injection 35
5.3 Discussion 38
6 CONCLUSIONS 39
REFERENCES 40

1 INTRODUCTION

Artificial Neural Networks (ANNs) are becoming a widely adopted computational approach in many fields, from data mining to pattern recognition, high performance computing, and data analysis (AMIN et al., 1997). Additionally, ANNs are extremely attractive for safety critical applications, such as space exploration, to reduce the communication strain with the Earth, and to disclose autonomous driving (NEAGOE et al., 2012). Both applications currently rely on multiple pattern recognition algorithms to identify and classify certain objects of interest based on captured frames or signals, as well as linear regression to predict behaviors and trajectories. Most of these algorithms can be efficiently implemented using ANNs.

The basic functional principle of ANNs is to compute a solution in a similar way that the biological brain solves problems. A number of neurons connected to each other and organized in layers, interact via synapses. Each neuron has an activation function that, based on the received stimulus, sends a signal to the connected neurons. The ANN needs to be trained using a sufficiently complex and representative set of inputs. During the training phase the weights of every connection between neurons are tuned. Hence, when an input is given the ANN computes a solution based on the training, using the previously established values for the connections.

Due to their intrinsic parallelism and the high number of connections, ANNs map efficiently on Field-Programmable Gate Arrays (FPGAs) (HE et al., 2009). In fact, FPGAs offer the possibility of configuring logic blocks, used to tune the neuron's activation function, and a hierarchy of interconnects, which can be used to create connections between neurons.

Unfortunately, while being low cost, extremely efficient, and flexible, FPGAs have been shown to be prone to be corrupted by radiation (WIRTHLIN, 2015). In particular, SRAM-based FPGAs may experience radiation-induced corruptions in the configuration memory, also called of Single Event Upsets (SEUs). In SRAM-based FPGAs, an SEU can change the configuration of a routing connection, the configuration of a Look-Up Table (LUT), or the configuration of an embedded Block RAM memory (BRAM). Moreover, SEUs have a persistent effect, which can only be corrected when a new bitstream is loaded to the FPGA. SEUs can also occur in a Flip-Flop (FF) of a Configuration Logic Block (CLB) used to implement the user's sequential logic. In this case, the SEU has a transient effect and the next load of the FF can correct it. It is also worth mentioning that as state-of-the-art SRAM-based FPGAs are built with cutting-edge manufacturing processes (sub-28nm) and as they are composed of millions of SRAM cells to

store their configuration, they are also very susceptible to Multiple Bit Upsets (MBU) (WIRTHLIN et al., 2014).

In this paper we specifically address the reliability of ANNs implemented on FPGAs. We consider two kinds of ANNs: Single-Layer Perceptron (SLP) and Multi-Layer Perceptron (MLP). SLP networks have only one hidden layer, while MLPs have more than one hidden layers. As examples, we consider the SLP *iris flower*, that identifies flowers, and the MLP *Boston housing*, that makes a prediction based on inputs approximation. Through controlled heavy-ions beams we evaluate the ANN expected output error rate and we analyze the impact of radiation-induced errors in the application's correctness. It is very important to highlight that the training phase was performed beforehand, thus without radiation influence. Our results show that only a small portion of faults affects is to be considered critical. In fact, while radiation perturbs computation, a significant portion of output errors does not impact the performed pattern recognition. Then, to identify the causes of critical errors, we perform an extensive fault injection campaign. As discussed in the paper, we find that faults injected on neurons in the early stages of the ANN are more likely to generate critical errors.

The case-study device we have selected for the proposed analysis is a Zynq-7000 (XILINX, 2015), which offers high configurability, stimulates strong interest in the scientific community, and is highly present on the market. Zynq-7000 is composed of two main parts: a *Processing System* (PS) formed around a dual-core ARM Cortex-A9 processor, and *Programmable Logic* (PL) based on a standard Xilinx Artix-7 FPGA. The PL section is ideal for implementing high-speed logic, arithmetic, and data processing subsystems, while the PS supports software routines and/or operating systems. Therefore, the overall functionality of any designed system can be appropriately partitioned between hardware and software. In this particular case, the network itself is in the PL section, while the PS is responsible for the final comparison between the outputs of the last layer of neurons.

2 BACKGROUND

This section introduces some main concepts like safety, reliability and heterogeneity. State-of-the-art devices and machine learning algorithms are also presented. Finally, we discuss why it is crucial to consider radiation effects on electronic devices and applications.

2.1 Safety-Critical Applications

A regular application is considered safety-critical if its failure can result in loss of human lives, harm to private property or environmental damage (IEC, 2017). Systems to be used in such applications should present the following properties: reliability, maintainability, availability, safety and security. This work focuses on reliability, which is the probability of a fail to occur, and safety, which is the probability that the system will not cause any harm in case of malfunction (IEEE, 1990). Further properties are defined by (BERNARDESCHI, 2015).

In order to reach functional safety, a given system must be compliant with given safety constraints by knowing how to handle failures, reaching tolerable levels. These constraints are defined to help designers on the development and the qualification of reliable systems.

One of the most actual and important examples of safety-critical applications is the fast-growing autonomous trend in the automotive sector. Every year, the level of reliability in partially self-driving cars rises, and we will inevitably reach fully autonomous conduction, sooner rather than later (ELECTREK, 2016). The aerospace sector is also massively dependent on the proper fault tolerance techniques, mainly to guarantee functional correctness of missions, where billions of dollars are at stake. The safety-critical area is, of course, highly regulated with norms such as the IEC 61508, which restricts the set of solutions that can be commercially adopted, but of course, does not forbid scientific research.

Figure 2.1: Tesla's autopilot

Font: (ELECTREK, 2016)

2.2 Machine Learning

One of the main subareas of Artificial Intelligence (AI) is Machine Learning (ML), which consists in a set of algorithms and methods that allow a computer to learn without being explicitly programmed (SAMUEL, 1959). Generally speaking, ML adoption is highly recommended in applications where the construction of a deterministic algorithm is impossible or very time consuming, such as computer vision and predictive systems.

The actual learning process involves several hours of supervised or unsupervised training, where the system is fed several times with large datasets, and adjusts its own behavior after each iteration. Essentially, ML algorithms learn from input patterns, and it eventually becomes able to spot details that humans are not able to deterministically describe.

This work explores Artificial Neural Networks (ANNs), which is one of the approaches within the ML field. Section 4.2 explains ANNs in detail, but the basic definition is: a network of interconnected units, propagating signals based on the received stimulus.

Recently, the research on ANNs has increased significantly, with large scale topologies (surpassing thousands of neurons) and Big Data usage in the training processes. Also, efforts towards the construction of dedicated Integrated Circuits (ICs) have been made by big market players, such as IBM with its *TrueNorth* chip, capable of simulating up to 256 million synapses in real-time (MEROLLA et al., 2014). In addition, neural networks are consistently proving to perform better than human professionals in a number of specific fields, which can be exemplified by Google's DeepMind AlphaGo beating the world's best player of Go (millenary Chinese table game) (QUARTZ, 2016).

Figure 2.2: Google's DeepMind AlphaGo vs World Champion Lee Sedol

Font: (QUARTZ, 2016)

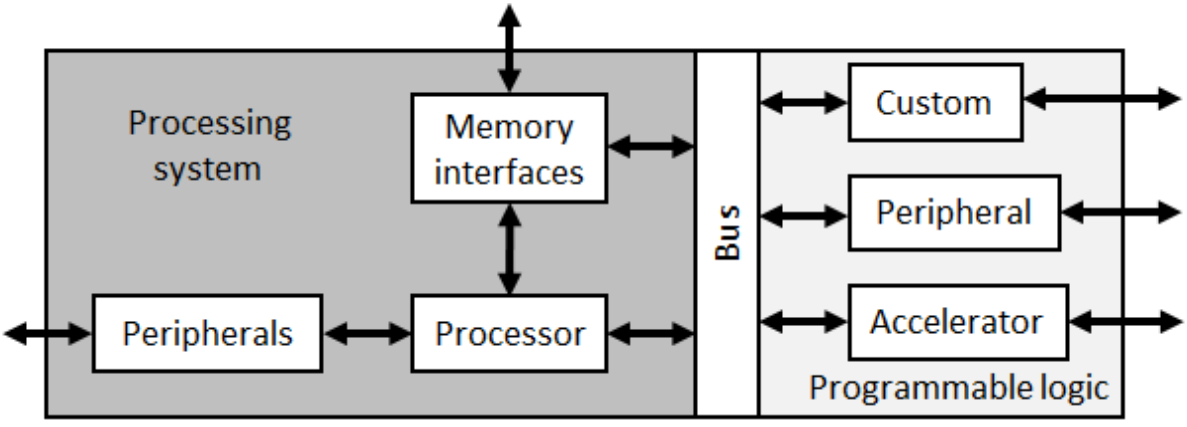
2.3 Heterogeneous Systems

As Moore's law (MOORE, 1965) states, the transistor density in processors doubles every 18 to 24 months, and this has been confirmed up until the past decade. Nowadays this evolution rate seems to be slowing down, but the highly increased complexity achieved from years of improvements led to significantly higher power consumption. One of the most promising ways researchers designed to increase performance while being as energy efficient as possible, is the adoption of heterogeneous architectures that combine traditional processors with other hardware resources or accelerators. This work focuses on heterogeneity by Field-Programmable Gate Array (FPGA) usage.

FPGAs enable high flexibility due to their ability to implement arbitrary logic circuits using Look-Up Tables (LUTs). These devices have been proven to be very suitable for parallel tasks and to deal with pipeline-friendly data flows. Today, this type of heterogeneous device is commercially called All-Programmable System-On-Chip (APSoC). In most cases, APSoCs are composed of two main parts: Programmable Logic (PL) and Processing System (PS), with the PL part being based on a FPGA and the PS part having single or multicore traditional processors. Finally, there are bus interfaces for communication between PS and PL, alongside optional peripherals.

Several state-of-the-art APSoCs are currently available on the market, such as the Cyclone V (ALTERA, 2015) from Altera, the SmartFusion (MICROSEMI, 2015) from Microsemi and the Zynq-7000 (XILINX, 2015) from Xilinx. This work uses as case-study the latter, which will be presented in Section 4.1.

Figure 2.3: Generic architecture of an APSoC



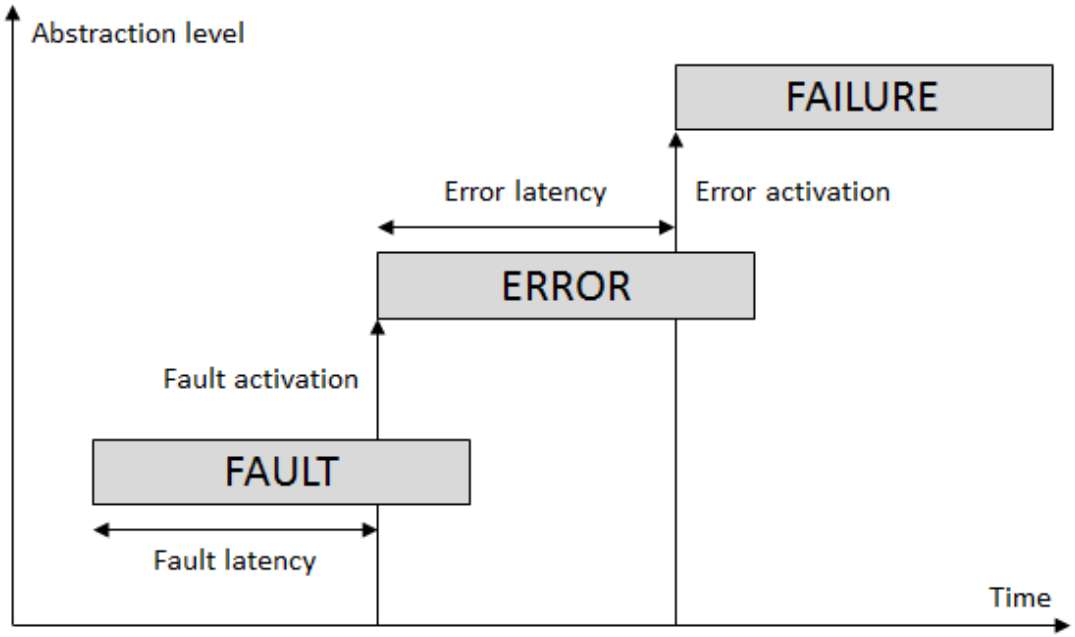
Font: (TAMBARA, 2017)

2.4 Radiation Effects on Electronic Devices

2.4.1 Fault, Error and Failure

According to (AVIZIENIS et al., 2004), a system is an entity composed by one or more subsystems that interact with each other, while the system itself interacts with other systems in its environment. Given this scenario, we can say that the service provided by a particular system is its behavior as perceived by its users. A system presents an error when its output is different from the expected one, and it fails whenever there is a behavior deviation. Finally, we define fault, error and failure as concepts linked by a cause-effect relationship, illustrated by Fig. 2.4.

Figure 2.4: Fault, error and failure propagation



Font: (TAMBARA, 2017)

2.4.2 Radiation Sources

Radiation is a naturally occurring phenomena which can basically be described as the transmission of energy through particles traveling at high speeds. There are many sources of radiation in the universe, such as stars (like our sun) and cosmic rays, but there are also man-made radiation rich environments like particle accelerators and nuclear reactors, for scientific purposes and energy supply, respectively.

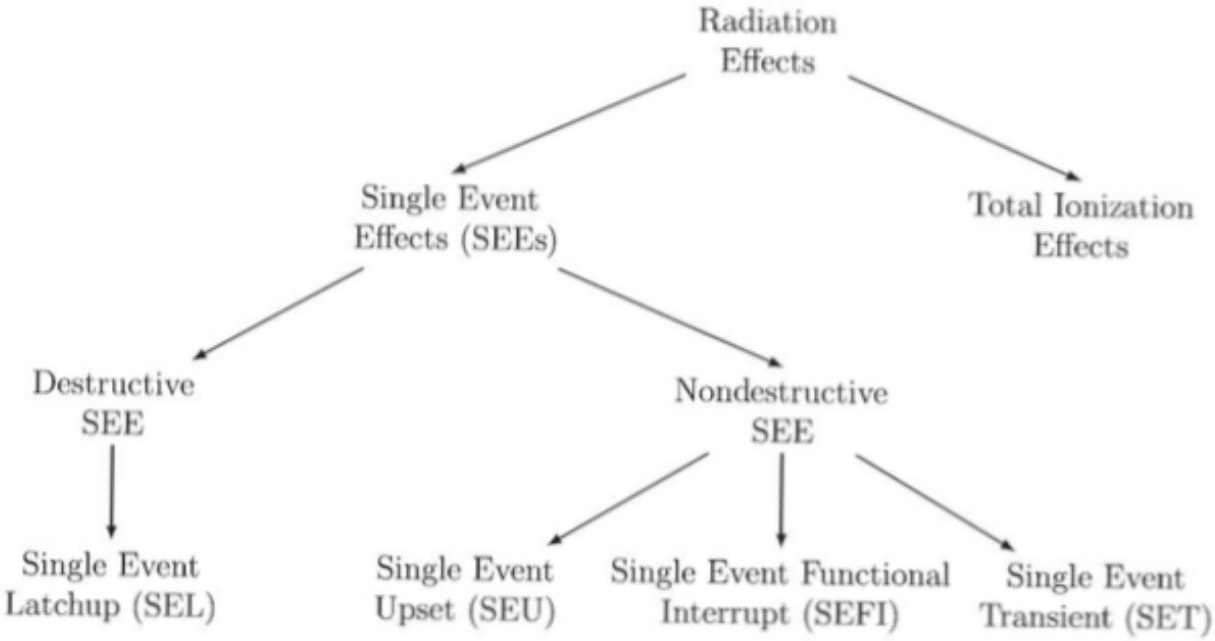
While most particles are deviated by the terrestrial magnetic field, the most energetic cosmic-rays reach the Earth. As they collide with the nuclei of atoms in our atmosphere, they produce a variety of secondary particles, including alpha particles, protons, gamma, and, mainly, neutrons. Each type of these particles affects the ICs in a different way.

2.4.3 Radiation Effects

The advances in both conception and fabrication processes of ICs, over the past two decades, has led to a considerable increase in transistor density and a reduction in power consumption (due to lower voltages). The combination of these two factors brings a greater radiation sensitivity (BAUMANN, 2005).

As said in Section 2.4.2, different particles interact with electronic devices in distinct manners. We classify as Single Event Effect (SEE), when a single particle deposits enough energy in the circuit to alter its normal functionality. SEEs can still be subdivided in Destructive or Non-Destructive. Destructive SEEs are those in which either the device gets permanently damaged or has to be reset. Non-Destructive SEEs on the other hand, comprise the cases where the particle-circuit interaction results in system/application error or failure. Fig. 2.5 shows the main possible radiation effects.

Figure 2.5: Radiation effects



Font: (SIEGLE et al., 2015)

3 EXPERIMENTAL METHODOLOGY

Among the methods to evaluate SEEs on electronic devices, the most realistic one is in its real application environment (satellites in space, for example). Sometimes however, due to practical and/or financial constraints, this approach is not an option. Moreover, it may take large periods of time to collect a statistically relevant amount of data. As a result, accelerated radiation tests are nowadays the most common way to qualify the reliability of integrated circuits for SEEs (JEDEC, 2006). In these experiments the devices are exposed to a radiation source with much higher density than the normal levels it would experience in its application environment, reducing drastically the amount of time necessary to obtain useful data. Accelerator facilities provide a variety of particles, such as neutrons, protons, and heavy ions. Neutron facilities like the Los Alamos National Science Center (LANSCE) in the United States, and the Rutherford Appleton Laboratory (RAL/ISIS) in the United Kingdom are generally used for testing devices to be used in terrestrial and avionic applications. Proton facilities as the Paul Scherrer Institut (PSI) in Switzerland are capable of generating protons with enough energy to simulate solar flares. Heavy ions facilities like the 8UD Pelletron at Universidade de São Paulo (USP), are usually for evaluating devices destined to space and deep space. Finally, there are facilities in which the Device Under Test (DUT) is exposed to a very high flux of a cocktail of particles. The best example of this facility is the CERN High Energy Accelerator Mixed-field (CHARM) located in Switzerland.

A third method of qualification is fault injection by emulation or simulation. This method is, at the same time, the less costly and the most flexible, but it does not waive the necessity of a radiation test. Usually, fault emulation is an attractive technique to predict the susceptibility of a system before submitting the device to an accelerated test, for example. This approach basically consists in flipping bits of memory components of FPGAs and processors through the assistance of an embedded circuit or a monitoring computer. Single Event Upsets (SEUs) can be emulated randomly or sequentially (when every configuration bit is flipped in a sequential order). Thus, fault injection provides deeper information when compared to radiation experiments, since the correlation between the flipped bit's location and the fault effect is known.

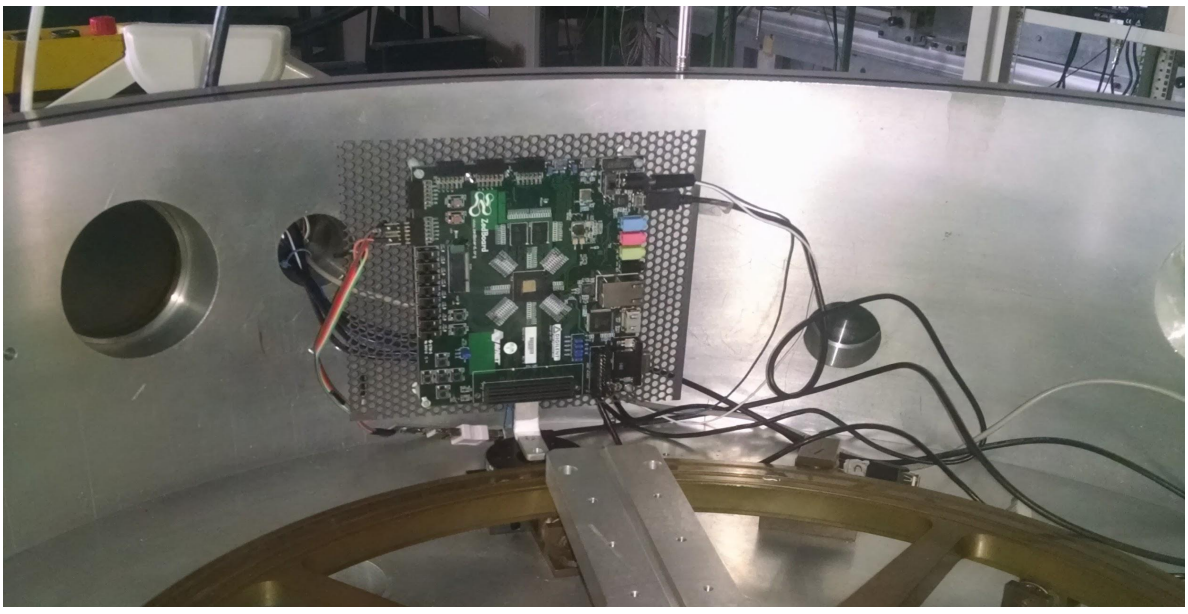
In this work, we exploit both radiation and fault injection experiments, which are described in Sections 3.1 and 3.2, respectively, along with the main metric of each experiment.

3.1 Radiation Experiment

3.1.1 Setup

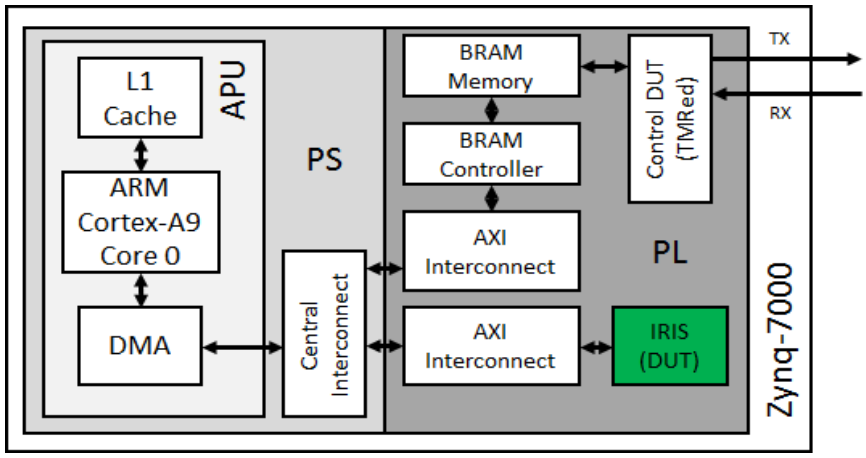
Radiation experiments were conducted with heavy ion particles at the Laboratório Aberto de Física Nuclear (LAFN), located in Universidade de São Paulo, Brazil (AGUIAR et al., 2014), where the ion beams are produced and accelerated by the 8UD Pelletron Accelerator. We included a standard Rutherford scattering setup using a gold foil to achieve a very low particle flux in the range from 10^2 to 10^5 particles.cm^{-2}.s^{-1}, as recommended by the European Space Agency (ESA) for SEU tests (ESA, 2005). The experiment was performed in vacuum. A silicon barrier detector was mounted inside the vacuum chamber at 45º to monitor the beam intensity. The SEU events were observed using a ^{16}O beam, scattered by a 184 μg/cm^2 gold target, with an energy of 56 MeV, which provides an effective Linear Energy Transfer (LET) on the active region of 5 MeV/mg/cm and a penetration in Si of 28.5 μm. To achieve the desired particle fluence, the DUT was positioned at a scattering angle of 90º, resulting in an average flux between 2.0x10^2 and 2.5x10^2 particles.cm^{-2}.s^{-1}. Such configuration was chosen based on several trials and it was the most suitable in terms of particle flux and number of errors for our purposes. For this analysis, the package of a Xilinx Zynq-7000 device, part XC7Z020-CLG484, was thinned to allow that irradiated particles could penetrate the active region of the silicon, as Fig. 3.1 illustrates.

Figure 3.1: The thinned Zynq-7000 device in the irradiation chamber



A heterogeneous setup based on the PS and PL parts of the Zynq-7000 was developed for testing the Design Under Test (DUT), as illustrated in Fig. 3.2. On the PS part, a software running on one of the ARM Cortex-A9 cores controls the DUT. It is worth highlighting that the L2 Cache of the PS part was disabled to not compromise the processor's reliability, while the L1 Cache of the processor was left enabled to not compromise the processor's performance. Such choice was based on the results obtained by (TAMBARA et al., 2016). The communication between PS and PL parts is performed through General Purpose (GP) ports, which provide a very high data throughput. On the PL part, the Control DUT hardware block monitors the memory space of the DUT. This block is implemented with Triple Modular Redundancy (TMR) aiming to mask SEUs. If the block detects errors, it sends them to a script running on a monitor computer, which time-stamps the errors and logs them for future analysis.

Figure 3.2: Block diagram of the developed setup for testing the DUT



3.1.2 Cross Section

In a radiation experiment, the most important metric is the cross section (σ), and it quantifies the sensitivity of a system to particles (JEDEC, 2006). The cross section is measured dividing the number of observed errors (N_{errors}) by the particles fluence ($\phi_{particles}$).

(Equation 3.1) $$\phi_{particles} = \left(particle\ flux\ \left[\frac{particles}{cm^2.\ s}\right]\right).(time\ [s])\quad \left[\frac{particles}{cm^2}\right]$$

(Equation 3.2) $$\sigma = \frac{N_{errors}}{\phi_{particles}}\quad [cm^2]$$

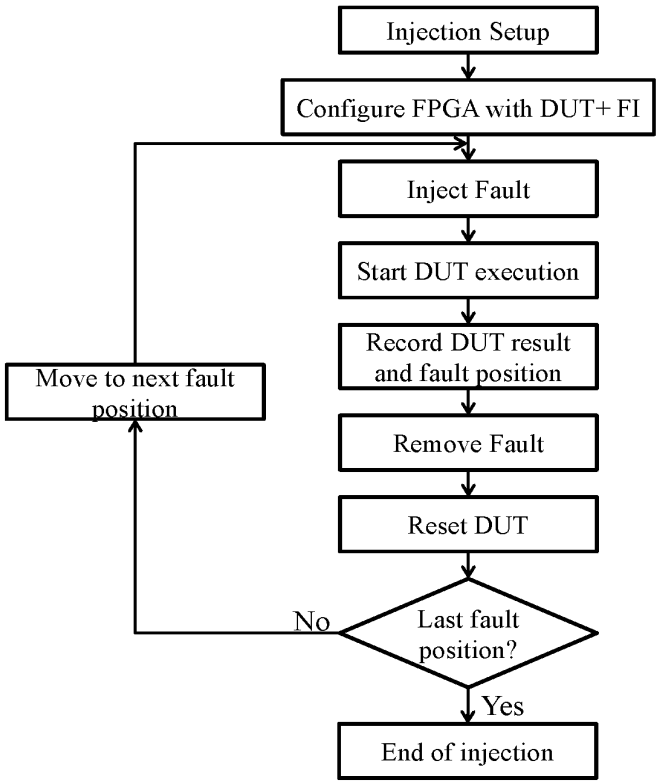
3.2 Fault Injection

3.2.1 Framework

This work uses the fault-injection framework presented in (TONFAT et al., 2016), which is based on the Internal Configuration Access Port (ICAP) block of Xilinx FPGAs, an ICAP controller and a monitor computer. The ICAP block can read and write the configuration memory of the FPGA using the readback and dynamic reconfiguration capabilities of the FPGA. Hence, the fault-injection framework can emulate SEUs in the configuration memory of the FPGA.

The flow of a fault-injection campaign is presented in Fig. 3.3. The first task is the definition of the injection area and the type of fault-injection campaign. For the case considered in this paper, the injection area is the area of a given neuron of the ANN, implemented in the FPGA and an exhaustive fault-injection campaign is selected to known which configuration bits produce errors on the outputs of the artificial neural network. The exhaustive fault-injection campaign produces bit-flips on all the configuration bits of the injection area, one at a time.

Figure 3.3: The fault injection flow



3.2.2 Architectural Vulnerability Factor (AVF)

The fault injection test, as in any higher abstraction level simulation, cannot measure the cross section, since it depends on the physical-level sensitivity to radiation, as defined in Section 3.1.2. An alternate metric for fault injection is the Architectural Vulnerability Factor (AVF), which represents the probability that a visible error will occur at the output of a system given a bit-flip in a hardware structure (MUKHERJEE et al., 2003). It is simply obtained by dividing the number of observed errors (N_{errors}) by the number of injected faults (N_{faults}) as show in Eq. 3.3.

(Equation 3.3) $$AVF = \frac{N_{errors}}{N_{faults}}$$

3.3 Comparison of Experiments

While in radiation experiments, the entire device is irradiated, leading to a more realistic prediction of the error rate, using fault injection, we corrupt specific portions of the hardware, in order to correlate the fault source to the observed effect in the output.

In addition, the cross section is a metric in terms of area [cm²], while the AVF is dimensionless, which means they provide complementary information about the studied system, thus, one experiment does not exempts the execution of the other.

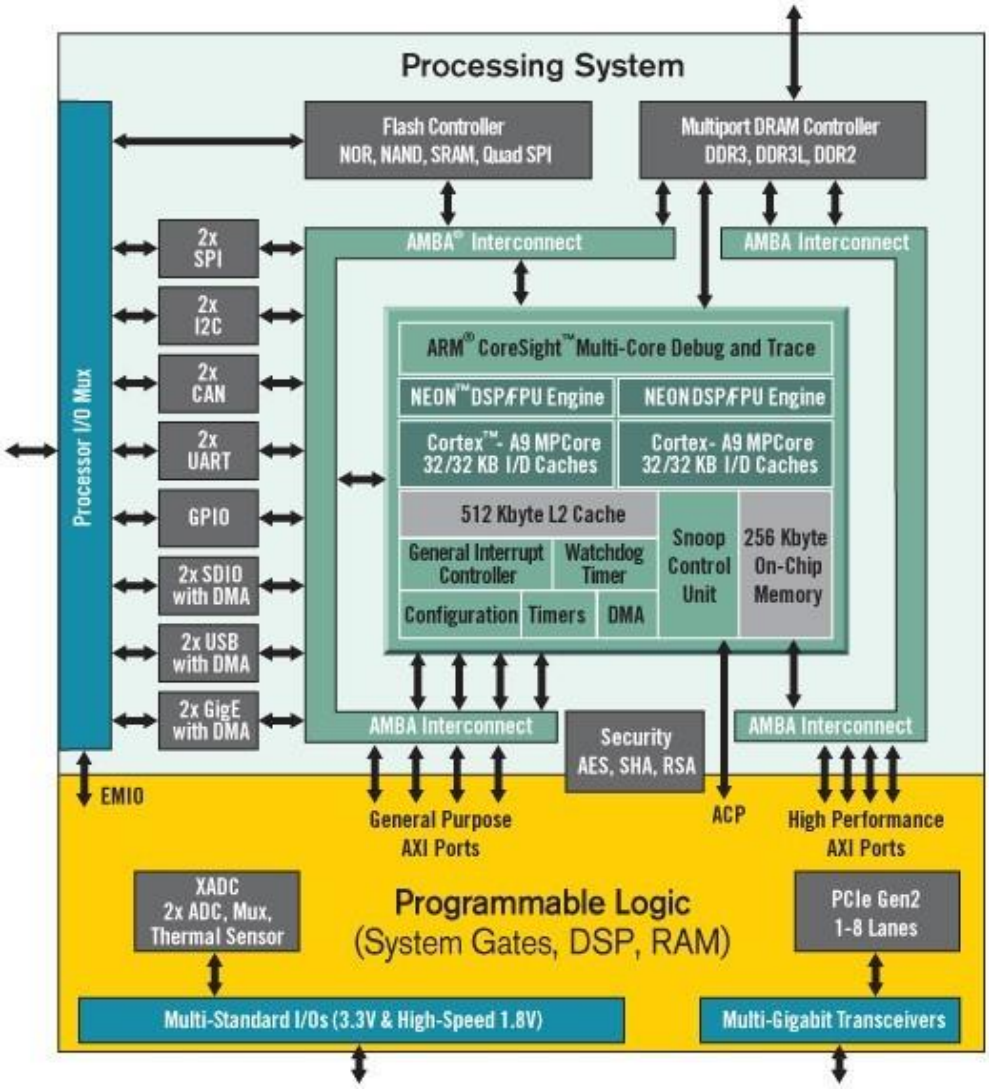
4 CASE STUDIES

This section presents the Zynq-7000 APSoC, the design strategy for the neuron's activation function, as well as the chosen ANNs and their correspondent datasets. Finally, we stablish the error criticality definition and the heuristics for the reliability evaluation.

4.1 Zynq-7000 APSoC

The Zynq-7000 is a device designed by Xilinx using a 28nm technology. In specific, this work utilizes the XC7Z020-CLG484 part, which is commercially available in the ZedBoard Development Board. Fig. 4.1 shows its block diagram.

Figure 4.1: Zynq-7000's block diagram

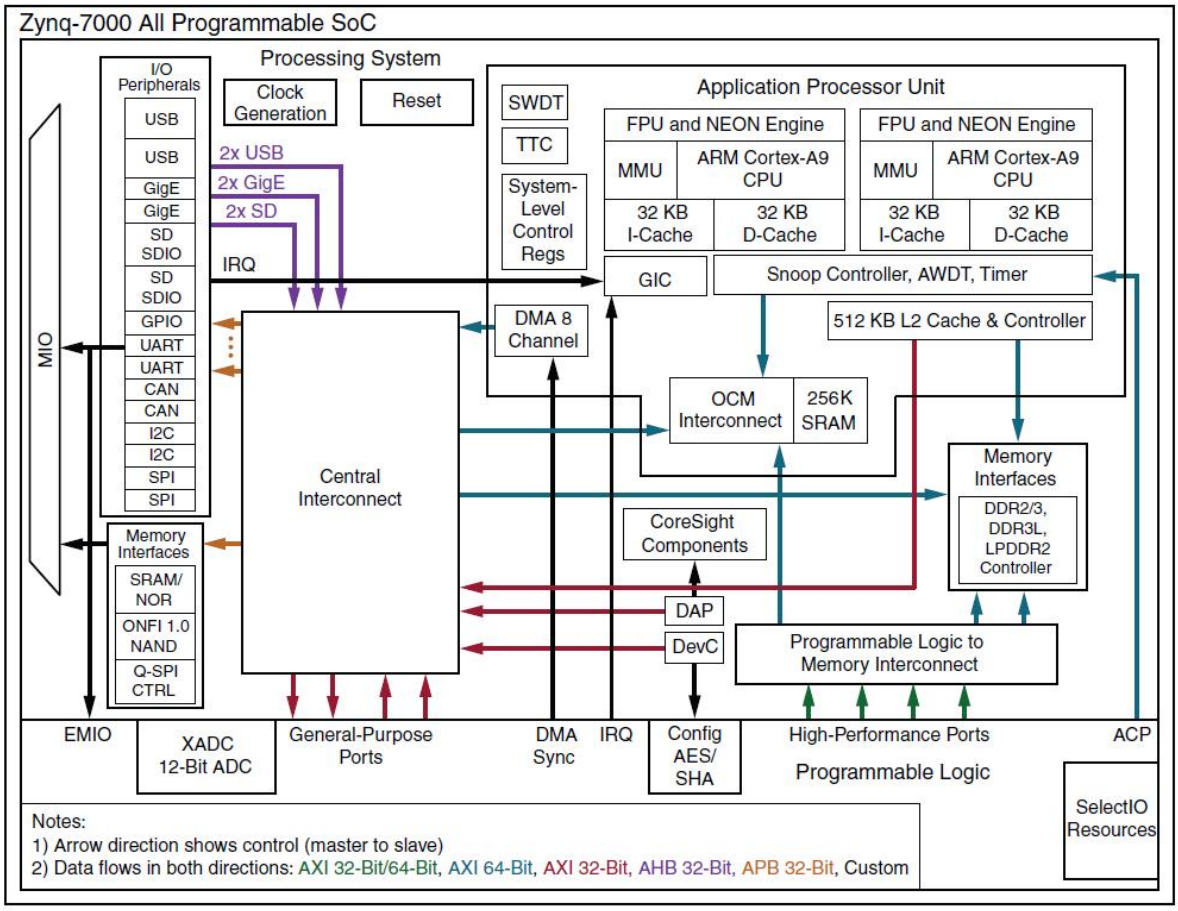


Font: (XILINX, 2017)

As we can see from Fig 4.1, there are two main parts in the Zynq-7000, which are the Processing System (PS) and the Programmable Logic (PL), and they are connected mainly through Advanced eXtensible Interface (AXI) ports.

The PS has a dual core ARM Cortex-A9 processor alongside other resources such as Floating Point Units (FPUs) and Memory Management Units (MMUs). The complete block diagram of the PS part is showcased by Fig 4.2.

Figure 4.2: Zynq-7000's PS part

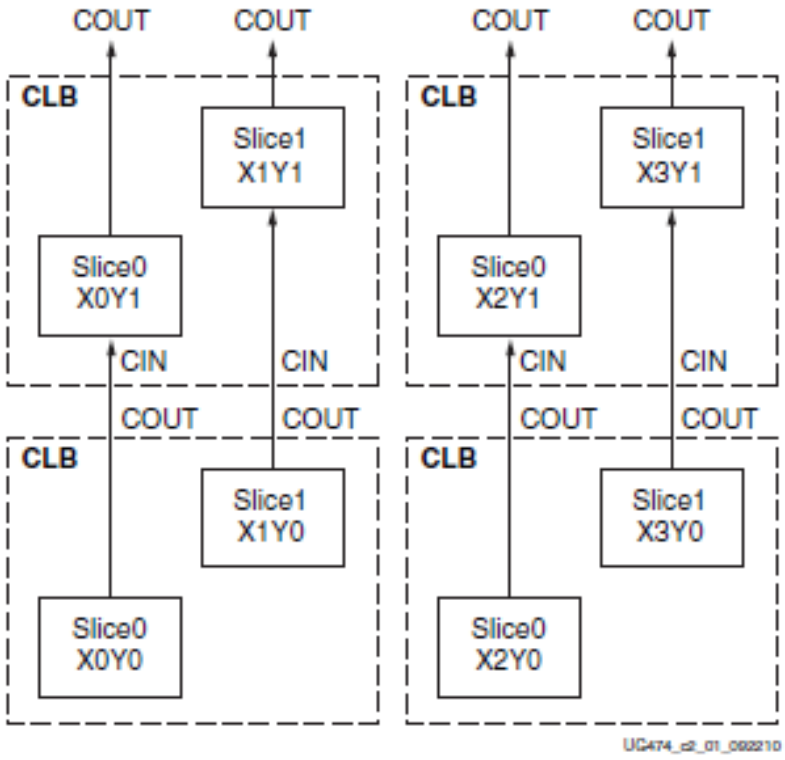


Font: (XILINX, 2015)

The PL part of the Zynq-7000 has the same internal architecture as two other families of Xilinx FPGAs: Artix-7 and Kintex-7. This architecture is composed mainly of Configurable Logic Blocks (CLBs), Digital Signal Processor (DSP) blocks and embedded memory blocks (BRAM). A set of programmable interconnections creates an array of programmable logic blocks of different types. All of these components are configured by the bitstream file, which loads into the configuration memory during the device power-up, ultimately defining a circuit previously described in a Hardware Description Language (HDL).

CLBs are used to implement the logic of the user's design. Each CLB is composed of one or more slices, and a slice contains one or more LUTs, Flip-Flops (FFs), and routing structures. Fig 4.3 illustrates the relationship between CLBs and slices.

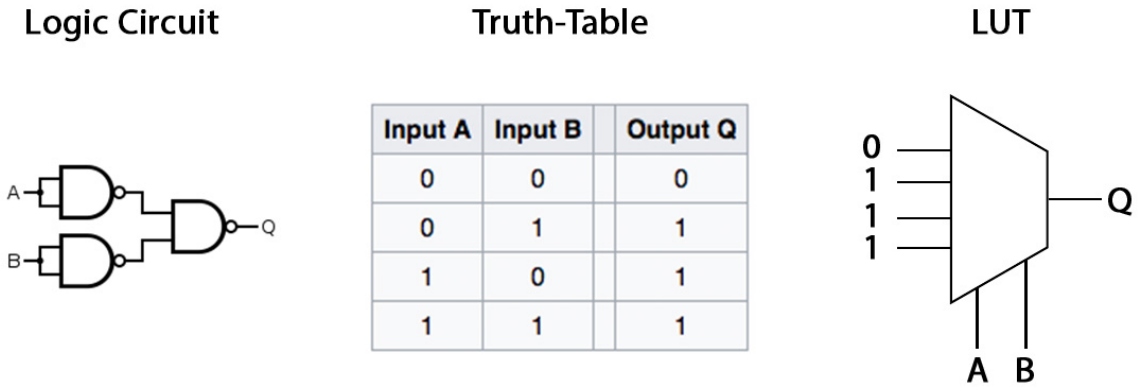
Figure 4.3: Relationship between CLBs and Slices in the Xilinx 7-Series FPGAs



Font (XILINX, 2014)

Logic functions, and their respective truth-tables, described in the HDL code are implemented in the LUTs as multiplexers with 2^n inputs and n selectors (Fig 4.4 serves as an example).

Figure 4.3: Example of a 2-input function implemented in the LUT

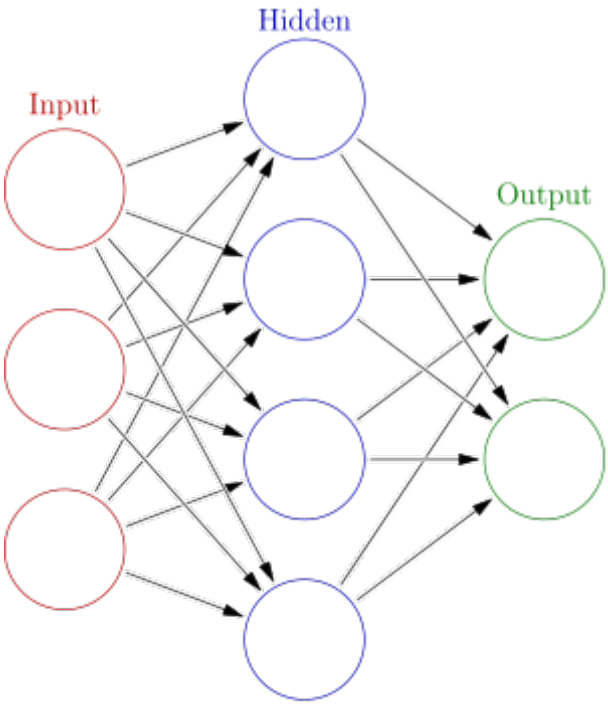


4.2 Artificial Neural Networks implemented in FPGAs

Artificial Neural Networks (ANNs) are a computational concept of the Machine Learning area, inspired on the biological structure of animal brains (MCCULLOCH et al., 1943). Analogously, the main component of an ANN is the neuron, which basically sums its inputs and applies an activation function, simulating the behavior of an axon, from a biology standpoint. The neurons are organized in layers, in a fully connected way. In other words, every neuron of a given layer *L* is connected to every neuron of the layers *L-1* and *L+1*. Every connection has an associated weight, which multiplies its input and passes it to the following neuron, similarly to what occurs during a synapse (ROSENBLATT, 1958). It is worth mentioning that the Input Layer does not perform any arithmetic operation.

The intelligence of a particular ANN varies with the topology (number of neurons and layers), and with the synaptic weights (for each connection between neurons). The topology is defined by the system designer, while the synaptic weights are defined during the training phase, where a large set of data is presented to the network, and the weights get updated depending on the discrepancy between the output and the expected value for each iteration. There are many variations of training algorithms, but the most commonly adopted nowadays is the Backpropagation (RUMELHART et al., 1986).

Figure 4.4: Sample structure of an ANN

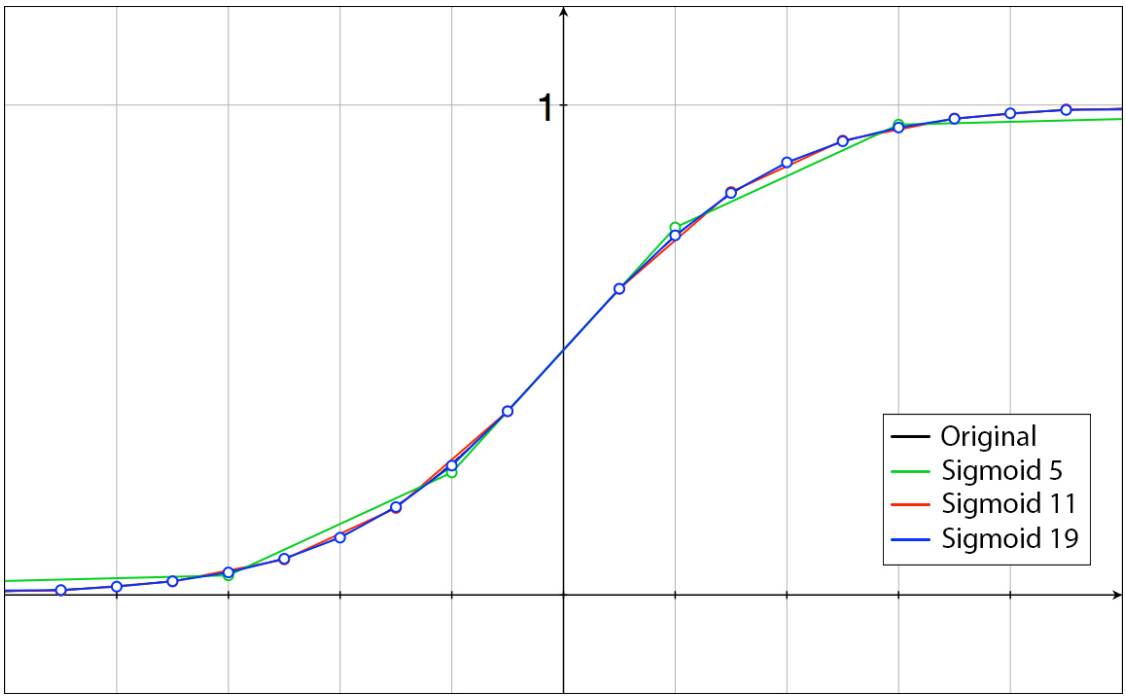


Font: (WIKIPEDIA, 2017)

4.2.1 Neuron's Activation Function

The activation function produces a non-linear decision boundary. It is an abstraction of the action potential concept from biology that, when applied to neurons, is also known as nerve impulse. The most used mathematical expressions of the activation function are sigmoidal, such as the hyperbolic tangent and the logistic. In this work, we analyze the logistic activation function shown in Fig. 4.5.

Figure 4.5: The original logistic function and the three levels of discretization



We can quickly notice that the sigmoid output is comprised between 0 and 1. Its mathematical expression involves an exponentiation as in Eq. 4.1.

(Equation 4.1)
$$sigmoid(x) = \frac{1}{1 + e^{-x}}$$

Implementing an exponential, for fixed or floating point data representations, in a HDL is not trivial and can lead to inefficiencies in the FPGA. A few solutions were proposed for this problem (BUI et al., 1999), but this work adopts the discretization strategy (AMIN et al., 2004). We use three levels of discretization of the activation function (shown in Fig. 4.5), resulting in three different neuron designs. As it can be noticed, *Sigmoid 19* (that divides the sigmoid in 19 steps) is very similar to the continuous function.

4.2.2 Boston Housing Dataset

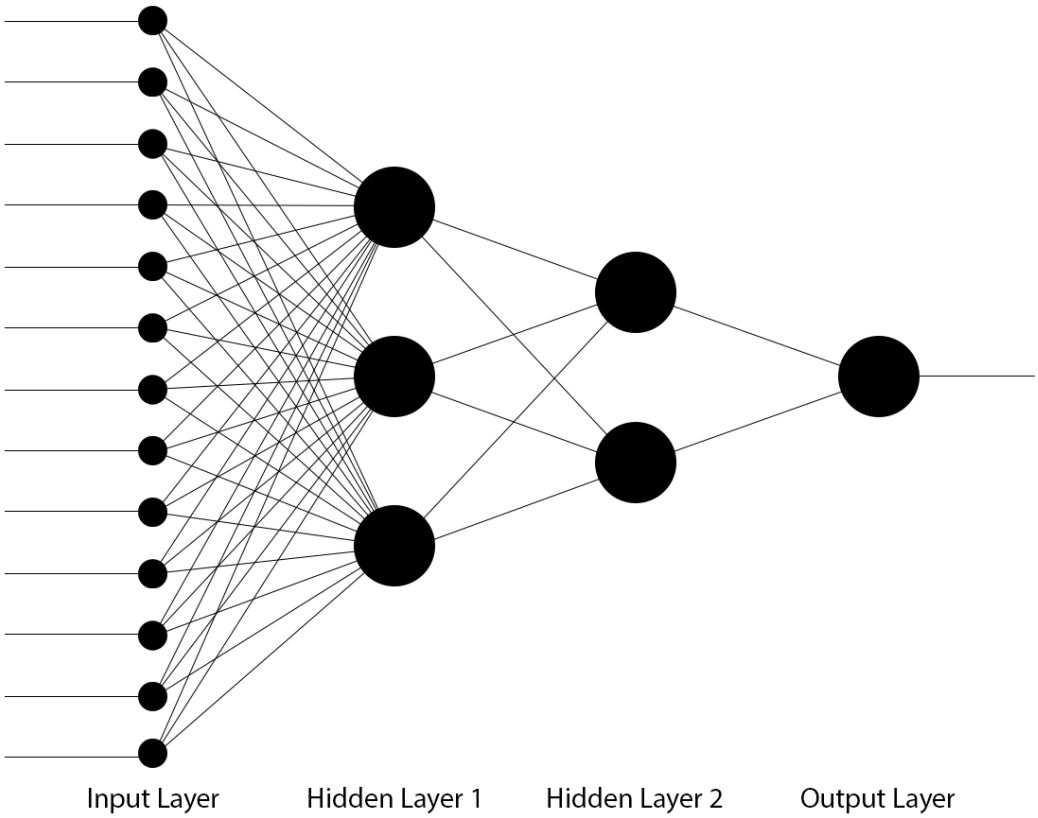
This dataset was first introduced by (HARRISON et al., 1978) and has been utilized in a number of relevant Machine Learning papers, such as (MERZ et al., 1999). It has a total of 506 instances, where each instance has fourteen attributes, thirteen of which being inputs and one being the output, as Table 4.1 shows.

Table 4.1: Instance attributes for the Boston Housing dataset

Attribute	**Description**
Input 1	Per capita crime rate
Input 2	Proportion of residential land zoned for lots over 25,000 sq.ft.
Input 3	Proportion of non-retail business acres
Input 4	Charles River dummy variable (=1 if tract bounds river, 0=otherwise)
Input 5	Nitric oxides concentration (parts per 10 million)
Input 6	Average numbers of rooms per dwelling
Input 7	Proportion of owner-occupied units built prior to 1940
Input 8	Weighted distances to five Boston employment centers
Input 9	Index of accessibility to radial highways
Input 10	Full-value property-tax rate per $10,000
Input 11	Pupil-teacher ration
Input 12	$1000(Bk-0.63)^2$ where Bk is the proportion of blacks
Input 13	Percentual lower status of the population
Output 1	Median value of owner-occupied homes in $1000's

The ANN implements a linear regression algorithm in order to predict the output value given the thirteen inputs. Its topology is illustrated in Fig 4.6. It is worth highlighting that for this particular case, the neuron in the output layer doesn't apply the activation function, since the expected value is not restricted to any subinterval of $\mathbb{R}$.

Figure 4.6: Boston Housing ANN's topology



In addition, Table 4.2 presents the Zynq's FPGA resource utilization by our implemented topology of the ANN. It is very noticeable that the FF and DSP percentages are equal across all cases. This is due to the fact that the pipeline structure (FFs) doesn't depend on the level of sigmoid discretization, nor do the connections between the neurons (DSPs).

Table 4.2: Zynq's FPGA resource utilization by the linear regression ANN

Resource (#)	**Sigmoid 19**	**Sigmoid 11**	**Sigmoid 5**
FF (106,400)	3.1%	3.1%	3.1%
DSP (220)	96.3%	96.3%	96.3%
LUT (53,200)	57.9%	42.1%	27.2%

4.2.3 Iris Flower Dataset

In order to evaluate another type of algorithm in ANNs (specifically: classification), we chose the Iris Flower dataset, which is well-know in the Pattern Recognition literature, and was first introduced by (FISCHER, 1936), but the actual data was collected by (ANDERSON, 1935)

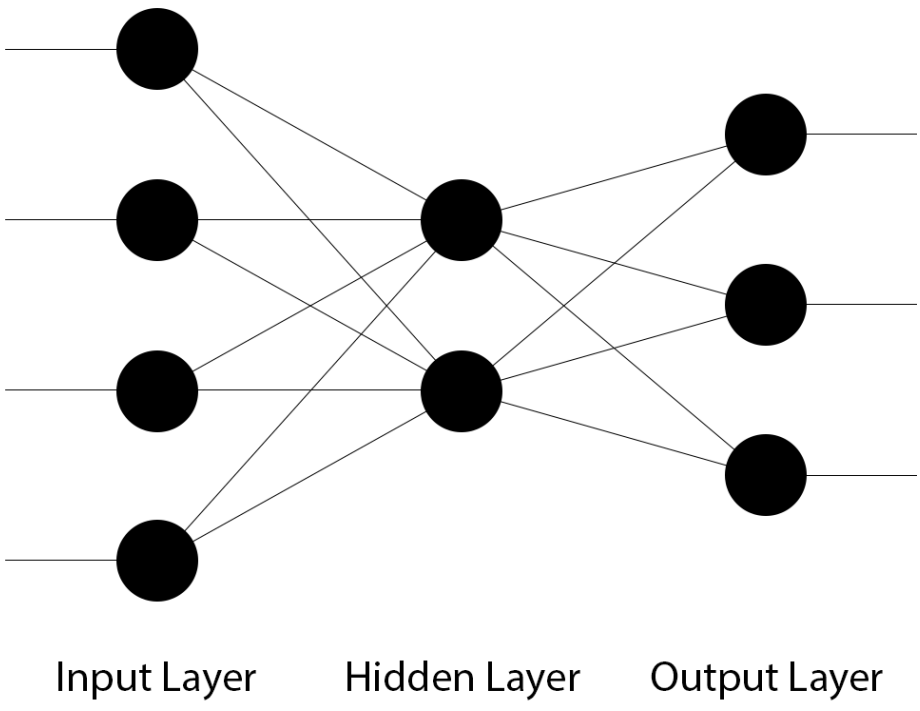
who wanted to quantify the morphologic variation of three related species of the Iris Flower: Setosa, Versicolor and Virginica. The dataset has a total of 150 instances, divided in three classes of 50, where each class refers to a species. Table 4.3 shows the seven attributes of each instance.

Table 4.3: Instance attributes for the Iris Flower dataset

Attribute	**Description**
Input 1	Sepal Length
Input 2	Sepal Width
Input 3	Petal Length
Input 4	Petal Width
Output 1	Iris Setosa
Output 2	Iris Versicolor
Output 3	Iris Virginica

These instances differ from the ones presented in Section 4.2.2 because there is more than one output for every set of inputs. The actual classification is obtained by verifying which output is higher. Fig. 4.7 illustrates this ANN's topology

Figure 4.7: Iris Flower ANN's topology



Finally, Table 4.4 presents the Zynq's FPGA resource utilization by our implemented topology of the ANN. Once again, the percentage of utilized FF and LUT resources is the same for all sigmoids.

Table 4.4: Zynq's FPGA resource utilization by the classification ANN

Resource (#)	**Sigmoid 19**	**Sigmoid 11**	**Sigmoid 5**
FF (106,400)	1.5%	1.5%	1.5%
DSP (220)	34.6%	34.6%	34.6%
LUT (53,200)	44.2%	29.2%	14.0%

4.3 Error Criticality and Reliability Evaluation

A radiation-induced output error is not always critical for the ANN. In fact, while the result is based on the output of the last neurons layer, a corrupted output can still lead to a correct behavior. In other words, the particle can induce the ANNs to produce wrong output values but, based on these (corrupted) values the system can still be considered properly functional. In this situation, we consider the error as *tolerable*. Otherwise, the error is marked as *critical.* Note that we adopt this nomenclature to facilitate the understanding, since the formal definition in Section 2.4.1 might be confusing to readers who are unfamiliar with the area.

As described in Sections 4.2.2 and 4.2.3, our datasets are composed of 506 and 150 instances each. As the radiation-induced error in SRAM-based FPGAs has a persistent effect, we want to measure how many of the input instances it affects (in a critical or tolerable way). To do so, when a mismatch between the ANN's output and the expected output is detected (either under radiation or in our fault-injection experiment), we run all the instances of the datasets and evaluate the error criticality.

Each application has its proper definition of a critical or tolerable error. For the Boston Housing ANN, we consider the Tolerated Relative Error (TRE), which defines an interval with boundaries of criticality, mathematically expressed as in Eq. 4.2:

(Equation 4.2)
$$criticality(x) = \begin{cases} critical, & x < G - TRE * G \\ tolerable, & G - TRE * G \leq x \leq G + TRE * G \\ critical, & x > G + TRE * G \end{cases}$$

where x is the ANN's output and G is the expected/gold value. As an example, with a TRE of 10%, the tolerable interval would be [0.9x, 1.1x].

For the Iris Flower ANN identifying a critical or tolerable error is easier: if any of the three outputs is different than expected, but the flower classification is still correct, the error is tolerable, otherwise it is critical.

Based on these considerations we identify four possible radiation effects on ANNs, indicated in Table 4.5:

Table 4.5: Radiation effects on ANNs

(1) Single Tolerable	one and only one element in the dataset produces output errors, but the application's behavior can still be considered correct
(2) Multiple Tolerable	more than one element in the dataset produces output errors, but the application's behavior can still be considered correct
(3) Single Critical	one and only one element in the dataset produces an output errors considered critical for the application (there may be one or more tolerable errors)
(4) Multiple Critical	more than one element in the dataset produces output errors which are considered critical for the application (there may be one or more tolerable errors)

In Section 5, we consider both radiation and fault injection results based on (1) to (4).

5 RESULTS

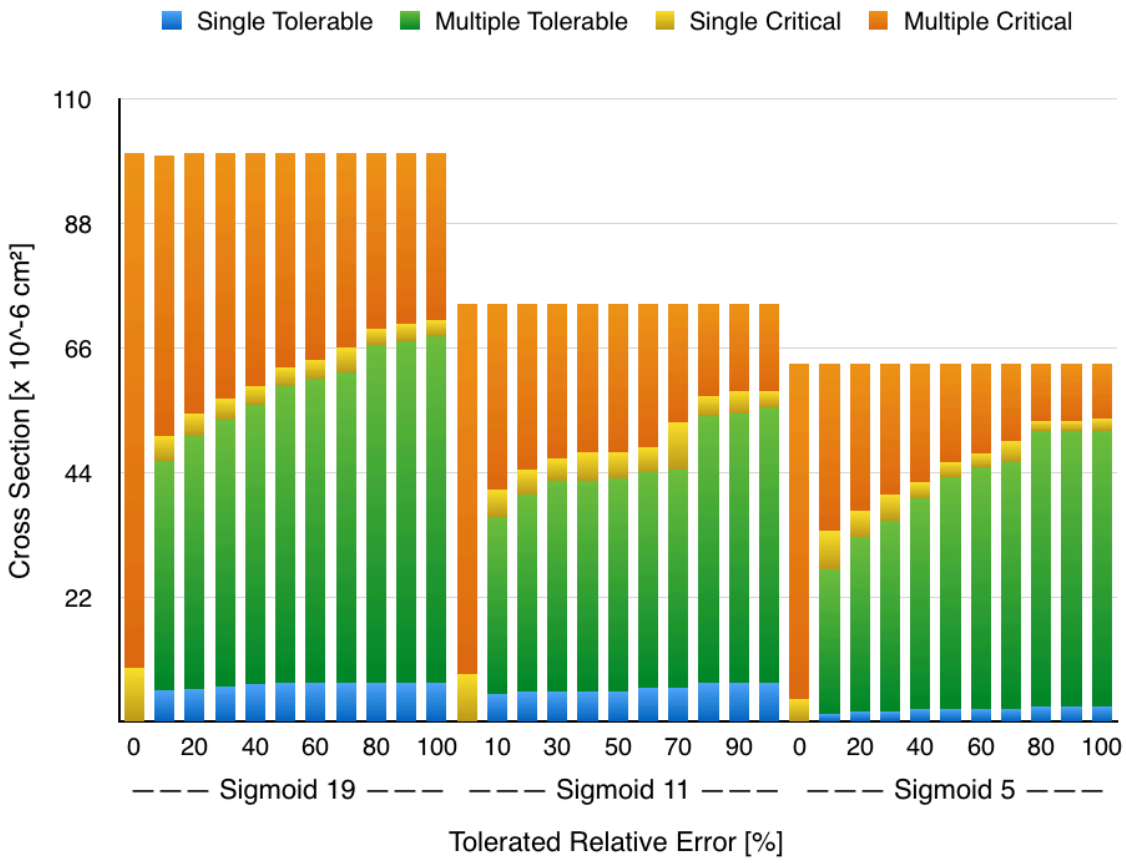
In this section we present and discuss the results obtained through our beam experiments and fault-injection framework. Then, we utilize both experimental methodologies to draw generic conclusions on ANN reliability.

5.1 Radiation Experiment

Using the setup described in Section 3.1.1 we have irradiated the device for about 15 hours, for a total fluence of 1.14×10^7 ions/cm². We have collected more than 100 errors for each configuration, and they are divided in the four categories discussed in Section 4.3.

Fig. 5.1 shows the heavy ions cross section of the Boston Housing ANN for the three different levels of discretization of the neuron's activation function. We divide the contribution of single/multiple tolerable/critical errors as a function of the TRE, which we vary from 0% (any difference from the expected value is critical) to 100% (a relative error as high as the output value is tolerable) in 10% increments.

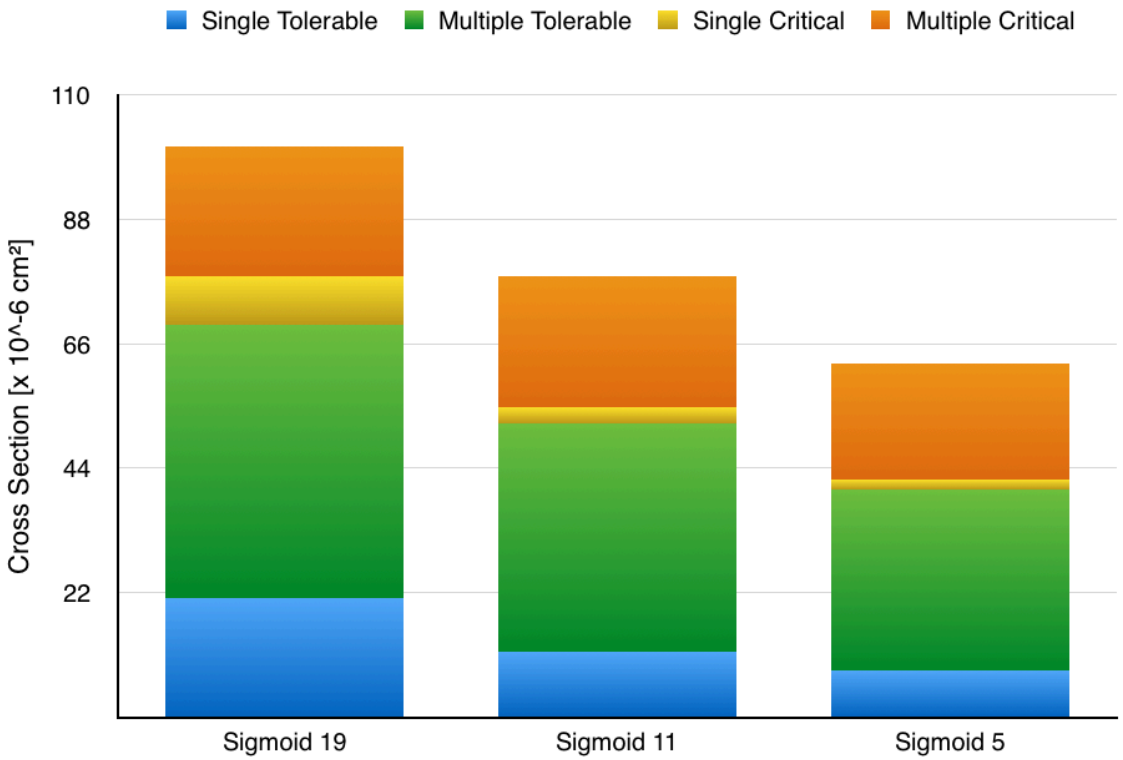
Figure 5.1: Heavy-ions cross section of the Boston Housing ANN



From Fig. 5.1 it is clear that a more complex and precise discretization of the sigmoid function makes the ANN more likely to be corrupted. In fact, Sigmoid 19 cross section is 30% higher than Sigmoid 11 and 40% higher than Sigmoid 5. An important observation from our experimental data is that the probability of all error types scales with the sigmoid complexity. Unfortunately, multiple errors (when more than one instance generates an unexpected output), are predominant over single occurrences. Multiple errors are likely to be caused by particles affecting the connections among neurons more than the neurons themselves. Also, when the TRE is 0%, by definition every error is considered critical, but as the TRE is incremented, the number of critical errors decreases.

The cross section of the Iris Flower ANN is plotted in Fig 5.2. It is interesting to notice that Iris Flower cross section has a very similar trend compared to Boston Housing. It is reasonable to state that less precise discretization leads to smaller cross section.

Figure 5.2: Heavy-ions cross section of the Iris Flower ANN



Experimental data also highlights that most of radiation-induced errors on the tested ANNs are not to be considered critical. As discussed in Section 4.3, both single and multiple tolerable errors identify those corruptions that do not compromise the application's correctness. In other words, those are not failures in the particular application. As shown in Fig. 5.2, about 65% of radiation-induced corruptions lead to single or multiple output errors, without any impact on classification of the Iris Flower.

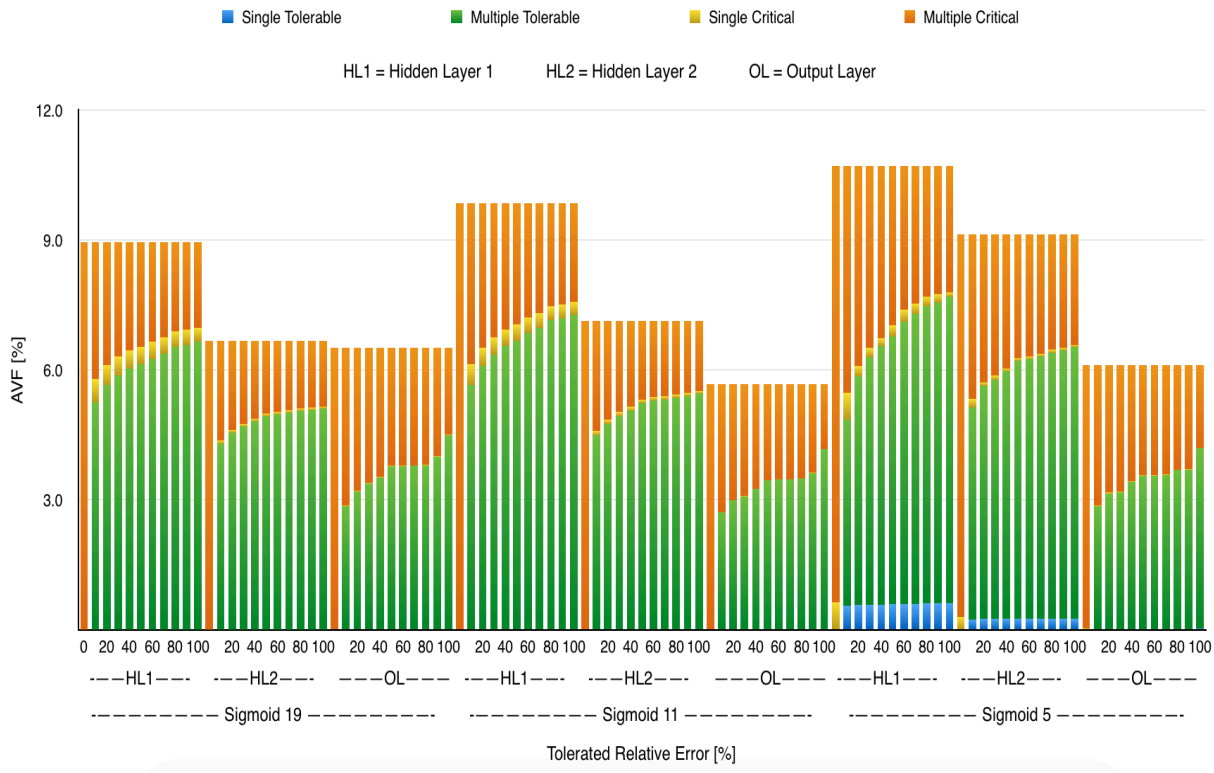
5.2 Fault Injection

While offering several advantages, radiation experiments allow little visibility of error propagation. As radiation-induced errors are detected only at the output, it is not possible to correlate the observed errors with the source of corruption. We take advantage of the fault-injection framework described in Section 3.2.1 to better understand the observed phenomena. We decide to inject faults all in neurons from different layers of the networks, and not in the connections between neurons. In fact, our intent is to comprehend the impact of faults in neurons and their propagation through the connections. This choice is also justified by the relevance of the neuron as the main and most original component of ANNs.

It is worth noting that a direct comparison between fault-injection and radiation-experiment results is not possible. In fact, while beam experiments provide the probability of errors to occur, fault-injection evaluates the probability of a fault to affect the output. As such, fault injection can provide deeper insights on ANN reliability.

Fig. 5.3 shows the results of our fault injection in the Boston Housing ANN, expressed as AVF. We divide the fault-injection outcomes based on the four categories discussed in Section 4.3. Faults were injected separately in the Hidden Layer 1 (HL1), Hidden Layer 2 (HL2) and Output Layer (OL) neurons of the chosen ANN.

Figure 5.3: AVF of the Boston Housing ANN

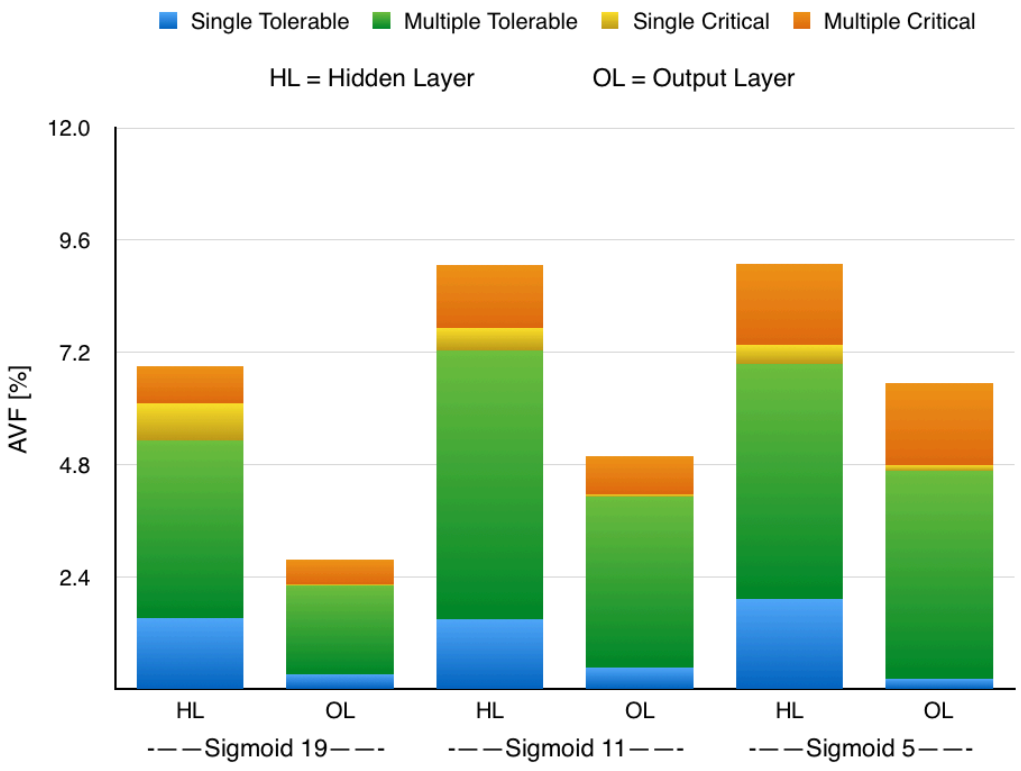


It is clear from Fig. 5.3 that injections in the initial layers are more likely to generate errors and failures. In addition, the trend of single/multiple tolerable/critical errors is very similar to the observed in the radiation experiment (Fig. 5.1).

In Fig. 5.3 the AVF is inversely proportional to the discretization level of the activation function, only in apparent contrast with radiation experiment results. In fact, the higher cross section for more precise discretization is mostly justified by the higher amount of resources used to implement the circuit. Fault-injection results, on the contrary, are dependent on the probability for a fault in a used resource to affect computation. The AVF, then, removes the dependence on the exposed area and is focused on the effect of radiation-induced faults. As shown in Fig. 5.3, the higher the precision of the neuron's response is, the lower its architectural vulnerability will be. This may seem odd at first sight, but it is due to the fact that a more precise discretization for the neuron's activation function implies in higher LUT utilization, and the number of essential configuration bits is proportional to the design's complexity. In the first studied case, for example, the corruption of a single bit through fault injection represents 0.00011% of the neuron's circuit using sigmoid 19, while the same corruption represents 0.00043% (4 times more) of the neuron using sigmoid 5.

As for the classification ANN, the faults were injected separately in the Hidden Layer (HL) and Output Layer (OL). Results are shown in Fig. 5.4.

Figure 5.4: AVF of the Iris Flower ANN



Once again the likeability of error and failure occurrence decreases as the affected layer is closer to the output. This means that hardening strategies should have priority of adoption in the earlier layers of the network.

To demonstrate that both experiments are in total accordance, we present Fig 5.5 and Fig 5.6. They show the absolute number of observed errors during the fault injection campaigns for the Boston Housing ANN and the Iris Flower ANN, respectively.

Figure 5.5: Total observed errors in fault injection (Boston Housing ANN, TRE= 10%)

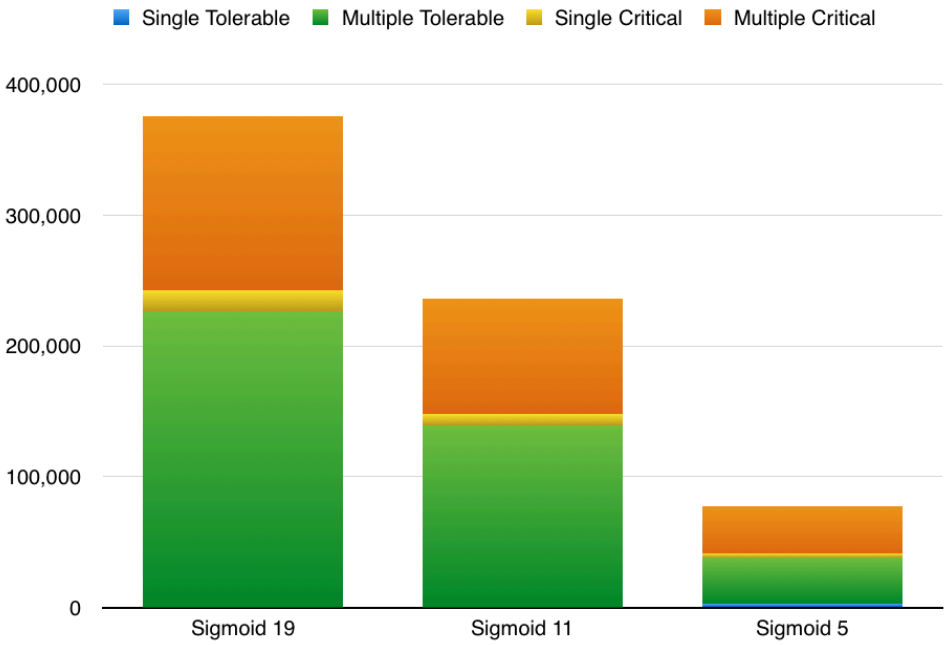
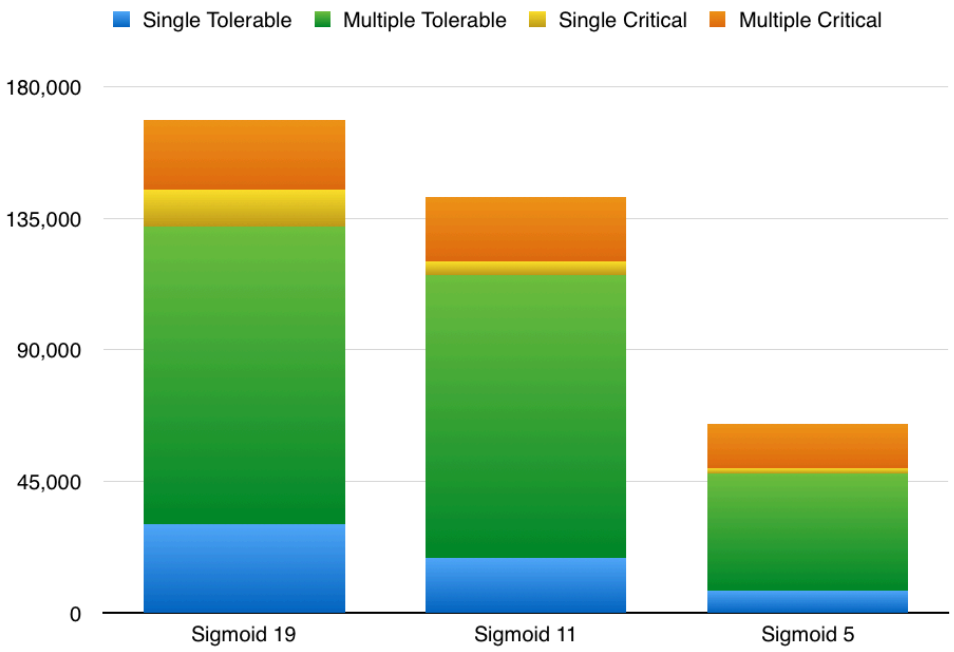


Figure 5.6: Total observed errors in fault injection (Iris Flower ANN)



5.3 Discussion

Analyzing the beam experiments and fault-injection results presented in Sections 5.1 and 5.2 respectively, we can primarily perceive that tolerable errors are dominant over critical errors. Hence, while transient faults affect the ANN execution, in most of the cases their effect is not sufficient to jeopardize the application's purpose. Interestingly, the portion of errors that are to be considered critical is similar between beam experiments and fault-injection. This suggests that the criticality of errors in the ANNs is an intrinsic property of the circuit, and does not depend on the source of error or the probability of corruption. Using fault-injection results it is then possible to harden the ANN circuit, protecting the sources of critical errors.

Furthermore, it is unlikely for radiation to affect the computation of one single instance at the time, for both beam experiment and fault-injection. In other words, in the majority of cases when corruption occurs, it appears in multiple instances of the dataset. This is because the ANNs use all neurons and connections to determine its output. Modifying one neuron or connection is then likely to affect more than one output.

6 CONCLUSIONS

Experimental results indicate an overall low probability of error occurrence in the ANNs implemented on an FPGA due to the intrinsic masking effect capability of the neurons. Additionally, the application's functionality is maintained even for most of the observed output errors, for all studied cases. The critical errors represent the smaller portion of the total number of events. Our results attest that not all the radiation-induced errors are critical in the ANNs, which is a promising result for safety-critical applications. Moreover, thanks to our fault-injection campaign we discovered that the inner layers of neurons may be more critical, and should be carefully hardened by fault tolerant techniques.

As a personal experience, we learned that fault injections are significantly more time consuming than standard radiation tests (in this particular case, 20 times more). This is due to the fact that fault injection campaigns are performed extensively (in every bit of the FPGA's configuration memory, that corresponds to the determined injection area), while beam experiments are conducted for just enough time to observe a statistically significant number of errors. But as briefly explained in Section 3.3, both results are necessary in order to perform a complete analysis of a system's reliability, since, by definition, they provide different perspectives on fault effects.

As future work, we intend to evaluate the behavior of more complex ANNs, as well as to reduce the resource utilization of the neuron's activation function by exploring its aspect of symmetry, and to implement fault tolerance techniques such as Triple Modular Redundancy (TMR) in the initial layers of the network.

REFERENCES

AGUIAR, V. A. P.; ADDED, N.; MEDINA, N. H.; MACCHIONE, E. L. A.; TABACNIKS, M. H.; AGUIRRE, F. R.; SILVEIRA, M. A. G; SANTOS, R. B. B.; SEIXAS, L. E. Experimental Setup for Single Event Effects at the São Paulo 8UD Pelletron Accelerator. **Nuclear Instruments & Methods in Physics Research**, vol. 332, 2014, pp. 397-400.

ALTERA. Cyclone V FPGAs & SoCs, 2017. Available at: <https://www.altera.com/products/fpga/cyclone-series/cyclone-v/overview.html>. Accessed on: July 2017.

AMIN, H.; CURTIS, K. M.; HAYES-GILL, B. R. Piecewise linear approximation applied to nonlinear function of a neural network. **IEEE Proceedings on Circuits Devices and Systems**, pp. 313–317, 1997.

AMIN, S. U.; AGARWAL, K.; BEG, R. Genetic Neural Network Based Data Mining in Prediction of Heart Disease Using Risk Factors. **IEEE Information and Communication Technologies (ICT)**, 2013.

ANDERSON, E. The Irises of the Gaspe Peninsula. **Bulletin of the American Iris Society**, vol. 59, pp 2-5, 1935.

AVIZIENIS, A.; LAPRIE, J. C.; RANDELL, B.; LANDWEHR, C. Basic Concepts and Taxonomy of Dependable and Secure Computing. **IEEE Transactions on Dependable and Secure Computing**, vol. 1, n. 1, Mar. 2004, pp. 11-33.

BAUMANN, R. Radiation-induced soft errors in advanced semiconductor technologies. **IEEE Transactions on Device and Materials Reliability**. v. 5, n. 3, p. 305–316, Sept. 2005. ISSN 1530-4388.

BERNARDESCHI, C.; CASSANO, L.; DOMENICI, A. SRAM-Based FPGA Systems for Safety-Critical Applications: A Survey on Design Standards and Proposed Methodologies. **Journal of Computer Science and Technology**. 30(20), pp. 372-390, Mar. 2015.

BUI, H.; TAHAR, S. Design and synthesis of an IEEE-754 exponential function. **IEEE Canadian Conference on Electrical and Computer Engineering**, pp. 450–455, May 1999.
HARRISON, D.; RUBINFIELD, D. L. Hedonic housing prices and demand for clean air. **Journal of Environmental Economics Management**, vol. 5, pp. 81-102, 1978.

ELECTREK. Elon Musk on Tesla fully autonomous car, 2016. Available at: <https://electrek.co/2016/08/03/elon-musk-tesla-fully-autonomous-car-blows-mind/>. Accessed on July 2017.

ESA. **ESA/SCC Basic Specification No. 25100: Single Event Effects Test Methods and Guidelines**. ESA, Noordwijk, Netherlands, 2005.

FISCHER, R. A. The use of multiple measurements in taxonomic problems. **Annals of Eugenics**, vol. 7, Part II, pp 179-188, 1936.

HE, C.; PAPAKONSTANTINOU, A.; CHEN, D. A novel SoC architecture on FPGA for ultra fast face detection. **ICCD**, pp. 412–418, 2009.

IEC. Functional Safety, 2017. Available at: < http://www.iec.ch/functionalsafety/faq-ed2/page5.htm>. Accessed on: July 2017.

IEEE. **IEEE Standard Computer Dictionary: A Compilation of IEEE Standard Computer Glossaries**. New York, NY, 1990.

JEDEC. Measurement and Reporting of Alpha Particle and Terrestrial Cosmic Ray-Induced Soft Errors in Semiconductor Devices, **JESD89A** (Revision of JESD89), Oct. 2006.

MCULLOCH; WARREN; PITTS, W. A Logical Calculus of Ideas Immanent in Nervous Activity. **Bulletin of Mathematical Biophysics**. 5 (4): 115–133, 1943.

MEROLLA, P. A. et al. A million spiking-neuron integrated circuit with a scalable communication network and interface. **Science**, 345 (6197): 668, 2014.

MERZ C. J.; PAZZANI, M. J. A Principal Components Approach to Combining Regression Estimates. **Machine Learning**, vol. 36, pp. 9-32, July 1999.

MICROSEMI. SmartFusion SoC FPGAs, 2017. Available at: <http://www.microsemi.com/products/fpga-soc/soc-fpga/smartfusion>. Accessed on: July 2017.

MOORE, G. E. Cramming More Components Onto Integrated Circuits. **Electronics**, vol. 38, n. 8, Apr. 1965, pp. 114-117.

MUKHERJEE, S. S.; WEAVER, C. T.; EMER, J.; REINHARDT, S. K.; AUSTIN, T. Measuring Architectural Vulnerability Factors. **IEEE Micro**, vol. 23, n. 6, Nov. 2003, pp. 70-75.

NEAGOE, V. E; CIOTEC, A. D.; ROPOT, A. A Concurrent Neural Network Approach to Pedestrian Detection in Thermal Imagery. **Communications (COMM) 9th International Conference**, 2012.

QUARTZ. Google's AI won the game Go by defying millennia of basic human instinct, 2016. Available at < https://qz.com/639952/>. Accessed on July 2017.

ROSENBLATT, F. The Perceptron: A Probabilistic Model For Information Storage And Organization In The Brain. **Psychological Review**. 65 (6): 386–408, 1958.

RUMELHART, D. E.; HINTON; GEOFFREY, E.; WILLIANS; RONALD, J. Learning representations by back-propagating errors. **Nature**. 323 (6088): 533–536, 1986.

SAMUEL, A. Some Studies in Machine Learning Using the Game Checkers. **IBM Journal**. July 1959.

SIEGLE, F. et al. Mitigation of radiation effects in sram-based fpgas for space applications. **ACM Computing Surveys (CSUR),** ACM, v. 47, n. 2, p. 37, 2015.

TAMBARA, L. A.; RECH, P.; CHIELLE, E.; TONFAT, J.; KASTENSMIDT, F. L. Analyzing the Impact of Radiation-induced Failures in Programmable SoCs. **IEEE Transactions on Nuclear Science**, vol. 63, n. 4, Aug. 2016, pp. 1-8.

TAMBARA, L. A. **Analyzing the Impact of Radiation-induced Failures in All Programmable System-on-Chip Devices**, 2017, 192 f. Thesis (Doctor of Philosophy in Microelectronics) – Instituto de Informática, Universidade Federal do Rio Grande do Sul, Porto Alegre, 2017.

TONFAT, J. L.; TAMBARA, L. A.; SANTOS, A.; KASTENSMIDT, F. L. Method to Analyze the Susceptibility of HLS Designs in SRAM-Based FPGAs Under Soft Errors. **Lecture Notes in Computer Science**, vol. 9625, Mar. 2016, pp. 132-143.

WIKIPEDIA. Artificial Neural Networks, 2017. Available at: <https://en.wikipedia.org/wiki/Artificial_neural_network>. Accessed on: July 2017.

WIRTHLIN, M. High-Reliability FPGA-Based Systems: Space, High-Energy Physics, and Beyond. **Proceedings of the IEEE**, vol. 103, n. 3, Apr. 2015, pp. 379-389.

WIRHTLIN, M.; LEE, D.; SWIFT, G.; QUINN, H. A Method and Case Study on Identifying Physically Adjacent Multiple-Cell Upsets Using 28-nm, Interleaved and SECDED-Protected Arrays. **IEEE Transactions on Nuclear Science**, vol. 61, n. 6, pp. 3500-3505, Dec. 2014.

XILINX. 7 Series FPGAs Configurable Logic Block – User Guide. UG474 (v.17) November 17, 2014.

XILINX. 7 Series FPGAs Configuration – User Guide. UG470 (v1.10) June 24, 2015.

XILINX. Zynq-7000 All Programmable SoC, 2017. Available at: <http://www.xilinx.com/products/silicon-devices/soc/zynq-7000.html>. Accessed on: July 2017.