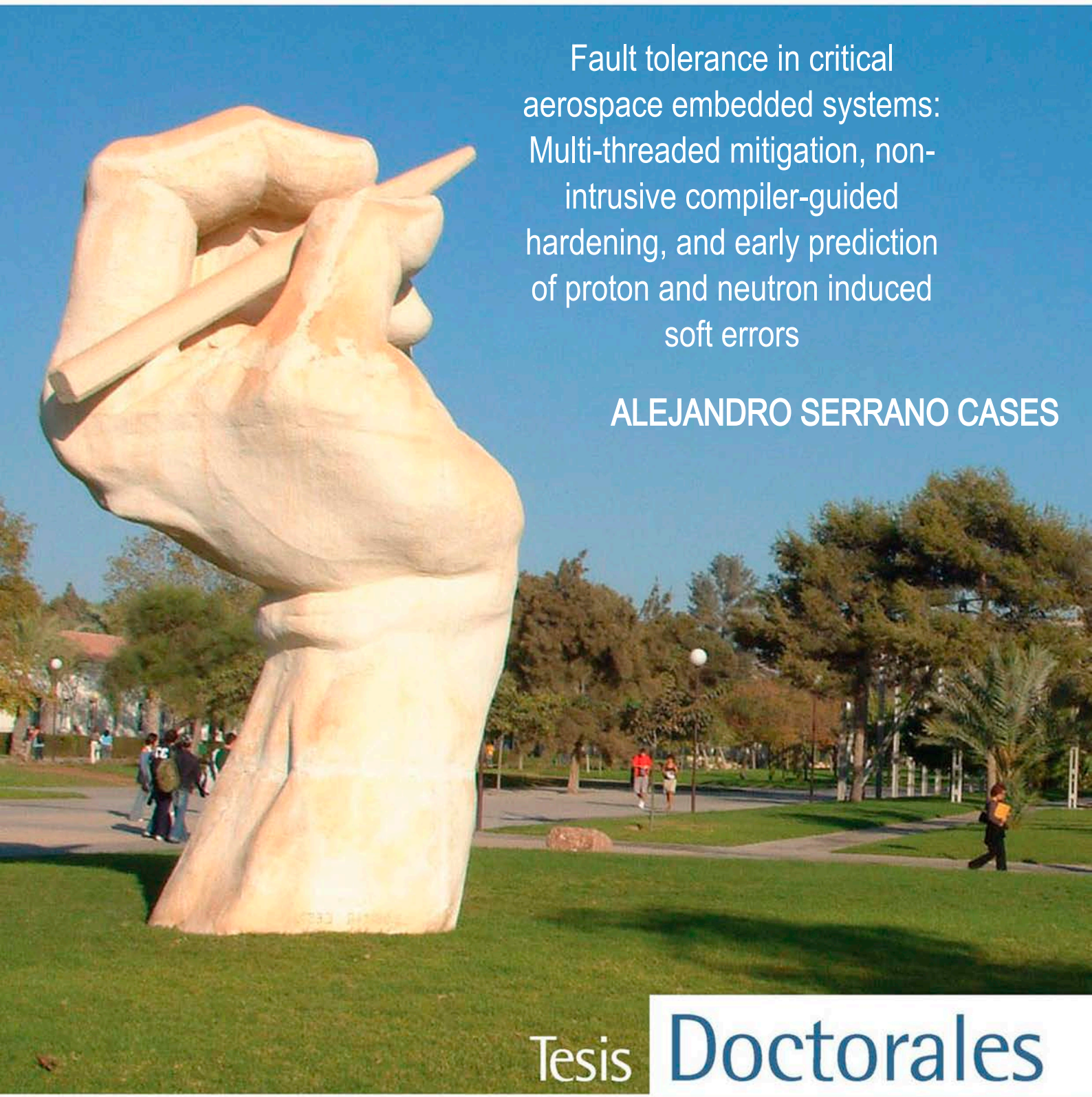
Universitat d'Alacant
Universidad de Alicante

Fault tolerance in critical aerospace embedded systems: Multi-threaded mitigation, non-intrusive compiler-guided hardening, and early prediction of proton and neutron induced soft errors

ALEJANDRO SERRANO CASES



Tesis Doctorales
UNIVERSIDAD de ALICANTE

Unitat de Digitalització UA
Unidad de Digitalización UA

Universitat d'Alacant
Universidad de Alicante

Alejandro Serrano Cases

Fault tolerance in critical aerospace embedded systems: Multi-threaded mitigation, non-intrusive compiler-guided hardening, and early prediction of proton and neutron induced soft errors

Alejandro Serrano Cases

ED|UA Escola de Doctorat
Escuela de Doctorado
edua.ua.es

Tesis doctoral
Alicante, agosto 2020

Tesis doctoral
Alicante, agosto 2020

Universitat d'Alacant
Universidad de Alicante

Acknowledgements

I would like to thanks all people who have contributed towards shaping this thesis. First and foremost, I would like to thank my supervisor Professor Antonio Martínez and advisors Professor Sergio Cuenca and Leonardo Reyneri from Politécnico de Torino, whose guidance, great support and advice made possible this thesis. As my supervisor and advisors, they have constantly forced me to remain focused on achieving my goal. I am also grateful for moral support I received from them.

It is also dedicated to all members of the Department of Computer Science and Technology and the members of the group UniCAD from Alicante University: Toni, Paco, Jose Luis, Marcelo and Isaza that put all their time and effort into make the development of this thesis easier. I also want to thank the great time we spent together and the fun we had with the technical staff, which made it a very enjoyable experience. It is also dedicated to the people of Federal University of Rio Grande do Sul: Fernanda, Gennaro and Iuri. Also, thanks to all the members of the group of Diseño Microlectrónica y Aplicaciones from Universidad Carlos III of Madrid: Luis Entrena, Mario García, Manuel Peña and Almudena Lindoso, for the great time I had during our collaboration, the radiation testing at Los Alamos and the SERESSA course at Seville. Thanks to all the team at Universidad de Sevilla, Hipolito and Maria, and the members of the Centro Nacional de Aceleradores (CNA), with special mention to Yolanda, Pedro and Begoña who supported us in the radiation facilities and make us spend a great time during the experiment. Also, I would like to thank to the staff in Los Alamos National Laboratory to their support during the accelerated neutron testing at the WNR.

I would like to thank all my colleagues and friends all time we spend together. Those moments made it more enjoyable the development of the thesis. Thanks, all of you: Fran, Victor, Albert, John, Sergiu, Rafa, Luis, Jose Maria. Thanks to all my family that supported me and reminded me that it is necessary to take a break from time to time. Thanks to Salva, Angel, Mari and Pilar for showing me that life is to enjoy it in every single way. Also, thanks to all people from Teruel: Paulino, Vicente, Faustino, Julio, Andres, Roberto, Carmen, Lola and Javier.

August 12, 2020

Resumen

Hoy día, existe una creciente demanda de las capacidades computacionales en sistemas críticos, donde los estados inesperados o inoperantes no son aceptables. Algunos de estos sistemas funcionan en entornos hostiles, sufriendo un comportamiento anómalo (*faults*), tanto en el software como en el hardware. Con objeto de solucionar esta problemática, se está recurriendo a la utilización de soluciones de computación, que explotan las nuevas características presentes en los microprocesadores de última generación. Entre estas características, destaca un mayor número de núcleos, mejor rendimiento computacional y menor consumo energético. Esta evolución de los microprocesadores es debida, entre otros factores, a la mejora en el proceso de fabricación fotolitográfico, sin embargo, este proceso está reduciendo progresivamente la tolerancia de los nuevos microprocesadores a los fallos inducidos por la radiación, conocidos como Efecto de Evento Único (*Single Event Effect* – SEE). Destacando, entre las fuentes que generan un comportamiento anómalo, las fuentes naturales de radiación, como los rayos cósmicos, o las fuentes de radiación artificial, como las producidas por máquinas de radio-diagnóstico.

En esta tesis, se propone varias estrategias para mejorar la fiabilidad de los sistemas críticos que operan en presencia de radiación ionizante, tanto en el espacio, como a nivel terrestre. En este contexto, la radiación ionizante puede alterar la salida de un sistema digital creando interferencias, fallos y alteraciones permanentes en los circuitos, entre otras muchas incidencias. Como resultado, los sistemas críticos pueden comportarse de manera inesperada, produciendo resultados erróneos o entrando en estados no operativos, que requieran un mecanismo externo para recuperar un funcionamiento correcto (*watchdogs*, interrupciones temporizadas). En la bibliografía, se encuentra que las técnicas basadas en redundancia, aplicables tanto a hardware como a software, son las soluciones más efectivas para detectar y mitigar este tipo de comportamiento inesperado. Estas técnicas basadas en redundancia, presentan una alta variabilidad, dado que puede aplicarse a estructuras de diferente complejidad. En el caso de redundancia software, se puede utilizar a nivel de instrucciones de ensamblador, accesos de memoria, funciones o métodos, incluso a nivel de procesos o hilos. Es importante destacar que, la variabilidad de las diferentes técnicas de mitigación de fallos, provoca una alta complejidad de la predicción del efecto de estas técnicas en el conjunto del sistema. De manera paradóji-

ca, la aplicación de estas técnicas de endurecimiento a algunos elementos de un sistema, en ocasiones, puede dar lugar a un aumento de la susceptibilidad del sistema a fallos inducidos por radiación, por consiguiente, a una reducción significativa de la fiabilidad. Esta paradoja, es debida, al aumento de los sobrecostes en los recursos utilizados, o al incremento computacional de dicha técnica de endurecimiento.

De manera general, con objeto de reducir la susceptibilidad a fallos inducidos por la radiación, en diferentes sistemas críticos, en esta tesis, se pretende mejorar la fiabilidad, adaptando o proporcionando nuevas técnicas y herramientas para el endurecimiento software en microprocesadores de última generación. Para ello, se han desarrollo dos técnicas, la primera se centra en la búsqueda automática de soluciones maximizando la fiabilidad; la segunda técnica desarrollada, consiste en un endurecimiento software basado en redundancia, optimizado para obtener un mayor rendimiento computacional. Además, se ha desarrollado un nuevo modelo matemático semi-empírico, para evaluar y predecir los fallos inducidos por radiación.

La primera de las técnicas desarrolladas, explora, de manera eficiente, soluciones que maximicen la fiabilidad, buscando optimizaciones y endurecimientos que aumenten el rendimiento del sistema, reduzcan los recursos utilizados y, al mismo tiempo, aumenten la cobertura frente a fallos. Con objeto de optimizar el rendimiento y el uso de recursos de las aplicaciones y circuitos, en esta tesis, se utilizan técnicas de aprendizaje automático y algoritmos de búsqueda meta-heurísticos, inspirados en los sistemas naturales (algoritmos genéticos), optimizados con técnicas de optimización multiobjetivo basadas en el concepto de eficiencia de Pareto. Este algoritmo de búsqueda optimizado, permite mejorar al mismo tiempo el rendimiento del sistema, el uso de los recursos y la cobertura de fallos, ya que es capaz de explorar un espacio de soluciones multidimensional de manera eficiente. La aplicación de este algoritmo, permite alterar la generación de las aplicaciones, por parte del compilador, logrando obtener aplicaciones más fiables de forma no intrusiva, es decir, sin necesidad de modificar el código.

La segunda de las técnicas desarrolladas, propone una mejora de las técnicas de endurecimiento clásicas, empleando esquemas paralelos basados en el multiprocesamiento simétrico y asimétrico (SMP y AMP, respectivamente). Con objeto de lograr un incremento en la fiabilidad, se han utilizado sistemas mononúcleo y multinúcleo, a los que se les ha eliminado la necesidad de un sistema operativo, con el fin de reducir los sobrecostes de recursos y aumentar el rendimiento, manteniendo la cobertura frente a fallos.

Por último, se ha desarrollado un modelo semi-empírico que permite la evaluación y selección de las configuraciones más fiables. Además, el modelo también permite realizar un endurecimiento selectivo de los recursos críticos, antes de acometer una campaña de radiación acelerada. En el desarrollo del modelo, se emplea tanto datos históricos de campañas de radiación real, como los resultados de las campañas de inyección simulada de las aplicaciones. Así, el modelo es capaz de realizar una predicción temprana de la fiabilidad de las nuevas soluciones, antes de probarlas bajo radiación real en aceleradores de

partículas. Para verificar el modelo, se realizaron pruebas de radiación acelerada de varias soluciones, desarrolladas en esta tesis, empleando protones en el Centro Nacional de Aceleradores (CNA) de Sevilla, y empleando neutrones en Los Álamos National Laboratory (LANL - USA).

Abstract

The increasing demand for computing capabilities in critical systems operating in hostile environments, and therefore subject to failure, is closely linked to the increasing use of computing solutions that exploit the new features of state-of-the-art microprocessors: an increasing number of cores, better computing performance, and lower power consumption. However, the photolitography manufacturing process that enables these improvements is progressively reducing the tolerance of microprocessors to radiation faults, known as Single Event Effects (SEE). Among the sources that generate anomalous behaviour, it can be found natural sources such as cosmic rays, or artificial sources such as those produced by radiodiagnostic machines.

In this thesis, several strategies are proposed to improve the reliability of critical systems operating, both at space and ground level, in the presence of ionizing radiation. In this context, ionizing radiation can alter the output of a digital system creating interference, failures, and permanent alterations in the circuits, among many other incidences. As a result, critical systems may behave in an unexpected way producing erroneous results or entering in non-operational states. As a result, these systems can require an external mechanism to recover a nominal operation (watchdogs, timed interruptions). The literature shows that techniques based on redundancy, applicable to both hardware and software, are the most effective solutions to detect and mitigate this type of unexpected behavior. These techniques, based on redundancy, present a high variability since they can be applied to structures of different complexity. In the case of software redundancy, it can be applied at assembler instruction level, memory accesses, functions, or methods, even at process or thread level. It is essential to highlight that the variability of the different fault mitigation techniques, causes a significant increase in the complexity to predict the effect of these techniques on the whole system. Paradoxically, the application of these hardening techniques to specific elements of a system can, on occasion, lead to an increase in the susceptibility of the system to radiation-induced faults and, consequently, to a significant reduction in reliability. This paradox is due to the increase in the overheads in the resources used, or to the computational increase of such hardening techniques.

This thesis aims to improve reliability by adapting or providing new techniques for software hardening on state-of-the-art microprocessors, reducing

Contents

Part I

Thesis synthesis

Chapter 1

Soft error mitigation on embedded critical systems exposed to radiation

This first chapter presents an overview of the thesis research entitled: "**Fault tolerance in critical aerospace embedded systems: Multi-threaded mitigation, non-intrusive compiler-guided hardening, and early prediction of proton and neutron induced soft errors**". This chapter is organized as follows: section 1.1 shows the description of the problem; section 1.2 presents the terminology used and describes the problem in depth; section 1.3 shows several approaches presented in the literature to overcome the problem exposed; section 1.4 and section 1.5 present the thesis research objectives and the hypothesis, respectively; the description of the methodology used during this thesis research is in section 1.6; the list of scientific articles published in high impact journals, as well as the communications to national and international conferences are in section 1.7; finally, the results obtained during this research are in section 1.8.

1.1 Motivation and problem definition

The increasing demand in computer processing has led microprocessor manufacturers to evolve from single-core processors in the 1990s, through parallel processors (vector and matrix computation) in the 2010s, to the new emerging trends in quantum computing [1], [2]. In this context, advances in recent decades have achieved performance improvements in computer processing, focusing mainly on three strategies. First, increasing the frequency at which the microprocessors operate, achieving an increase in processing speed. However, overheating of the microprocessors limited further implementation of this strategy. Next, increasing the number of processing units per chip, improving the computational performance. However, power consumption and over-

heating again limited this strategy. Finally, recent advances in semiconductor manufacturing processes have led to significant reductions in manufacturing technology over the last decade. This technique has allowed the development of functional microprocessors, using designs with a 7 nm photolithography process. This reduction in the manufacturing process allows higher rates of component integration, reduction of power levels, improving the problem of system overheating. As a result, a higher number of structures are tightly integrated into a single chip [3]. However, this strategy is close to reaching the limit due to physical constraints that cause a decrease in the microprocessor's reliability. In this sense, the latest evaluations of microprocessor reliability, presented in the literature, show significant reductions in the tolerable noise margins by the circuits, causing greater susceptibility to environmental [4], [5]. Indeed, there is a growing trend to mitigate the faults of microprocessors exposed to radiation. Several of these radiation sources are common in everyday life, such as artificial sources of radiation present in medical diagnostic equipment, or natural radiation from cosmic rays. One of the effects of radiation is altering the functionality of new microprocessor designs, making them less reliable, either degrading or causing unexpected behavior. The most important radiation effects are the so-called Single-event effects (SEE) [6]–[8]. These effects are particularly relevant for critical systems, where unexpected or inoperative states are not acceptable. In this sense, autonomous essential decision systems, such as those integrated into aircraft, must provide correct behavior even in the presence of faults, such as those produced by cosmic radiation [9]–[11]. Consequently, critical systems must implement various countermeasures to detect when a fault has occurred and provided ways to mitigate or correct it. The procedures for implementing these design modifications are called hardening techniques, which attempt to achieve reliable computing in an unreliable design.

In this context, the industry increasingly demands reliable, low-cost, and high-performance solutions. However, reliability is a concept that is usually opposed to objectives such as cost and computational performance. An example of solutions with high reliability is the "`Rad-Hard`" custom processors, such as the `LEON-FT`, which are hardened solutions with high production costs. In this context, a new research trend is to enable reliable computation on state-of-the-art *Comercial-Of-The-Shelf* (`COTS`) devices, due to their higher computational performance, and reduced costs. The difficulty of this strategy lies in reducing the performance penalty associated with hardening techniques while increasing reliability. In this sense, fault mitigation techniques can be applied at different software levels (assembler instructions, memory access, methods or functions, processes or threads), or hardware resources (single or multiple cores). Therefore, this diversity allows a significant variability of hardened solutions, resulting in solutions with different degrees of reliability. The objective of this thesis is to find methods, models, and solutions that optimize the of systems reliability, using hardware or software hardening techniques.

1.2 Introduction

This doctoral thesis is part of the research project: *"Early evaluation of radiation effects by simulation and virtualization. Mitigation strategies in processor advanced architectures"* (Ref. ESP2015-68245-C4-3-P, MINECO/FEDER, UE), aiming to enable reliable computation in COTS components.

This thesis's main objective is to provide reliable computing against radiation-induced faults in electronic devices, known as SEE (*Single Event Effects*), and more specifically, the mitigation of effects considered as transient, on these devices. In this sense, the thesis analyzes the behavior of COTS microprocessors under the effects of ionizing radiation. For this purpose, multiple programs and hardware configurations are used, focused on identifying the origin of the faults, increasing the system's reliability through the so-called hardening techniques. Although hardening techniques are aimed at obtaining reliable computing, their widespread use is unfeasible in most cases, due to the high cost in resources or computing performance. Thus, it is necessary to develop partial and optimized hardening techniques, which reduce the cost and, at the same time, preserve the fault tolerance of the solutions. This last point is another main objective of the thesis.

The remaining section is devoted to introducing the main concerns of the doctoral work: section 1.2.1, shows the taxonomy of the single event effects (SEE), and their effects on a system; section 1.2.2, presents the different routines to control a radiation event; section 1.2.3, identifies the most common methods to estimate the reliability of the solution.

1.2.1 Single-event effects (SEE)

Most critical systems operate in an environment with a higher or lower rate of radiation due to human activity or natural sources. Until now, the old critical systems were not very sensitive to the effects of radiation. However, in order to meet the new performance requirements of the industry, there has been a reduction in the size of the components, increasing the susceptibility to radiation. In particular, the literature shows how cosmic radiation can interfere with the atmosphere, generating particles such as heavy ions, neutrons, and protons, which interact with electronic components, causing undesirable effects [12], [13]. These particles involve one-tenth of the natural radiation observed, at sea level, causing the malfunctioning of the devices (see Figure 1.1). This radiation can induce a parasitic charge on the transistors by the passage of high-energy particles through them, or by proton/neutron collisions with the transistor substrate. A recurring case, to illustrate this behavior, are the errors produced in a 256 Mb SRAM memory, being able to register a fault every three weeks, due to the radiation at sea level.

As described, cosmic rays can cause ionization in circuits, causing components to change their conductivity, producing alterations that generate erroneous outcomes, including the destruction of the affected component [14], [15]. According to the severity of the damage (see Figure1.2), the effects are

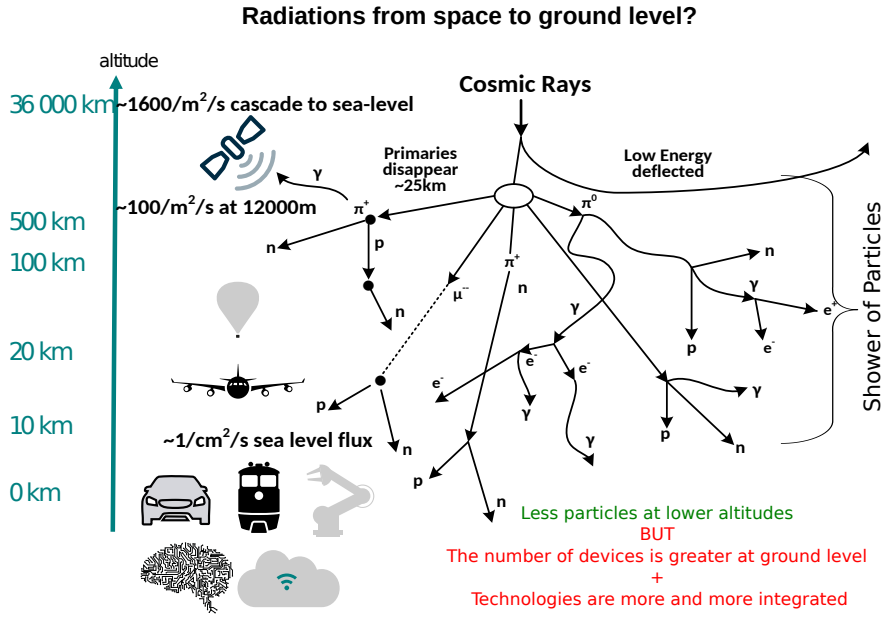


Figure 1.1: Effects of cosmic radiation and particle flux observed at different altitudes. The particle flux decreases with decreasing altitude, while the number of susceptible devices increases.

classified as soft errors or hard errors. Depending on the persistence of the disturbances, they can also be classified as transient or permanent.

The faults known as hard error, involve the destruction of part of the circuit permanently, due to an overload or breakage of the crystalline substrate. The different types of hard error that exist are:

- Single Event Latch-up (SEL)[16] are caused by a shortcut, in a parasitic PNPN structure, of the *Complementary Metal-Oxide-Semiconductor* (CMOS) circuits, when high energy particles (typically silicon dioxide SiO_2) pass through the substrates. This event is usually detected by a malfunction of the system or excessive consumption of energy, caused by the shortcut. Thus, it is encouraged to avoid the use of elements sensitive to this type of event in critical designs

- Single Event Burnout (SEB) [17] occurs when a highly ionizing particle, transmits energy to the depletion region of the transistor. As a result, the transistor acquires a voltage, higher than the breakdown voltage of the parasitic structures, creating, as a consequence, a parasitic transistor, which amplifies the current entering the transistor. An example of this phenomenon can be seen in an NMOS transistor, where electrons are injected into the collector, generating a feedback loop, causing local overheating that destroys the transistor.

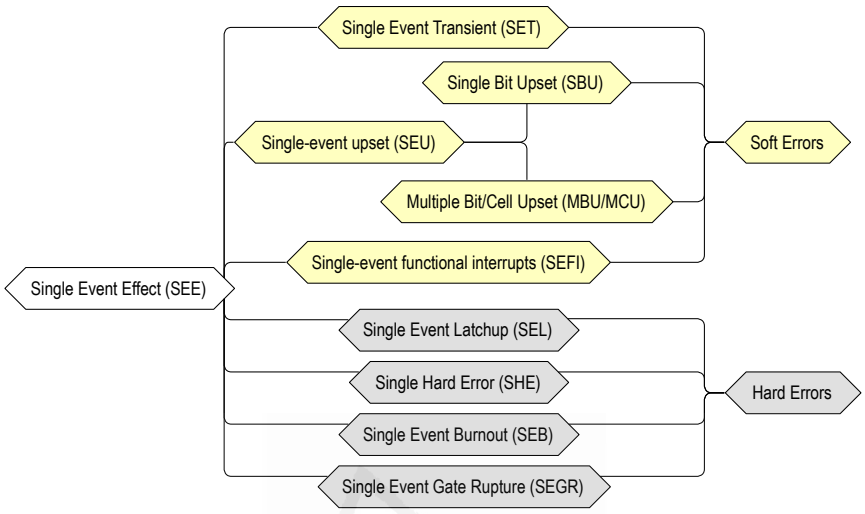


Figure 1.2: Single Event Effects Classification (SEE): Transient Faults (yellow), Permanent Faults (grey).

- Single Event Gate Rupture (SEGR)[18] occurs when high energy ions hit an active gate. As a result, the silicon dioxide insulation layer is damaged by overvoltage and overheating, generating an undesirable partial connection.
- Single-Event Hard Error (SEHE)[19] is produced by an alteration in the gate that controls the transistor, causing a permanent change in the operation of the transistor. SEHE affects all types of memory where some bit is stacked, and it is impossible to invert the stored value.

On the other hand, the faults known as soft errors, show a non-destructive behavior, characterized by a change in the logical state of a node (bit-flip) in any component. Usually, they are memory cells or configuration registers that change the internal state of their bits, from 1 to 0 or *vice versa*. In most cases, these faults can go unnoticed if they affect the resources not used by the system. In those cases, the system recovers itself, without any intervention. An example of this type of fault occurs when a bit-flip changes a resource not used by a program. As a consequence, the program terminates with the expected outcome without any performance penalty. However, supposing that the fault affects an active (in-use) resource. In that case, the effects may differ slightly, depending on the criticality of the resource. In this last case, the faults follow the next classification:

- Single-event functional interrupt (SEFI)[20], occurs when the device reaches an unknown state, due to an alteration in the configuration bits. This effect is detected by the loss of some functionality of the device. A possible

solution to this problem is to perform a system reboot (*soft reset*). However, there are cases where the *soft reset* may not completely recover the system, making it necessary to drain the power (*hard reset*). An example of these cases is observed when a real-time system operated by interruptions suffers a "*bit-flip*", masking the activation of the interruptions, entering the device in an inoperative state, as the configuration of the platform is compromised. In this case, a simple solution to restore the default configuration is to activate the *watchdog* subsystem that triggers a *hard reset*.

- Single Event Transient (SET)[21] is produced by the discharge of an ionized circuit node that introduces a pulse into the system. In this case, the fault can propagate to other components, becoming a more severe fault (see Figure 1.3).
- Single Even-Upset (SEU) [22] is produced by changing the saved state (*bit-flip*) in a storage unit (see Figure 1.3). Depending on the number of affected bits they are called as:
  - Single Bit-Upset (SBU)[23], if only one bit is changed.
  - Multiple Bit-Upset (MBU)[7], [15], [24], in case of affecting, in devices with high sensitivity, multiple adjacent bits.

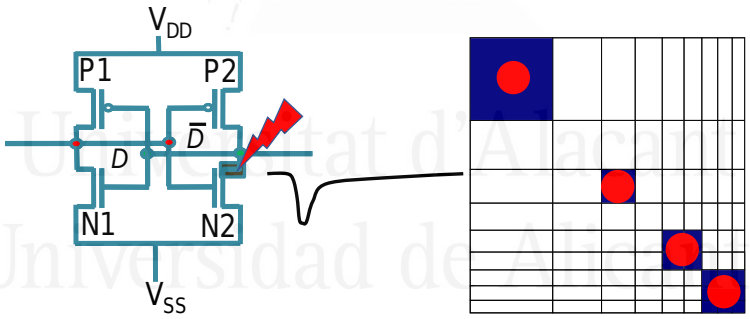
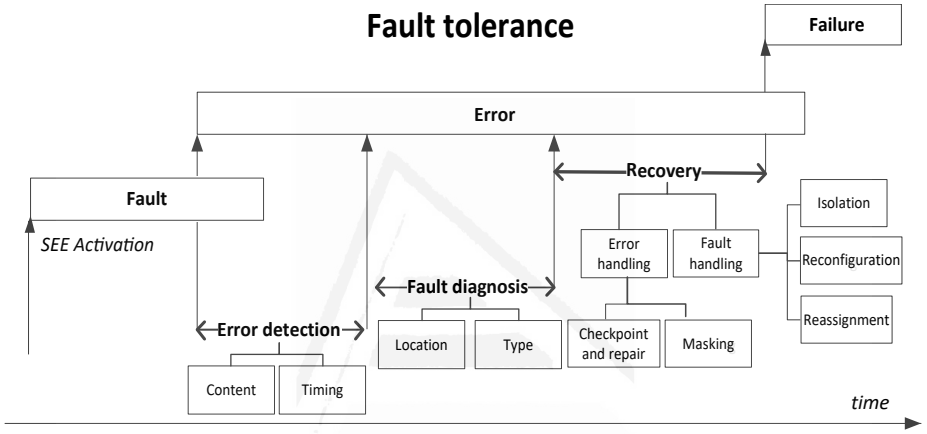


Figure 1.3: Example of soft error when a particle hits the *N2* transistor of the SRAM cell. It is called SET, if the fault is transmitted as a pulse. It is called SEU, if the fault alters the cell contents. Also, the increased probability of suffering an MBU, due to the reduction of manufacturing technology, is schematized.

1.2.2 Fault tolerance

By definition, a critical system must provide correct behavior in every environment, even in hostile ones. In particular, in environments where any of the effects mentioned above can degrade its functionality. In order to avoid catastrophic situations, developers include recovery functions in their designs, generating a fault-tolerant system. The purpose of these recovery functions is

to monitor the correction of the system and to activate the support routines, as soon as a fault occurs, to correct it, avoiding its propagation. Early detection and contention of errors are essential since a long detection and recovery time can increase the severity of the fault, or even propagate it to other system components. For a fault-tolerant system to be effective, error detection and correction routines must be low-intrusive. Understanding by intrusiveness, the new elements added to obtain a reliable computation. Also, they must have high efficiency and low complexity, which ensures the system's progression, avoiding the blocking of the task under protection.



  - Checking and searching for defective system components.

- **Recovery** restoring normal system operation by correction. This correction is made by
  - Error handling, are corrected by restoring the mirror consensus (*Masking*) or returning to a previously saved error-free state (*Checkpoint and repair*).
  - Fault handling, involving isolation of the faulty component (*isolation*) or reconfiguration or reassignment of system resources to avoid it (*reconfiguration* and *reassignment*).

In addition to the classification described above, support routines or hardening techniques can also be classified according to the level to which they apply: hardware, software, or hybrid. Regardless of the classification, the support routines have high versatility, allowing them to be applied to different components and levels of the system to be protected. For instance, the routines of content detection or recovery can be applied both in hardware and software. When applied at a hardware level, the replication is done over several logic gates and soft-cores. In contrast, at a software level, it is done by replicating assembler instructions, variables, and instruction flows.

1.2.3 Accelerated radiation test and hardening validation

As described above, hardening techniques introduce mechanisms to detect and correct faults affecting system components. However, to be considered reliable or immune to radiation (*RadHard*), a component must pass extensive tests to verify its behavior under radiation [25]. Usually, these radiation tests follow a strict standard to ensure correctness and statistical relevance, allowing comparisons between different measurements and tests performed. The European standards of mandatory compliance are:

- ESCC Basic Specification 22500 (displacement damage)
- ESCC Basic Specification 22900 (total ionizing dose – TID)
- ESCC Basic Specification 23100 (evaluation and procurement)
- ESCC Basic Specification 25100 (single event effects – SEE)

It is common to use facilities where access is limited to a few days to perform this assessment, often involving high costs to perform the tests mentioned above [26], [27]. The radiation used is obtained from particle accelerators and other radiation sources such as cobalt, which estimate the system's behavior during the operating phase (lifetime). In the development of this thesis, the particle accelerators known as cyclotron and tandem, belonging to the Centro Nacional de Aceleradores – CNA, have been used, which allow testing of protons and neutrons, respectively. Likewise, to carry out additional tests using

neutrons, the Weapons Neutron Research – WNR at Los Alamos Neutron Laboratory – LANL, which presents a similar neutron spectrum to that produced in the atmosphere by cosmic rays, was used (Figure 1.5). In this context, an

Figure 1.5: Experimental setup using proton and neutron irradiation test.

exhaustive evaluation of all possible solutions in radiation campaigns is not feasible due to the considerable variability in the solutions of hardening techniques. Thus, it is essential to select a representative set of solutions that will maximize the useful information obtained from the accelerated radiation test. Because of this restriction, it is essential to rely on faster, but less accurate tests to select the best candidates for irradiation.

Injection campaigns with simulators emerge as a low-cost alternative during the early stages of evaluation. These tests usually provide an upper limit to the fault events observable in subsequent radiation tests. Simulations perform the reliability tests, approximating the effects of radiation through the use of models. An example of a fault injection model, in simulation, is the so-called bit-flip. This model simulates the effects of SEUs, affecting a memory cell, changing its contents. However, the simulation results depend on the precision of both the fault model and the microprocessor model to provide reliable predictions. In some cases, the simulated results may differ slightly compared to the results of radiation experiments, because the microprocessor simulation does not take into account the different sensitivities of the components to radiation.

1.2.4 Thesis scope

The work developed in this thesis is focused on improving the reliability of classical hardening techniques. These techniques use *n-modular* replication, both software, and hardware, to detect and correct radiation-induced faults. These techniques introduce high-cost overheads in terms of resources or computational performance, even affecting the reliability of the solution. Therefore, this thesis focuses on improving performance and reducing the number of resources exposed to radiation, while maintaining fault coverage of the techniques.

It is necessary to reduce the replicas' size, decreasing their accuracy to reduce the size of the *n-modular* solutions using hardware approaches. These

reductions can be achieved by using partial computing, also known as approximate computing. This technique provides highly reliable hardware with lower resource consumption. In the case of software solutions, the increase in performance is obtained by taking advantage of the unused resources in the system to perform the replica's computation. In this case, different instruction flows are used in multi-core systems to compute the replicas simultaneously. In both cases, the reduction of temporal and spatial overheads reduces the probability of a radiation-induced event and increases reliability.

Furthermore, this thesis presents tools and methods for the validation of the developed hardening approaches. In this sense, simulated injection tests and accelerated radiation tests have been used to evaluate the proposed solutions, thus verifying the increase in reliability. Furthermore, the results of both campaigns have been combined to develop a mathematical model that allows greater accuracy in predicting future reliability measures, with the capacity to determine the resources that most influence the reliability of the solution.

1.3 Related Work

The literature shows numerous solutions for correcting or detecting faults in microprocessors, microcontrollers, circuits, FPGA soft-cores, and solutions aimed at custom products. As already mentioned, this thesis focuses on the development and validation of several hardening techniques, aimed at COTS devices. This choice limits the set of hardening techniques, reducing it to those techniques that do not require the development of specific hardware, being reduced to the set shown below:

- Modification of circuits or soft-cores programmed in FPGAs.
- Modification of software aimed at increasing reliability.
- Hybrid techniques that reconfigure the hardware to detect and correct faults.

1.3.1 Hardware hardening techniques

Hardware protection approaches, gain reliability by introducing circuit-level redundancy [28], [29]. Redundancy can be obtained by applying both information theory techniques [30] and component replication. Consequently, it is possible to protect and ensure the correctness of the circuit functionality.

Protection techniques based on information theory use a function that describes the data to be protected. These functions usually present high performance, characterized by low spatial overhead. Three examples of the functions described above are *Decimal Matrix Codes* (DMC) [31], *Error Detection And Correction* (EDAC) [32], [33] and *Error Correction Codes* (ECC) [30]. The application of these techniques reduces the resources available for storage by dedicating part of them to maintaining the data in the summary functions. An example of a summary function is Hamming error detection and correction codes (see Figure 1.6). This summary function achieves its protection capabilities by adding parity bits, mixed with the data to be protected. As a result, the solution can detect faults affecting a few bits and correct them in some cases. However, the increase of the exposed surface and the reduced correction capacity increases the probability of suffering radiation-induced event. In this sense, the latest trends in this field seek to reduce storage overheads while increasing redundancy and correction capabilities.

The techniques based on hardware replication, such as the *n-modular* technique, are applied to different hardware structures or components, such as logic gates, complete components (memories), and processing units (*lockstep* microprocessors or complete systems). These techniques aim to develop specific and expensive components, presenting an elevated use of resources and energy consumption. In a recent publication [25], the authors show the evaluation of a *Triple-Lock-Step* (TLS) processor, comparing it with other hardened hardware solutions *Dual-Lock-Step* (DLS) processors. Also, they analyze the differences in the solution with *RadHard* processors. The authors conclude that

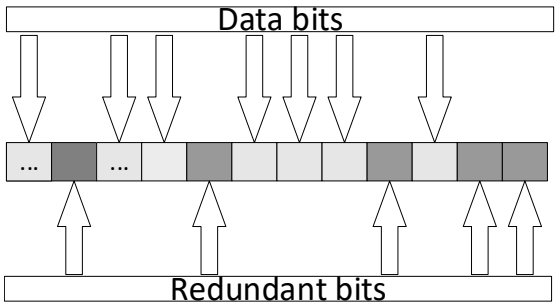


Figure 1.6: Schematic of a Hamming detector and corrector code. Redundant bits are represented in dark grey, while protected data bits are represented in light grey. Redundant bits are associated with positions where their binary representation has only one active bit. The parity function protects the data bits, using the redundant bits, associated with the binary decomposition of their position.

TLS processor has a performance increase of about 1000× compared to its DLS analog. Additionally, the authors analyze the DLS processor, comparing it to its non-redundant (non-hardened) version. The authors found significant increases in resource and energy use. Also, decreases in performance and increases in power consumption of hardened processors (*RadHard*) are shown.

The latest research trends in hardware redundancy, focus on reducing the over costs associated with this type of solutions while preserving reliability levels. An example of a highly reliable technique is *Triple Modular Redundancy* (TMR). However, its implementation involves a significant increase in resources. Because of this, there are occasions when it is not feasible to replicate all or most of the solution's components to be protected [34]. Despite the high cost of TMR solutions, some systems consider adopting this technique, because accuracy in them is not a critical feature. In this way, by simplifying or approximating replicas' calculation, it is possible to reduce the overheads inherent in TMR techniques. In this sense, the use of the approximate computation [35]–[37] applied to TMR techniques causes a reduction of the computational effort, as well as of the resources, maintaining high reliability [38]–[40]. In Figure 1.7, it is shown how the approximation of a circuit is made by reducing the number of logic gates. The new circuit continues to operate correctly in most cases. However, there is one case where the result of the approximated circuit differs from the original circuit. Assuming that this circuit replaces one of the replicas of a TMR, it will still provide the expected behavior with a reduced logic. This new approximate TMR, presents a reduced temporal vulnerability, where the expected result of the TMR could be affected.

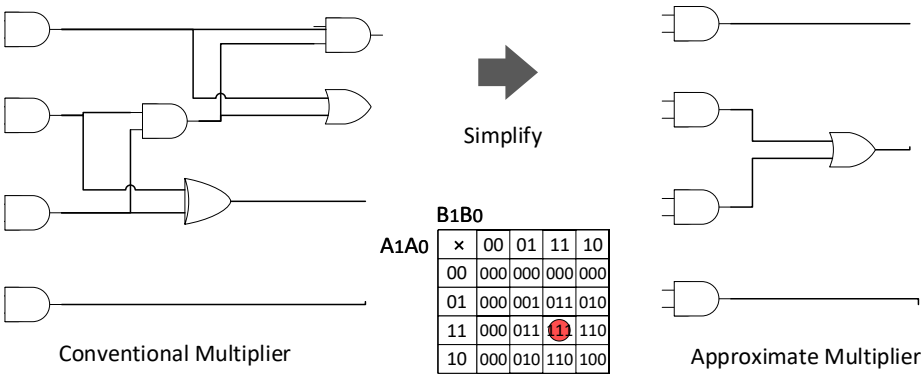


Figure 1.7: Conversion by approximate computation of a conventional multiplier to an approximate one, reducing the number of logic gates. The approximate multiplier decreases the area used and increases the throughput. However, the simplified circuit presents a case where the output is wrong, and the table highlights the only such case.

1.3.2 Software hardening techniques

In order to mitigate the effects of radiation and avoiding costly hardware modifications, software-hardening approaches emerged. These techniques can quickly replicate different software structures to achieve resilience. In this sense, a hardening technique based on replication can make use of structures, simple, with a single instruction, or complex, such as procedures or even complete programs, to build the replicas [41], [42]. The modifications to the code, which these techniques make, make them intrusive. Within this set of techniques, one of the most remarkable is *Software-Implemented Hardware Fault Tolerance* (`SIHFT`) technique. Also, there is another set of techniques, based on software, which is capable of increasing reliability, without the need to introduce new elements into the solution. To this end, the techniques optimize the software, increasing reliability, during the application generation phase, by modifying the mapping of resources, task planning, and thus reducing the radiation exposure of the different program resources. One way to achieve this goal is to guide the generation and optimization phase of the compilers [43], [44]. This set of techniques, as they do not make use of modifications within the code, are known as non-intrusive.

Within the intrusive set, the literature shows several approaches that try to increase the system's reliability, introducing redundancy based on *n-modular* [29], [45]. Two of the most widely used applications of this technique for detecting and correcting radiation-induced faults are *Triple Modular Redundancy* (`TMR`) and *Duplication with Comparison and Re-Execution* (`DWC-R`) [46]. The hardening of the software using this methodology is characterized by being easily exportable to different hardware architectures and presenting low development and verification times. However, these approaches tend to present a

high consumption in terms of resources (spatial replication), as well as a low computational performance (temporal replication). In recent publications [47]–[49], the authors test and evaluate an approach that reduces both overheads by selectively enabling unused processor resources to host TMR replicas.

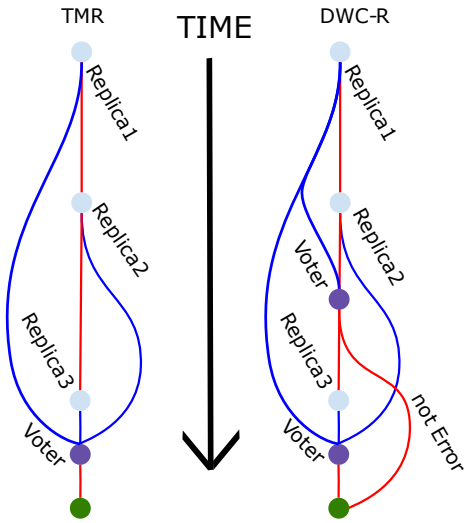


Figure 1.8: The functioning of *n-modular* techniques: TMR and DWC−R The instruction flow is indicated in red, while the data flow is indicated in blue. The TMR shows the execution of the three software replicas under protection for later verification in the voter. In the case of the DWC−R, two of the three replicas are computed when no error is detected in the first voter. In case of error, the technique behaves similarly to the TMR technique.

The *n-version* techniques are an extension of the *n-modular* techniques, which protect the program using different resources in the system, to detect faults coupled to a specific configuration. This technique improves *n-modular* techniques by using different hardware components, adding diversity. This modification ensures the detection of the defective component, avoiding compromising the results of all the replicas. An example of *n-version* applied to a multiplication consists of modifying the calculation of the replicas by using different algorithms. One of these algorithms (replica) can perform multiplication by successive additions. The other replica can make use of a digital signal processor (DSP) to perform the calculation. In this context, running both replicas using different procedures ensures that the faulty component is not used in both replicas. Furthermore, the execution time differs between them, avoiding any running conditions in the algorithm that reproduce the replicas fault.

Another way to achieve software resilience is through control flow checking. This technique helps to avoid unexpected program behavior by taking advantage of the instruction flow information defined during the program design phase. The information obtained from the instructions flow is used to de-

tect abnormal bifurcations using annotations [50], [51]. These annotations label each of the different processing blocks of a program, which can be formed by multiple instruction flows. The annotations are used at the beginning of each block to verify the signature (annotation) of the preceding calculation block. In case an annotation does not match any of the expected annotations (preceding blocks), anomalous behavior is detected. However, these techniques based on control flow, present reductions in performance, due to the constant verification made within each block. They also present an increase in the use of resources due to the storage needed to maintain the annotations. To clarify the behavior of this technique, Figure 1.9, shows an example where a program with six blocks and five instruction flows is used. Considering the block S4.2, an initial verification task, previous to processing the block, analyzes if the computation made by the algorithm is correct. This way, before processing the block, it is verified that one of the preceding blocks (S3.2 or S3.4) has been processed. If this restriction is not met, because the previous block is S2.3, an error is detected in the program flow. This technique is highly intrusive, besides presenting a high consumption of resources. Some recent research works present improvements in this technique's performance, using *assertions* to detect errors that affect the program flow, reducing the number of resources used, and increasing the performance [52].

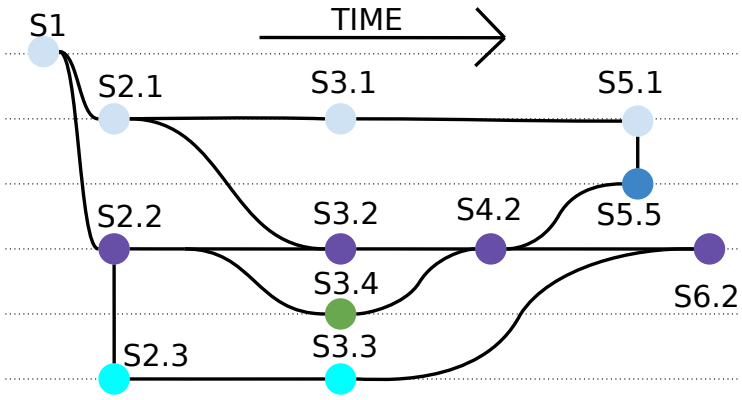


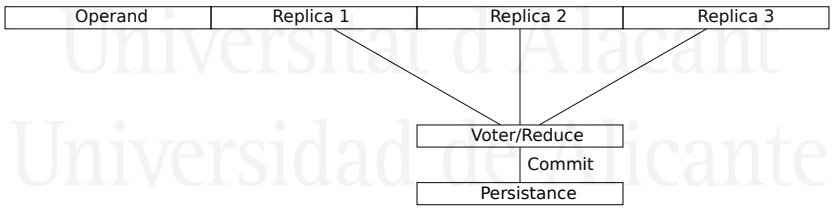
Figure 1.9: Representation of the hardening of the instruction flow using annotations of the program blocks. Each annotation uniquely defines a processing block, taking into account the different blocks and instruction flows that compose the program.

Within the SIHFT techniques, there is a trend that tries to reduce the over costs present in *n-modular* techniques [53], [54]. This trend takes advantage of resources and structures, not used in most state-of-the-art processors. These investigations are focused on achieving an increase in performance, as well as obtaining different versions with different degrees of reliability (diversity). Another trend is investigating the improvement in the maintenance and de-

bugging of the code generated by the redundancy techniques. Consequently, new researches are focused on reducing the intrusiveness of hardening techniques and increasing the code's quality. To this end, these investigations seek to improve reliability by using automatic hardening based on annotations or substitutions with low intrusiveness.

1.3.3 Hybrid hardening techniques

The proposals of the hybrid approaches aim to overcome the shortcomings of the software-only and hardware-only strategies, getting the benefits of both. These approaches offer solutions with high performance, similar to the hardware approaches, and fast and flexible development, similar to the software approaches. However, these techniques are not able to offer completely fault-free solutions. Recent research shows how a combination of software and hardware fault mitigation techniques can improve the reliability of a system. This technique improves the software, by using `SIHFT` techniques, and the hardware, by selectively applying redundancy to a *soft-core* synthesized in an `FPGA` [55]. A recent trend is to improve the reliability of a system by reusing part of the existing resources in a way that was not intended. An example of this type of technique is the reconversion of vector type processing units, present in most of the latest generation processors. These processing units are reconverted to speed up replicas' computation in *n-version* techniques (Figure 1.10). Three examples of this technique, which allows parallel processing of replicas, are found in: `VLIW` microprocessor [56], `ARM NEON` vector units [57], and the `AVX` and `SSE` instruction set present in most `x86` systems.



1.3.4 Models and statistical reliability analysis

Fault mitigation techniques have to be tested using expensive and extensive testing on the particle accelerators to ensure that they present a correct behavior in the presence of faults [26], [27]. These expensive accelerated radiation campaigns provide a statistical analysis of the device's behavior over its lifetime. However, these campaigns are blind and do not provide the cause of the fault. As mentioned above, hardening techniques offer a variety of solutions with varying degrees of reliability. Therefore, it is essential to select a set of solutions, which is representative, that maximizes the information obtained from the radiation campaigns, avoiding low performance or defective versions.

To analyze reliability at an early stage, simulated or emulated fault injection campaigns arise as a way of estimating and approximating the behavior of a solution. For this purpose, this type of campaign makes use of `RTL` functional simulations, cycle or logic level accurate simulations, and emulated simulations. Although very accurate, `RTL` functional simulations show degraded performance due to a large number of variables simulated and the high level of detail in the models. Due to this problem, solution evaluation is limited, and a deep reliability exploration is not feasible. Instructional level models or functional level simulations instead show high performance, because the simulations use simplifications in the models, causing a degradation in the accuracy of the results. On the other hand, fault injections using real hardware (emulated injections), show modifications in the solution that are not present in a production environment, compromising the accuracy of the measures. An example of these additional elements is the debugging infrastructures that can be reconfigured to perform the injections.

As mentioned above, the microprocessor models and fault models, used in the fault injection campaigns, approximate the behavior of the actual hardware. However, the resulting loss of accuracy due to the use of a model with improved performance may limit the reliability analysis. In these cases, more complex models, such as the *Architectural Vulnerability Factor* (`AVF`), help obtain more accurate estimations of the solution's behavior under radiation [58]. These models use early reliability results and historical data obtained from accelerated radiation campaigns to improve the estimation of early reliability tests.

1.4 Objectives

The progressive miniaturization of the manufacturing process makes microprocessors more and more sensitive to ionizing radiation and less and less reliable. Thus, it is necessary to implement routines, techniques, and methods that increase reliability, allowing for error-free computing. In this sense, this thesis's main objective is to mitigate and improve the tolerance to faults induced by radiation in different microprocessor-based systems. This thesis aims to improve reliability by, adapting, or providing new SIHFT techniques in last-generation microprocessors (COTS). To this end, resources involving different instruction flows (threads or processes) executed in one or more cores of a microprocessor, have been used. Due to this, this thesis develops and evaluates new parallel schemes with low intrusion, being this last feature mandatory to avoid increasing the sensitivity of the solution to faults. This thesis also aims to develop a model for predicting the sensitivity to radiation of an application. Therefore, the model can help to evaluate both the improvements and the effectiveness of the techniques developed. Furthermore, the model developed will select the best candidates to be evaluated using radiation tests in accelerated radiation facilities, where the user is only allowed to evaluate a limited set of solutions. Therefore, the model needs to make accurate predictions of the behavior of solutions under radiation. In this respect, a comparative study between the data of the model and the radiation experiments' data will be necessary to correct and improve the predictions of the model.

In order to meet the general objectives, the following objectives and subobjectives are established:

O1: To review the most advanced fault tolerance solutions, techniques, and metrics to evaluate the performance, resource utilization, fault coverage, and reliability of electronic devices subjected to ionizing radiation. Focusing on the following points:

1. Study the basic notions of reliability in systems exposed to ionizing radiation
2. Study the most common fault tolerance techniques in software, hardware, and hybrid systems
3. Identify the most relevant reliability metrics (AVF, MWTF), to analyze the proposals of state-of-the-art and the proposals presented in this thesis.

O2: Evaluate the suitability and feasibility of state-of-the-art microprocessor simulators for performing fault injection into complex programs (multiple threads, multiple processes or operating systems). For this purpose will:

1. Design or modify a fault injection tool.

2 Evaluate different relevant architectures and their associated simulators, giving special attention to the following architectures:
   - **ARM** 32bit multi-core instruction-accurate simulators: Xilinx-QEMU, Gem-V, OVPsim, Simics, fast-models
   - **MSP430** 16bit microcontroller cycle-accurate simulator: SHE, NAKEN

3 Evaluate different *High-level Description Languages* (HDL) and their associated simulators, taking into account the following alternative:
   - ***Verilog***: ABC simulator and the *nandgate* 45nm gate library.

O3: Develop an automatic tool to explore a large space of hardening solutions effectively, with the ability to:

1 Use a meta-heuristic algorithm, based on machine learning, to perform a blind and unguided search, in a huge non-linear multidimensional solution.

2 Optimize the search algorithm to take into account a multi-objective optimization.

3 Accelerate the convergence of the searching algorithm.

O4: Analyze the current state of multilevel fault mitigation techniques, with special emphasis on multi-core and multi-threaded systems. Considering:

1 Investigate state of the art in multi-threaded fault mitigation techniques in the following areas:
   - Techniques in the field of Operating Systems level techniques using APIs as *pThreads*, *MPI* or *OpenMP*
   - Techniques in the field of RTOS level
   - Techniques in the field of *bare-metal*

2 Propose a new multilevel mitigation strategy in:
   - Multi-threaded / Multi-Core
   - Protection of the program code

O5: Estimate the contribution of each microprocessor resource to the system's overall reliability, identifying which resources can be accessed or measured directly, to obtain their fault tolerance. For this purpose will:

1 Identify and order resources according to their criticality.

2 Estimate through a semi-empirical model the sensitivity of each resource.

1.5 Hypothesis

A methodology to increase the reliability of electronic components to radiation-induced faults is employing redundancy techniques. Within this set, the techniques based on *n-modular* present high rates of detection and correction of faults induced by radiation. However, these techniques present a decrease in the solutions' performance, when replicas are computed sequentially, or area increases due to the replicated resources. In this sense, it is considered that the optimization of *n-modular* techniques can increase reliability. This optimization aims to increase performance and reduce the area exposed while maintaining the improvements in coverage against faults obtained by the original techniques. In this context, it is proposed as a hypothesis that:

`H1`: Techniques based on machine learning should efficiently explore the space of improved solutions. To this end, the technique should:

1. Perform an intelligent exploration, finding solutions with better compromises between area, performance and fault coverage.
2. Consider a multidimensional solution space based on the objectives to be improved.
3. Present the solutions in terms of the degree of compliance, within the objectives of interest (Pareto efficiency)

`H2`: Hardware `TMR` techniques should present reductions in area exposure. To this end:

1. The approximation in the computation of the replicas should reduce the sizes of the circuit.
2. Reductions in complexity, due to the approximate replica calculation, should not affect the algorithm calculation.
3. The optimization of the replica circuits should not decrease the reliability rate of the `TMR`.

`H3`: Compilers could produce, during the optimization and resource allocation stages, reliability improvements as a side effect. Because:

1. The large number of compiler optimizations should produce a set of similar size applications without modifying the source code.
2. It is highly probable that there will be applications with better fault coverage than the non-optimized version.

`H4`: Using infra-utilized resources presented in most of the state-of-the-art microprocessors, could accelerate the computation of replicas in *n-modular* techniques. Based on this hypothesis:

1. Improved performance of n-modular techniques should be achieved by spreading replications across multiple instruction fluxes.

2 The code for replication should be minimally intrusive and reduce the introduction of new points of failure.

H5: The constituent parts of the device being evaluated may not share the same sensitivity to radiation. Because of this:

1 The implementation of each component may differ slightly. Therefore each bit that makes up the platform may not share the same radiation susceptibility.

2 Each active area of the components and their exposure time should contribute to the platform's reliability differently.

1.6 Methodology

The methodology used in this thesis is a combination of deductive, experimental, and quantitative research methods. In this sense, the first task was to analyze the literature to identify the most relevant problems in microprocessors exposed to ionizing radiation. Also, novel ways to increase fault tolerance in critical systems were identified. A set of objectives and hypotheses was then established, focusing on ways to obtain early predictions of reliability and new methods to mitigate faults induced by ionizing radiation. Subsequently, methods and algorithm approaches were developed to mitigate radiation-induced faults and validate them through simulated faults injection campaigns and accelerated radiation tests. Finally, the results of both campaigns are used to create a semi-empirical analysis. This model uses statistical models to generalize the results of the research and predict the behavior of new hardened solutions under the effects of radiation.

Below is an overview of the most relevant strategies, methods and algorithms used:

`M1` A search method based on a multi-objective optimization guided by genetic algorithms to find solutions with variations in the different objectives under evaluation: reliability, performance, and resource use. Thus, this algorithm searches:

  1 Improve reliability by generating optimized applications using the compiler.

  2 Reduce the area of `TMR` hardening techniques, using the approximate computation.

`M2` Threaded strategy applied to *n-modular* approaches to reduce the overhead induced by the sequential calculation of replications.

`M3` Reliability prediction using simulated fault coverage and radiation measurements of the sensitivity of platform resources. For that purpose, it will be used:

  1 Low-intrusive hardening substitution focus on fast and easy development mechanism to implement *n-modular* techniques.

The methodologies that have been followed are detailed and discussed below:

1.6.1 `M1` Multi-objective optimization process guided by genetic algorithms

It is common for hardening techniques to cover different scenarios, using different parameters or configurations. Exploring all possible configurations that increase reliability may require significant computational effort, which is

not always feasible. In this context, *Genetic Algorithms* (GA), are a solution for efficient and automatic exploration of the solution space. GA belongs to a set of algorithms called *Evolutionary Algorithms* (EAs), inspired by biological evolution. They encode one of the solutions to the problem as the genes of an individual belonging to a population (solution set). Each individual of the population is evaluated according to the degree of fulfillment of the problem's objective. The individual is then classified (selected), to check if it has the skills to remain within the evolutionary loop (see Fig 1.11).

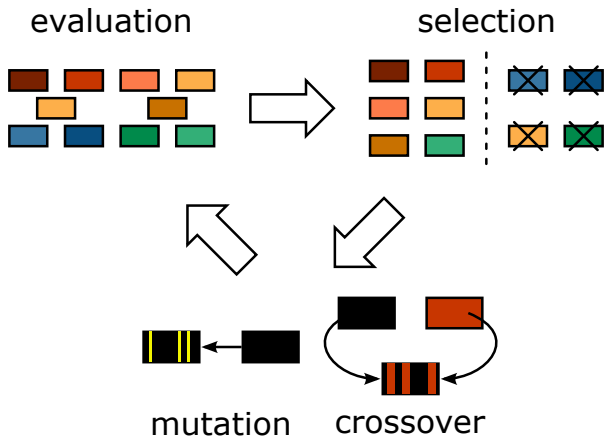


Figure 1.11: Stages in the cycle of genetic algorithms: Evaluation, Selection, Crossover and Mutation.

The selection stage allows GAs to make a classification. The most suitable individuals are selected to continue in the evolutionary loop. The last stage is performed using two operators: crossing and mutation. The crossover operator allows the algorithm to perform an optimization based on gradient descent, which only considers the best solutions to generate the new population. However, this operator, by itself, does not guarantee the best set of solutions. An example of this problem occurs when the population degenerates with similar or equal individuals, reaching a local minimum, in which the individuals do not present variation (inbreeding). For this reason, genetic algorithms use the mutation operator to avoid the local minimum. This operator selects an individual to change randomly, with low probability, the genes that identify its solution. As a result, an individual can obtain a competitive ability that improves the rest of the population, allowing the search algorithm's progression. The disadvantage of this methodology is that the optimization process is based on a single objective function.

In the case of reliability problems, not only is a system sought to increase fault coverage, but also to reduce the exposed area (size) and increase the computational performance of the problem. For this reason, the problem of increasing fault tolerance (reliability) is far from being mono-objective. In this context, the selection of a composite function, which fairly evaluates the population, is

not always feasible because the objectives may be unrelated or opposite to each other. Solutions must be considered within a multidimensional space, defined by the objectives being evaluated, to make a fair assessment. As a consequence, the improvement of a single objective can lead to a worsening of the others. In this sense, to compare the solutions using algorithms based on the concept of Pareto efficiency elicit to evaluate the trade-off among the objectives selected. This algorithm is useful to create a set of solutions with improved qualities, called Pareto frontier. These solutions are characterized by not being able to improve any objective without worsening the others (see Figure 1.12).

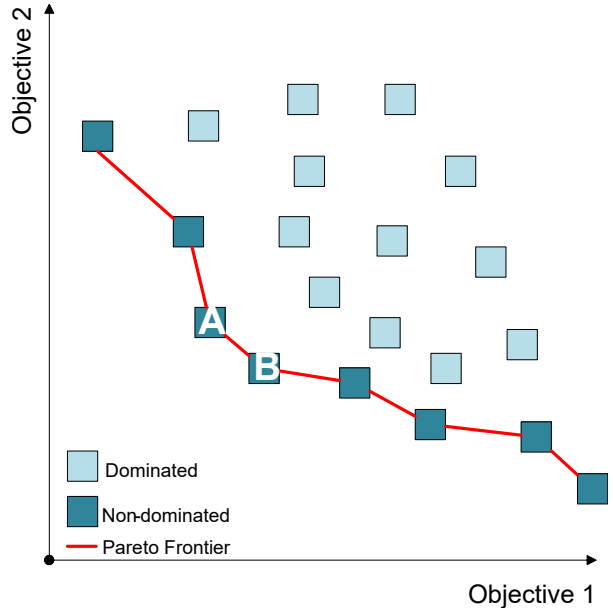


Figure 1.12: Representation of solutions in a multi-objective space based on the concept of Pareto's Frontier. The graph shows two objectives under evaluation, where the best solutions, not dominated, conform the Pareto frontier (red line). Within the Pareto frontier, the best-balanced objectives (A and B) are highlighted.

The combination of both algorithms (*Genetic Algorithms* and *Multi-Objective Optimization Algorithms*), in the so-called `MOOGA` algorithm, allows a blind search in a multidimensional solution space (Figure 1.13) This algorithm is configured using three parameters established in the initial configuration of the algorithm: initial population, `GA` parameters, and the number of iterations or other convergence criteria. Depending on the problem being evaluated, these parameters are responsible for the acceleration and convergence of the search algorithm. Assuming an algorithm has a low initial population, it is necessary to increase the GA mutation rate to increase the diversity of the population, avoiding inbreeding. However, this configuration, maintained during a high number of

iterations (generations), does not guarantee the convergence of the search algorithm. For this reason, the MOOGA algorithm is composed of two stages. The first stage tries to maximize the unique individuals in the population. In the second stage, the mutation rate is reduced, allowing the convergence of the algorithm.

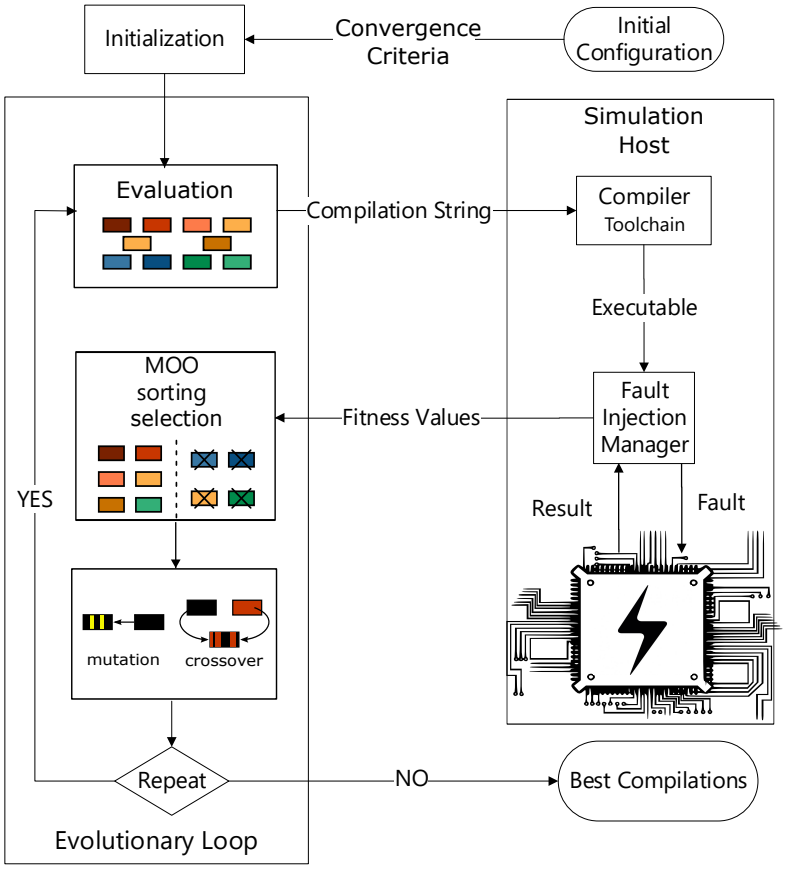
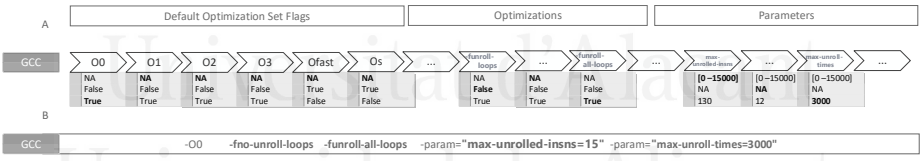


Figure 1.13: Combination of genetic algorithms and multi-objective optimization algorithms in the so-called MOOGA algorithm. The diagram shows an example focused on optimizing applications using the compiler and a test tool (fault injection manager) in charge of evaluating and reporting the fitness values.

M1−1 Improve reliability by optimizing compiler parameter selection

Compilers can produce different programs that behave identically, making small changes to the instructions scheduling, or replacing some of the instructions with higher-performance ones. As a result, it is possible to achieve the

same computation, using similar processing flows. These modifications can result in programs with improved performance and high resource usage, or applications with reduced performance and high resource availability. Achieving improved performance, or producing programs with reduced resource use, is relatively simple to obtain. However, general-purpose compilers do not offer a fast and easy way to produce reliable code transformations. In this sense, using compiler optimizations that make extensive use of system resources produces performance increases. However, it can lead to significantly higher resource exposure. In contrast, optimizations aimed at reducing resource use, increase computation time. Thus, increase the exposure time to the harmful environment. The literature shows works where only by adjusting the process of generating a program by the compiler can it be possible to achieve different levels of reliability [44]. These works show that it is possible to improve performance and reliability, adjusting the compilation by using the most common optimization provided by compilers. However, this set of optimizations, commonly used by developers, triggers a wider range of optimizations (see Figure 1.14). Compilers have an extensive set of optimizations that, in some cases, interfere with each other. Due to the large number of options, it is not possible to know the effects of all of them. It is extremely costly, from a computational point of view, to evaluate each of the possible combinations. Also, the compiler parameter responsible for reliability increases may be application-dependent. To address this problem, MOOGA performs a blind search for all compiler optimizations, looking for optimizations that improve size, performance, and fault coverage, improving system reliability.



MOOGA uses simulators as a way to obtain quality metrics, such as the computational performance of a program, to evaluate solutions. However, to estimate metrics, such as fault coverage, it is necessary to perform simulated fault injection campaigns. Fault injection managers (FIM), obtain the statistical behavior of the platform under radiation (Figure 1.13). This statistical campaign is accomplished by performing a robust random injection, avoiding compromising the results obtained. After evaluating each injection, FIM classifies the result obtained according to the effect caused by the failure in the system. In case of having a microprocessor-based system running a specific program, the effect of the failure is modeled according to the following classification: unACE, SDC, and HANG. If the program succeeds in completing the calculation correctly, then the injection is labeled as unACE. In the case of an erroneous result, the fault is labelled as *Silent Data Corruption* (SDC). Finally, if the platform becomes inoperative or does not comply with the time limitations (critical time) is labeled as HANG. These campaigns use conventional simulators modified to extend their functionality through the use of plugins or by modifying their source code. In these cases, the new injection capabilities provided to the simulators must consider non-intrusive and deterministic behavior. Non-intrusive behavior is intended to ensure that the simulation remains unchanged during the activation of the platform fault. Whereas, deterministic is intended that the evaluation is replicable.

M1−2 Reduce the resources of TMR techniques by using approximate computing

Similarly to compilers that optimize applications for improved reliability, circuits can be optimized, approximating their functionality for better performance or reduced occupancy. Circuit approximations can be made by eliminating logic gates, which define the circuit and replacing them with simpler and higher performance ones, achieving a reduction in circuit complexity. As a result, these approximations can present a reduction in accuracy or even provide erroneous results in some cases. Evaluating all possible approaches and circuit combinations to obtain the best approximate circuit is not feasible, due to the large number of logic gates that a circuit can have and the varying degrees of complexity of each gate. MOOGA search method can solve this problem by selecting the most efficient circuits. Although they are not able to produce reliable behavior, approximate circuits can be combined between them, to produce a reliable solution (approximate TMRs). Due to the existence of cases where the result of the replications is wrong, the correcting (voter) circuit, is activated more frequently. Therefore, the approximation of the TMR replicas should produce the same results as the original circuit, excluding some cases. This way, the period of vulnerability of TMR is limited. This feature is exploited by the MOOGA algorithm to narrow the huge space of possible solutions further, accelerating the convergence of the algorithm.

1.6.2 M2 Threaded n-modular strategies in bare-metal

State-of-the-art microprocessors seek to increase software applications' performance, using parallel programming schemes such as *Asymmetric Multiprocessing* (AMP) and *Symmetric Multiprocessing* (SMP). These schemes make use of multiple instructions flows to accelerate the processing tasks when the data to be processed is independent between the flows. For instance, *n-modular* strategies can perform the same computation using an independent set of data (replicas). Because of this, these techniques can increase their performance by distributing the calculation of replica over different instruction flows (Figure 1.15).

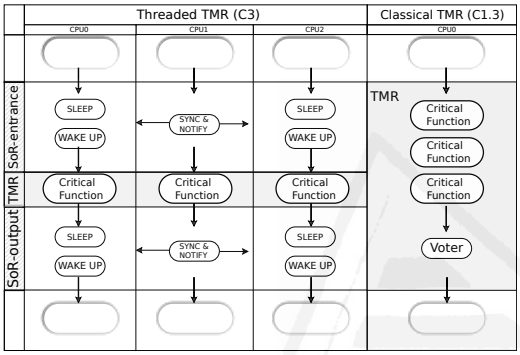


Figure 1.15: Schemes of TMR and DWC−R techniques, using a parallelized version with threads and its sequential analog. The parallel versions show the synchronization and correction block surrounding the critical function to be protected. In contrast, the sequential versions present a voter to obtain the expected result once the replicas are calculated.

The literature shows several approaches where replicas' spreading in different instruction flows was successfully applied. A recent publication shows the use of high-level libraries for the dissemination of replicas of *n-modular* techniques, requiring the use of resources such as the operating system for their operation [59]. One of the conclusions of this work highlights that reliability is reduced due to the increase of unprotected artifacts to enable parallel computing (the operating system and thread libraries). In another publication, it is shown that the spread of replicas can lead to performance degradation, not complying with the solution's requirements and time limitations [60]. The authors overcame this problem by developing a strategy that modifies the scheduling and granularity to which redundancy is applied, reducing performance penalties. From these publications, it is concluded that the use of multiple instructions flows increases the solution's overheads (size and time) due to the artifacts in charge of data distribution and synchronization. Therefore, a parallelized software hardening solution, aiming at increasing reliability, should limit and minimize the inclusion of these unprotected artifacts. In this context, solutions without an operating system (*bare-metal*), allow to ex-

ploit low-level optimizations and do high-performance synchronization routines, avoiding the inclusion of sizeable unprotected software pieces. In this context, the dissemination of replicas in *n-modular* techniques using different instruction flows transforms the solution into a software lockstep. Unlike its hardware analog, this solution allows for easier adjustment of the consistency checking frequency and the number of protected resources in the solution, allowing for improved performance while maintaining the fault coverage of the solution. This technique allows for independent processing of replicas on platforms with shared resources, using isolated memory areas (partitions), thus avoiding collisions in the use of resources. Due to this restriction, the verification routines will be performed after all replicas have been calculated, accessing the other replicas' results in read-only mode. This way, the solution only includes these synchronization codes as the artifact that differentiates it from the original code (Figure 1.15).

1.6.3 `M3` Mathematical model for soft error rate prediction

Critical systems, exposed to ionizing radiation, must be validated by expensive radiation tests. In this way, the correct operation and high reliability of the system are ensured against radiation-induced faults. Therefore, obtaining early and highly accurate measurements of reliability is essential to know the effectiveness of a hardened solution before accelerated radiation campaigns evaluate it. In this context, the simulators can provide early measurements of reliability on microprocessors. However, even with high precision, the simulators model approximates the real hardware operation. Therefore, the accuracy of the model affects the measured results, making them slightly different from the results obtained when using the actual hardware. Furthermore, the use of high accuracy models limits the number of solutions to evaluate due to the penalties in terms of simulation performance. Consequently, to carry out an effective study of hardware behavior under radiation, it is mandatory to find the best trade-off between performance and accuracy. In this context, slightly reducing the accuracy of the model and approximating the effects of faults can increase the model's performance, allowing a statistical evaluation of the hardware. An example of reducing accuracy to speed up simulations is approximating the effects of radiation (`SEU`) using the bit-flips model. This model simulates when a particle hits a transistor, generating a parasitic charge that can change the state of the bits in memory. However, the model makes an unrealistic assumption, considering all events equally likely, regardless of the underlying technology. This assumption can lead to erroneous reliability assessments because each platform component may be of different technology or use a different logical combination. Since each component has a different sensitivity to radiation, it is necessary to assess each component of the platform's reliability in isolation. Therefore, to provide a measure of reliability, it is necessary to develop a semi-empirical model, which uses the different fault sensitivity assets in simulation and modulates them with the sensitivity of each component, which is obtained from previous radiation experiments

(historical radiation tests). This model is based on a multiple linear regression (equation 1.1), where the stimulus (X_i) correspond to the simulated fault injection measurements of each component, the radiation sensitivity of the platform components are (β_i), and intercept terms (ϵ_1 and ϵ_2) corresponds to the estimation of unmodelled elements.

$$\mathbf{Y} = \sum_i \beta_i \mathbf{X} + \epsilon_1 + \epsilon_2 \tag{1.1}$$

M3−1 Generic C++ programming for transparent software hardening

Hardening the code usually involves introducing new software blocks (artifacts) into the programs, to allow reliable calculation. These artifacts are highly intrusive and, in some cases, difficult to understand and debug. Code intended to protect software from the effects of radiation usually increases the complexity of development and debugging time, increasing production costs.

In this context, it is possible using the generic programming paradigm (templates) to make highly reusable algorithms for a large number of problems, applying only a few modifications, to adapt to new specifications. This paradigm allows developers to focus on the generation of algorithms, generalizing the solution, and abstracting from the type of data used. This type of tool obtains the basic structure of the algorithm, and, once the functionality is verified, it is possible to export it to other more complex elements. An example of application is found in the hardening frameworks based on templates, aimed at avoiding the introduction of defects in the program to be protected, allowing faster development. Also, another feature they introduce, is the ease maintenance, due to the similarity of the parts of the program with the original code. These modifications present low intrusiveness, because they only make a substitution of the basic types of the language, by a hardened one. The hardened language types, using the concept of the *Sphere of Replication* (SoR) [61], create and compute three replicas of the variable to be protected. Eventually, to restore consensus, the hardened type takes a majority vote (Figure 1.16).

```
template <typename DataType>
class TD {
    private volatile DataType d1;
    private volatile DataType d2;
    private volatile DataType d3;
}
```

Figure 1.16: Template-based replication code. DataType is the basic (unhardened) language type to be replaced.

Given the versatility of this technique, it is essential to identify the platform's less reliable elements. The empirical model described above helps to

quickly assess the impact of the solution by identifying the elements on which to focus the hardening effort.

1.7 Articles and contributions supporting this thesis

The work developed in this thesis is focused on the fulfillment of the proposed objectives, validating the hypotheses raised. In this sense, the methods, algorithms, and strategies proposed focused on the achievement of a semi-empirical mathematical model. This model predicted the effects of ionizing radiation on microprocessors, taking into account aspects of both the hardware (cache memory, registers) and software (binary file structure) technological structure. As a result, this thesis performed 6 publications in high impact JCR ranked journals (J1 to J6), 11 communications to international conferences (C1 to C11) and 3 communications to national conferences (N1 a N3). The contributions to the journal called **J1**, **J2**, **J3** and **J4**, constitute the compendium of publications presented for the defence of the thesis.

Below, all the contributions made, within the scope of this thesis, are detailed according to the type of publication: journals indexed in the JCR, communications to international conferences, and communications to national conferences:

1.7.1 Contributions to peer-reviewed journals articles

J1 A. Serrano-Cases, L. M. Reyneri, Y. Morilla, S. Cuenca-Asensi, and A. Martínez-Álvarez "Empirical mathematical model of microprocessor sensitivity and early prediction to proton and neutron radiation-induced soft errors", *IEEE Transactions on Nuclear Science* vol. 67 no. 7, pp. 1511–1520 2020. DOI: 10.1109/tns.2020.2993637 JCR 2019 impact factor: 1.575

J2 A. Serrano-Cases, F. Restrepo-Calle, S. Cuenca-Asensi, and A. Martínez-Álvarez "Multi-threaded mitigation of radiation-induced soft errors in bare-metal embedded systems", *Journal of Electronic Testing: Theory and Applications (JETTA)* vol. 36 no. 1, pp. 47–57 2020. DOI: 10.1007/s10836-019-05846-4 JCR 2019 impact factor: 0.596

J3 L. M. Reyneri, A. Serrano-Cases, Y. Morilla, S. Cuenca-Asensi, and A. Martínez-Álvarez "A compact model to evaluate the effects of high level c code hardening in radiation environments", *Electronics* vol. 8 no. 6, p. 653 Jun. 2019. DOI: 10.3390/electronics8060653 JCR 2019 impact factor: 2.412

J4 A. Serrano-Cases, Y. Morilla, P. Martín-Holgado, S. Cuenca-Asensi, and A. Martínez-Álvarez "Nonintrusive automatic compiler-guided reliability improvement of embedded applications under proton irradiation", *IEEE Transactions on Nuclear Science* vol. 66 no. 7, pp. 1500–1509 Jul. 2019. DOI: 10.1109/tns.2019.2912323 JCR 2019 impact factor: 1.575

J5 M. Peña-Fernández, A. Serrano-Cases, A. Lindoso, M. García-Valderas, L. Entrena, A. Martínez-Álvarez, *et al.* "Dual-core lockstep enhanced with redundant multithread support and control-flow error detection", *Microelectronics Reliability* vol. 100-101, p. 113 447 Sep. 2019. DOI: 10.1016/j.microrel.2019.113447 JCR 2019impact factor: 1.535

J6 I. Albandes, A. Serrano-Cases, M. Martins, A. Martínez-Álvarez, S. Cuenca-Asensi, and F. Kastensmidt "Design of approximate-TMR using approximate library and heuristic approaches", *Microelectronics Reliability* vol. 88-90, pp. 898–902 Sep. 2018. DOI: 10.1016/j.microrel.2018.07.115 JCR 2018 impact factor: 1.483

1.7.2 Contributions to a conference proceedings

C1 A. Aponte-Moreno, J. Isaza-Gonzalez, A. Serrano-Cases, A. Martínez-Álvarez, S. Cuenca-Asensi, and F. Restrepo-Calle, "An experimental comparison of fault injection tools for microprocessor-based systems", in *2020 IEEE Latin-American Test Symposium (LATS)*, IEEE, Mar. 2020

C2 D. R. Falco, A. Serrano-Cases, A. Martínez-Álvarez, and S. Cuenca-Asensi, "Soft error reliability predictor based on a deep feedforward neural network", in *2020 IEEE Latin-American Test Symposium (LATS)*, IEEE, Mar. 2020

C3 L. M. Reyneri, A. Serrano-Cases, Y. Morilla, S. Cuenca-Asensi, and A. Martínez-Álvarez, "A mathematical model to predict microprocessors fault tolerance under proton and neutron irradiation", in *2019 RADiation and its Effects on Components and Systems (RADECS)*, Sep. 2019

C4 A. Serrano-Cases, F. Restrepo-Calle, S. Cuenca-Asensi, and A. Martínez-Álvarez, "Softerror mitigation for multi-core processors based on thread replication", in *2019 IEEE Latin American Test Symposium (LATS)*, IEEE, Mar. 2019. DOI: 10.1109/latw.2019.8704614

C5 M. Peña-Fernandez, A. Serrano-Cases, A. Lindoso, M. Garcia-Valderas, L. Entrena, A. Martínez-Álvarez, *et al.*, "Dual-Core Lockstep Enhanced with Redundant MultiThread Support and Control-Flow Error Detection", in *30th European Symposium on Reliability of Electron Devices, Failure physics and Analysis (ESREF)*, Sep. 2019

C6 I. Albandes, A. Serrano-Cases, M. Martins, A. Martínez-Álvarez, S. Cuenca-Asensi, and F. Kastensmidt, "Design of Approximate-TMR using Approximate Library and Heuristic Approaches", in *29th European Symposium on Reliability of Electron Devices, Failure physics and Analysis (ESREF)*, Sep. 2018

C7 A. Serrano-Cases, Y. Morilla, P. Martin-Holgado, S. Cuenca-Asensi, and A. Martínez-Álvarez, "Automatic compiler-guided reliability improvement of embedded processors under proton irradiation", in *2018 RADiation and its Effects on Components and Systems (RADECS)*, Sep. 2018

C8 I. Albandes, A. Serrano-Cases, A. Sanchez-Clemente, M. Martins, A. Martínez-Álvarez, S. Cuenca-Asensi, *et al.*, "Improving approximate-TMR using multi-objective optimization genetic algorithm", in *2018 IEEE 19th Latin-American Test Symposium (LATS)*, IEEE, Mar. 2018. DOI: 10.1109/latw.2018.8349665

C9 J. Isaza-Gonzalez, A. Serrano-Cases, A. Martínez-Álvarez, S. Cuenca-Asensi, H. Guzman-Miranda, and M. A. Aguirre, "Contrast of a HDL model and COTS version of a microprocessor for soft-error testing", in *2017 18th IEEE Latin American Test Symposium (LATS)*, IEEE, Mar. 2017. DOI: 10.1109/latw.2017.7906771

C10 A. Serrano-Cases, J. Isaza-Gonzalez, S. Cuenca-Asensi, and A. Martínez-Álvarez, "On the influence of compiler optimizations in the fault tolerance of embedded systems", in *2016 IEEE 22nd International Symposium on On-Line Testing and Robust System Design (IOLTS)*, IEEE, Jul. 2016. DOI: 10.1109/iolts.2016.7604701

C11 J. Isaza-Gonzalez, A. Serrano-Cases, F. Restrepo-Calle, S. Cuenca-Asensi, and A. Martínez-Álvarez, "Dependability evaluation of COTS microprocessors via on-chip debugging facilities", in *2016 17th Latin-American Test Symposium (LATS)*, IEEE, Apr. 2016. DOI: 10.1109/latw.2016.7483335

1.7.3 Contributions to a national conference proceedings

N1 A. Serrano-Cases, S. Cuenca-Asensi, and A. Martínez-Álvarez, "Estrategia multi-hilo para la mitigación de fallos software inducidos por radiación en sistemas empotrados carentes de sistema operativo", Sep. 2019. [Online]. Available: http://hdl.handle.net/10662/9626

N2 A. Serrano-Cases, L. M. Reyneri, S. Cuenca-Asensi, and A. Martínez-Álvarez, "Protección De Software Frente A Radiación En Procesadores Multi-Núcleo Sin Sistema Operativo", Sep. 2018. DOI: 10.5281/ZENODO.1442364

N3 A. Serrano-Cases, S. Cuenca-Asensi, and A. Martínez-Álvarez, "Mejorando La Tolerancia A Fallos De Sistemas Embebidos Cambiando La Compilación", Sep. 2017. DOI: 10.5281/ZENODO.996065

1.7.4 Objectives and hypotheses covered by each publication

The following figures show the relationships between research publications in journals and conference papers, the hypotheses put forward, and the objectives pursued in this thesis (Figure 1.17, Figure 1.18 and Figure 1.19). As can be seen, all the objectives and hypotheses raised by the contributions made during this thesis have been covered.

Figure 1.17: Objectives and hypothesis covered by the published research on international JCR-indexed journals

	J1	J2	J3	J4	J5	J6
O1: State-of-the-art Analysis	✔	✔	✔	✔	✔	✔
O2: Simulators evaluation		✔		✔	✔	
O3: Automatic space exploration				✔		✔
O4: Multi-level Mitigation		✔			✔	
O5: Component Radiation Sensitivity	✔		✔			
H1: Machine Learning				✔		✔
H2: Approximate Hardware TMR						✔
H3: Compilers improve reliability				✔		
H4: Improving n-modular performance		✔	✔		✔	
H5: Component Radiation sensitivity	✔		✔			

Figure 1.18: Objectives and hypothesis covered by the published research on international conferences

	C1	C2	C3	C4	C5	C6	C7	C8	C9	C10	C11
O1: State-of-the-art Analysis	✔	✔	✔	✔	✔	✔	✔	✔	✔	✔	✔
O2: Simulators evaluation	✔			✔	✔				✔		✔
O3: Automatic space exploration						✔	✔	✔		✔	
O4: Multi-level Mitigation		✔	✔	✔	✔						
O5: Component Radiation Sensitivity			✔								
H1: Machine Learning	✔					✔	✔	✔	✔	✔	
H2: Approximate Hardware TMR						✔		✔			
H3: Compilers improve reliability		✔					✔			✔	
H4: Improving n-modular performance			✔	✔	✔						✔
H5: Component Radiation sensitivity	✔		✔						✔		✔

Figure 1.19: Objectives and hypothesis covered by the published research on national conference

		N2	**N3**
O1: State-of-the-art Analysis	✔	✔	✔
O2: Simulators evaluation			✔
O3: Automatic space exploration			✔
O4: Multi-level Mitigation	✔	✔	
O5: Component Radiation Sensitivity			
H1: Machine Learning		✔	✔
H2: Approximate Hardware TMR			
H3: Compilers improve reliability			✔
H4: Improving n-modular performance	✔	✔	
H5: Component Radiation sensitivity			

1.8 Research results

In this thesis, the optimization and hardening of applications and circuits, in embedded devices, to increase their reliability was studied. In the manuscripts mentioned above, different ways to optimize applications, reducing their exposure to ionizing radiation in time and resources, in order to increase reliability are shown (contributions called **J2** and **J4**). Also, several papers show, from simulated fault injections, how to obtain better reliability estimations. For this purpose, an empirical model is used, created from the results obtained in real radiation campaigns (contributions called **J1** and **J3**). This model has also made it possible to optimize the hardening techniques developed by selectively guiding the hardening of the resources to be protected.

1.8.1 Reliability improvements by using Multi-objective optimizations genetics algorithms

The manuscript named **J4**, is the continuation of the contributions named **J6**, **C7**, **C6**, **C8**, **C10** and **N3**. These papers show a method capable of exploring a non-linear, multidimensional solution space (MOOGA). This method uses two algorithms to classify and explore the solution space. The first algorithm is a multi-objective optimization (MOO), based on the concept of Pareto Efficiency. The algorithm evaluates the solutions found and the degree of compliance with the different objectives selected (performance, resource use, and fault coverage). The second case uses a genetic algorithm (GA), to explore the solution space automatically and efficiently, using gradient descent. The so-called MOOGA algorithm is designed to find a set of applications, with different degrees of reliability.

The contributions named as **J6**, **C8**, and **C6**, show a part of the work of this thesis, consisting of the optimization of the TMR hardware redundancy algorithm. This optimization process focuses on reducing area and consumption

overheads, replacing some of the logic gates that compose the hardened circuit, using others from an approximate gate library. This approximate gate library offers a set of gates with higher performance and less complexity than the gates used in the original circuit. However, this library achieves the performance improvements by reducing the similarity to the original gate. Due to the high number of logic gates that a circuit can have, the MOOGA method studies numerous combinations and approximations that can be generated. These combinations tend to grow exponentially with the number and complexity of the gates that compose the circuit. The MOOGA algorithm was configured to evaluate the solution space efficiently, with a single individual as the initial population, corresponding to the original TMR circuit. The first generation of the MOOGA algorithm was configured to produce a high number of mutated individuals, accelerating the population growth. As a result, the population of unique individuals grew rapidly. After a few generations, the mutation rate was reduced to allow the convergence of the algorithm. As a result, the MOOGA algorithm found a set of combinations, where the area is reduced by about 50%. The fault coverage is maintained by about 70%, compared to the full protection of the TMR circuit 1.20). The MOOGA algorithm was also able to obtain a more extensive set of smaller TMR circuit combinations with better fault coverage than other published search methods [82].

Similarly, **J4**, **C7**, and **C10** show how the MOOGA algorithm can improve the reliability of a software application by adjusting, blindly, the process of compilation and generation of the executables. In this case, two architectures were evaluated: MSP430 and ARM. The MSP430 microcontroller has a reduced set of capabilities, limiting the achievement of relevant improvements in the execution of the program. As a result, improvements in reliability can only proceed from improved scheduling or pre-processing during compilation. Evaluating the reliability of an application uses some of the tools presented in the following publications **C2**, **C9**, and **C11** where several simulators were validated to measure increases in performance and reliability. In the paper presented at the international congress **C10**, the search capabilities of the MOOGA algorithm are shown. The paper shows that the algorithm found a considerable set of solutions, taking into account, the low complexity of the architecture.

In contrast, the works called **J4** and **C7** show how the compiler can optimize the instruction flow when some of the elements of the architecture of an ARM microprocessor are activated: the *Out-of-Order* pipeline, the caches, and the pre-search of instructions. Because of these elements, the computation can be accelerated using the appropriate compiler's set of optimizations. As a result, the MOOGA algorithm generated a large set of applications, using the same source code. Also, the MOOGA algorithm improved the behavior of the applications, optimized using the default options of the compiler. The best solutions found increased performance, and coverage against faults while reducing the exposed area (hardware resources). In this sense, in the publication called **J4**, it was shown that the optimizations, by default of the compiler, focused on the increase of the performance, generate less reliable programs. In contrast, the optimization, known as O0, showed greater reliability (Figure 1.21. However,

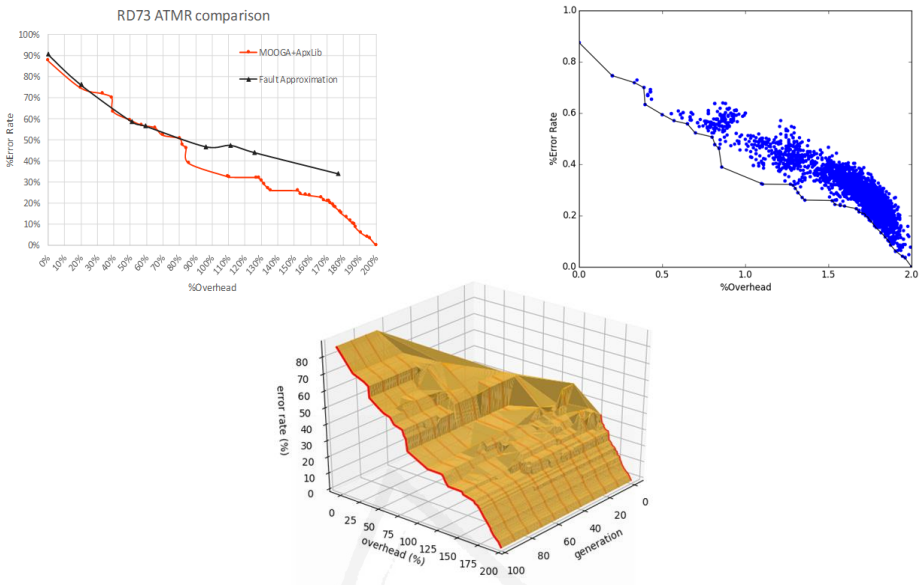


Figure 1.20: The images show the output of the optimization process applied to the RD73 circuit. The upper left figure compares the Pareto frontier solutions with other methods used to search for approximate TMRs [82]. The upper right side shows the population that MOOGA can find. Finally, the figure below shows the error rate's behavior and the cost overhead in the size of the circuits, as a function of the number of generations computed (algorithm convergence).

the increase in exposure time and the greater use of resources make it more prone to events under radiation. Part of the results obtained in this research were used in the international congress called **C1**. This work created a set of descriptors, through deep learning techniques, allowing early predictions of the reliability of an application.

The manuscript called **J4** shows the selection of several of the applications generated by MOOGA. This set of applications were exposed to proton irradiation in the cyclotron of the CNA facilities (Sevilla). The results obtained from the radiation campaign (table 1.1) showed high similarities between the dynamic cross-sections obtained from the radiation campaign and the results obtained from the simulation. The results show that the worst solutions tested are the MaxAce and the MinMWTF, presenting the worst fault coverage in the simulation. Consequently, these versions obtained the highest cross-sections (3.8 and 4.8, respectively). On the contrary, O0 and P.Pareto show lower cross-sections, following the trend observed in the simulation. P.Pareto and MaxMWTF, are the programs with higher performance. They also present similar cross-sections, resulting in programs that can perform more executions before a failure (MWTF).

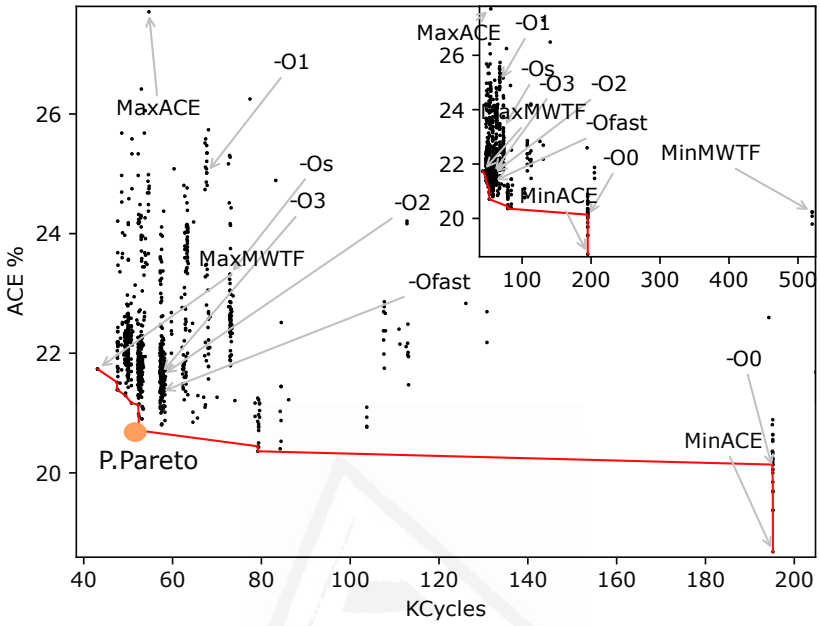


Figure 1.21: Results of the MOOGA optimization process applied to improve fault coverage, performance and resource usage by adjusting compiler options. The view shows the fault coverage versus performance represented on the vertical axis and the horizontal axis, respectively. MinACE and O0 show the best fault coverage; however, they also achieve the worst performance results, excluding the MinMWTF solution. In contrast, performance-enhancing optimizations such as O3 show relatively high performance with a significant reduction in fault coverage. MOOGA's most interesting solution (P.Pareto) shows the best balance between objectives, fault coverage, and high performance. Also, this build is the most reliable solution found during the early evaluation phase.

Table 1.1: Summary of radiation results for each version of the Bubblesort applications. The flux uncertainty is $\pm 10\%$.

$Version$	$Flux$ $p/cm^2 \cdot s$	$Fluence$ p/cm^2	$Cycles$	SDC	$Hang$	σ_{SDC} $(10^{-11})cm^2$	σ_{Hang} $(10^{-11})cm^2$	σ_{Total} $(10^{-11})cm^2$	$MWTF$ (10^{-14})
MaxACE	$7.8 \cdot 10^8$	$3.4 \cdot 10^{12}$	44409	112	14	3.3(2.7, 3.9)	0.42(0.20, 0.64)	3.8(3.2, 4.4)	3.96
-O0	$7.4 \cdot 10^8$	$4.9 \cdot 10^{12}$	220842	76	40	1.6(1.3, 1.9)	0.82(0.57, 1.1)	2.4(2.0, 2.8)	1.26
MaxMWTF	$8.3 \cdot 10^8$	$3.1 \cdot 10^{12}$	38412	79	36	2.5(1.9, 3.1)	1.2(0.82, 1.6)	3.7(3.0, 4.4)	4.67
MinMWTF	$1.0 \cdot 10^9$	$2.4 \cdot 10^{12}$	389806	69	45	2.9(2.2, 3.6)	1.9(1.4, 2.4)	4.8(3.9, 5.7)	35.4
P.Pareto	$1.1 \cdot 10^9$	$3.4 \cdot 10^{12}$	39635	65	38	1.9(1.4, 2.4)	1.1(0.74, 1.5)	3.1(2.5, 3.7)	5.45

1.8.2 Reliability improvements with bare-metal multithreading

The contribution, called **J4**, revealed that program performance is a critical factor in improving reliability. In this sense, the methods used in the literature

for hardening, such as the *n-modular* software technique, calculate the replicas extending them in time over a single-core, degrading the performance. However, several microprocessors have evolved to multi-core architectures, allowing them to perform simultaneous calculations on different instruction flows. As a result of this research, it is now possible to accelerate the calculation of such replicas to gain performance and reliability simultaneously.

The paper called **J2**, is the continuation of the contributions called **C4**, **J5**, **N2**, and **N1**, in which the optimization of the processing of the replicas, using different cores and *bare-metal* threads is evaluated. In order to validate the optimization performed, this work used simulated injection campaigns, as well as a preliminary version of an empirical model, to improve the reliability of the technique. As a result, the work called **J2** showed a reduction in the overheads associated with TMR techniques and a notable increase in the solution's performance. However, this programming model can introduce new points of failure, which are not negligible. For this reason, in the publication called **J5**, a hybrid technique is used to analyze and solve the communication problems between processes of the first proposals. One of the conclusions obtained showed that the synchronization process was extremely costly, even nullifying the increase in performance obtained in some cases. This work identified the synchronization frequency adjustment as a critical factor in obtaining improvements in reliability. The paper called **C4**, and its extension called **J2**, demonstrated that parallelized TMR, highly reduces the temporal overheads, compared to the single-thread approach (Figure 1.22). Also, research shows that parallelizing the DWC−R technique or an *n-modular* variant, with a number of replicas factor of two, increases radiation sensitivity due to the high cost of the mitigation routine. In the latter case, context restorations and reruns nullify the performance increase.

1.8.3 Early reliability prediction using an empirical model

The papers called **J2** and **J3**, use the model presented in the paper called **J1**. This model is used to estimate the reliability of a software application and predict the behavior of software applications under radiation. This model uses the simulation results (fault injection campaign) to estimate the fault coverage and modulate them with the system components' sensitivity, obtained from historical data of radiation campaigns. The sensitivity parameters of the model use proton radiation measurements obtained from the CNA and the neutron measurements from the Los Alamos Neutron Laboratory – LANL, to train and adjust it empirically. Thus, the model allows predicting the behavior of other applications on accelerated radiation tests, using simulated injection campaigns, targeting the same device used for training the model.

This model can estimate the reliability of an application, using instructions accurate simulator, such as those used in the works called **C11**, **C10**, and **C7**. The results of the model show that it is possible to predict the effects of radiation on a device and obtain the sensitivity to radiation of the components of both SoC devices and microcontrollers. An early version of the model was

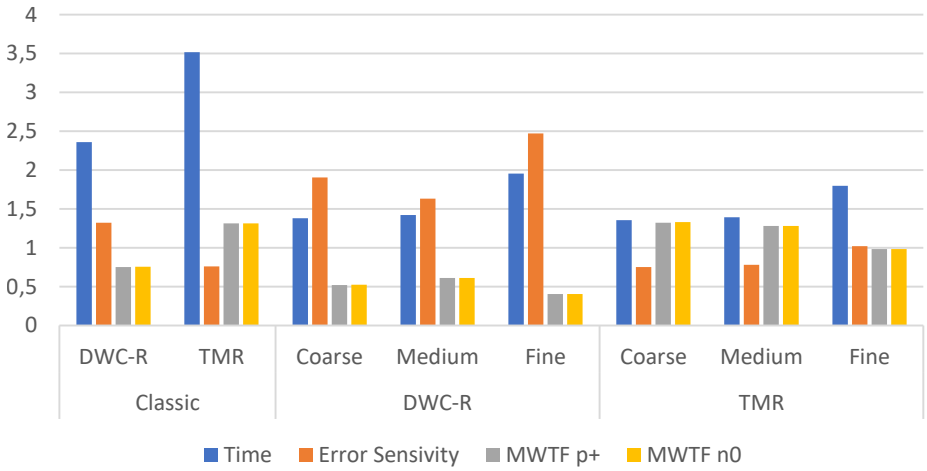


Figure 1.22: Evaluation of the *n-modular* versions using multiple threads, compared to the single-threaded versions. All versions are standardized with the unhardened version. The multi-threaded versions show a reduction in the time needed to calculate replicas; however, all parallelized versions of the `DWC-R` increase their sensitivity to faults compared to the original version. In contrast, `TMR` versions can reduce the sensitivity to errors while showing increases in performance and the amount of work completed before failure (`MWTF`).

presented in the paper called **`J3`**. It was later extended in the paper called **`J1`**. This model was used to improve a `TMR` strategy, based on generic programming, increasing the reliability of hardening software applications. The objective of the experiment carried out in work called **`J3`**, was twofold. On the one hand, to test a highly reliable but low-performance protection technique. On the other hand, to compare it with the solutions, optimized and unprotected, obtained with the `MOOGA` method, presented in work called **`J4`**. The results obtained show that the `TMR` technique, based on generic programming, improves the application fault coverage. At the same time, performance is reduced due to temporary replication. Furthermore, the results obtained from simulation and radiation, clearly show, that not only the failure coverage affects reliability, but also the performance and the exposed area. This behavior is shown in the results obtained from the model presented in **`J1`** (Figure 1.23), where the hardened solutions (denoted with `H`), show their correction capacity, presenting a reduced number of `SDC` events. The model also shows a large number of `HANG` events due to the exposure time and the increase in sensitive resources. The work called **`J3`**, shows that partial replication is the best hardening strategy. The results of the developed model helped to identify the most sensitive program variables, where to focus the hardening efforts. These coincided with the variables with a long *lifetime*. In this sense, the model and the injection campaigns show that avoiding a complete replacement of the basic

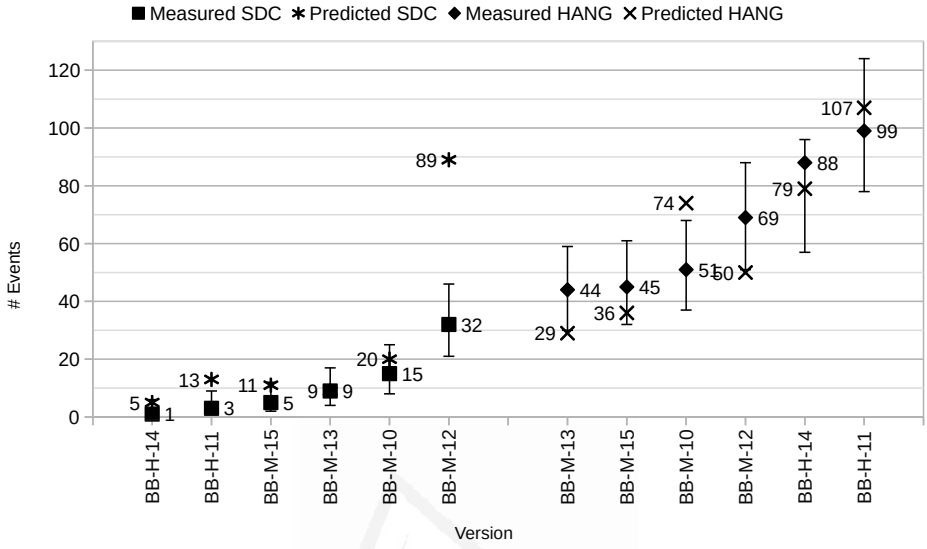


Figure 1.23: Erroneous outcomes (SDC) and HANG events for the Bubblesort (BB) application. The figure shows the template hardening strategies denoted by H, and several MOOGA enhanced builds denoted by M. The graph shows the measurements obtained from the accelerated radiation tests performed at LANL with a 95% confidence interval and the number of events predicted by the model for each test.

types by hardened types reduces area and time overheads. The work shows that TMR techniques should avoid unnecessary redundancy, while preserving the improvements in fault coverage obtained, as a determining factor to increase reliability.

Chapter 2

Published articles

This chapter includes the collection of articles that supports this thesis. These four publications in high impact JCR ranked journals represents and summarizes the results of the research performed during the last five years.

2.1 Nonintrusive automatic compiler-guided reliability improvement of embedded applications under proton irradiation

Reference
A. Serrano-Cases, Y. Morilla, P. Martín-Holgado, S. Cuenca-Asensi, and A. Martínez-Álvarez "Nonintrusive automatic compiler-guided reliability improvement of embedded applications under proton irradiation", *IEEE Transactions on Nuclear Science* vol. 66 no. 7, pp. 1500–1509 Jul. 2019. DOI: 10.1109/tns.2019.2912323 JCR 2019 impact factor: 1.575

Abstract
A method is presented for automated improvement of embedded application reliability. The compilation process is guided using genetic algorithms and a multiobjective optimization approach (MOOGA). Even though modern compilers are not designed to generate reliable builds, they can be tuned to obtain compilations that improve their reliability, through simultaneous optimization of their fault coverage, execution time, and memory size. Experiments show that relevant reliability improvements can be obtained from an efficient exploration of the compilation solutions space. Fault-injection simulation campaigns are performed to assess our proposal against different benchmarks, and the results are assessed against a real Advanced RISC Machines-based system-on-chip under proton irradiation.

2.2 A compact model to evaluate the effects of high level c++ code hardening in radiation environments

Reference
L. M. Reyneri, A. Serrano-Cases, Y. Morilla, S. Cuenca-Asensi, and A. Martínez-Álvarez "A compact model to evaluate the effects of high level c code hardening in radiation environments", *Electronics* vol. 8 no. 6, p. 653 Jun. 2019. DOI: 10.3390/electronics8060653 JCR 2019 impact factor: 2.412

Abstract
A high-level C++ hardening library is designed for the protection of critical software against the harmful effects of radiation environments that can damage systems. A mathematical and empirical model to predict system behavior in the presence of radiation induced faults is also presented. This model generates a quick evaluation and adjustment of several reliability vs. performance trade-offs, to optimize radiation hardening based on the proposed C++ hardening library. Several simulations and irradiation campaigns with protons and neutrons are used to build the model and to tune it. Finally, the effects of our hardening approach are compared with other hardened and non-hardened approaches.

2.3 Multi-threaded mitigation of radiation-induced soft errors in bare-metal embedded systems

Reference
A. Serrano-Cases, F. Restrepo-Calle, S. Cuenca-Asensi, and A. Martínez-Álvarez "Multi-threaded mitigation of radiation-induced soft errors in bare-metal embedded systems", *Journal of Electronic Testing: Theory and Applications (JETTA)* vol. 36 no. 1, pp. 47–57 2020. DOI: 10.1007/s10836-019-05846-4 JCR 2019 impact factor: 0.596

Abstract
This article presents a software protection technique against radiation-induced faults which is based on a multi-threaded strategy. Data triplication and instructions flow duplication or triplication techniques are used to improve system reliability and thus, ensure a correct system operation. To achieve this objective, a relaxed lockstep model to synchronize the execution of both, redundant threads and variables under protection on different processing units is defined. The evaluation was performed by means of simulated fault injection campaigns in a multi-core ARM system. Results show that despite being considered techniques that imply an evident overhead in memory and instructions (Duplication With Comparison and Re-Execution – DWC-R and Triple Modular Redundancy – TMR), spreading the replicas in different instruction flows not only produce similar results than classic techniques, but also improves the computational and recovery time in presence of soft-errors. In addition, this paper highlights the importance of protecting memory-allocated data, since the instruction flow triplication is not enough to improve the overall system reliability.

2.4 Empirical mathematical model of microprocessor sensitivity and early prediction to proton and neutron radiation-induced soft errors

Reference
A. Serrano-Cases, L. M. Reyneri, Y. Morilla, S. Cuenca-Asensi, and A. Martínez-Álvarez "Empirical mathematical model of microprocessor sensitivity and early prediction to proton and neutron radiation-induced soft errors", *IEEE Transactions on Nuclear Science* vol. 67 no. 7, pp. 1511–1520 2020. DOI: 10.1109/tns.2020.2993637 JCR 2019 impact factor: 1.575

Abstract
A mathematical model is described to predict microprocessor fault tolerance under radiation. The model is empirically trained by combining data from simulated fault injection campaigns, and radiation experiments, both with protons (at the CNA facilities, Seville, Spain) and with neutrons (at the LANSCE Weapons Neutron Research facility at Los Alamos, USA). The sensitivity to soft errors of different blocks of commercial processors is identified to estimate the reliability of a set of programs that had previously been optimized, hardened, or both. The results showed a standard error under 0.1, in the case of the ARM processor, and 0.12, in the case of the MSP430 microcontroller.

Chapter 3

Conclusions

This chapter summarizes the conclusions obtained from the research performed. This chapter is organized as follows: The section 3.1 shows the general conclusions of this research; the section 3.2 summarizes the contributions of the model that provides early predictions of the reliability of applications; the section 3.3 exposes the contributions and improvements in hardening techniques by spreading of replicas in multiple cores using threads without operating system; the section 3.4 shows the main conclusion obtained by applying the blind search optimization process in digital circuits and programs; finally, the section 3.5 shows the new lines of research derived from this thesis work.

3.1 General conclusions

This thesis evaluates new hardening techniques and methods for reliability evaluation using two integrated platforms, the MSP430 microcontroller from Texas Instruments and the Zybo (Zynq-7000) from Xilinx. The platform evaluation was done using a set of applications that were hardened or optimized to obtain reliability measurements. This evaluation was able to identify the main sources of faults of the solutions, which made it possible to determine where to focus the effort to increase the reliability of the platform. Therefore, early reliability predictions were made, using simulated fault injection campaigns as a fast statistical method of obtaining application fault coverage. These simulation results were used to select and verify the behavior of the applications using real radiation campaigns (protons and neutrons) in the particle accelerators in the CNA and LANL. The radiation measurements also contributed to the development of a semi-empirical model. This model can predict the radiation events of a solution using simulation measurements and the sensitivity to radiation of the platform's components under evaluation. Furthermore, this thesis shows that applications reliability can be improved by fine-tuning the compilation process. Compilers can perform several optimizations that alter the mapping of application resources, the sequence of instructions, and hardware resources

used, improving the reliability. Thus, it was necessary to select the appropriate set of optimizations to obtain these reliability increases, performing an efficient search for each solution. Also, it has been demonstrated that reliability can be increased by selectively hardening the resources to be protected. In this sense, the model developed can increase reliability by reducing the replication of resources with a low lifetime and constantly initialized. As a result, the solution can increase the performance and reduce resource usage while fault coverage is maintained. Finally, a new software protection technique using multiple threads has been developed to increase reliability and performance. This technique showed that it is possible to improve the performance without worsening the fault coverage on the hardening techniques based on replication by performing the replicas calculation in parallel flows. The results obtained in this thesis also demonstrated the importance of adjusting the frequency of verifications to maintain system consistency.

3.2 Early predictions of application reliability using a mathematical model

It is common for early estimates of reliability to predict the results of actual radiation measurements using models that approximate the behavior of the devices under evaluation. Thus, it is complicated to obtain precise reliability estimations for devices exposed to radiation. Therefore, this thesis developed a semi-empirical model capable of determining and explaining the slight differences between the expected and measured results. The model showed that several microprocessor components were more sensitive to radiation-induced faults. The first results of the radiation campaigns showed that the most sensitive components matched with those with a smaller exposed area, and were, therefore, less prone to be affected. Additionally, the model proved to be capable of finding which elements are most prone to errors and offers the opportunity to focus hardening efforts on them. The model, also, showed that the low-performance was responsible for the similarities in terms of reliability between the hardened applications (templates) and optimized applications (`MOOGA`). In this context, the model showed that hardening each variable in a program was counterproductive. As a result of the analysis, it was concluded that selecting the appropriate set of elements to be hardened is critical to improving reliability.

3.3 Optimization of the replica computation using multiples instruction fluxes on classical hardening techniques

Since performance is a critical feature for reliable solutions, the use of multiple cores for replication computation allows for faster processing. In this sense,

extending the computation of replication techniques over the different instruction flows of a microprocessor increases the solution's performance. The literature shows several approaches that tried to solve this problem. However, these solutions make use of large amounts of unprotected artifacts to achieve parallel computing that affected the reliability of the solutions. Therefore, this thesis develops a new multithreaded technique that manages to eliminate a large part of these artifacts, establishing a parallel replica computation without needing to add an operating system (*bare-metal*). The multithreaded hardening technique showed that by increasing the performance of the *n-modular* hardening techniques, it is possible to increase the reliability. However, depending on the optimized *n-modular* hardening version (`TMR` or `DWC-R`), the technique can introduce performance penalties that nullify the improvements obtained. On the one hand, the `TMR` techniques present have higher performance because the replicas are calculated simultaneously, and the correction routine has a low penalty. On the other hand, `DWC-R` presents a high reduction in performance due to heavy computing tasks, such as saving the context of the platform and re-execution in case of error.

3.4 Improved reliability through efficient solution space exploration

The search algorithm `MOOGA` developed in this thesis proved to be able to find and improve the reliability of different programs and circuits, reducing them in size and accelerating the computation. By applying the technique for optimizing approximate circuits, the `MOOGA` algorithm was able to find an improved set, with a significantly higher number of solutions than other search methods presented in the literature. On the other hand, by applying the `MOOGA` technique to reliability optimization using compilers, the algorithm can find a set of applications that increase reliability to levels similar to those obtained by hardened applications. Unlike the hardened applications developed in the thesis, these optimized applications increase reliability by improving performance and reducing the use of resources. Thus, the best solutions that `MOOGA` obtained are less likely to suffer a radiation-induced event. These reliability improvements were evaluated using the developed empirical model and accelerated radiation campaigns.

3.5 Future works

In this thesis, different methods have been evaluated to increase reliability and provide new methods for early reliability estimation. One of the investigations enables new ways of hardening using threads in non-operating system environments. In this research, the use of parallelization using threads have been studied to establish a reliable computation by optimizing the *n-modular* techniques, reusing part of the program code. It is considered important to

extend the research to include a study of the optimization of the *n-version* techniques, by parallelizing them using threads. Furthermore, since the set of techniques and methods evaluated generates a considerable amount of information, it would be desirable to use more sophisticated techniques for a more in-depth analysis. In this sense, machine-learning and deep-learning techniques can be used to analyze and to obtain new metrics for early reliability analysis.

In recent years, *deep-learning* has emerged as a new computing paradigm, adopted in most *High-Performance Computing* (HPC) environments. Applications employing this paradigm, use matrix processing units to achieve high performance, maximizing the number of resources used. A new research trend is trying to adapt to this type of computing to operate in critical environments due to its high performance. Since in this research, hardening techniques based on parallelization and automatic hardening have been developed, to extend this technique to operate in these new devices can help to establish a more reliable y performance computation, trying to maximize the number of resources available in the system.

Parte II

Resumen de la Tesis

Capítulo 1

Mitigación de *soft errors* en sistemas empotrados críticos expuestos a radiación

En este primer capítulo se presenta una visión general de la investigación realizada en la tesis con título: "**Fault tolerance in critical aerospace embedded systems: Multi-threaded mitigation, non-intrusive compiler-guided hardening, and early prediction of proton and neutron induced soft errors**". Este capítulo presenta la siguiente organización: en la sección 1.1 se muestra la descripción del problema; en la sección 1.2 se presenta la terminología empleada y se describe el problema en profundidad; en la sección 1.3 se muestran varios enfoques presentados en la bibliografía para superar el problema expuesto; en la sección 1.4 y en la sección 1.5 se presentan los objetivos de la investigación y las hipótesis, respectivamente; la descripción de la metodología utilizada durante el desarrollo de esta tesis, se encuentra en la sección 1.6; una enumeración de articulos científicos publicados en revistas de alto impacto, asi como las comunicaciones a congresos nacionales e internacionales, se presentan en la sección 1.7; por último, los resultados obtenidos durante esta investigación se encuentran en la sección 1.8.

1.1 Motivación y definición del problema

La creciente demanda de procesamiento informático, ha llevado a los fabricantes de microprocesadores a evolucionar desde los procesadores de un solo núcleo, en los años 90, pasando por los procesadores paralelos (computación vectorial y matricial), en los años 2010, hasta las nuevas tendencias emergentes en la computación cuántica [1], [2]. En este contexto, los avances en las últimas décadas, han logrado mejoras de rendimiento en el procesamiento informático, centrándose principalmente en tres estrategias. En la primera, se consigue

las mejoras aumentado la frecuencia a la que operan los microprocesadores, consiguiendo un incremento en la velocidad de procesamiento. Sin embargo, el sobrecalentamiento de los microprocesadores limitó la aplicación de esta estrategia. A continuación, se propuso aumentar el número de unidades de procesamiento por chip, mejorando el rendimiento computacional. Sin embargo, el consumo energético y el sobrecalentamiento, nuevamente, limitaron esta estrategia. Finalmente, los últimos avances en los procesos de manufactura de los semiconductores, de la última década, han dado lugar a importantes reducciones en la tecnología de fabricación. El empleo de esta técnica ha permitido el desarrollo de microprocesadores, funcionales, utilizando diseños con un proceso fotolitográfico de 7 nm. Esta reducción en el proceso de fabricación permite mayores tasas de integración de los componentes, reducción de los niveles de potencia, mejorando el problema de sobrecalentamiento del sistema. Como resultado, un mayor número de estructuras se integran estrechamente en un solo chip [3]. Sin embargo, esta estrategia está cerca de alcanzar, nuevamente, el límite, debido a las restricciones físicas que provocan un descenso en la fiabilidad del microprocesador. En este sentido, las últimas evaluaciones de fiabilidad de los microprocesadores, presentadas en la bibliografía, muestran reducciones significativas en los márgenes de ruido tolerables por los circuitos, provocando una mayor susceptibilidad al ruido ambiental [4], [5]. De hecho, existe una creciente tendencia a mitigar los fallos de los microprocesadores expuestos a la radiación. Varias de estas fuentes de radiación, son comunes en la vida cotidiana, como las fuentes artificiales de radiación presentes en los equipos médicos de radiodiagnóstico, o la radiación natural procedente de los rayos cósmicos. Uno de los efectos de la radiación consiste en la alteración de la funcionalidad de los nuevos diseños de microprocesadores, haciéndolos menos fiables, ya sea, degradando, o causando un comportamiento inesperado. Los efectos más importantes de la radiación son los llamados *Single Events Effect* (SEE) [6]-[8]. Estos efectos tienen especial relevancia en los sistemas críticos, donde los estados inesperados o inoperantes no son aceptables. En este sentido, los sistemas esenciales de toma de decisión autónomos, como los integrados en aeronaves, deben de proporcionar un comportamiento correcto, incluso, en presencia de fallos, como los producidos por la radiación cósmica [9]-[11]. Por este motivo, los sistemas críticos deben implementar varias contramedidas, permitiéndoles detectar, cuándo se ha producido un fallo, y proporcionar formas de mitigarlo o corregirlo. Los procedimientos para implementar estas modificaciones en el diseño son las llamadas técnicas de endurecimiento (*hardening*), que intentan lograr una computación fiable en un diseño poco fiable.

En este contexto, la industria exige cada vez más soluciones fiables, de bajo coste y con un alto rendimiento. Sin embargo, la fiabilidad es un concepto que se opone, habitualmente, a objetivos como coste y rendimiento computacional. Un ejemplo de soluciones, con una alta fiabilidad, son los procesadores personalizados "RadHard", como el LEON-FT, que son soluciones endurecidas, con altos costes de producción. Así mismo, una nueva tendencia de investigación, consiste en habilitar una computación fiable en dispositivos comerciales

de última generación (COTS), debido a su mayor rendimiento computacional, y reducidos costes. La dificultad de esta estrategia radica en reducir la penalización de rendimiento asociada a las técnicas de endurecimiento, al tiempo que se aumenta la fiabilidad. La mitigación de fallos puede aplicarse a diferentes niveles software (instrucciones de ensamblador, acceso a la memoria, métodos o funciones, procesos o hilos), o recursos hardware (núcleos individuales o múltiples). Por consiguiente, esta diversidad, permite una gran variabilidad de soluciones endurecidas, con diferentes grados de fiabilidad. El objetivo de esta tesis, es encontrar, métodos, modelos y soluciones que optimicen el endurecimiento de los sistemas, empleando técnicas de hardware o software.

1.2 Introducción

Esta tesis doctoral, forma parte del proyecto de investigación: *"Evaluación temprana de los efectos de la radiación mediante simulación y virtualización. Estrategias de mitigación en las arquitecturas avanzadas de los procesadores"*. (Ref. ESP2015-68245-C4-3-P, MINECO/FEDER, UE), que tiene como objetivo habilitar un cálculo fiable en componentes COTS.

El principal objetivo de esta tesis, es proporcionar una computación fiable frente a fallos, inducidos por radiación en los dispositivos electrónicos, conocidos como SEE (*Single Event Effects*), y más concretamente, la mitigación de los efectos considerados como transitorios, en estos dispositivos. En este sentido, la tesis analiza el comportamiento de los microprocesadores COTS, bajo los efectos de la radiación ionizante. Para este fin, se utilizan múltiples programas y configuraciones hardware, enfocados a la identificación del origen de los fallos, con objeto de aumentar la fiabilidad del sistema a través de las llamadas técnicas de endurecimiento. Aunque las técnicas de endurecimiento están dirigidas a la obtención de una computación fiable, su uso generalizado es inviable en la mayoría de los casos, debido al alto coste en recursos o rendimiento computacional. Por este motivo, es necesario, el desarrollo de técnicas de endurecimiento parciales y optimizadas, que reduzcan el coste y, al mismo tiempo, preserven la tolerancia a los fallos de la solución. Siendo este último punto, otro objetivo principal de la tesis.

El resto de la sección se distribuye de la siguiente manera: La sección 1.2.1, muestra la taxonomía de los efectos de evento único (SEE), y sus efectos sobre un sistema; la sección 1.2.2, presenta las diferentes rutinas para controlar un evento de radiación; la sección 1.2.3, identifica los métodos más comunes para estimar la fiabilidad de la solución.

1.2.1 Efectos de evento único (SEE)

La mayoría de los sistemas críticos, funcionan en un entorno con mayor o menor tasa de radiación, debido a la actividad humana o a fuentes naturales. Hasta ahora, los antiguos sistemas críticos, eran poco sensibles a los efectos de la radiación. Sin embargo, con objeto de alcanzar los nuevos requisitos de

rendimiento de la industria, se ha producido una reducción del tamaño de los componentes, aumentando de la susceptibilidad a la radiación. En particular, en la bibliografía, se muestra, como la radiación cósmica puede interferir con la atmósfera, generando partículas como: iones pesados, neutrones y protones, siendo éstos, los que interaccionan con los componentes electrónicos, llegando a causar efectos indeseables [12], [13]. Estas partículas implican una décima parte de la radiación natural observada al nivel del mar, causantes de los fallos de funcionamiento de los dispositivos (Figura 1.1). Esta radiación puede in-

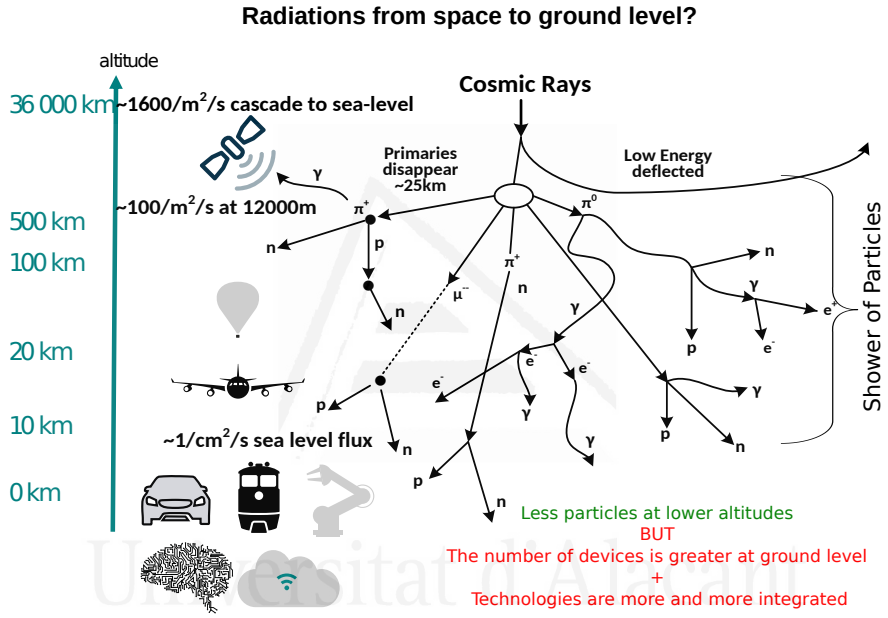


Figura 1.1: Efectos de la radiación cósmica y el flujo de partículas observados a diferentes altitudes. El flujo de partículas se decrece con la reducción de la altitud, mientras que el número de dispositivos susceptibles aumenta.

ducir una carga parasitaria en los transistores, debido al paso de partículas de alta energía a través de ellos, o bien por colisiones de protones/neutrones con el sustrato de los transistores. Un caso recurrente, para ilustrar este comportamiento, son los errores producidos en una memoria `SRAM` de 256 Mb, pudiendo registrar un fallo cada tres semanas, debido a la radiación a nivel del mar.

Tal como se ha descrito, los rayos cósmicos pueden causar ionización en los circuitos, haciendo que los componentes cambien su conductividad, produciendo alteraciones que generan resultados erróneos, incluso la destrucción del componente afectado [14], [15]. De acuerdo con la gravedad del daño (Figura 1.2), los efectos se clasifican como *soft error* o *hard error*. Atendiendo a la persistencia de las alteraciones, también pueden clasificarse como transitorios o permanentes.

Los fallos conocidos como *hard error*, implican la destrucción de parte del

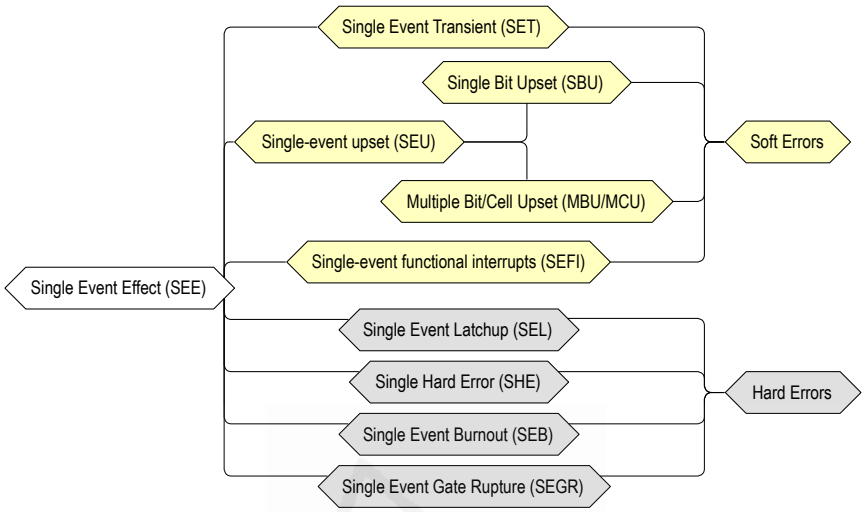


Figura 1.2: Clasificación de efectos de evento único (SEE): Fallos transitorios (amarillo), fallos permanentes (gris).

circuito de forma permanente, debido a una sobrecarga o rotura del sustrato cristalino. Los distintos tipos de *hard error* existentes son:

- *Single Event Latch-up* (SEL) [16], es causado por un cortocircuito en una estructura parasitaria PNPN de los circuitos *Complementary Metal-Oxide - Semiconductor* (CMOS), al atravesar partículas de alta energía los sustratos (típicamente dióxido de silicio SiO_2). Este evento se detecta, generalmente, por un mal funcionamiento del sistema o por un consumo excesivo de energía, causado por el cortocircuito. Por este motivo, es aconsejable evitar el empleo de elementos sensibles, a este tipo de evento, en diseños críticos.

- *Single Event Burnout* (SEB) [17], ocurre al incidir una partícula, altamente ionizante, transmitiendo energía a la región de depleción del transistor. Como resultado, el transistor adquiere un voltaje, más alto, que el voltaje de ruptura de las estructuras parásitas, creando, como consecuencia, un transistor parásito, que amplifica la corriente que entra en el transistor. Un ejemplo de este fenómeno se puede observar en un transistor NMOS, donde los electrones se inyectan en el colector generando un bucle de retroalimentación, causando un sobrecalentamiento local que destruye el transistor.

- *Single-event Gate Rupture* (SEGR) [18], se produce cuando iones de alta energía golpean una puerta activa. Como resultado, la capa aislante de dióxido de silicio se daña por la sobrecarga y el sobrecalentamiento, generando una conexión parcial indeseable.

- *Single-Event Hard Error* (SEHE) [19], se produce por una alteración en la puerta que controla el transistor, causando un cambio permanente en el funcionamiento del transistor. El SEHE afecta a todos los tipos de memoria en los que se bloquea algún bit, siendo imposible invertir el valor almacenado.

Por otro lado, los fallos conocidos como *soft errors*, muestran un comportamiento no destructivo, caracterizado por un cambio en el estado lógico de un nodo (*bit-flip*) en cualquier componente. Normalmente, son celdas de memoria o registros de configuración que cambian el estado interno de sus bits, de 1 a 0 o *viceversa*. En la mayoría de los casos, estos fallos pueden pasar desapercibidos, si afectan a los recursos no utilizados por el sistema. En esos casos, el sistema se recupera por sí mismo, sin ninguna intervención. Un ejemplo de este tipo de fallo se produce cuando un *bit-flip* cambia un recurso no utilizado por un programa. Como consecuencia, el programa terminará con el resultado esperado sin ninguna penalización de rendimiento. Sin embargo, si el fallo afecta un recurso activo (en uso), entonces los efectos pueden diferir ligeramente, dependiendo de lo crítico que sea el recurso. En este último caso, los fallos siguen la siguiente clasificación:

- *Single Event Functional Interrupt* (SEFI) [20], se produce cuando el dispositivo alcanza un estado desconocido, debido a una alteración en los bits de configuración. La detección de este efecto, se realiza por la pérdida de alguna funcionalidad del dispositivo. Una posible solución a este problema, es realizar un reinicio del sistema (*soft reset*). Sin embargo, hay casos, donde los *soft reset* pueden no recuperar completamente el sistema, siendo necesario, un drenado de energía (*hard reset*). Un ejemplo de estos casos, se observa cuando, un sistema de tiempo real operado por interrupciones, sufre un "*bit-flip*", enmascarando la activación de las interrupciones, entrando el dispositivo en un estado inoperante, al quedar comprometida la configuración de la plataforma. En este caso, una solución sencilla para restablecer la configuración por defecto, es activar el subsistema de *watchdog* que desencadena un *hard reset*.

- *Single Event Transient* (SET) [21], se produce por la descarga de un nodo ionizado de un circuito, introduciendo un pulso en el sistema. En este caso, el pulso puede propagarse a otros componentes, convirtiéndose en un fallo más grave (Figura 1.3).

- *Single Even-Upset* (SEU) [22], se produce por el cambio del estado guardado (*bit-flip*), en una unidad de almacenamiento (Figura 1.3). Dependiendo de número de bits afectados se denominan como:

  - *Single Bit-Upset* (SBU) [23], en caso de cambiar sólo un bit.
  - *Multiple Bit-Upset* (MBU) [7], [15], [24], en caso de afectar, en dispositivos con una alta sensibilidad, a multiples bits adyacentes.

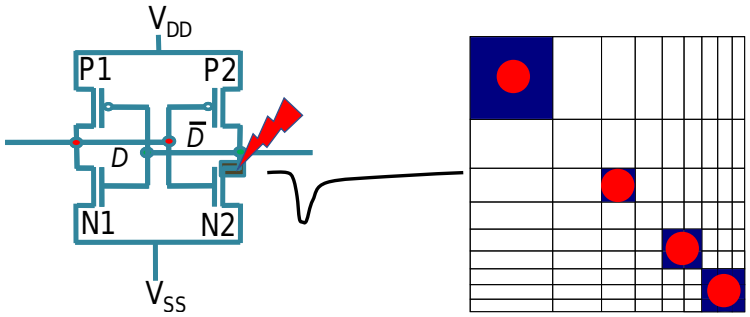


Figura 1.3: Soft error al incidir una partícula en el transistor *N2* de la celda SRAM. Se denomina SET, si el fallo se transmite como un pulso. Se denomina SEU, si el fallo altera el contenido de la celda. Además, se esquematiza, el aumento de la probilidad de sufrir un MBU, debido a la reducción de la tecnología de fabricación.

1.2.2 Tolerancia a los fallos

Por definición, un sistema crítico, debe proporcionar un comportamiento correcto en todo entorno, incluso en los entornos hostiles. En especial, en entornos, donde cualquiera de los efectos, anteriormente mencionados, puede degradar su funcionalidad. Con objeto de evitar situaciones catastróficas, los desarrolladores incluyen funciones de recuperación en sus diseños, generando un sistema tolerante a los fallos. Estas funciones de recuperación, tiene por objeto vigilar la corrección del sistema y activar las rutinas de soporte, tan pronto como se produzca un fallo, para corregirlo, evitando la propagación del mismo. La detección temprana y contención de errores es fundamental, dado que un largo tiempo de detección y recuperación puede aumentar la gravedad del fallo, o, incluso, puede propagarlo a otros componentes del sistema. Para que un sistema tolerante a fallos, sea efectivo, las rutinas de detección y corrección de errores, deben presentar una baja intrusión. Entendiendo por intrusivas, los nuevos elementos añadidos, para obtener un cómputo fiable. Además, deben de tener una alta eficiencia y baja complejidad, que asegure la progresión del sistema, evitando el bloqueo de la tarea bajo protección.

Uno de los objetivos de esta tesis es desarrollar rutinas de tolerancia a fallos aplicables, tanto a sistemas basados en microprocesadores, como a microcontroladores, expuestos a la radiación. Por este motivo, las soluciones propuestas necesitan abarcar alguna de las rutinas básicas de soporte, para generar un sistema tolerante a fallos. Esta rutinas son: detección de errores, diagnóstico de fallos y recuperación (Figura 1.4).

- Rutinas de **detección de errores**, se encargan de identificar un estado anormal en el sistema, realizando:
  - Detección de errores en el flujo de control, cuando las tareas críticas no cumplen con las limitaciones temporales requeridas (*Timing*).

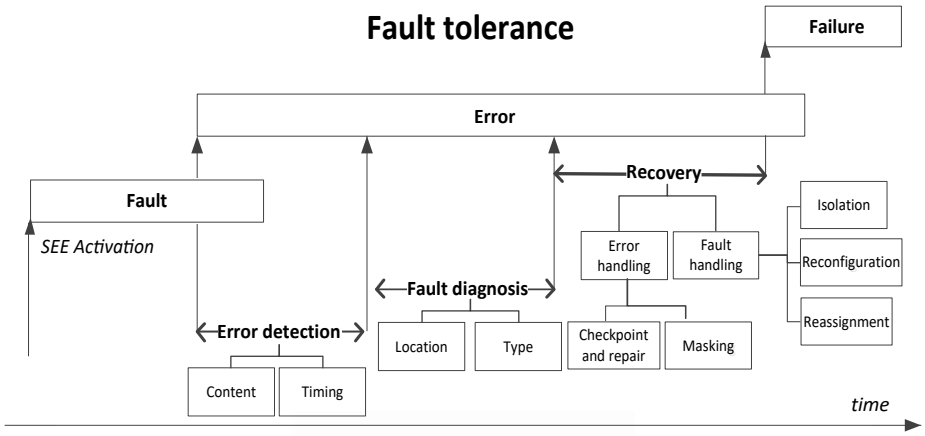


Figura 1.4: Fases de los sistema tolerantes a fallos, desde la activación del SEE hasta el fallo del sistema: *Fault*, *Error* y *Failure*. Además, se indican las rutinas de soporte para la corrección errores que tienen lugar mientras el sistema está en un estado inconsistente: *Error detection*, *Fault diagnosis* y *Recovery*.

  - Detección de errores en el cómputo, mediante el uso de replicación del cálculo (*Content*)

- Rutinas de **diagnóstico de fallos**, identifican la fuente del error (*location*) y su naturaleza (*type*). Este diagnóstico se realiza mediante:
  - Comprobación del estado de las réplicas y búsqueda de discrepancias.
  - Verificación y búsqueda componentes defectuosos del sistema.

- Rutinas de **recuperación**, restauración del funcionamiento normal del sistema mediante corrección. Esta corrección se realiza mediante:
  - *Error handling*, son corregidos restaurando el consenso de la réplica (*Masking*) o volviendo a un estado previamente guardado libre de fallos (*Checkpoint and repair*).
  - *Fault handling*, implican el aislamiento del componente defectuoso (*isolation*) o la reconfiguración o reasignación de recursos del sistema para evitarlo (*reconfiguration* y *reassignment*).

Además de la clasificación antes descrita, las rutinas de soporte o técnicas de endurecimiento, también pueden ser clasificadas según el nivel al que se aplican: hardware, software o híbrido. Independientemente de la clasificación, las rutinas de soporte tienen una alta versatilidad, que permitiendo aplicarlas a diferentes componentes y niveles del sistema a proteger. Tomando como ejemplo, las rutinas de detección o recuperación de contenido, éstas pueden aplicarse tanto en hardware, como en software. Cuando se aplican a nivel de

hardware, la replicación se realiza sobre varias puertas lógicas y *soft-cores*. En cambio, a nivel de software, se hace replicando instrucciones de ensamblador, variables y flujos de instrucciones.

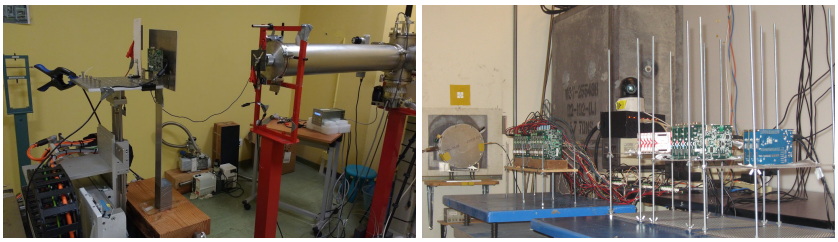
1.2.3 Pruebas de radiación acelerada y validación del endurecimiento.

Como se ha descrito anteriormente, las técnicas de endurecimiento introducen mecanismos para detectar y corregir los fallos que afectan a los componentes del sistema. Sin embargo, para considerar que un componente es fiable o inmune a la radiación (*RadHard*), debe ser sometido a pruebas exhaustivas que verifiquen su comportamiento bajo radiación [25]. Normalmente, estas pruebas de radiación, siguen una normativa estricta con objeto de asegurar la corrección y la relevancia estadística, permitiendo realizar comparaciones entre las diferentes mediciones y pruebas realizadas. Las normativas europeas de obligado cumplimiento son:

- ESCC Basic Specification 22500 (displacement damage)
- ESCC Basic Specification 22900 (total ionizing dose – TID)
- ESCC Basic Specification 23100 (evaluation and procurement)
- ESCC Basic Specification 25100 (single event effects – SEE)

Para realizar esta evaluación, es común el uso de instalaciones donde el acceso esta limitado a unos pocos días, y a menudo, implicando altos costes la realización de las pruebas anteriormente mencionadas [26], [27]. La radiación utilizada es obtenida de aceleradores de partículas, además de otras fuentes de radiación, como el cobalto, simulando el comportamiento del sistema durante la fase de explotación (vida útil). En el desarrollo de la presente tesis, se ha utilizado los aceleradores de partículas conocidos como ciclotrón y tándem, perteneciente al Centro Nacional de Aceleradores – `CNA`, que permiten realizar pruebas de protones y neutrones respectivamente. Asimismo, con objeto de realizar pruebas adicionales, mediante neutrones, se ha empleado el Weapons Neutron Research – `WNR` en el Laboratorio de Neutrones de Los Álamos – `LANL`, que presenta un espectro de neutrones similar al producido en la atmósfera por los rayos cósmicos(Figure **??**). La realización de una evaluación exhaustiva de todas las posibles soluciones en campañas de radiación, no es factible, debido a la gran variabilidad de soluciones, que presentan las técnicas de endurecimiento. Por este motivo, es esencial la selección de un conjunto representativo de soluciones que maximice la información útil, obtenida de la prueba de radiación acelerada. Debido a esta restricción, es fundamental basarse en pruebas más rápidas, aunque menos precisas, para seleccionar los mejores candidatos a ser irradiados.

Las campañas de inyección con simuladores, surgen como una alternativa de bajo coste durante las primeras etapas de evaluación. Estas pruebas, suelen



por la radiación, incrementando la fiabilidad.

Además, en esta tesis se presentan herramientas y métodos, para la validación de los enfoques de endurecimiento desarrollados. En este sentido, se han utilizado pruebas de inyección simulada, así como, pruebas de radiación acelerada, para evaluar las soluciones propuestas, verificando así, el aumento de la fiabilidad. Además, se han combinado los resultados de ambas campañas para desarrollar un modelo matemático, que permita una mayor precisión en la predicción de futuras medidas de fiabilidad, con capacidad de determinar los recursos que más influyen en la fiabilidad de la solución.

1.3 Trabajos relacionados

En la bibliografía se muestran numerosas soluciones dirigidas a la corrección o detección de fallos, en microprocesadores, microcontroladores, circuitos, *soft-cores* en FPGA, y soluciones dirigidas a productos personalizados. Como ya se ha mencionado, esta tesis se centra en el desarrollo y validación de varias técnicas de endurecimiento, dirigidas a los dispositivos COTS. Esta elección, limita el conjunto de técnicas de endurecimiento, reduciéndolo a aquellas técnicas que no requieren del desarrollo de un hardware específico, quedando reducido al conjunto mostrado a continuación:

- Modificación de circuitos o *soft-cores* programados en FPGAs.
- Modificaciones de software destinadas a aumentar la fiabilidad.
- Técnicas híbridas que reconfiguran el hardware para detectar y corregir los fallos.

1.3.1 Técnicas de endurecimiento de hardware

Los enfoques de protección hardware, ganan fiabilidad introduciendo redundancia a nivel de circuito [28], [29]. La redundancia, se puede obtener aplicando, tanto técnicas basadas en la teoría de la información [30], como realizando replicación de los componentes. De esta forma, se consigue proteger y asegurar la corrección de la funcionalidad del circuito.

Las técnicas de protección basadas en la teoría de la información, utilizan una función que describe los datos a proteger. Estas funciones, suelen presentar un alto rendimiento, caracterizándose por presentar un bajo sobrecoste espacial. Tres ejemplos de las funciones antes descritas son: *Decimal Matrix Codes* (DMC) [31], *Error Detection And Correction* (EDAC) [32], [33] y *Error Correction Codes* (ECC) [30]. La aplicación de estas técnicas, disminuye los recursos disponibles para el almacenamiento, al dedicar parte de ellos a mantener los datos de las funciones de resumen. Un ejemplo de función de resumen, son los códigos de detección y corrección de errores de Hamming (Figura 1.6). Esta función resumen, logra sus capacidades de protección al agregar bits de paridad, mezclados con los datos a proteger. Como resultado, la solución es capaz de detectar fallos que afectan a unos pocos bits y, en algunos casos, corregirlos. Sin embargo, el aumento de la superficie expuesta y la reducida capacidad de corrección, aumenta probabilidad de sufrir efectos inducidos por la radiación. En este sentido, las últimas tendencias en este campo, tratan de reducir los sobrecostes de almacenamiento, al tiempo que aumentan la redundancia y las capacidades de corrección.

Las técnicas basadas en hardware redundante, como la técnica *n-modular*, son aplicadas a diferentes estructuras o componentes hardware como: puertas lógicas, componentes completos (memorias) y unidades de procesamiento (microprocesadores *lockstep* y sistemas completos). Estas técnicas están destinadas al desarrollo de componentes específicos y costosos, presentando un

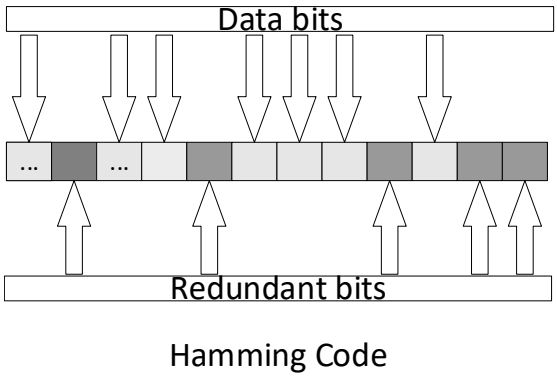


Figura 1.6: Esquema de un código detector y corrector de Hamming. En gris oscuro se representan los bits redundantes, mientras que los bits de datos protegidos se representan en gris claro. Los bits redundantes son asociados a posiciones donde su representación binaria solo tiene un bit activo. La función de paridad protege los bits de datos, utilizando los bits redundantes, asociados a la descomposición binaria de su posición.

elevado uso de recursos y/o consumo energético. En una publicación reciente [25], los autores muestran la evaluación de un procesador *Triple-Lock-Step* (`TLS`), comparándolo con otras soluciones hardware endurecidas, como los procesadores *Dual-Lock-Step* (`DLS`), además de, analizar las diferencias de la solución con los procesadores *RadHard*. Los autores concluyen que el procesador `TLS` tiene un aumento de rendimiento de alrededor de 1000× en comparación con su análogo `DLS`. Adicionalmente, los autores analizan el procesador `DLS`, realizando una comparativa con su versión sin redundancia (no endurecida). Los autores encontraron incrementos significativos en el uso de recursos y energía. Además, se muestran, las disminuciones en el rendimiento y el aumento energético de los procesadores endurecidos (*RadHard*).

Las últimas tendencias de investigación en redundancia hardware, se centran en la reducción de los sobrecostes asociados a este tipo de soluciones, preservando al mismo tiempo los niveles de fiabilidad. Un ejemplo de técnica altamente fiable es la *Triple Modular Redundancy* (`TMR`), sin embargo su implementación conlleva un aumento significativo de los recursos. Debido a ello, hay ocasiones que es inviable replicar la totalidad o la mayoría de los componentes de la solución a proteger [34]. A pesar del alto coste de las soluciones `TMR`, existen sistemas que se plantean adoptar esta técnica, debido a que la precisión no es una característica crítica en ellos. De esta forma, simplificando o aproximando el cálculo de las réplicas, se puede reducir los sobrecostes inherentes de las técnicas `TMR`. En este sentido, el uso de la computación aproximada [35]-[37] aplicada a las técnicas `TMR`, provoca una reducción del esfuerzo computacional, así como de los recursos, manteniendo una alta fiabilidad [38]-[40]. En la Figura 1.7, se muestra cómo se realiza la aproximación de un circuito mediante la

reducción del número de puertas lógicas. El nuevo circuito continúa operando correctamente en la mayoría de los casos. Sin embargo, existe un caso en el que el resultado del circuito aproximado difiere del circuito original. Suponiendo que este circuito reemplaza una de las réplicas de un `TMR`, este seguirá proporcionando el comportamiento esperado con una lógica más reducida. Este nuevo `TMR` aproximado, presenta una reducida vulnerabilidad temporal, donde el resultado esperado del `TMR` pude verse afectado.

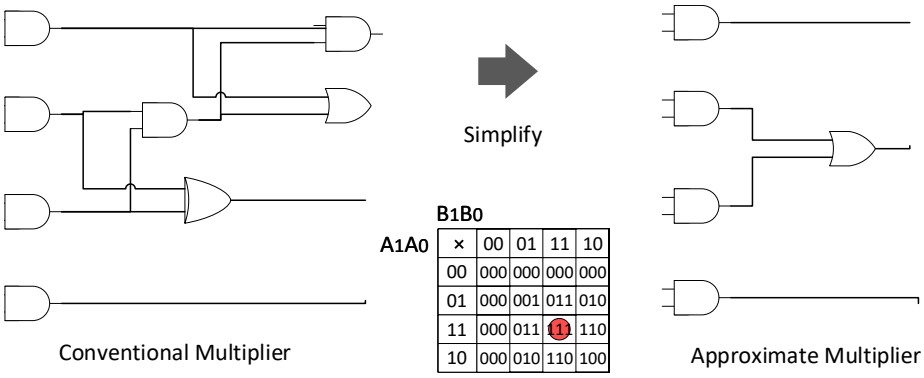


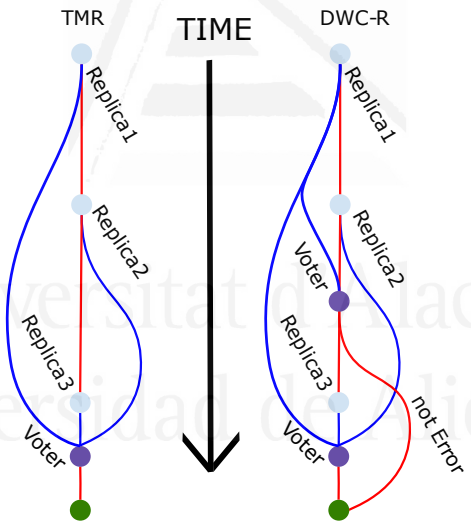
Figura 1.7: Conversión mediante computación aproximada de un multiplicador convencional a uno aproximado, reduciendo el número de puertas lógicas. El multiplicador aproximado disminuye el área utilizada y aumenta el rendimiento. Sin embargo, el circuito simplificado presenta un caso en el que la salida es errónea, en la tabla se destaca el único dicho caso.

1.3.2 Técnicas de endurecimiento de software

Con objeto de mitigar los efectos de la radiación, evitando costosas modificaciones hardware, surgen los enfoques basado en endurecimiento software. Estas técnicas pueden replicar, rápidamente, diferentes estructuras de software, para lograr resiliencia. En este sentido, una técnica de endurecimiento basada en la replicación, puede hacer uso de estructuras, simples, con una sola instrucción, o complejas, como procedimientos o incluso programas completos, para construir las réplicas [41], [42]. Las modificaciones del código, que realizan estas técnicas, hacen que tengan un carácter intrusivo. Dentro de este conjunto de técnicas, una de las más destacables son las técnicas *Software-Implemented Hardware Fault Tolerance* (`SIHFT`). Además, existe otro conjunto de técnicas basadas en software, que son capaces de aumentar la fiabilidad, sin necesidad de introducir elementos nuevos en la solución. Para este fin, las técnicas optimizan el software, aumentando la fiabilidad durante la fase de generación de las aplicaciones, mediante la modificación del mapeado de recursos, la planificación de tareas y la reducción de la exposición a la radiación de los distintos recursos del programa. Una forma de conseguir este objetivo,

es guiar la fase de generación y optimización de los compiladores [43], [44]. Este conjunto de técnicas, al no hacer uso de modificaciones dentro del código, se conocen como no intrusivas.

Dentro del conjunto de las intrusivas, en la bibliografía se encuentran varios enfoques que intentan aumentar la fiabilidad del sistema, introduciendo redundancia basada en *n-modular* [29], [45]. Dos de las aplicaciones más utilizadas de esta técnica para detectar y corregir fallos inducidos por radiación son: *Triple Modular Redundancy* (`TMR`) y *Duplication with Comparison and Re-Execution* (`DWC-R`) [46]. El endurecimiento del software mediante esta metodología, se caracteriza por ser fácilmente exportables a diferentes arquitecturas hardware, así como presentar bajos tiempos de desarrollo y verificación. Sin embargo, estos enfoques tienden a presentar un alto consumo en términos de recursos (replicación espacial), además de un bajo rendimiento computacional (replicación temporal). En publicaciones recientes [47]-[49], los autores prueban y evalúan un enfoque que reduce ambos sobrecostes, habilitando, selectivamente, los recursos no utilizados del procesador destinados a albergar las réplicas del `TMR`.



técnicas *n-modular* mediante el uso de diferentes componentes hardware, añadiendo diversidad. Esta modificación, asegura la detección del componente defectuoso, evitando comprometer los resultados de todas las réplicas. Un ejemplo de *n-version* aplicado a una multiplicación consiste en modificar el cálculo de las réplicas mediante el uso de distintos algoritmos. Uno de esto algoritmos (réplica) puede realizar la multiplicación mediante sumas sucesivas. La otra réplica, por su parte, puede hacer uso de un procesador de señales digitales (`DSP`) para realizar el cálculo. En este contexto, la ejecución ambas réplicas usando procedimientos distintos asegura que el componente defectuoso no se utilice en las dos réplicas. Además, el tiempo de ejecución difiere entre ambas réplicas, evitando cualquier condición de carrera, en el algoritmo, que reproduzca el fallo de la réplica.

Otra forma de lograr la resiliencia del software, es a través de la comprobación del flujo de control. Este tipo de técnicas ayudan a evitar el comportamiento inesperado del programa, aprovechando la información del flujo de instrucciones definida durante la fase de diseño del mismo. La información obtenida del flujo de instrucciones es usada para detectar bifurcaciones anormales, usando anotaciones [50], [51]. Estas anotaciones, etiquetan cada uno de los distintos bloques de procesamiento de un programa, que puede estar formado por múltiples flujos de instrucciones. Las anotaciones se utilizan al inicio de cada bloque, para verificar la firma (anotación) del bloque de cálculo precedente. En caso de que una anotación no coincida con ninguna de las anotaciones esperadas (bloques precedentes), se detecta un comportamiento anómalo (fallo). Sin embargo, estas técnicas basadas en control de flujo, presentan reducciones en el rendimiento, debido a las constantes verificaciones realizadas dentro de cada bloque. Además, presentan un incremento en el uso de recursos, debido al almacenamiento necesario para mantener las anotaciones. Para clarificar el comportamiento de esta técnica, en la Figura 1.9, se muestra un ejemplo donde se utiliza un programa con seis bloques y cinco flujos de instrucciones. Considerando el bloque S4.2, una tarea de verificación inicial, previa al procesamiento del bloque, analiza si el cómputo realizado por el algoritmo es correcto. De esta forma, antes de procesar el bloque, se verifica que uno de los bloques precedentes (S3.2 o S3.4) ha sido procesados. En caso de no cumplirse esta restricción, debido a que el bloque previo sea S2.3, se detecta un error en el flujo del programa. Esta técnica es altamente intrusiva, además de presentar un alto consumo de recursos. Algunos trabajos de investigación recientes, presentan mejoras el rendimiento de esta técnica, utilizando *assertions* para detectar errores que afectan al flujo del programa, reduciendo la cantidad de recursos utilizados y aumentando el rendimiento [52].

Dentro de las técnicas *SIHFT*, existe una tendencia que intenta reducir los sobrecostes presentes en las técnicas *n-modular*, aprovechando recursos y estructuras no utilizados en la mayoría de procesadores de última generación [53], [54]. Estas investigaciones se centran en lograr un aumento en el rendimiento, además de obtener diferentes versiones con distinto grado de fiabilidad (diversidad). Otra tendencia, investiga la mejora en el mantenimiento y la depuración del código generado por las técnicas de redundancia. Por con-

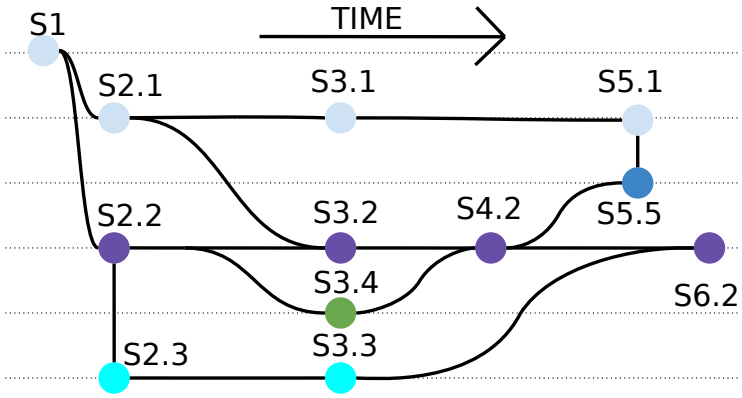


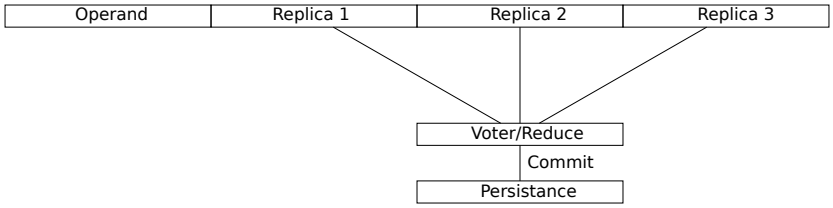
Figura 1.9: Representación del endurecimiento del flujo de instrucciones usando un etiquetado de los bloques del programa. Cada etiqueta define de forma única un bloque de procesamiento, teniendo en cuenta los diferentes bloques y flujos de instrucciones que componen el programa.

siguiente, las nuevas investigaciones se centran en reducir la intrusividad de las técnicas de endurecimiento, aumentando la calidad del código. Para conseguir este fin, estas investigaciones buscan mejorar la fiabilidad usando endurecimientos automáticos basados en anotaciones o sustituciones con una baja intrusividad.

1.3.3 Técnicas híbridas de endurecimiento

Las propuestas de los enfoques híbridos tienen por objeto superar las deficiencias de las estrategias de solo-software y solo-hardware, obteniendo los beneficios de ambas. Estos enfoques ofrecen soluciones con un alto rendimiento, similar a los enfoques de hardware, y un desarrollo rápido y flexible, similar a los enfoques de software. Sin embargo, estas técnicas no son capaces de ofrecer soluciones completamente libres de fallos. En investigaciones recientes, se muestra como una combinación de técnicas de mitigación de fallos software y hardware es capaz de mejorar la fiabilidad de un sistema. Esta técnica mejora el software mediante el uso de técnicas *SIHFT*, además de aplicar una técnica de redundancia hardware de forma selectiva un *soft-core* sintetizado en una `FPGA` [55]. Una reciente tendencia, trata de mejorar la fiabilidad de un sistema, reutilizando, parte de los recursos existentes, de una forma no prevista para ello. Un ejemplo de este tipo de técnicas, es la reconversión de las unidades de procesamiento de tipo vectorial, presentes en la mayoría de los procesadores de última generación. Estas unidades de procesamiento, se reconvierten para acelerar el cómputo de las réplicas en las técnicas *n-version* (Figura 1.10). Tres ejemplos de esta técnica, que permiten realizar el procesamiento de las réplicas en paralelo, se encuentran en: el microprocesador `VLIW` [56], las unidades vectoriales `ARM NEON` [57], y el conjunto de instrucciones `AVX` y `SSE` presente

en la mayoría de los sistemas x86.



las inyecciones de fallos utilizando hardware reales (inyecciones emuladas), presentan modificaciones en la solución a evaluar que no están presentes en un entorno de producción, comprometiendo la exactitud de las estimaciones. Un ejemplo de estos elementos adicionales, son las infraestructuras de depuración reconfiguradas para realizar las inyecciones.

Como se ha mencionado anteriormente, los modelos de los microprocesadores, así como los modelos de fallos, usados en las campañas de inyección de fallos, aproximan al comportamiento del hardware real. Sin embargo, la pérdida de precisión resultante debido a la utilización de un modelo con rendimiento mejorado, puede limitar el análisis de fiabilidad. En estos casos, modelos más complejos, como el *Architectural Vulnerability Factor* (AVF), ayuda a obtener estimaciones más precisas del comportamiento de la solución bajo la radiación [58]. Estos modelos, utilizan los resultados de fiabilidad temprana, así como los datos históricos obtenidos de campañas de radiación acelerada para mejorar la estimación de las pruebas de fiabilidad temprana.

1.4 Objetivos

La progresiva miniaturización del proceso de fabricación, provoca que los microprocesadores sean cada vez más sensibles a la radiación ionizante, siendo cada vez menos fiables. Por este motivo, es necesario implementar rutinas, técnicas y métodos que aumente la fiabilidad, permitiendo un cómputo libre de fallos. En este sentido, el principal objetivo de esta tesis, es mitigar y mejorar la tolerancia a los fallos inducidos por la radiación, en diferentes sistemas basados en microprocesadores. De manera más específica, esta tesis pretende mejorar la fiabilidad, adaptando o proporcionando nuevas técnicas SIHFT, en microprocesadores de última generación (COTS). Para este fin, se han utilizado recursos que implican diferentes flujos de instrucciones (hilos o procesos), que se ejecutan en uno o más núcleos de un microprocesador. Debido a ello, esta tesis desarrolla y evalúa nuevos esquemas paralelos con baja intrusión, siendo esta última característica obligatoria para evitar incrementar la sensibilidad de la solución a los fallos. Además, esta tesis pretende, desarrollar un modelo de predicción de la sensibilidad a la radiación de una aplicación y, por lo tanto, que ayude a evaluar tanto las mejoras, como la eficacia de las técnicas desarrolladas. El modelo desarrollado, podrá seleccionar los mejores candidatos a ser evaluados mediante pruebas de radiación, en instalaciones de radiación acelerada, donde únicamente se permite al usuario evaluar un número limitado de soluciones. Por lo tanto, el modelo necesita realizar predicciones precisas del comportamiento de las soluciones bajo radiación. En este sentido, será preciso un estudio comparativo entre los datos del modelo y los datos de los experimentos de radiación, con objeto de corregir y mejorar las predicciones del modelo.

Con objeto de cumplir con los objetivos generales, se establecen los siguientes objetivos y subobjetivos:

O1: Revisar las soluciones, técnicas y métricas más avanzadas de tolerancia a fallos, para evaluar el rendimiento, la ocupación de recursos, la cobertura frente a fallos y la fiabilidad de los dispositivos electrónicos sometidos a la radiación ionizante. Prestando especial atención a los siguientes puntos:

1. Estudiar las nociones básicas de fiabilidad en los sistemas expuestos a radiación ionizante.
2. Estudiar las técnicas más comunes de tolerancia a fallos en software, hardware y sistemas híbridos (hardware/software).
3. Identificar las métricas de fiabilidad más relevantes (AVF, MWTF), para analizar las propuestas de estado del arte, analizando las propuestas presentadas en esta tesis.

O2: Evaluar la idoneidad y viabilidad de los simuladores de microprocesadores, de última generación, para realizar inyecciones de fallos en programas complejos (múltiples hilos, múltiples procesos o sistemas operativos). Para este fin se va a:

1 Diseñar o modificar una herramienta de inyección de fallos.

2 Evaluar diferentes arquitecturas relevantes y sus simuladores asociados, prestando especial atención a las siguientes arquitecturas:

- **ARM**: Simuladores multinúcleo de 32 bits con precisión de instrucción: Xilinx-QEMU, Gem5, OVPsim, Simics, fast-models
- **MSP430**: Simulador de microcontroladores de 16 bits con precisión de ciclo: SHE, NAKEN

3 Evaluar la idoneidad de los diferentes Lenguajes de Descripción de Alto Nivel (HDL) y sus simuladores asociados, teniendo en cuenta la siguiente alternativa:

- ***Verilog***: El simulador ABC y la biblioteca de puertas *nandgate* de 45nm.

O3: Desarrollar una herramienta automática para explorar eficazmente un extenso espacio de soluciones de endurecimiento, con capacidad de:

1 Usar un algoritmo meta-heurístico, basado en *machine learning*, para realizar una búsqueda ciega y no guiada, en un enorme extenso no lineal multidimensional de soluciones.

2 Optimizar el algoritmo de búsqueda para tener en cuenta una optimización basada en múltiples objetivos.

3 Acelerar la convergencia del algoritmo de búsqueda.

O4: Analizar el estado actual de las técnicas de mitigación de fallos multinivel, con especial énfasis en los sistemas multinúcleo y multihilo. Atendiendo a:

1 Investigar el estado de las técnicas de mitigación de fallos multihilo en los siguientes ámbitos:

- Técnicas en el ámbito de los Sistemas Operativos utilizando `APIs` como *pThreads*, *MPI* o *OpenMP*
- Técnicas en el ámbito de *RTOS*
- Técnicas en el ámbito de *bare-metal*

2 Proponer una nueva estrategia de mitigación en:

- Multihilo / Multinúcleo
- Protección del código del programa

O5: Estimar la contribución de cada recurso del microprocesador a la fiabilidad general del sistema, identificando a qué recursos se puede acceder o medir directamente, para obtener su tolerancia a los fallos. Para este fin se va a:

1 Identificar y ordenar los recursos según su criticidad.

2 Estimar mediante un modelo semi-empírico la sensibilidad de cada recurso.

1.5 Hipótesis

Una metodología para incrementar la fiabilidad de los componentes electrónicos, a los fallos inducidos por radiación, emplea técnicas de redundancia. Dentro de este conjunto, las técnicas basadas en *n-modular*, presentan altas tasas de detección y corrección de fallos inducidas por la radiación. Sin embargo, estas técnicas presentan un descenso en el rendimiento de las soluciones, cuando las réplicas se calculan de forma secuencial, o incrementos de área debido a los recursos replicados. En este sentido, se considera que la optimización de las técnicas *n-modular*, puede incrementar la fiabilidad. Para lograr esta optimización, se plantea aumentar el rendimiento y reducir el área expuesta, manteniendo al mismo tiempo, las mejoras de la cobertura frente a fallos, obtenida por las técnicas originales. En este contexto, se plantea como hipótesis que:

`H1`**:** Las técnicas basadas en el aprendizaje automático, deberían explorar eficientemente el espacio de las soluciones mejoradas. Para este fin, la técnica tendría que:

1. Realizar una exploración inteligente, encontrando soluciones con mejores compromisos entre área, rendimiento y cobertura de fallos.
2. Considerar un espacio de soluciones multidimensionales basado en los objetivos a mejorar.
3. Presentar las soluciones en términos de grado de cumplimiento, dentro de los objetivos de interés (eficiencia de Pareto)

`H2`**:** Las técnicas hardware `TMR`, deberían presentar reducciones en la exposición del área. Para este fin:

1. La aproximación en el cómputo de las réplicas, debería reducir los tamaños del circuito.
2. Las reducciones de la complejidad, debido al cálculo de la réplica aproximada, no debería afectar al cálculo del algoritmo.
3. La optimización de los circuitos de réplica, no debe disminuir la tasa de confiabilidad del `TMR`.

`H3`**:** Los compiladores podrían producir, durante las etapas de optimización y de asignación de recursos, mejoras rentables de la fiabilidad, como efecto secundario. Debido a que:

1. El gran número de optimizaciones de los compiladores, deberían producir un conjunto de aplicaciones de similar tamaño, sin modificar el código fuente.
2. Es altamente probable, la existencia de aplicaciones con una mejor cobertura de fallos, en comparación con la versión no optimizada de referencia.

H4: El uso de recursos infrautilizados, presentes en la mayoría de los microprocesadores de última generación, podría acelerar el cómputo de las réplicas en técnicas *n-modular*. En base a esta hipótesis:

1. Se debería lograr una mejora en el rendimiento de las técnicas *n-modular*, mediante la difusión de las réplicas en múltiples flujos de instrucciones.
2. El código para la replicación, debe ser mínimamente intrusivo, y además, reducir la introducción de nuevos puntos de fallo.

H5: Es posible que las partes constitutivas del dispositivo que se está evaluando, no compartan la misma sensibilidad a la radiación. Debido a ello:

1. La implementación de cada componente, puede diferir ligeramente y, por lo tanto, es posible que cada bit que compone la plataforma, no comparta la misma susceptibilidad a la radiación.
2. Cada área activa de los componentes, así como su tiempo de exposición, deberían contribuir a la fiabilidad de la plataforma, de manera diferente.

1.6 Metodología

La metodología utilizada en esta tesis es una combinación de métodos de investigación deductivos, experimentales y cuantitativos. En este sentido, la primera tarea fue analizar la bibliografía, para identificar los problemas más relevantes, en microprocesadores expuestos a la radiación ionizante. Además, se identificó formas novedosas de incrementar la tolerancia a los fallos, en sistemas críticos. A continuación, se estableció un conjunto de objetivos e hipótesis, centrándose en formas de obtener predicciones tempranas de la fiabilidad, así como nuevos métodos para mitigar los fallos inducidos por la radiación ionizante. Posteriormente, se desarrollaron las propuestas de algoritmos y métodos, enfocados a la mitigación de fallos inducidos por la radiación, validándolos, mediante campañas de inyección de fallos simuladas y test de radiación acelerados. Por último, los resultados, de ambas campañas, se utilizan para crear un análisis semi-empírico, utilizando modelos estadísticos, generalizando los resultados de la investigación, para predecir el comportamiento de nuevas soluciones endurecidas bajo los efectos de la radiación.

A continuación, se muestra una perspectiva general de las estrategias, métodos y algoritmos más relevantes utilizados:

`M1` Un método busqueda basado en, una optimización multiobjetivo guiada por algoritmos genéticos, para encontrar soluciones con variaciones en los distintos objetivos bajo evaluacion: fiabilidad, rendimiento y uso de recursos. Así, este algoritmos busca:

  `1` Mejorar la fiabilidad, generando aplicaciones optimizadas, utilizando el compilador.
  `2` Reducir el área de las técnicas de endurecimiento `TMR`, utilizando la computación aproximada.

`M2` Estrategia de múltiples flujos de instrucciones, aplicadas a aproximaciones *n-modular*, para reducir los sobrecostes inducidos por el cálculo secuencial de las réplicas.

`M3` Predicción de la fiabilidad, utilizando tasas de fallo simulados y medidas de radiación, para calcular la sensibilidad de los recursos de la plataforma. Para ese fin, se utilizará:

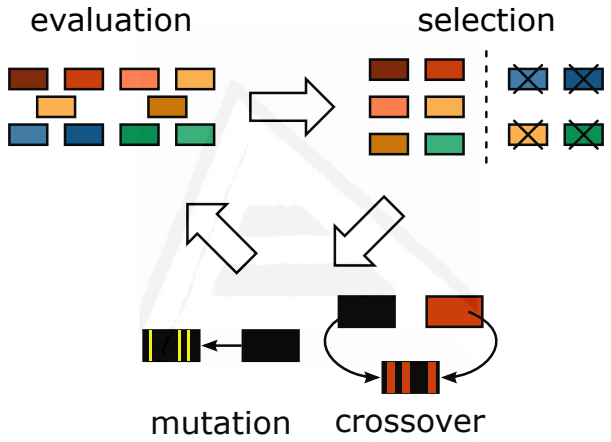
  `1` Una técnica de endurecimiento software, de baja intrusión, para aplicar las técnicas *n-modular* de forma rápida y fácil.

A continuación, se detallan y discuten las metodologías que se han seguido:

1.6.1 `M1` Proceso de optimización multiobjetivo guiado por algoritmos genéticos

Es común que las técnicas de endurecimiento cubran diferentes escenarios, utilizando diferentes parámetros o configuraciones. Explorar todas las confi-

guraciones posibles que aumenten la fiabilidad, puede requerir un importante esfuerzo computacional, que no siempre es factible. En este contexto, los *Genetic Algorithms* (GA), son una solución para la exploración eficiente y automática del espacio de soluciones. Los GA pertenecen al conjunto de algoritmos llamados *Evolutionary Algorithms* (EAs), inspirados en la evolución biológica. Estos algoritmos, codifican una de las soluciones del problema, como los genes de un individuo perteneciente a una población (conjunto de soluciones). Cada individuo de la población, se evalúa según el grado de cumplimiento del objetivo del problema. A continuación, se clasifica el individuo (selección), para comprobar si tiene las propiedades para permanecer dentro del bucle evolutivo (Figura 1.11).



En el caso de los problemas de fiabilidad, no sólo se busca un sistema que aumente la cobertura de fallos, sino que también, se busca reducir el área expuesta (tamaño), así como aumentar el rendimiento computacional del problema. Por este motivo, el problema de aumentar la tolerancia a los fallos (fiabilidad), está lejos de ser mono-objetivo. En este contexto, la selección de una función compuesta, que evalúe justamente la población, no siempre es factible, porque los objetivos pueden estar no relacionados o ser opuestos entre sí. Para poder realizar una evaluación justa, debe considerarse que las soluciones se encuentran dentro de un espacio multidimensional, definido por los objetivos que se evalúan. Como consecuencia, la mejora de un solo objetivo, puede conllevar un empeoramiento de los demás. En este sentido, la comparación de las soluciones usando algoritmos basados en el concepto de eficiencia de Pareto, permite evaluar el equilibrio de los diferentes objetivos seleccionados. Estos algoritmos, son útiles para crear un conjunto de soluciones con cualidades mejoradas, llamado frontera de Pareto. Estas soluciones se caracterizan por no poder mejorar ningún objetivo sin empeorar los restantes (Figura 1.12).

La combinación de ambos algoritmos (Algoritmos Genéticos y Algoritmos de Optimización Multiobjetivo), en el llamado algoritmo `MOOGA`, permite realizar una búsqueda ciega en un espacio de soluciones multidimensional (Figura 1.13). Este algoritmo, se configura usando tres grupos de parámetros, establecidos en la configuración inicial del algoritmo: población inicial, parámetros del `GA` y la configuración de parada, definida por el número de iteraciones u otros criterios de convergencia. Estos parámetros, dependientes del problema evaluado, son responsables de la aceleración y la convergencia del algoritmo de búsqueda. Si un algoritmo presenta una población inicial baja, es necesario incrementar la tasa de mutación del `GA`, con el fin de aumentar rápidamente la diversidad de la población, evitando la endogamia. Sin embargo, esta configuración mantenida durante un número alto de iteraciones (generaciones), no garantiza la convergencia del algoritmo de búsqueda. Con objeto de solucionar este problema, el algoritmo `MOOGA` está compuesto por dos etapas. La primera etapa trata de maximizar los individuos únicos en la población, mientras que la segunda, reduce la tasa de mutación, permitiendo la convergencia del algoritmo.

`M1-1` Mejorar la fiabilidad optimizando la selección de parámetros del compilador

Los compiladores pueden producir distintos programas que se comporten de forma idéntica, introduciendo pequeños cambios en la planificación de las instrucciones, o reemplazando alguna de las instrucciones con otras de mayor rendimiento. Como resultado, es posible lograr el mismo cómputo, usando flujos de procesamiento similares. Estas modificaciones pueden dar lugar a programas con un rendimiento mejorado y un alto uso de recursos, o aplicaciones con un rendimiento reducido y una alta disponibilidad de recursos. Conseguir una mejorar en el rendimiento o producir programas con un uso reducido de recursos, es relativamente sencillo de obtener. Sin embargo, los

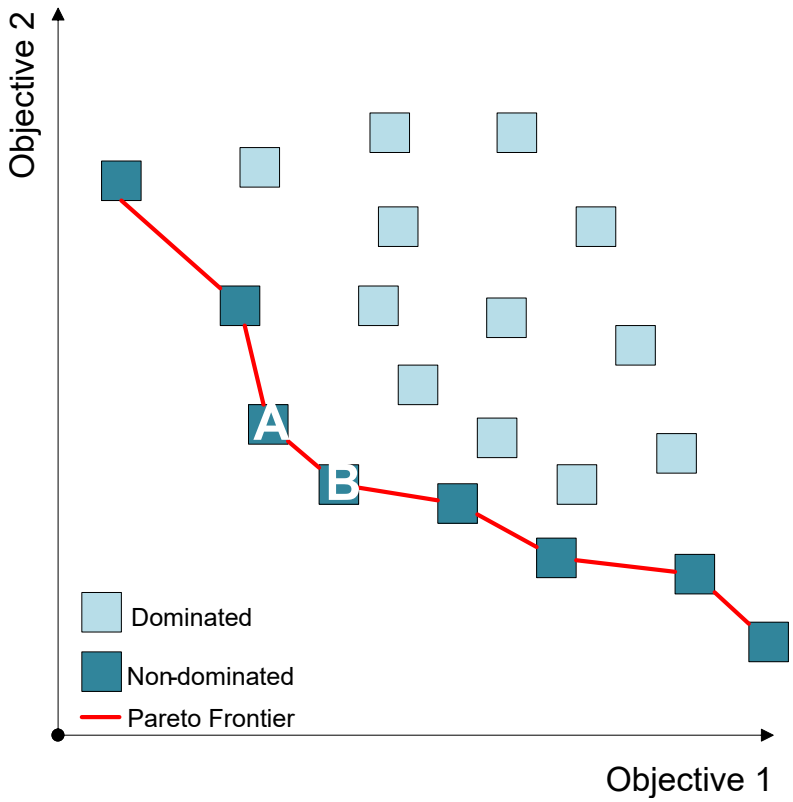


Figura 1.12: Representación de soluciones en un espacio multiobjetivo basada en el concepto de frontera de Pareto. La gráfica muestra dos objetivos bajo evaluación, donde las mejores soluciones, no dominadas, conforman la frontera de Pareto (línea roja). Dentro de frontera de Pareto se destacan los objetivos mejor balanceados (A y B).

compiladores de proposito general no ofrecen una forma inmediata y sencilla de producir transformaciones de código fiables. En este sentido, el uso opciones del compilador que hacen un uso extensivo de los recursos del sistema, produce incrementos en el rendimiento, pero puede conducir a una exposición de recursos significativamente mayor a la radiación. En cambio, el uso de opciones destinadas a la reducción del uso de recursos, aumenta el tiempo de cómputo, aumentando el tiempo de exposición al medio ambiente nocivo. La bibliografía muestra trabajos donde sólo ajustando el proceso de generación de un programa, por parte del compilador, es posible lograr diferentes niveles de fiabilidad [44]. Estos trabajos muestran que es posible mejorar el rendimiento y la fiabilidad, ajustando la compilación, mediante el uso de las opciones de optimización más comunes proporcionadas por los compiladores. Sin embargo, este conjunto de optimizaciones, comúnmente utilizado por los desarrolla-

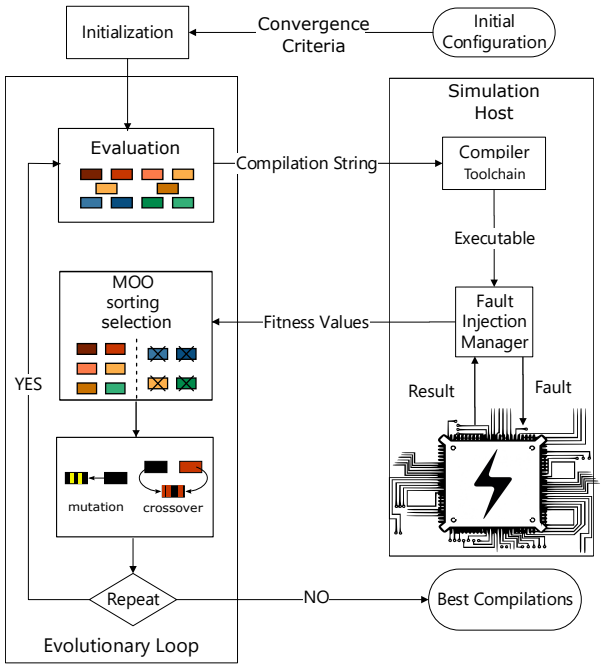


Figura 1.13: Combinación de algoritmos genéticos y algoritmos de optimización multiobjetivo en el llamado algoritmo MOOGA. El diagrama muestra un ejemplo centrado en la optimización de aplicaciones utilizando el compilador y una herramienta de test (gestor de inyección de fallos) a cargo de la evaluación y del reporte de los valores de fitness.

dores, desencadena una gama más amplia de optimizaciones (Figura 1.14). De hecho, los compiladores presentan un extenso conjunto de optimizaciones que, en algunos casos, interfieren entre sí. Debido al gran número de opciones, no es posible conocer los efectos de todas ellas, siendo extremadamente costoso, desde el punto de vista computacional, evaluar cada una de las posibles combinaciones. Además, es posible que el parámetro del compilador responsable de los aumentos de fiabilidad dependa de la aplicación. Para dar solución a este problema, MOOGA realizar una búsqueda ciega de todas las optimizaciones del compilador, buscando las optimizaciones que mejoran en conjunto el tamaño, el rendimiento y la cobertura de fallos, mejorando la fiabilidad del sistema.

Para poder realizar la evaluación de las soluciones, MOOGA utiliza simuladores como medio de obtener métricas de calidad, por ejemplo, el rendimiento computacional de un programa. Sin embargo, para estimar métricas, como la cobertura de fallos, es necesario realizar campañas de inyecciones de fallos simuladas. Los gestor de inyección de fallos (FIM), obtienen el comportamiento estadístico de la plataforma bajo radiación (Figura 1.13). Para conseguir este fin, se realiza una inyección fuertemente aleatoria, evitando comprometer los

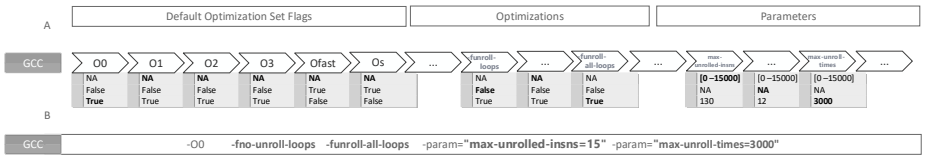


Figura 1.14: Cadena de optimizaciones ofrecidas por el compilador para mejorar la generación de código y la representación de una solución codificada como los genes de un individuo durante la optimización de MOOGA.

resultados obtenidos. Después de evaluar cada inyección, FIM clasifica el resultado obtenido atendiendo al efecto causado por el fallo en el sistema. En caso de tener un sistema basado en microprocesador ejecutando una determinado programa, el efecto del fallo se modela siguiendo las siguiente clasificación: unACE, SDC y HANG. Si el programa logra completar el cálculo correctamente, entonces la inyección se etiqueta como unACE. En el caso de obtener un resultado erróneo, el fallo se etiqueta como Silent Data Corruption (SDC). Finalmente, si la plataforma se vuelve inoperante o no cumple con las limitaciones de tiempo (tiempo crítico), el cálculo se marca como HANG. Estas capampañas, hacen uso de simuladores convencionales modificados para ampliar su funcionalidad, mediante el uso de *plugins* o modificando su código fuente. En estos casos, las nuevas capacidades de inyección aportadas a los simuladores, deben considerar la inclusión de un comportamiento no intrusivo y determinístico. Con el comportamiento no intrusivo, se pretende que la simulación permanezca inalterada durante la activación del fallo en la plataforma. Mientras que, con el comportamiento determinístico, se pretende que la evaluación de cada configuración probada sea reproducible, incluso considerando una arquitectura con un comportamiento especulativo.

M1−2 Reducir los recursos de las técnicas TMR mediante el uso de la computación aproximada

De forma similar a los compiladores que optimizan las aplicaciones para mejorar la fiabilidad, los circuitos pueden optimizarse, aproximando su funcionalidad para obtener un mejor rendimiento o una reducción de su ocupación. Las aproximaciones de los circuitos, pueden realizarse eliminando algunas puertas lógicas, que definen el circuito, y sustituirlas por otras más sencillas y de mayor rendimiento, logrando una reducción de la complejidad del mismo. Como resultado, estas aproximaciones pueden presentar una reducción de la precisión o, incluso, proporcionar resultados erróneos, en algunos casos. La evaluación de todas las posibles aproximaciones y combinaciones de circuitos, para obtener el mejor circuito aproximado, no es factible. Esto es debido al gran número de puertas lógicas que puede presentar un circuito, además de los distintos grados de complejidad de cada puerta. Para solucionar este problema, el método de búsqueda MOOGA, puede seleccionar los circuitos más eficientes

y, aunque no sean capaces de producir un comportamiento totalmente fiable, los circuitos aproximados pueden combinarse entre ellos, para producir una solución fiable (TMRs aproximados). Debido a la existencia de casos donde el resultado de las réplicas es erróneo, el circuito corrector (votador), se activa con mayor frecuencia. Por ello, la aproximación de las réplicas del TMR, deberían producir los mismos resultados que el circuito original, excluyendo algún caso. De esta manera, se limita el período de vulnerabilidad del TMR. Esta característica, es explotada por el algoritmo MOOGA, para acotar, aún más, el enorme espacio de posibles soluciones, acelerando la convergencia del algoritmo.

1.6.2 M2 Estrategia *bare-metal* de paralelización de flujos de instrucciones en técnicas *n-modular*

Los microprocesadores de última generación buscan aumentar el rendimiento de las aplicaciones software, utilizando esquemas de programación paralela como *Asymmetric Multiprocessing* (AMP) y *Symmetric Multiprocessing* (SMP). Estos esquemas, hacen uso de varios flujos de instrucciones para acelerar las tareas de procesamiento, cuando la computación sobre los datos a procesar es independiente entre los flujos. Un ejemplo de independencia de datos se puede encontrar en las estrategias *n-modular*, donde se realiza el mismo cálculo sobre un conjunto diferente de datos (réplicas). Debido a ello, estas técnicas pueden aumentar su rendimiento distribuyendo el cálculo de las réplicas sobre diferentes flujos de instrucciones (Figura 1.15).

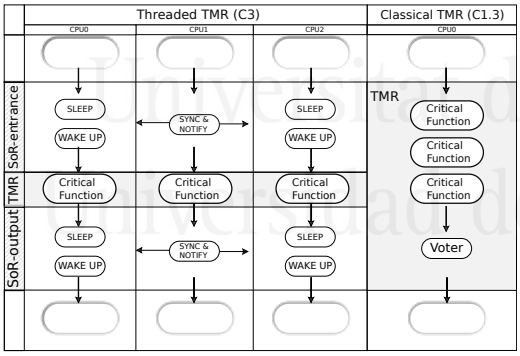


Figura 1.15: Esquemas de técnicas TMR y DWC−R, usando una versión paralelizada con hilos y su análoga secuencial. Las versiones paralelas se muestran el bloque de sincronización y corrección rodeando la función crítica a proteger. Mientras que las versiones secuenciales presentan un votador para la obtención del resultado esperado, una vez realizado el cálculo de las réplicas.

En la bibliografía se encuentra varios enfoques donde la difusión de las réplicas, en distintos flujos de instrucciones, se aplicó con éxito. En una reciente publicación, se muestra el uso de bibliotecas de alto nivel para la difusión de las réplicas de las técnicas *n-modular*, siendo necesario el uso de recursos

como el sistema operativo para su funcionamiento [59]. Una de las conclusiones de este trabajo, destaca que la fiabilidad se reduce debido al incremento de artefactos no protegidos para habilitar la computación paralela (el sistema operativo y las bibliotecas de hilos). En otra publicación, se muestra que la difusión de las réplicas puede conducir a la degradación del rendimiento, incumpliendo los requisitos y limitaciones temporales de la solución [60]. Para evitar estos problemas, los autores desarrollaron una estrategia que modifica la planificación y la granularidad a la que se aplica la redundancia, disminuyendo las penalizaciones del rendimiento. De estas publicaciones se concluye que, el uso de múltiples flujos de instrucciones incrementa los sobrecostes de la solución (tamaño y tiempo), debido a los artefactos encargados de la distribución y sincronización de los datos. Por lo tanto, una solución de endurecimiento software paralelizada, que pretenda aumentar la fiabilidad, debe limitar y minimizar la inclusión de estos artefactos desprotegidos. En este contexto, las soluciones sin sistema operativo (*bare-metal*), permiten explotar optimizaciones de bajo nivel, así como realizar rutinas de sincronización con un alto rendimiento, evitando la inclusión de grandes piezas de software sin protección. En este contexto, la difusión de réplicas en las técnicas *n-modular* usando diferentes flujos de instrucciones, transforman la solución en un *lockstep* software. Esta solución, al contrario que su análoga hardware, permite ajustar, más fácilmente, la frecuencia de comprobación y la cantidad de recursos protegidos en la solución, permitiendo mejorar el rendimiento y, al mismo tiempo, mantener la cobertura de fallos de la solución. Además, esta técnica permite un procesamiento de las réplicas de forma independiente en plataformas con recursos compartidos, usando zonas de memoria aisladas (particiones), evitando de este modo colisiones en el uso de recursos. Debido a esta restricción, las rutinas de verificación se realizarán una vez finalizado el cálculo de todas las réplicas, accediendo en modo de sólo lectura a los resultados de las otras réplicas. De esta forma, la solución sólo incluye estos puntos de sincronización como único artefacto de código que lo diferencia con programa el programa original, reduciendo así introducción de nuevos puntos de fallos. (Figura 1.15)

1.6.3 `M3` Modelo matemático para la predicción de la tasa de *soft errors*

Los sistemas críticos, expuestos a la radiación ionizante, deben ser validados mediante costosas pruebas de radiación. De esta manera, se asegura frente a fallos inducidos por la radiación, el correcto funcionamiento y la alta fiabilidad del sistema. Debido a ello, la obtención de medidas tempranas de la fiabilidad, altamente precisas, es imprescindible para conocer la eficacia de una solución endurecida, antes de ser evaluada mediante campañas de radiación acelerada. En este contexto, los simuladores son capaces de ofrecer medidas tempranas de la fiabilidad en los microprocesadores. Sin embargo, aun siendo de alta precisión, los simuladores modelan y aproximan el funcionamiento del hardware real. En consecuencia, la precisión del modelo afecta a los resultados medidos, llegando a diferir levemente con los resultados obtenidos al utilizar

hardware real. Además, el uso de modelos de alta precisión limita el número de soluciones a evaluar, debido a las penalizaciones en términos de rendimiento de la simulación. Como consecuencia, para realizar un estudio eficaz del comportamiento del hardware bajo la radiación, es obligatorio encontrar el mejor equilibrio entre rendimiento y precisión. En este contexto, reducir ligeramente la precisión del modelo y aproximar los efectos de los fallos puede aumentar el rendimiento del modelo, permitiendo una evaluación estadística del hardware. Un ejemplo de reducción de la precisión para acelerar las simulaciones, es aproximar los efectos de la radiación (*Single Events Upset*) usando el modelo de *bit-flips*. Este modelo simula cuándo una partícula incide sobre un transistor, generando una carga parásita que puede cambiar el estado de los bits en la memoria. Sin embargo, el modelo hace una suposición no realista, considerar todos los eventos igualmente probables, con independientemente de la tecnología subyacente. Esta suposición puede conducir a evaluaciones erróneas de la fiabilidad, debido a que cada componente de la plataforma puede ser de tecnología distinta o utilizar una combinación lógica diferente. Al presentar cada componente una sensibilidad distinta a la radiación, es necesario evaluar la fiabilidad de los componentes la plataforma de forma aislada. Por consiguiente, para ofrecer una medida de la fiabilidad, es necesario desarrollar un modelo semi-empírico, que utilice las distintas estimaciones de la sensibilidad a fallos obtenidas mediante simulación, además de modularlas con la sensibilidad de cada componente, obtenida de experimentos previos de radiación (pruebas históricas de radiación). Este modelo se basa en una regresión lineal múltiple (ecuación 1.1), donde los estímulos (X_i) corresponden con las medidas de inyección de fallas simuladas de cada componente, la sensibilidad a la radiación de los componentes de la plataforma son (β_i), y los términos de ajuste (ϵ_1 y ϵ_2) corresponde con la estimación de elementos no modelados.

$$\mathbf{Y} = \sum_i \beta_i \mathbf{X} + \epsilon_1 + \epsilon_2 \tag{1.1}$$

`M3-1` Endurecimiento transparente de código C++ empleando programación genérica

El endurecimiento del código, suele implicar la introducción de nuevo bloques de software (artefactos) en los programas, para permitir un cálculo fiable. Estos artefactos son altamente intrusivos y, en algunos casos, difícil de entender y depurar. El código destinado a proteger el software, de los efectos de la radiación, normalmente, aumenta la complejidad del desarrollo y el tiempo de depuración, incrementando los costes de producción.

Hacer algoritmos altamente reutilizables, para un gran número de problemas, aplicando sólo unas pocas modificaciones, para adaptarse a las nuevas especificaciones, es posible utilizando el paradigma de programación genérica (*templates*). Este paradigma permite a los desarrolladores, centrarse en la generación de algoritmos, generalizando la solución y abstrayéndose del tipo de datos utilizados. Este tipo de herramientas obtiene la estructura básí-

ca del algoritmo, y, una vez, verificada la funcionalidad, es posible exportarlo a otros elementos más complejos. Un ejemplo de aplicación, se encuentra en los *frameworks* de endurecimiento basados en plantillas, destinados a evitar la introducción de defectos, en el programa a proteger, permitiendo un desarrollo más rápido. Además, otra característica que introducen, es la facilidad del manteniendo, debido a la similitud de las partes del programa con el código original. Estas modificaciones presentan una baja intrusividad, debido a que sólo realizan una sustitución de los tipos básicos del lenguaje, por un tipo de endurecido. Los tipos del lenguaje endurecido, usando el concepto de *Sphere of Replication* (SoR) [61], crean y computan tres réplicas de la variable a proteger. Eventualmente, para restaurar el consenso, el tipo endurecido realiza una votación mayoritaria.(Figura 1.16).

```
template <typename DataType>
class TD {
    private volatile DataType d1;
    private volatile DataType d2;
    private volatile DataType d3;
}
```

Figura 1.16: Código de replicación basado en plantillas. DataType es el tipo básico del lenguaje (no endurecido) para ser sustituido.

Dada la versatilidad de esta técnica, es imprescindible identificar los elementos menos fiables de la plataforma. Para ello, el modelo empírico descrito anteriormente, ayuda a evaluar, rápidamente, el impacto de la solución, identificando los elementos dónde centrar el esfuerzo de endurecimiento.

1.7 Colección de artículos y contribuciones que apoyan esta tesis

El trabajo desarrollado en esta tesis, se centra en el cumplimiento de los objetivos propuestos, validando las hipótesis planteadas. En este sentido, los métodos, algoritmos y estrategias propuestos, se centraron en la consecución de un modelo matemático semi-empírico. Este modelo predijo los efectos de la radiación ionizante, en microprocesadores, teniendo en cuenta aspectos tanto de la estructura tecnológica del hardware (memoria caché, registros, etc.), como del software (estructura del fichero binario). Como resultado de esta tesis se realizaron 6 publicaciones en revistas de alto impacto clasificadas por la JCR (J1 a J6), 11 comunicaciones a congresos internacionales (C1 a C11) y 3 comunicaciones a congresos nacionales (N1 a N3). Las contribuciones a revista denominadas **J1**, **J2**, **J3** y **J4**, constituyen el compendio de publicaciones que se presentan para la defensa de la tesis.

A continuación, se detallan todas las contribuciones realizadas en el marco de la presente tesis en función del tipo de publicación: revistas indexadas en el JCR, comunicaciones a congresos internacionales, y comunicaciones a congresos nacionales:

1.7.1 Artículos publicados en revistas indexadas en el JCR

J1 A. Serrano-Cases, L. M. Reyneri, Y. Morilla, S. Cuenca-Asensi y A. Martínez-Álvarez "Empirical mathematical model of microprocessor sensitivity and early prediction to proton and neutron radiation-induced soft errors", *IEEE Transactions on Nuclear Science* vol. 67 n.º 7, págs. 1511-1520 2020. DOI: 10.1109/tns.2020.2993637 JCR 2019 impact factor: 1.575

J2 A. Serrano-Cases, F. Restrepo-Calle, S. Cuenca-Asensi y A. Martínez-Álvarez "Multi-threaded mitigation of radiation-induced soft errors in bare-metal embedded systems", *Journal of Electronic Testing: Theory and Applications (JETTA)* vol. 36 n.º 1, págs. 47-57 2020. DOI: 10.1007/s10836-019-05846-4 JCR 2019 impact factor: 0.596

J3 L. M. Reyneri, A. Serrano-Cases, Y. Morilla, S. Cuenca-Asensi y A. Martínez-Álvarez "A compact model to evaluate the effects of high level c code hardening in radiation environments", *Electronics* vol. 8 n.º 6, pág. 653 jun. de 2019. DOI: 10.3390/electronics8060653 JCR 2019 impact factor: 2.412

J4 A. Serrano-Cases, Y. Morilla, P. Martín-Holgado, S. Cuenca-Asensi y A. Martínez-Álvarez "Nonintrusive automatic compiler-guided reliability improvement of embedded applications under proton irradiation", *IEEE Transactions on Nuclear Science* vol. 66 n.º 7, págs. 1500-1509 jul. de 2019. DOI: 10.1109/tns.2019.2912323 JCR 2019 impact factor: 1.575

J5 M. Peña-Fernández, A. Serrano-Cases, A. Lindoso, M. García-Valderas, L. Entrena, A. Martínez-Álvarez y col. "Dual-core lockstep enhanced with redundant multithread support and control-flow error detection", *Microelectronics Reliability* vol. 100-101, pág. 113 447 sep. de 2019. DOI: 10.1016/j.microrel.2019.113447 JCR 2019 impact factor: 1.535

J6 I. Albandes, A. Serrano-Cases, M. Martins, A. Martínez-Álvarez, S. Cuenca-Asensi y F. Kastensmidt "Design of approximate-TMR using approximate library and heuristic approaches", *Microelectronics Reliability* vol. 88-90, págs. 898-902 sep. de 2018. DOI: 10.1016/j.microrel.2018.07.115 JCR 2018 impact factor: 1.483

1.7.2 Comunicaciones a congresos internacionales

C1 A. Aponte-Moreno, J. Isaza-Gonzalez, A. Serrano-Cases, A. Martínez-Álvarez, S. Cuenca-Asensi y F. Restrepo-Calle, "An experimental comparison of fault injection tools for microprocessor-based systems", en *2020 IEEE Latin-American Test Symposium (LATS)*, IEEE, mar. de 2020

C2 D. R. Falco, A. Serrano-Cases, A. Martínez-Álvarez y S. Cuenca-Asensi, "Soft error reliability predictor based on a deep feedforward neural network", en *2020 IEEE Latin-American Test Symposium (LATS)*, IEEE, mar. de 2020

C3 L. M. Reyneri, A. Serrano-Cases, Y. Morilla, S. Cuenca-Asensi y A. Martínez-Álvarez, "A mathematical model to predict microprocessors fault tolerance under proton and neutron irradiation", en *2019 RADiation and its Effects on Components and Systems (RADECS)*, sep. de 2019

C4 A. Serrano-Cases, F. Restrepo-Calle, S. Cuenca-Asensi y A. Martínez-Álvarez, "Softerror mitigation for multi-core processors based on thread replication", en *2019 IEEE Latin American Test Symposium (LATS)*, IEEE, mar. de 2019. DOI: 10.1109/latw.2019.8704614

C5 M. Peña-Fernandez, A. Serrano-Cases, A. Lindoso, M. Garcia-Valderas, L. Entrena, A. Martínez-Álvarez y col., "Dual-Core Lockstep Enhanced with Redundant MultiThread Support and Control-Flow Error Detection", en *30th European Symposium on Reliability of Electron Devices, Failure physics and Analysis (ESREF)*, sep. de 2019

C6 I. Albandes, A. Serrano-Cases, M. Martins, A. Martínez-Álvarez, S. Cuenca-Asensi y F. Kastensmidt, "Design of Approximate-TMR using Approximate Library and Heuristic Approaches", en *29th European Symposium on Reliability of Electron Devices, Failure physics and Analysis (ESREF)*, sep. de 2018

C7 A. Serrano-Cases, Y. Morilla, P. Martin-Holgado, S. Cuenca-Asensi y A. Martínez-Álvarez, "Automatic compiler-guided reliability improvement

of embedded processors under proton irradiation", en *2018 RADiation and its Effects on Components and Systems (RADECS)*, sep. de 2018

C8 I. Albandes, A. Serrano-Cases, A. Sanchez-Clemente, M. Martins, A. Martínez-Álvarez, S. Cuenca-Asensi y col., "Improving approximate-TMR using multi-objective optimization genetic algorithm", en *2018 IEEE 19th Latin-American Test Symposium (LATS)*, IEEE, mar. de 2018. DOI: 10.1109/latw.2018.8349665

C9 J. Isaza-Gonzalez, A. Serrano-Cases, A. Martínez-Álvarez, S. Cuenca-Asensi, H. Guzman-Miranda y M. A. Aguirre, "Contrast of a HDL model and COTS version of a microprocessor for soft-error testing", en *2017 18th IEEE Latin American Test Symposium (LATS)*, IEEE, mar. de 2017. DOI: 10.1109/latw.2017.7906771

C10 A. Serrano-Cases, J. Isaza-Gonzalez, S. Cuenca-Asensi y A. Martínez-Álvarez, "On the influence of compiler optimizations in the fault tolerance of embedded systems", en *2016 IEEE 22nd International Symposium on On-Line Testing and Robust System Design (IOLTS)*, IEEE, jul. de 2016. DOI: 10.1109/iolts.2016.7604701

C11 J. Isaza-Gonzalez, A. Serrano-Cases, F. Restrepo-Calle, S. Cuenca-Asensi y A. Martínez-Álvarez, "Dependability evaluation of COTS microprocessors via on-chip debugging facilities", en *2016 17th Latin-American Test Symposium (LATS)*, IEEE, abr. de 2016. DOI: 10.1109/latw.2016.7483335

1.7.3 Comunicaciones congresos nacionales

N1 A. Serrano-Cases, S. Cuenca-Asensi y A. Martínez-Álvarez, "Estrategia multi-hilo para la mitigación de fallos software inducidos por radiación en sistemas empotrados carentes de sistema operativo", sep. de 2019. dirección: http://hdl.handle.net/10662/9626

N2 A. Serrano-Cases, L. M. Reyneri, S. Cuenca-Asensi y A. Martínez-Álvarez, "Protección De Software Frente A Radiación En Procesadores Multi-Núcleo Sin Sistema Operativo", sep. de 2018. DOI: 10.5281/ZENODO.1442364

N3 A. Serrano-Cases, S. Cuenca-Asensi y A. Martínez-Álvarez, "Mejorando La Tolerancia A Fallos De Sistemas Embebidos Cambiando La Compilación", sep. de 2017. DOI: 10.5281/ZENODO.996065

1.7.4 Objetivos e hipótesis que abarca cada publicación

Las siguientes figuras muestran las relaciones entre las publicaciones de investigación realizadas en revistas y comunicaciones en congresos, las hipótesis

planteadas y los objetivos que se llevaron a cabo en esta tesis (Figura 1.17, Figura 1.18 y Figura 1.19). Como puede observarse, se han cubierto todos los objetivos e hipótesis planteados con las contribuciones realizadas durante esta tesis.

Figura 1.17: Objetivos e hipótesis cubiertos por la investigación publicada en revistas

	J1	J2	J3	J4	J5	J6
O1: Análisis del estado del arte	✔	✔	✔	✔	✔	✔
O2: Evaluación de simuladores		✔		✔	✔	
O3: Exploración automática del espacio				✔		✔
O4: Mitigación multi-nivel		✔			✔	
O5: Sensibilidad de componentes a radiación	✔		✔			
H1: Machine Learning				✔		✔
H2: Hardware aproximado TMR						✔
H3: Mejoras de fiabilidad con compiladores				✔		
H4: Mejoras en el rendimiento de n-modular		✔	✔		✔	
H5: Sensibilidad de componentes a radiación	✔		✔			

Figura 1.18: Objetivos e hipótesis cubiertos por la investigación publicada en congresos internacionales

	C1	C2	C3	C4	C5	C6	C7	C8	C9	C10	C11
O1: Análisis del estado del arte	✔	✔	✔	✔	✔	✔	✔	✔	✔	✔	✔
O2: Evaluación de simuladores	✔			✔	✔				✔		✔
O3: Exploración automática del espacio						✔	✔	✔		✔	
O4: Mitigación multi-nivel		✔	✔	✔	✔						
O5: Sensibilidad de componentes a radiación			✔								
H1: Machine Learning	✔					✔	✔	✔	✔	✔	
H2: Hardware aproximado TMR						✔		✔			
H3: Mejoras de fiabilidad con compiladores		✔					✔			✔	
H4: Mejoras en el rendimiento de n-modular			✔	✔	✔						✔
H5: Sensibilidad de componentes a radiación	✔		✔						✔		✔

Figura 1.19: Objetivos e hipótesis cubiertos por la investigación publicada en congresos nacionales

		N2	N3
O1: Análisis del estado del arte	✔	✔	✔
O2: Evaluación de simuladores			✔
O3: Exploración automática del espacio			✔
O4: Mitigación multi-nivel	✔	✔	
O5: Sensibilidad de componentes a radiación			
H1: Machine Learning		✔	✔
H2: Hardware aproximado TMR			
H3: Mejoras de fiabilidad con compiladores			✔
H4: Mejoras en el rendimiento de n-modular	✔	✔	
H5: Sensibilidad de componentes a radiación			

1.8 Resultados de la investigación

En esta tesis se estudió la optimización y el endurecimiento de aplicaciones y circuitos, en dispositivos empotrados, para aumentar su fiabilidad. En los manuscritos mencionados anteriormente, se muestran diferentes formas de optimizar las aplicaciones, reduciendo su exposición a la radiación ionizante en tiempo y recursos, con objeto de aumentar la fiabilidad (contribuciones denominadas **J2** y **J4**). Además, en varios trabajos, se muestra, a partir de inyecciones de fallos simuladas, cómo obtener mejores estimaciones de la fiabilidad. Utilizando, para este fin, un modelo empírico, creado a partir de los resultados obtenidos en campañas de radiación real (contribuciones denominadas **J1** y **J3**). Este modelo, también, ha permitido optimizar las técnicas de endurecimiento desarrolladas, dirigiendo de manera selectiva, el endurecimiento de los recursos a proteger.

1.8.1 Mejoras en la fiabilidad mediante el uso de algoritmos genéticos de optimización multiobjetivo

El manuscrito denominado **J4**, es la continuación de las contribuciones denominadas como **J6**, **C7**, **C6**, **C8**, **C10** y **N3**. Estos trabajos, muestran un método capaz de explorar un espacio de soluciones no lineal y multidimensional (MOOGA). Este método, utiliza dos algoritmos para clasificar y explorar el espacio de soluciones. El primer algoritmo, es una optimización multiobjetivo (MOO), basado en el concepto de Eficiencia de Pareto, que evalúa las soluciones encontradas, y el grado de cumplimiento de los diferentes objetivos seleccionados (rendimiento, uso de recursos y cobertura frente a fallos). El segundo caso, se utiliza un algoritmo genético (GA), para explorar el espacio de soluciones de forma automática y eficiente, utilizando el descenso de gradiente. El llamado algoritmo MOOGA, está diseñado para encontrar un conjunto de aplicaciones, con distintos grados de fiabilidad.

La contribuciones denominadas como **J6**, **C8** y **C6**, muestran una parte del trabajo de esta tesis, consistente en la optimización del algoritmo de redundancia hardware TMR. Este proceso de optimización, está enfocado a la reducción de los sobrecostes de área y consumo, reemplazando algunas de las puertas lógicas que componen el circuito endurecido, usando otras, de una librería de puertas aproximada. Esta biblioteca de puertas aproximadas, ofrece un conjunto de puertas con un mayor rendimiento y menor complejidad, que las puertas utilizadas en el circuito original. Sin embargo, esta librería, logra las mejoras en el rendimiento, reduciendo la similitud con la puerta original. Debido al elevado número de puertas lógicas, que puede llegar a presentar un circuito, el método MOOGA, realiza un estudio, sobre las numerosas combinaciones y aproximaciones que se pueden generar. Estas combinaciones, tienden a crecer exponencialmente, con el número y la complejidad de las puertas, que componen el circuito. Para evaluar eficientemente el espacio de soluciones, el algoritmo de MOOGA, fue configurado con un solo individuo como población inicial, correspondiente al circuito TMR original. Además, la primera generación del algoritmo MOOGA, se configuró para producir un alto número de individuos mutados, acelerando el crecimiento de la población. Como resultado, la población de individuos únicos creció rápidamente. Después de unas pocas generaciones, la tasa de mutación fue reducida para permitir la convergencia del algoritmo. Como resultado, el algoritmo MOOGA, encontró un conjunto de combinaciones, donde, el área se reduce alrededor del 50 %, y la cobertura de la fallos se mantiene alrededor del 70 %, en comparación con la protección completa del circuito TMR (Figura 1.20). El algoritmo MOOGA, también fue capaz de obtener un conjunto, más extenso, de combinaciones de circuitos TMR, más pequeños y con mejor cobertura frente a fallos, en comparación con otros métodos de búsqueda publicados [82].

De manera similar, los trabajos denominados **J4**, **C7** y **C10**, muestran cómo el algoritmo MOOGA, puede mejorar la fiabilidad de una aplicación software, ajustando, a ciegas, el proceso de compilación y generación de los ejecutables. En este caso, se evaluaron dos arquitecturas: MSP430 y ARM. El microcontrolador MSP430, posee un reducido conjunto de capacidades, limitando la obtención de mejoras relevantes, en la ejecución del programa. Como resultado, las mejoras de fiabilidad, sólo pueden proceder, de una mejora en la planificación o en el preprocesamiento, realizado durante la compilación. La evaluación de la fiabilidad de una aplicación, utiliza parte del trabajo presentado en las siguientes publicaciones **C2**, **C9** y **C11**, donde son validados simuladores, cuya finalidad es comprobar los incrementos de rendimiento y fiabilidad. En el trabajo presentado en el congreso internacional **C10**, se muestra las capacidades de búsqueda del algoritmo MOOGA, para encontrar un conjunto considerable de soluciones, teniendo en cuenta, la baja complejidad de la arquitectura.

En cambio, los trabajos denominados **J4** y **C7**, muestran como el compilador es capaz de optimizar el flujo de instrucciones, cuando se activan algunos de los elementos de la arquitectura de un microprocesador ARM: el cauce de segmentado fuera de orden (*Out-of-Order pipeline*) y las cachés y la pre-búsqueda de instrucciones. Debido a estos elementos, el cómputo puede ser acelerado,

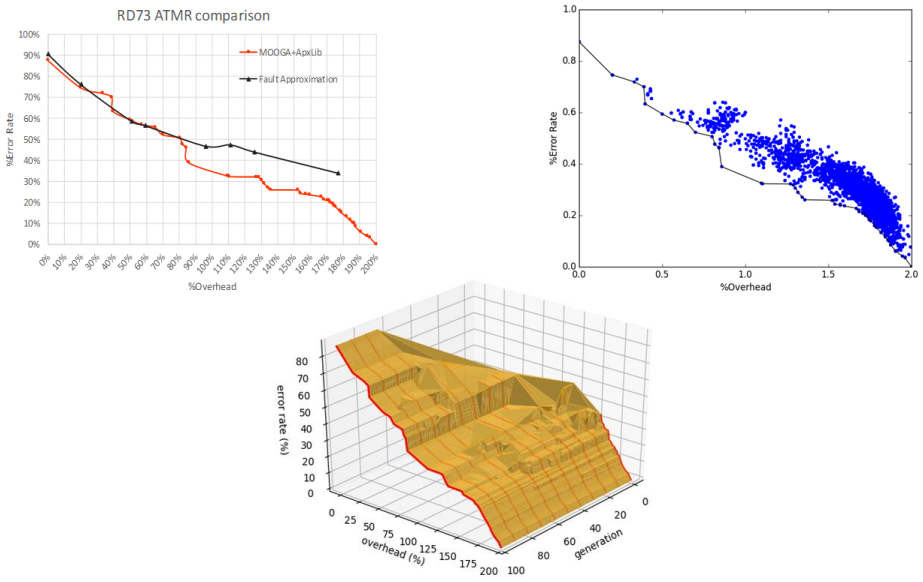


Figura 1.20: Las imágenes muestran la salida del proceso de optimización aplicado al circuito `RD73`. La figura superior izquierda compara las soluciones de la frontera de Pareto con otros métodos utilizados para buscar `TMRs` aproximados [82]. La parte superior derecha muestra la población que `MOOGA` es capaz de encontrar. Finalmente, la figura de abajo muestra el comportamiento de la tasa de error y el sobrecoste en el tamaño de los circuitos, en función del número de generaciones computadas (convergencia del algoritmo).

utilizando el conjunto de optimizaciones ofrecidas por el compilador. Como resultado, el algoritmo `MOOGA`, generó un gran conjunto de aplicaciones, utilizando el mismo código fuente. Además, el algoritmo `MOOGA`, fue capaz de mejorar el comportamiento de las aplicaciones, optimizadas usando las opciones por defecto del compilador. Las mejores soluciones encontradas, aumentaron el rendimiento y la cobertura frente a fallos, mientras se reducía el área expuesta (recursos hardware). En este sentido, en la publicación denominada **`J4`**, se mostró que las optimizaciones, por defecto del compilador, enfocadas al aumento del rendimiento, producen programas menos fiables. Por otro lado, la optimización conocida como `O0`, mostró una mayor fiabilidad (Figura 1.21). Sin embargo, el aumento del tiempo de exposición, así como el mayor uso de recursos, la hace más propensa a sufrir eventos bajo la radiación. Parte de los resultados obtenidos en esta investigación, fueron usados en el congreso internacional denominado **`C1`**, para crear un conjunto de descriptores, mediante técnicas de aprendizaje profundo, permitiendo realizar predicciones tempranas de la fiabilidad de una aplicación.

El manuscrito denominado **`J4`**, muestra la selección de varias de las aplicaciones generadas por `MOOGA`, y su evaluación, exponiéndolas a radiación de

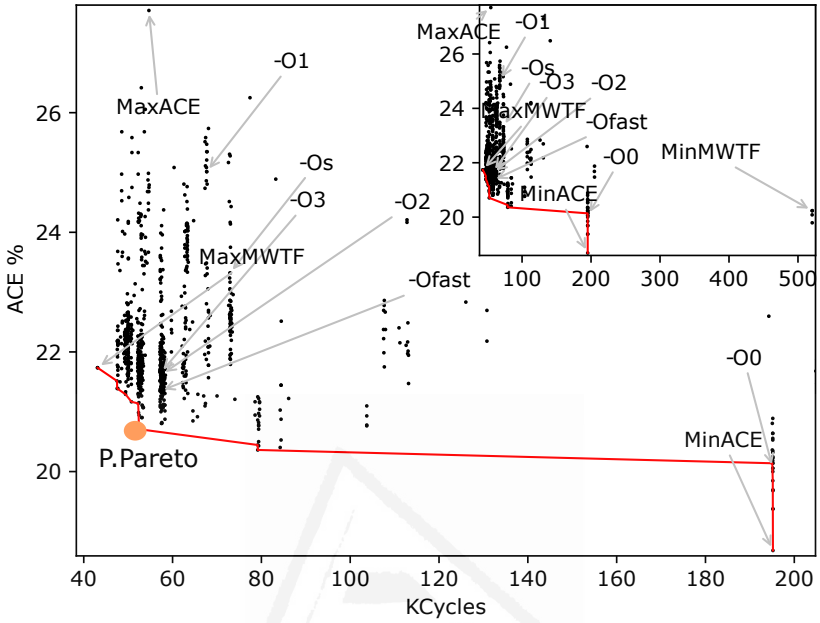


Figura 1.21: Resultados del proceso de optimización `MOOGA` aplicado a la mejora de la cobertura frente a fallos, rendimiento y el uso de recursos mediante el ajuste de las opciones del compilador. La vista muestra la cobertura de fallos frente al rendimiento representado en el eje vertical y el eje horizontal, respectivamente. `MinACE` y `O0` muestran la mejor cobertura de frente a fallos, sin embargo, también logran los peores resultados en rendimiento, excluyendo la solución `MinMWTF`. Por el contrario, las optimizaciones enfocadas a mejorar el rendimiento como `O3` muestran un rendimiento relativamente alto con una notable reducción en la cobertura frente a fallos. La solución más interesante de `MOOGA` (`P.Pareto`), muestra el mejor equilibrio entre ambos objetivos, cobertura frente a fallos y un alto rendimiento, siendo esta la solución más fiable encontrada durante la fase de evaluación temprana.

protones, en el ciclotrón de las instalaciones del `CNA` (Sevilla). Los resultados obtenidos de la campaña de radiación (Tabla 1.1), mostraron altas similitudes entre, las secciones transversales dinámicas, obtenidas de la campaña de radiación, y los resultados obtenidos de la simulación. En los resultados, se observa que las peores soluciones probadas son el `MaxAce` y el `MinMWTF`, presentando también la peor cobertura de fallos en la simulación. En consecuencia, estas versiones obtuvieron las secciones transversales más altas (3.8 y 4.8, respectivamente). Por el contrario, `O0` y `P.Pareto` muestran secciones transversales más baja, siguiendo la tendencia observada en la simulación. `P.Pareto` y `MaxMWTF`, son las programas con mayor rendimiento, y con secciones transversales similares, dando como resultado, programas que puedan realizar más

ejecuciones antes de que se produzca un fallo (MWTF).

Tabla 1.1: Resumen de los resultados de la radiación para cada versión de las aplicaciones de Bubblesort. La incertidumbre del flujo es $\pm 10\,\%$.

	$Flujo$ $p/cm^2 \cdot s$	$Fluencia$ p/cm^2	$Ciclos$	SDC	$HANG$	σ_{SDC} $(10^{-11})cm^2$	σ_{HANG} $(10^{-11})cm^2$	σ_{Total} $(10^{-11})cm^2$	$MWTF$ (10^{-14})
MaxACE	$7{,}8 \cdot 10^8$	$3{,}4 \cdot 10^{12}$	44409	112	14	3,3(2,7, 3,9)	0,42(0,20, 0,64)	3,8(3,2, 4,4)	3,96
-O0	$7{,}4 \cdot 10^8$	$4{,}9 \cdot 10^{12}$	220842	76	40	1,6(1,3, 1,9)	0,82(0,57, 1,1)	2,4(2,0, 2,8)	1,26
MaxMWTF	$8{,}3 \cdot 10^8$	$3{,}1 \cdot 10^{12}$	38412	79	36	2,5(1,9, 3,1)	1,2(0,82, 1,6)	3,7(3,0, 4,4)	4,67
MinMWTF	$1{,}0 \cdot 10^9$	$2{,}4 \cdot 10^{12}$	389806	69	45	2,9(2,2, 3,6)	1,9(1,4, 2,4)	4,8(3,9, 5,7)	35,4
P.Pareto	$1{,}1 \cdot 10^9$	$3{,}4 \cdot 10^{12}$	39635	65	38	1,9(1,4, 2,4)	1,1(0,74, 1,5)	3,1(2,5, 3,7)	5,45

1.8.2 Mejoras en la fiabilidad en bare-metal usando multi-hilo

La contribución denominada **J4**, reveló que el rendimiento del programa es un factor crítico, para mejorar la fiabilidad. En este sentido, los métodos utilizados en la bibliografía para el endurecimiento, como la técnica software *n-modular*, hacen el cálculo de las réplicas extendiéndolas en el tiempo sobre un solo núcleo, degradando el rendimiento. Sin embargo, muchos microprocesadores, han evolucionado hacía arquitecturas multi-núcleo, permitiéndoles realizar cálculos simultáneos, en distintos flujos de instrucciones. Como resultado de la investigación realizada, ahora es posible acelerar el cálculo de dichas réplicas para ganar rendimiento y fiabilidad simultáneamente.

El trabajo denominado **J2**, es la continuación de las contribuciones denominadas como **C4**, **J5**, **N2**, y **N1**, en las que se evalúa la optimización del procesamiento de las réplicas, usando diferentes núcleos, mediante hilos en *bare-metal*. Para validar la optimización realizada, en este trabajo utilizó campañas de inyección simuladas, además de, una versión preliminar de un modelo empírico, para mejorar la fiabilidad de la técnica. Como resultado, los experimentos realizados en el trabajo denominado **J2**, mostraron una reducción en los sobrecostes asociados a las técnicas TMR, además de un notable incremento en el rendimiento de la solución. Sin embargo, este modelo de programación, puede introducir nuevos puntos de fallo, que no son despreciables. Por este motivo, en la publicación denominada **J5**, se utiliza una técnica híbrida, para analizar y resolver, los problemas de comunicación entre procesos, de las primeras propuestas. Una de las conclusiones obtenidas, mostró que el proceso de sincronización era extremadamente costoso, llegando a anular el aumento del rendimiento obtenido, en algunos casos. Este trabajo identifico, el ajuste de la frecuencia de las sincronizaciones, como factor crítico para obtener mejoras de la fiabilidad. El trabajo denominado **C4**, y su extensión denominada **J2**, demostraron que el TMR paralelizado, reduce altamente los sobrecostes temporales, en comparación con el enfoque de un solo hilo (Figura 1.22). Además, la investigación realizada muestra, que paralelizar la técnica DWC−R, o una variante de *n-modular*, con número de réplicas múltiplo de dos, aumenta la sensibilidad a la radiación, debido al alto coste de la rutina de mitigación. En este último

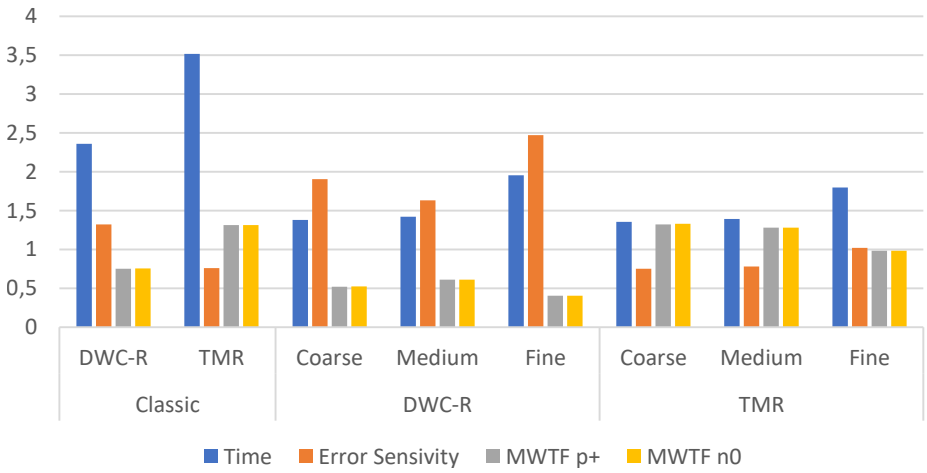


Figura 1.22: Evaluación de las versiones *n-modular* usando múltiples hilos, comparadas con las versiones de un solo hilo. Todas las versiones se normalizan con la versión sin endurecimiento. Las versiones multihilo muestran una reducción del tiempo necesario para calcular las réplicas, sin embargo, en todas las versiones de la `DWC-R` paralelizadas aumentan su sensibilidad a sufrir un fallo comparada con la versión original. Por el contrario, las versiones `TMR` son capaces de reducir la sensibilidad a los errores mientras que muestran aumentos en el rendimiento y la cantidad de trabajo completado antes de sufrir un fallo (`MWTF`).

caso, las restauraciones de contexto y la re-ejecución, anulan el aumento de rendimiento.

1.8.3 Predicción temprana de la fiabilidad mediante un modelo empírico

Los trabajos denominados **`J2`** y **`J3`**, utilizan el modelo presentado en el trabajo denominado **`J1`**. Este modelo, es usado para estimar la fiabilidad de una aplicación software, así como predecir el comportamiento de las aplicaciones software bajo radiación. Este modelo, usa los resultados de simulación (campaña de inyección de fallos), para estimar la cobertura a fallos, y modularlas con la sensibilidad de los componentes del sistema, obtenidos a partir de datos históricos de campañas de radiación. Los parámetros de sensibilidad del modelo, utilizan medidas de radiación de protones obtenidas del `CNA`, así como los resultados de neutrones de Los Alamos Neutron Laboratory – `LANL`, para entrenarlo y ajustarlo empíricamente. Así, el modelo, permite la predicción del comportamiento de las pruebas de radiación acelerada de otras aplicaciones, usando campañas de inyección simuladas, dirigidas al mismo dispositivo usado en el entrenamiento del modelo.

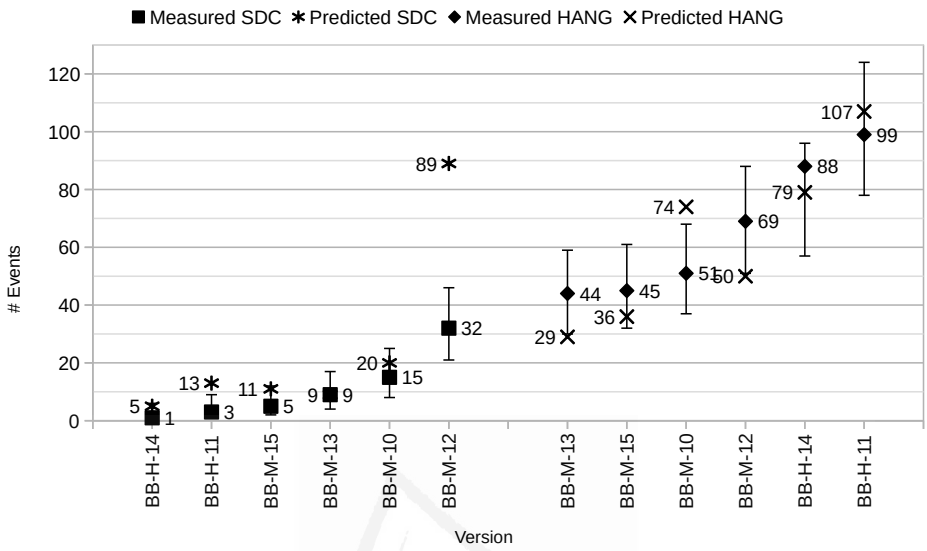


Figura 1.23: Resultados erróneos (SDC) y eventos HANG para la aplicación Bubblesort (BB). La figura muestra las estrategias de endurecimiento de la plantilla denotadas con H y varias construcciones de MOOGA denotadas con M. El gráfico muestra las medidas obtenidas de las pruebas de radiación acelerada realizadas en LANL con un intervalo de confianza del 95 % y el número de eventos predichos por el modelo para cada una de las pruebas.

Este modelo, es capaz de estimar la fiabilidad de una aplicación, a través de simuladores con precisión de instrucción, como los utilizados en los trabajos denominados **C11**, **C10**, y **C7**. Los resultados del modelo, muestran que es posible predecir los efectos de la radiación en un dispositivo, así como obtener la sensibilidad a la radiación de los componentes, tanto de los dispositivos SoC, como de los microcontroladores. Una versión incipiente del modelo, fue presentada en el trabajo denominado **J3**, y posteriormente extendida en el trabajo denominado **J1**. Este modelo, fue empleado para mejorar una estrategia de TMR, basada en programación genérica, aumentando la fiabilidad del endurecimiento de las aplicaciones software. El objetivo del experimento realizado, en el trabajo denominado **J3**, era doble. Por una parte, probar una técnica de protección altamente fiable, pero de bajo rendimiento. Y por otra parte, compararla con las soluciones, optimizadas y sin protección, obtenidas con el método MOOGA, presentado en el trabajo denominado **J4**. Los resultados obtenidos, muestran que la técnica TMR, basada en programación genérica, mejora la cobertura frente a fallos de la aplicación, mientras que se reduce el rendimiento debido a la replicación temporal. Además, los resultados obtenidos de simulación y radiación, muestran claramente, que no sólo la cobertura de fallos afecta a la fiabilidad, sino también al rendimiento y a la zona expuesta. Este comportamiento, se muestra en los resultados obtenidos del modelo presentado en **J1**

(Figura 1.23), donde las soluciones endurecidas (denotadas con `H`), muestran su capacidad de corrección, presentando un número reducido de eventos `SDC`. El modelo también muestra, un gran número de eventos `HANG` debido al tiempo de exposición y al aumento de los recursos sensibles. El trabajo denominado **`J3`**, pone de manifiesto, que la replicación parcial es la mejor estrategia de endurecimiento. Los resultados del modelo desarrollado, ayudaron a identificar las variables del programa más sensibles, donde enfocar los esfuerzos de endurecimiento. Éstas, coincidieron con las variables con un largo tiempo de vida (*lifetime*). En este sentido, el modelo y las campañas de inyección, muestran que evitar una sustitución completa, de los tipos básico por tipos endurecidos, es crítico para reducir los sobrecostes de area y temporales. El trabajo, muestra que las técnicas de `TMR`, deben evitar la redundancia innecesaria, preservando al mismo tiempo las mejoras de cobertura a fallos obtenidas, como factor determinante para aumentar la fiabilidad.

Capítulo 2

Artículos publicados

Este capítulo incluye la colección de artículos que respaldan esta tesis. Estas cuatro publicaciones en revistas de alto impacto clasificadas por la JCR representan y resumen los resultados de la investigación realizada durante los últimos cinco años.

2.1 Nonintrusive automatic compiler-guided reliability improvement of embedded applications under proton irradiation

Referencia
A. Serrano-Cases, Y. Morilla, P. Martín-Holgado, S. Cuenca-Asensi y A. Martínez-Álvarez "Nonintrusive automatic compiler-guided reliability improvement of embedded applications under proton irradiation", *IEEE Transactions on Nuclear Science* vol. 66 n.º 7, págs. 1500-1509 jul. de 2019. DOI: 10.1109/tns.2019.2912323 JCR 2019 impact factor: 1.575

Resumen
Se presenta un método para la mejora automatizada de la fiabilidad de las aplicaciones integradas. El proceso de compilación se guía mediante algoritmos genéticos y un enfoque de optimización multiobjetivo (MOO-GA). Aunque los compiladores modernos no están diseñados para generar compilaciones fiables, pueden ser afinados para obtener compilaciones que mejoren su fiabilidad, mediante la optimización simultánea de su cobertura de fallos, el tiempo de ejecución y el tamaño de la memoria. Los experimentos demuestran que pueden obtenerse mejoras relevantes de la fiabilidad mediante una exploración eficiente del espacio de soluciones de compilación. Se realizan campañas de simulación de inyección de fallas para evaluar nuestra propuesta contra diferentes puntos de referencia, y los resultados se evalúan contra un sistema real basado en Máquinas RISC Avanzadas en un chip bajo irradiación de protones.

2.2 A compact model to evaluate the effects of high level c++ code hardening in radiation environments

Referencia

L. M. Reyneri, A. Serrano-Cases, Y. Morilla, S. Cuenca-Asensi y A. Martínez-Álvarez "A compact model to evaluate the effects of high level c code hardening in radiation environments", *Electronics* vol. 8 n.º 6, pág. 653 jun. de 2019. DOI: 10.3390/electronics8060653 JCR 2019 impact factor: 2.412

Resumen

Una biblioteca de alto nivel de endurecimiento C++ diseñada para la protección del software crítico contra los efectos nocivos de los entornos de radiación que pueden dañar los sistemas. También se presenta un modelo matemático y empírico para predecir el comportamiento del sistema en presencia de fallas inducidas por la radiación. Este modelo genera una rápida evaluación y ajuste de varios compromisos de fiabilidad frente a rendimiento, para optimizar el endurecimiento por radiación basado en la biblioteca de endurecimiento C++ propuesta. Se utilizan varias simulaciones y campañas de irradiación con protones y neutrones para construir el modelo y ajustarlo. Finalmente, los efectos de nuestro enfoque de endurecimiento se comparan con otros enfoques endurecidos y no endurecidos.

2.3 Multi-threaded mitigation of radiation-induced soft errors in bare-metal embedded systems

Referencia
A. Serrano-Cases, F. Restrepo-Calle, S. Cuenca-Asensi y A. Martínez-Álvarez "Multi-threaded mitigation of radiation-induced soft errors in bare-metal embedded systems", *Journal of Electronic Testing: Theory and Applications (JETTA)* vol. 36 n.º 1, págs. 47-57 2020. DOI: 10.1007/s10836-019-05846-4 JCR 2019 impact factor: 0.596

Resumen
En este artículo se presenta una técnica de protección de software contra las fallas inducidas por la radiación, que se basa en una estrategia de múltiples hilos. Se utilizan técnicas de triplicación de datos y de duplicación o triplicación del flujo de instrucciones para mejorar la fiabilidad del sistema y, por lo tanto, asegurar un funcionamiento correcto del sistema. Para lograr este objetivo, se define un modelo de lockstep relajado para sincronizar la ejecución de ambos, los hilos redundantes y las variables bajo protección en diferentes unidades de procesamiento. La evaluación se realizó mediante campañas simuladas de inyección de fallas en un sistema ARM multi-núcleo. Los resultados muestran que a pesar de ser consideradas técnicas que implican un evidente overhead en la memoria y en las instrucciones (Duplicación con comparación y reejecución – DWC-R y Triple Redundancia Modular – TMR), la difusión de las réplicas en diferentes flujos de instrucciones no sólo produce resultados similares a las técnicas clásicas, sino que también mejora el tiempo de cálculo y de recuperación en presencia de soft errors. Además, en este documento se destaca la importancia de proteger los datos asignados en memoria, ya que la triplicación del flujo de instrucciones no es suficiente para mejorar la fiabilidad general del sistema.

2.4 Empirical mathematical model of microprocessor sensitivity and early prediction to proton and neutron radiation-induced soft errors

Referencia

A. Serrano-Cases, L. M. Reyneri, Y. Morilla, S. Cuenca-Asensi y A. Martínez-Álvarez "Empirical mathematical model of microprocessor sensitivity and early prediction to proton and neutron radiation-induced soft errors", *IEEE Transactions on Nuclear Science* vol. 67 n.º 7, págs. 1511-1520 2020. DOI: 10.1109/tns.2020.2993637 JCR 2019 impact factor: 1.575

Resumen

Se describe un modelo matemático para predecir la tolerancia a las fallos del microprocesador bajo la radiación. El modelo se entrena empíricamente combinando datos de campañas simuladas de inyección de fallas y experimentos de radiación, tanto con protones (en las instalaciones del CNA, Sevilla, España) como con neutrones (en las instalaciones de Investigación de Neutrones de Armas LANSCE en Los Álamos, EE.UU.). La sensibilidad a los soft errors de diferentes bloques de procesadores comerciales se identifica para estimar la fiabilidad de un conjunto de programas que habían sido previamente optimizados, endurecidos, o ambos. Los resultados mostraron un error estándar inferior a 0,1, en el caso del procesador ARM, y 0,12, en el caso del microcontrolador MSP430.

Capítulo 3

Conclusiones

En el presente capítulo se presentan las conclusiones obtenidas de las investigaciones realizadas. Este capítulo está organizado de la siguiente manera: la sección 3.1 muestra la conclusiones generales de esta investigación; la sección 3.2 presenta las contribuciones del modelo que proporciona predicciones tempranas de la fiabilidad de aplicaciones; la sección 3.3 expone las contribuciones y mejoras en las técnicas de endurecimiento mediante la difusión de las réplicas en múltiples núcleos usando hilos sin sistema operativo; la sección 3.4 muestra la principal conclusión obtenida al aplicar el proceso de optimización de búsqueda ciega en circuitos digitales y programas; finalmente, la sección 3.5 muestra las nuevas líneas de investigación derivadas de este trabajo de tesis.

3.1 Conclusiones generales

En esta tesis se evalúan, nuevas técnicas de endurecimiento y métodos, destinados a evaluación de la fiabilidad, haciendo uso de dos plataformas integradas, el microcontrolador MSP430 de Texas Instruments y la placa Zybo (Zynq-7000) de Xilinx. La evaluación de las plataformas, se hizo usando de un conjunto de aplicaciones, que fueron endurecidas u optimizadas, para obtener medidas de fiabilidad. Esta evaluación, permitió identificar las principales fuentes de fallos de las soluciones, lo que permitió determinar dónde centrar el esfuerzo, para aumentar la fiabilidad de la plataforma. Para conseguir este fin, se realizaron predicciones tempranas de fiabilidad, usando campañas de inyección de fallos simuladas, como método estadístico rápido, de obtención de la cobertura de fallos de las aplicaciones. Estos resultados de simulación, fueron utilizados para seleccionar y verificar el comportamiento de las aplicaciones, utilizando campañas de radiación (protones y neutrones), en los aceleradores de partículas del CNA y del LANL. Además, las medidas de radiación, contribuyeron a crear un modelo experimental, que predice los eventos de radiación, a partir de las medidas de simulación, así como la contribución de las partes estructurales que componen los dispositivos que se están evaluando.

Adicionalmente, en esta tesis se muestra, que es posible mejorar la fiabilidad de las aplicaciones usando métodos que: afinen el proceso de compilación, realicen un endurecimiento selectivo y aumenten el rendimiento del cálculo de réplicas, mediante varios flujos de instrucciones. Las mejoras de fiabilidad obtenidas al utilizar los compiladores, son debidas a optimizaciones que alteran, tanto el mapeo de los recursos de la aplicación, como la secuencia de instrucciones y los recursos hardware utilizados. Para obtener estos aumentos de fiabilidad, fue necesario seleccionar el conjunto de optimizaciones adecuado, realizando una búsqueda eficaz, para cada solución evaluada. Por otro lado, se ha demostrado que, la fiabilidad puede aumentarse endureciendo selectivamente los recursos a proteger. En este sentido, el modelo experimental, demostró que al evitar endurecer recursos con una vida útil baja, y una continua inicialización, se mantiene la cobertura de fallos, se aumenta el rendimiento y se reduce el uso de recursos, traduciéndose en un aumento de la fiabilidad. Finalmente, para aumentar la fiabilidad y el rendimiento, se ha desarrollado una nueva técnica de protección software, usando múltiples hilos. Esta técnica, mostró que es posible, mejorar el rendimiento, sin empeorar la cobertura de fallos, en las técnicas de endurecimiento basadas en la replicación, realizando el cálculo de la réplica en flujos paralelos. Además, los resultados demostraron, la importancia de ajustar la frecuencia de las comprobaciones, para mantener la consistencia del sistema.

3.2 Predicciones tempranas mediante un modelo matemático de la fiabilidad de aplicaciones

Es común que las estimaciones tempranas de la fiabilidad, realizadas para predecir los resultados de las medidas reales de radiación, utilicen modelos que aproximen el comportamiento de los dispositivos evaluados. Por este motivo, existe una alta dificultad para obtener estimaciones precisas de la fiabilidad, en dispositivos expuestos a radiación. Con objeto de solucionar esta problemática, en esta tesis se elaboró un modelo semi-empírico, capaz de determinar y explicar las ligeras diferencias entre los resultados esperados y medidos. El modelo puso de manifiesto, que varios componentes de los microprocesadores eran más sensibles a los fallos inducidos por la radiación. Los primeros resultados de las campañas de radiación, mostraron que los componentes más sensibles, coincidían con los que presentaban una zona de exposición más reducida, por tanto, con una menor probabilidad de ser afectados. Adicionalmente, el modelo demostró ser capaz de encontrar qué elementos son más propensos a errores, ofreciendo la oportunidad de centrar los esfuerzos de endurecimiento en ellos. Además, identificó el rendimiento como un parámetro crítico. En este sentido, el modelo señalaba que la reducción del rendimiento, era responsable de la obtención de resultados similares, en términos de fiabilidad, entre: las aplicaciones endurecidas mediante (*templates*), y las aplicaciones optimizadas utilizando la técnica `MOOGA`. En este contexto, el modelo mostró que endure-

cer cada variable de un programa era contraproducente. Como resultado del análisis, se concluyó que seleccionar el conjunto apropiado de elementos a ser endurecidos, es crítico para mejorar la fiabilidad.

3.3 Optimización del cálculo de las réplicas usando múltiples flujos de instrucciones en técnicas clásicas de endurecimiento

Dado que el rendimiento es una característica crítica, para obtener soluciones fiables, la utilización de múltiples núcleos para acelerar las técnicas de replicación, permite lograr un procesamiento más rápido. En este sentido, paralelizar las réplicas de las técnicas *n-modular*, sobre los diferentes flujos de instrucciones de un microprocesador, aumenta el rendimiento de la solución. En la bibliografía, existen varias investigaciones que utilizan grandes artefactos sin proteger para lograr un cómputo paralelo, mostrando penalizaciones en la cobertura de fallos. Con objeto de solucionar esta problemática, en esta tesis se desarrolla una nueva técnica multihilo, que consigue eliminar gran parte de estos artefactos, estableciendo un cómputo paralelo, sin la necesidad de añadir un sistema operativo (*bare-metal*). La técnica de endurecimiento multihilo, demostró ser capaz de aumentar el rendimiento de las técnicas de endurecimiento *n-modular*, manteniendo la tolerancia a fallos. Sin embargo, esta técnica, dependiendo de la versión de endurecimiento optimizada (`TMR` o `DWC-R`), puede introducir penalizaciones de rendimiento. Las técnicas `TMR`, presentan un mayor rendimiento, debido a que las réplicas se calculan al mismo tiempo y la rutina de corrección tiene una baja penalización. Por otro lado, la técnicas de corrección `DWC-R`, presenta una alta reducción del rendimiento, debido al tiempo necesario para guardar el estado de la plataforma y la re-ejecución en caso de error.

3.4 Mejoras de la fiabilidad mediante la exploración eficiente del espacio de soluciones

El algoritmo de búsqueda `MOOGA`, desarrollado en esta tesis, demostró ser capaz de encontrar y mejorar la fiabilidad de diferentes programas y circuitos, reduciéndolos en tamaño y acelerando el cómputo. Al aplicar la técnica, para la optimización de circuitos `TMR` aproximados, el algoritmo `MOOGA` fue capaz de encontrar un conjunto mejorado, con un número significativamente mayor de soluciones, que otros métodos de búsqueda presentados en la bibliografía. Por otro lado, al aplicar la técnica `MOOGA` a la optimización de la fiabilidad usando compiladores, el algoritmo es capaz de encontrar un conjunto de aplicaciones, que aumentan la fiabilidad, a niveles similares a los obtenidos por aplicaciones protegidas mediante endurecimiento. Estas aplicaciones optimizadas, a diferencia de las aplicaciones endurecidas, desarrolladas en la tesis, aumentan la

fiabilidad mejorando el rendimiento y reduciendo el uso de recursos. Debido a ello, las mejores soluciones que MOOGA obtuvo, son menos propensas a sufrir un evento inducido por radiación. Estas mejoras en la fiabilidad, se evaluaron usando el modelo empírico desarrollado y campañas de radiación acelerada.

3.5 Trabajos futuros

En esta tesis se han evaluado distintos métodos para aumentar la fiabilidad, además de proporcionar nuevas herramientas para la estimación temprana de la fiabilidad. Una de las investigaciones habilita nuevas formas de endurecer usando hilos en entornos sin sistema operativo. En esta investigación se ha estudiado el uso de la paralelización mediante hilos para establecer un cómputo fiable optimizando las técnicas *n-modular*, reutilizando parte del código del programa. Con objeto de analizar y potenciar este nuevo enfoque, se considera necesario extender la investigación para incluir un estudio de la optimización las técnicas *n-version*, paralelizándolas usando hilos. Además, dado que el conjunto de técnicas y métodos evaluados genera una cantidad considerable de información, sería deseable uso de técnicas más sofisticadas como el machine-learning y las técnicas de deep-learning, para la realización de un análisis más profundo, con objeto de obtener nuevas métricas de análisis temprano de la fiabilidad.

En los últimos años ha surgido *deep-learning* como un nuevo paradigma informático, adoptado en la mayoría de los entornos de computación de alto rendimiento (HPC). Las aplicaciones que emplean este paradigma, utilizan unidades de procesamiento matricial para lograr un alto rendimiento, maximizando la cantidad de recursos utilizados. Una nueva tendencia de investigación intenta adaptar este tipo de computación para operar en entornos críticos, debido a su alto rendimiento. Dado que en esta investigación se han desarrollado técnicas de endurecimiento basadas en paralelización y endurecimiento automático, es el paso lógico la extensión de esta técnica para operar en estos nuevos dispositivos, estableciendo un cómputo fiable intentando maximizar la cantidad de recursos disponibles en el sistema.

Bibliography

[1] S. Borkar and A. A. Chien, "The future of microprocessors", *Communications of the ACM*, vol. 54, no. 5, pp. 67–77, May 2011. DOI: 10.1145/1941487.1941507 (cit. on pp. 4, 57).

[2] K. S. Mohamed, *Neuromorphic Computing and Beyond*. Springer International Publishing, 2020. DOI: 10.1007/978-3-030-37224-8 (cit. on pp. 4, 57).

[3] R. Courtland, "Transistors could stop shrinking in 2021", *IEEE Spectrum*, vol. 53, no. 9, pp. 9–11, Sep. 2016. DOI: 10.1109/mspec.2016.7551335 (cit. on pp. 5, 58).

[4] M. Nicolaidis, Ed., *Soft Errors in Modern Electronic Systems*, ser. Frontiers in Electronic Testing. Boston, MA: Springer US, 2011, vol. 41, p. 368, ISBN: 978-1-4419-6992-7. DOI: 10.1007/978-1-4419-6993-4 (cit. on pp. 5, 58).

[5] D. M. Fleetwood, "Evolution of total ionizing dose effects in MOS devices with moore's law scaling", *IEEE Transactions on Nuclear Science*, vol. 65, no. 8, pp. 1465–1481, Aug. 2018. DOI: 10.1109/tns.2017.2786140 (cit. on pp. 5, 58).

[6] T. Karnik, P. Hazucha, and J. Patel, "Characterization of soft errors caused by single event upsets in CMOS processes", *IEEE Transactions on Dependable and Secure Computing*, vol. 1, no. 2, pp. 128–143, Apr. 2004, ISSN: 15455971. DOI: 10.1109/TDSC.2004.14 (cit. on pp. 5, 58).

[7] R. Edwards, C. Dyer, and E. Normand, "Technical Standard for Atmospheric Radiation Single Event Effects, (SEE) on Avionics Electronics", in *IEEE Radiation Effects Data Workshop*, IEEE, 2004, pp. 1–5, ISBN: 0780386973. DOI: 10.1109/redw.2004.1352895 (cit. on pp. 5, 9, 58, 62).

[8] R. C. Baumann, "Radiation-induced soft errors in advanced semiconductor technologies", *IEEE Transactions on Device and Materials Reliability*, vol. 5, no. 3, pp. 305–315, Sep. 2005, ISSN: 15304388. DOI: 10.1109/TDMR.2005.853449 (cit. on pp. 5, 58).

[9] F. Wang and V. D. Agrawal, "Single event upset: An embedded tutorial", in *Proceedings of the IEEE International Frequency Control Symposium and Exposition*, IEEE, 2008, pp. 429–434, ISBN: 0769530834. DOI: 10.1109/VLSI.2008.28 (cit. on pp. 5, 58).

[10] R. Gaillard, "Single Event Effects: Mechanisms and Classification", in *SOFT ERRORS IN MODERN ELECTRONIC SYSTEMS*, ser. Frontiers in Electronic Testing, M Nicolaidis, Ed., vol. 41, PO BOX 17, 3300 AA DORDRECHT, NETHERLANDS: SPRINGER, 2011, pp. 27–54, ISBN: 978-1-4419-6992-7. DOI: 10.1007/978-1-4419-6993-4_2 (cit. on pp. 5, 58).

[11] S. S. Rathod, A. K. Saxena, and S. Dasgupta, "Radiation effects in MOS-based devices and circuits: A review", *IETE Technical Review (Institution of Electronics and Telecommunication Engineers, India)*, vol. 28, no. 6, pp. 451–469, 2011, ISSN: 02564602. DOI: 10.4103/0256-4602.90747 (cit. on pp. 5, 58).

[12] T. Câlin, M. Nicolaidis, and R. Velazco, "Upset hardened memory design for submicron CMOS technology", *IEEE Transactions on Nuclear Science*, vol. 43, no. 6 PART 1, pp. 2874–2878, 1996, ISSN: 00189499. DOI: 10.1109/23.556880 (cit. on pp. 6, 60).

[13] R. C. Baumann, "Soft errors in advanced semiconductor devices-part i: the three radiation sources", *IEEE Transactions on Device and Materials Reliability*, vol. 1, no. 1, pp. 17–22, 2001, ISSN: 15304388. DOI: 10.1109/7298.946456 (cit. on pp. 6, 60).

[14] R. D. Schrimpf, K. M. Warren, R. A. Weller, R. A. Reed, L. W. Massengill, M. L. Alles, D. M. Fleetwood, X. J. Zhou, L. Tsetseris, and S. T. Pantelides, "Reliability and radiation effects in IC technologies", in *IEEE International Reliability Physics Symposium Proceedings*, ser. INTERNATIONAL RELIABILITY PHYSICS SYMPOSIUM, IEEE Electron Devices Soc; IEEE Reliabil Soc, 345 E 47TH ST, NEW YORK, NY 10017 USA: IEEE, ELECTRON DEVICES SOC & RELIABILITY GROUP, 2008, pp. 97–106, ISBN: 9781424420506. DOI: 10.1109/RELPHY.2008.4558869 (cit. on pp. 6, 60).

[15] B. Vibishna, K. S. Beenamole, and A. K. Singh, "Understanding single-event effects in FPGA for Avionic system design", *IETE Technical Review (Institution of Electronics and Telecommunication Engineers, India)*, vol. 30, no. 6, pp. 497–505, 2013, ISSN: 02564602. DOI: 10.4103/0256-4602.125674 (cit. on pp. 6, 9, 60, 62).

[16] R. R. Troutman, *Latchup in CMOS Technology*. Boston, MA: Springer US, 1986, ISBN: 978-1-4419-5199-1. DOI: 10.1007/978-1-4757-1887-4 (cit. on pp. 7, 61).

[17] C. D. Davidson, E. W. Blackmore, and J. I. Hess, "Failures of MOSFETs in terrestrial power electronics due to single event burnout", in *INTELEC, International Telecommunications Energy Conference (Proceedings)*, IEEE, 2004, pp. 503–507, ISBN: 0-7803-8458-X. DOI: 10.1109/intlec.2004.1401516 (cit. on pp. 7, 61).

[18] D. Nichols, J. Coss, and K. McCarty, "Single event gate rupture in commercial power MOSFETs", in *RADECS 93. Second European Conference on Radiation and its Effects on Components and Systems (Cat. No.93TH0616-3)*, IEEE, 2002, pp. 462–467, ISBN: 0-7803-1793-9. DOI: 10.1109/RADECS.1993.316559 (cit. on pp. 8, 61).

[19] C. Poivey, T. Corriere, J. Beaucour, and T. R. Oldham, "Characterization of single hard errors (SHE) in 1Mbit SRAMs from single ion", *IEEE Transactions on Nuclear Science*, vol. 41, no. 6, pp. 2235–2239, Dec. 1994, ISSN: 15581578. DOI: 10.1109/23.340568 (cit. on pp. 8, 62).

[20] R. Koga, S. H. Penzin, K. B. Crawford, and W. R. Crain, "Single event functional interrupt (SEFI) sensitivity in microcircuits", in *Proceedings of the European Conference on Radiation and its Effects on Components and Systems, RADECS*, IEEE, 1998, pp. 311–318, ISBN: 0-7803-4071-X. DOI: 10.1109/radecs.1997.698915 (cit. on pp. 8, 62).

[21] J. M. Benedetto, P. H. Eaton, D. G. Mavis, M. Gadlage, and T. Turflinger, "Digital single event transient trends with technology node scaling", *IEEE Transactions on Nuclear Science*, vol. 53, no. 6, pp. 3462–3465, Dec. 2006, ISSN: 00189499. DOI: 10.1109/TNS.2006.886044 (cit. on pp. 9, 62).

[22] P. T. McDonald, W. J. Stapor, A. B. Campbell, and L. W. Massengill, "Non-random single event upset trends", *IEEE Transactions on Nuclear Science*, vol. 36, no. 6, pp. 2324–2329, 1989, ISSN: 15581578. DOI: 10.1109/23.45443 (cit. on pp. 9, 62).

[23] A. D. Tipton, J. A. Pellish, P. R. Fleming, R. D. Schrimpf, R. A. Reed, R. A. Weller, M. H. Mendenhall, and L. W. Massengill, "High Energy Neutron Multiple-Bit Upset", in *2007 IEEE International Conference on Integrated Circuit Design and Technology*, IEEE, May 2007, pp. 1–3, ISBN: 1-4244-0756-7. DOI: 10.1109/ICICDT.2007.4299575 (cit. on pp. 9, 62).

[24] H. Quinn, P. Graham, J. Krone, M. Caffrey, and S. Rezgui, *IEEE Transactions on Nuclear Science*, vol. 52, no. 6, pp. 2455–2461, 2005, ISSN: 00189499. DOI: 10.1109/TNS.2005.860742 (cit. on pp. 9, 62).

[25] X. Iturbe, B. Venu, E. Ozer, J.-L. Poupat, G. Gimenez, and H.-U. Zurek, "The arm triple core lock-step (TCLS) processor", *ACM Transactions on Computer Systems*, vol. 36, no. 3, pp. 1–30, Aug. 2019. DOI: 10.1145/3323917 (cit. on pp. 11, 14, 65, 69).

[26] J. L. Autran, D. Munteanu, P. Roche, G. Gasiot, S. Martinie, S. Uznanski, S. Sauze, S. Semikh, E. Yakushev, S. Rozov, P. Loaiza, G. Warot, and M. Zampaolo, "Soft-errors induced by terrestrial neutrons and natural alpha-particle emitters in advanced memory circuits at ground level", *Microelectronics Reliability*, vol. 50, no. 9-11, pp. 1822–1831, 2010, ISSN: 00262714. DOI: 10.1016/j.microrel.2010.07.033 (cit. on pp. 11, 20, 65, 74).

[27] E. H. Ibe, "Irradiation Test Methods for Single Event Effects", in *Terrestrial Radiation Effects in ULSI Devices and Electronic Systems*, Singapore: John Wiley & Sons Singapore Pte. Ltd, Nov. 2014, pp. 61–105, ISBN: 978-1-118-47930-8; 978-1-118-47929-2. DOI: 10.1002/9781118479308.ch5 (cit. on pp. 11, 20, 65, 74).

[28] F. Kastensmidt, L. Carro, and R. Reis, *Fault-Tolerance Techniques for SRAM-based FPGAs*. Boston, MA: Springer US, 2006, vol. 32, pp. 1–183, ISBN: 978-0-387-31068-8. DOI: 10.1007/978-0-387-31069-5 (cit. on pp. 14, 68).

[29] S. Mukherjee, *Architecture Design for Soft Errors*. Elsevier Inc., 2008, pp. 1–4, ISBN: 9780123695291. DOI: 10.1016/B978-0-12-369529-1.X5001-0 (cit. on pp. 14, 16, 68, 71).

[30] K. Preethi and T. Karthik, "Protection of memory using code redundancies a survey", in *IC-GET 2015 - Proceedings of 2015 Online International Conference on Green Engineering and Technologies*, IEEE, Nov. 2016, pp. 1–4, ISBN: 9781467397810. DOI: 10.1109/GET.2015.7453847 (cit. on pp. 14, 68).

[31] T. E. Santhia, R. H. R. Bharathi, and M. Revathy, "Error detection and correction using decimal matrix code: Survey", in *Proceedings - 2017 IEEE International Conference on Electrical, Instrumentation and Communication Engineering, ICEICE 2017*, K Sundararaju, Ed., IEEE; M Kumarasamy Coll Engn; IEEE Madras Sect; M Kumarasamy Coll Engn, Fac Elect Engn, vol. 2017-Decem, 345 E 47TH ST, NEW YORK, NY 10017 USA: IEEE, 2017, pp. 1–5, ISBN: 9781509049967. DOI: 10.1109/ICEICE.2017.8191867 (cit. on pp. 14, 68).

[32] C. Colodro-Conde and R. Toledo-Moreo, "Design and analysis of efficient synthesis algorithms for EDAC functions in FPGAs", *IEEE Transactions on Aerospace and Electronic Systems*, vol. 51, no. 4, pp. 3332–3347, Oct. 2015, ISSN: 00189251. DOI: 10.1109/TAES.2015.140823 (cit. on pp. 14, 68).

[33] A. J. Olazabal and J. Pleite Guerra, "Multiple cell upsets inside aircrafts. New fault-tolerant architecture", *IEEE Transactions on Aerospace and Electronic Systems*, vol. 55, no. 1, pp. 332–342, Feb. 2019, ISSN: 15579603. DOI: 10.1109/TAES.2018.2852198 (cit. on pp. 14, 68).

[34] C. Frenkel, J. D. Legat, and D. Bol, "Comparative analysis of redundancy schemes for soft-error detection in low-cost space applications", in *2016 IFIP/IEEE International Conference on Very Large Scale Integration, VLSI-SoC 2016*, Int Federation Informat Proc TC 10 Working Grp 10 5; Inst Elect & Elect Engineers; IEEE Circuits & Syst Soc; Tallinn Univ Technol; Testonica Lab; IEEE Council Elect Design Automat, 345 E 47TH ST, NEW YORK, NY 10017 USA: IEEE, 2016, ISBN: 9781509035618. DOI: 10.1109/VLSI-SoC.2016.7753573 (cit. on pp. 15, 69).

[35] V. K. Chippa, S. T. Chakradhar, K. Roy, and A. Raghunathan, "Analysis and characterization of inherent application resilience for approximate computing", in *Proceedings of the 50th Annual Design Automation Conference on - DAC 2013*, ACM Press, 2013. DOI: 10.1145/2463209.2488873 (cit. on pp. 15, 69).

[36] K. Roy and A. Raghunathan, "Approximate computing: An energy-efficient computing technique for error resilient applications", in *2015 IEEE Computer Society Annual Symposium on VLSI*, IEEE, Jul. 2015. DOI: 10.1109/isvlsi.2015.130 (cit. on pp. 15, 69).

[37] A. J. Sanchez-Clemente, L. Entrena, and M. Garcia-Valderas, "Partial TMR in FPGAs using approximate logic circuits", *IEEE Transactions on Nuclear Science*, vol. 63, no. 4, pp. 2233–2240, Aug. 2016. DOI: 10.1109/tns.2016.2541700 (cit. on pp. 15, 69).

[38] I. A. Gomes, M. Martins, F. L. Kastensmidt, A. Reis, R. Ribas, and S. P. Novales, "Methodology for achieving best trade-off of area and fault masking coverage in ATMR", in *LATW 2014 - 15th IEEE Latin-American Test Workshop*, ser. Latin American Test Workshop, IEEE Comp Soc; Test Technol Tech Council; IEEE Council Elect Design Automat; Catholic Univ Rio Grande Sul; Fed Univ Ceara; Brazilian Natl Sci Fdn; CAPES; CEITEC, 345 E 47TH ST, NEW YORK, NY 10017 USA: IEEE, 2014, ISBN: 9781479947119. DOI: 10.1109/LATW.2014.6841916 (cit. on pp. 15, 69).

[39] I. A. Gomes, M. G. Martins, A. I. Reis, and F. L. Kastensmidt, "Exploring the use of approximate TMR to mask transient faults in logic with low area overhead", *Microelectronics Reliability*, vol. 55, no. 9-10, pp. 2072–2076, 2015, ISSN: 00262714. DOI: 10.1016/j.microrel.2015.06.125 (cit. on pp. 15, 69).

[40] R. Yao, Q. Chen, Z. Li, and Y. Sun, "Multi-objective evolutionary design of selective triple modular redundancy systems against SEUs", *Chinese Journal of Aeronautics*, vol. 28, no. 3, pp. 804–813, Jun. 2015, ISSN: 10009361. DOI: 10.1016/j.cja.2015.03.005 (cit. on pp. 15, 69).

[41] G. A. Reis, J. Chang, N. Vachharajani, R. Rangan, and D. I. August, "SWIFT: Software implemented fault tolerance", in *Proceedings of the 2005 International Symposium on Code Generation and Optimization, CGO 2005*, vol. 2005, 2005, pp. 243–254, ISBN: 076952298X. DOI: 10.1109/CGO.2005.34 (cit. on pp. 16, 70).

[42] J. A. Blome, S. Gupta, S. Feng, and S. Mahlke, "Cost-efficient soft error protection for embedded microprocessors", in *Proceedings of the 2006 International Conference on Compilers, Architecture and Synthesis for Embedded Systems*, ser. CASES '06, New York, NY, USA: ACM, 2006, pp. 421–431, ISBN: 1-59593-543-6. DOI: 10.1145/1176760.1176811 (cit. on pp. 16, 70).

[43] A. Martínez-Álvarez, F. Restrepo-Calle, L. A. V. Tejuelo, and S. Cuenca-Asensi, "Fault tolerant embedded systems design by multi-objective optimization", *Expert Systems with Applications*, vol. 40, no. 17, pp. 6813–6822, Dec. 2013. DOI: 10.1016/j.eswa.2013.06.060. [Online]. Available: https://doi.org/10.1016/j.eswa.2013.06.060 (cit. on pp. 16, 71).

[44] F. Lins, L. Tambara, F. L. Kastensmidt, and P. Rech, "Register file criticality and compiler optimization effects on embedded microprocessors reliability", *IEEE Transactions on Nuclear Science*, pp. 1–1, 2017. DOI: 10.1109/tns.2017.2705150 (cit. on pp. 16, 29, 71, 83).

[45] O. Goloubeva, M. Rebaudengo, M. S. Reorda, and M. Violante, *Software-implemented hardware fault tolerance*. Boston, MA: Springer US, 2006, pp. 1–227, ISBN: 0387260609. DOI: 10.1007/0-387-32937-4 (cit. on pp. 16, 71).

[46] H. Quinn, Z. Baker, T. Fairbanks, J. L. Tripp, and G. Duran, "Robust duplication with comparison methods in microcontrollers", *IEEE Transactions on Nuclear Science*, vol. 64, no. 1, pp. 338–345, Jan. 2017. DOI: 10.1109/tns.2016.2634781 (cit. on pp. 16, 71).

[47] F. Restrepo-Calle, A. Martínez-Álvarez, S. Cuenca-Asensi, and A. Jimeno-Morenilla, "Selective SWIFT-r", *Journal of Electronic Testing*, vol. 29, no. 6, pp. 825–838, Nov. 2013. DOI: 10.1007/s10836-013-5416-6. [Online]. Available: https://doi.org/10.1007/s10836-013-5416-6 (cit. on pp. 17, 71).

[48] F. Restrepo-Calle, S. Cuenca-Asensi, A. Martínez-Álvarez, E. Chielle, and F. L. Kastensmidt, "Application-based analysis of register file criticality for reliability assessment in embedded microprocessors", *Journal of Electronic Testing*, vol. 31, no. 2, pp. 139–150, Feb. 2015. DOI: 10.1007/s10836-015-5513-9. [Online]. Available: https://doi.org/10.1007/s10836-015-5513-9 (cit. on pp. 17, 71).

[49] J. Isaza-González, F. Restrepo-Calle, A. Martínez-Álvarez, and S. Cuenca-Asensi, "SHARC: An efficient metric for selective protection of software against soft errors", *Microelectronics Reliability*, vol. 88-90, pp. 903–908, Sep. 2018. DOI: 10.1016/j.microrel.2018.07.008. [Online]. Available: https://doi.org/10.1016/j.microrel.2018.07.008 (cit. on pp. 17, 71).

[50] C. A. L. Lisboa, L. Carro, M. S. Reorda, and M. Violante, "Online hardening of programs against seus and sets", in *2006 21st IEEE International Symposium on Defect and Fault Tolerance in VLSI Systems*, 2006, pp. 280–290 (cit. on pp. 18, 72).

[51] O. Goloubeva, M. Rebaudengo, M. Sonza Reorda, and M. Violante, "Soft-error detection using control flow assertions", in *Proceedings 18th IEEE Symposium on Defect and Fault Tolerance in VLSI Systems*, 2003, pp. 581–588 (cit. on pp. 18, 72).

[52] E. Chielle, G. S. Rodrigues, F. L. Kastensmidt, S. Cuenca-Asensi, L. A. Tambara, P. Rech, and H. Quinn, "S-SETA: Selective software-only error-detection technique using assertions", *IEEE Transactions on Nuclear Science*, vol. 62, no. 6, pp. 3088–3095, Dec. 2015. DOI: 10.1109/tns.2015.2484842. [Online]. Available: https://doi.org/10.1109/tns.2015.2484842 (cit. on pp. 18, 72).

[53] S. K. Reinhardt and S. S. Mukherjee, "Transient fault detection via simultaneous multithreading", in *Conference Proceedings - Annual International Symposium on Computer Architecture, ISCA*, Vancouver, BC, Can: IEEE, 2000, pp. 25–36. DOI: 10.1145/342001.339652 (cit. on pp. 18, 72).

[54] F. Restrepo-Calle, A. Martínez-Álvarez, S. Cuenca-Asensi, and A. Jimeno-Morenilla, "Selective SWIFT-R: A flexible software-based technique for soft error mitigation in low-cost embedded systems", *Journal of Electronic Testing: Theory and Applications (JETTA)*, vol. 29, no. 6, pp. 825–838, Dec. 2013, ISSN: 09238174. DOI: 10.1007/s10836-013-5416-6 (cit. on pp. 18, 72).

[55] S. Cuenca-Asensi, A. Martínez-Álvarez, F. Restrepo-Calle, F. R. Palomo, H. Guzman-Miranda, and M. A. Aguirre, "A novel co-design approach for soft errors mitigation in embedded systems", *IEEE Transactions on Nuclear Science*, vol. 58, no. 3, pp. 1059–1065, Jun. 2011. DOI: 10.1109/tns.2011.2112379. [Online]. Available: https://doi.org/10.1109/tns.2011.2112379 (cit. on pp. 19, 73).

[56] Y.-Y. Chen and K.-L. Leu, "Reliable data path design of VLIW processor cores with comprehensive error-coverage assessment", *Microprocessors and Microsystems*, vol. 34, no. 1, pp. 49–61, Feb. 2010. DOI: 10.1016/j.micpro.2009.11.004 (cit. on pp. 19, 73).

[57] A. Lindoso, M. Garcia-Valderas, L. Entrena, Y. Morilla, and P. Martin-Holgado, "Evaluation of the suitability of NEON SIMD microprocessor extensions under proton irradiation", *IEEE Transactions on Nuclear Science*, vol. 65, no. 8, pp. 1835–1842, Aug. 2018. DOI: 10.1109/tns.2018.2823540 (cit. on pp. 19, 73).

[58] S. S. Mukherjee, C. Weaver, J. Emer, S. K. Reinhardt, and T. Austin, "A systematic methodology to compute the architectural vulnerability factors for a high-performance microprocessor", in *Proceedings of the Annual International Symposium on Microarchitecture, MICRO*, vol. 2003-January,

IEEE Comput. Soc, 2003, pp. 29–40, ISBN: 076952043X. DOI: 10.1109/MICRO.2003.1253181 (cit. on pp. 20, 75).

[59] G. S. Rodrigues, F. Rosa, A. B. de Oliveira, F. L. Kastensmidt, L. Ost, and R. Reis, "Analyzing the impact of fault-tolerance methods in arm processors under soft errors running linux and parallelization apis", *IEEE Transactions on Nuclear Science*, vol. 64, no. 8, pp. 2196–2203, 2017. DOI: 10.1109/tns.2017.2706519 (cit. on pp. 31, 87).

[60] K. Chen, G. v. der Bruggen, and J. Chen, "Reliability optimization on multi-core systems with multi-tasking and redundant multi-threading", *IEEE Transactions on Computers*, vol. 67, no. 4, pp. 484–497, 2018. DOI: 10.1109/tc.2017.2769044 (cit. on pp. 31, 87).

[61] S. K. Reinhardt and S. S. Mukherjee, "Transient fault detection via simultaneous multithreading", *ACM SIGARCH Computer Architecture News*, vol. 28, no. 2, pp. 25–36, May 2000. DOI: 10.1145/342001.339652. [Online]. Available: https://doi.org/10.1145/342001.339652 (cit. on pp. 33, 89).

[62] A. Serrano-Cases, L. M. Reyneri, Y. Morilla, S. Cuenca-Asensi, and A. Martínez-Álvarez, "Empirical mathematical model of microprocessor sensitivity and early prediction to proton and neutron radiation-induced soft errors", *IEEE Transactions on Nuclear Science*, vol. 67, no. 7, pp. 1511–1520, 2020. DOI: 10.1109/tns.2020.2993637 (cit. on pp. 35, 50, 90, 106).

[63] A. Serrano-Cases, F. Restrepo-Calle, S. Cuenca-Asensi, and A. Martínez-Álvarez, "Multi-threaded mitigation of radiation-induced soft errors in bare-metal embedded systems", *Journal of Electronic Testing: Theory and Applications (JETTA)*, vol. 36, no. 1, pp. 47–57, 2020. DOI: 10.1007/s10836-019-05846-4 (cit. on pp. 35, 49, 90, 105).

[64] L. M. Reyneri, A. Serrano-Cases, Y. Morilla, S. Cuenca-Asensi, and A. Martínez-Álvarez, "A compact model to evaluate the effects of high level c code hardening in radiation environments", *Electronics*, vol. 8, no. 6, p. 653, Jun. 2019. DOI: 10.3390/electronics8060653 (cit. on pp. 35, 48, 90, 104).

[65] A. Serrano-Cases, Y. Morilla, P. Martín-Holgado, S. Cuenca-Asensi, and A. Martínez-Álvarez, "Nonintrusive automatic compiler-guided reliability improvement of embedded applications under proton irradiation", *IEEE Transactions on Nuclear Science*, vol. 66, no. 7, pp. 1500–1509, Jul. 2019. DOI: 10.1109/tns.2019.2912323 (cit. on pp. 35, 47, 90, 103).

[66] M. Peña-Fernández, A. Serrano-Cases, A. Lindoso, M. García-Valderas, L. Entrena, A. Martínez-Álvarez, and S. Cuenca-Asensi, "Dual-core lockstep enhanced with redundant multithread support and control-flow error detection", *Microelectronics Reliability*, vol. 100-101, p. 113 447, Sep. 2019. DOI: 10.1016/j.microrel.2019.113447 (cit. on pp. 36, 91).

[67] I. Albandes, A. Serrano-Cases, M. Martins, A. Martínez-Álvarez, S. Cuenca-Asensi, and F. Kastensmidt, "Design of approximate-TMR using approximate library and heuristic approaches", *Microelectronics Reliability*, vol. 88-90, pp. 898–902, Sep. 2018. DOI: 10.1016/j.microrel.2018.07.115 (cit. on pp. 36, 91).

[68] A. Aponte-Moreno, J. Isaza-Gonzalez, A. Serrano-Cases, A. Martínez-Álvarez, S. Cuenca-Asensi, and F. Restrepo-Calle, "An experimental comparison of fault injection tools for microprocessor-based systems", in *2020 IEEE Latin-American Test Symposium (LATS)*, IEEE, Mar. 2020 (cit. on pp. 36, 91).

[69] D. R. Falco, A. Serrano-Cases, A. Martínez-Álvarez, and S. Cuenca-Asensi, "Soft error reliability predictor based on a deep feedforward neural network", in *2020 IEEE Latin-American Test Symposium (LATS)*, IEEE, Mar. 2020 (cit. on pp. 36, 91).

[70] L. M. Reyneri, A. Serrano-Cases, Y. Morilla, S. Cuenca-Asensi, and A. Martínez-Álvarez, "A mathematical model to predict microprocessors fault tolerance under proton and neutron irradiation", in *2019 RADiation and its Effects on Components and Systems (RADECS)*, Sep. 2019 (cit. on pp. 36, 91).

[71] A. Serrano-Cases, F. Restrepo-Calle, S. Cuenca-Asensi, and A. Martínez-Álvarez, "Softerror mitigation for multi-core processors based on thread replication", in *2019 IEEE Latin American Test Symposium (LATS)*, IEEE, Mar. 2019. DOI: 10.1109/latw.2019.8704614 (cit. on pp. 36, 91).

[72] M. Peña-Fernandez, A. Serrano-Cases, A. Lindoso, M. Garcia-Valderas, L. Entrena, A. Martínez-Álvarez, and S. Cuenca-Asensi, "Dual-Core Lockstep Enhanced with Redundant MultiThread Support and Control-Flow Error Detection", in *30th European Symposium on Reliability of Electron Devices, Failure physics and Analysis (ESREF)*, Sep. 2019 (cit. on pp. 36, 91).

[73] I. Albandes, A. Serrano-Cases, M. Martins, A. Martínez-Álvarez, S. Cuenca-Asensi, and F. Kastensmidt, "Design of Approximate-TMR using Approximate Library and Heuristic Approaches", in *29th European Symposium on Reliability of Electron Devices, Failure physics and Analysis (ESREF)*, Sep. 2018 (cit. on pp. 36, 91).

[74] A. Serrano-Cases, Y. Morilla, P. Martin-Holgado, S. Cuenca-Asensi, and A. Martínez-Álvarez, "Automatic compiler-guided reliability improvement of embedded processors under proton irradiation", in *2018 RADiation and its Effects on Components and Systems (RADECS)*, Sep. 2018 (cit. on pp. 37, 91).

[75] I. Albandes, A. Serrano-Cases, A. Sanchez-Clemente, M. Martins, A. Martínez-Álvarez, S. Cuenca-Asensi, and F. L. Kastensmidt, "Improving approximate-TMR using multi-objective optimization genetic algorithm", in *2018 IEEE 19th Latin-American Test Symposium (LATS)*, IEEE, Mar. 2018. DOI: 10.1109/latw.2018.8349665 (cit. on pp. 37, 92).

[76] J. Isaza-Gonzalez, A. Serrano-Cases, A. Martínez-Álvarez, S. Cuenca-Asensi, H. Guzman-Miranda, and M. A. Aguirre, "Contrast of a HDL model and COTS version of a microprocessor for soft-error testing", in *2017 18th IEEE Latin American Test Symposium (LATS)*, IEEE, Mar. 2017. DOI: 10.1109/latw.2017.7906771 (cit. on pp. 37, 92).

[77] A. Serrano-Cases, J. Isaza-Gonzalez, S. Cuenca-Asensi, and A. Martínez-Álvarez, "On the influence of compiler optimizations in the fault tolerance of embedded systems", in *2016 IEEE 22nd International Symposium on On-Line Testing and Robust System Design (IOLTS)*, IEEE, Jul. 2016. DOI: 10.1109/iolts.2016.7604701 (cit. on pp. 37, 92).

[78] J. Isaza-Gonzalez, A. Serrano-Cases, F. Restrepo-Calle, S. Cuenca-Asensi, and A. Martínez-Álvarez, "Dependability evaluation of COTS microprocessors via on-chip debugging facilities", in *2016 17th Latin-American Test Symposium (LATS)*, IEEE, Apr. 2016. DOI: 10.1109/latw.2016.7483335 (cit. on pp. 37, 92).

[79] A. Serrano-Cases, S. Cuenca-Asensi, and A. Martínez-Álvarez, "Estrategia multi-hilo para la mitigación de fallos software inducidos por radiación en sistemas empotrados carentes de sistema operativo", Sep. 2019. [Online]. Available: http://hdl.handle.net/10662/9626 (cit. on pp. 37, 92).

[80] A. Serrano-Cases, L. M. Reyneri, S. Cuenca-Asensi, and A. Martínez-Álvarez, "Protección De Software Frente A Radiación En Procesadores Multi-Núcleo Sin Sistema Operativo", Sep. 2018. DOI: 10.5281/ZENODO.1442364 (cit. on pp. 37, 92).

[81] A. Serrano-Cases, S. Cuenca-Asensi, and A. Martínez-Álvarez, "Mejorando La Tolerancia A Fallos De Sistemas Embebidos Cambiando La Compilación", Sep. 2017. DOI: 10.5281/ZENODO.996065 (cit. on pp. 37, 92).

[82] A. J. Sanchez-Clemente, L. Entrena, R. Hrbacek, and L. Sekanina, "Error mitigation using approximate logic circuits: A comparison of probabilistic and evolutionary approaches", *IEEE Transactions on Reliability*, vol. 65, no. 4, pp. 1871–1883, Dec. 2016. DOI: 10.1109/tr.2016.2604918 (cit. on pp. 40, 41, 95, 96).

[83] A. Lindoso, M. Garcia-Valderas, and L. Entrena, "Analysis of neutron sensitivity and data-flow error detection in ARM microprocessors using NEON SIMD extensions", *Microelectronics Reliability*, vol. 100-101, p. 113 346, Sep. 2019. DOI: 10.1016/j.microrel.2019.06.038 (cit. on p. 74).