UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL
INSTITUTO DE INFORMÁTICA
PROGRAMA DE PÓS-GRADUAÇÃO EM COMPUTAÇÃO

THIAGO CABERLON SANTINI

Increasing Embedded Software Radiation Reliability Through Cache Memories

Thesis presented in partial fulfillment
of the requirements for the degree of
Master of Computer Science

Advisor: Prof. Dr. Flávio Rech Wagner
Coadvisor: Prof. Dr. Paolo Rech

Porto Alegre
February 2015

CIP – CATALOGING-IN-PUBLICATION

Santini, Thiago Caberlon

Increasing Embedded Software Radiation Reliability Through Cache Memories / Thiago Caberlon Santini. – Porto Alegre: PPGC da UFRGS, 2015.

73 f.: il.

Thesis (Master) – Universidade Federal do Rio Grande do Sul. Programa de Pós-Graduação em Computação, Porto Alegre, BR–RS, 2015. Advisor: Flávio Rech Wagner; Coadvisor: Paolo Rech.

1. Reliability. 2. Performance. 3. Embedded systems. 4. Cache. 5. Radiation. I. Wagner, Flávio Rech. II. Rech, Paolo. III. Increasing Embedded Software Radiation Reliability Through Cache Memories.

“ I give it to you not that you may remember time, but that you may forget it now and then for a moment and not spend all your breath trying to conquer it. Because no battle is ever won he said. They are not even fought. The field only reveals to man his own folly and despair, and victory is the illusion of philosophers and fools. ”

— WILLIAM FAULKNER

ACKNOWLEDGMENTS

First and foremost, I would like to thank my parents Gilberto Santini and Maria Inês Caberlon Santini and my brother Diego Caberlon Santini, for their continuous support throughout my life.

Many thanks to my advisor Prof. Dr. Flávio Rech Wagner and *grazie di tutto* to my coadvisor Prof. Dr. Paolo Rech. I really appreciate the incredible guidance (academic, scientific, and otherwise) that you have provided me throughout the course of my studies.

Thanks to my family, friends, colleagues, and to the UFRGS' staff; you know how awful my memory is, so no names or otherwise I would forget to mention someone :-)

Of course there is one exception, special thanks to my dear friend Nicole Gehrke.

I would also like to thank Heather Quinn, Thomas Fairbanks, Steve Wender, and Tanya Herrera from Los Alamos National Laboratory, Los Alamos, NM, USA for the help in scheduling beam time in LANSCE and for the support during experiments.

ABSTRACT

Cache memories are traditionally disabled in space-level and safety-critical applications since it is believed that the sensitive area they introduce would compromise the system reliability. As the technology has evolved, the speed gap between logic and main memory has increased in such a way that disabling caches slows the code much more than in the past. As a result, the processor is exposed for a much longer time in order to compute the same workload. In this work we demonstrate that, on modern embedded processors, enabling caches may bring benefits to critical systems: the larger exposed area may be compensated by the shorter exposure time, leading to an overall improved reliability. We propose an intuitive metric and a mathematical model to evaluate system reliability in spatial (i.e., radiation-sensitive area) and temporal (i.e., performance) terms, and prove that minimizing radiation-sensitive area does not necessarily maximize application reliability. The proposed metric and model are experimentally validated through an extensive radiation test campaign using a $28\,\mathrm{nm}$ off-the-shelf ARM-based System-on-Chip as a case study. The experimental results demonstrate that, while executing the considered application at military aircraft altitude, the probability of executing a two-year mission workload without failures is increased by 5.85% if L1 caches are enabled (thus, increasing the radiation-sensitive area), when compared to no cache level being enabled. However, if both L1 and L2 caches are enabled the probability is decreased by 31.59%.

Keywords: Reliability. performance. embedded systems. cache. radiation.

Aumentando a Confiabilidade de Software Embarcado sob Radiação através de Memórias Cache

RESUMO

Memórias cache são tradicionalmente desabilitadas em aplicações espaciais e críticas porque acredita-se que a área sensível por elas introduzida comprometeria a confiabilidade do sistema. Conforme a tecnologia tem evoluído, a diferença de velocidade entre lógica e memória principal tem aumentado de tal maneira que desabilitando as caches a execução do código é retardada muito mais do que no passado. Como resultado, o processador fica exposto por um tempo muito maior para computar a mesma carga de trabalho. Neste trabalho nós demonstramos que, em processadores embarcados modernos, habilitar as caches pode trazer benefícios para sistemas críticos: a área exposta maior pode ser compensada pelo tempo de exposição mais curto, levando a uma melhora total na confiabilidade. Nós propomos uma métrica intuitiva e um modelo matemático para avaliar a confiabilidade de um sistema em termos espaciais (i.e., área sensível à radiação) e temporais (i.e., desempenho), e provamos que minimizar a área sensível à radiação não necessariamente maximiza a confiabilidade da aplicação. A métrica e modelo propostos são experimentalmente validados através de uma campanha extensiva de testes de radiação utilizando um Sistema-em-Chip de prateleira fabricado em $28\,\mathrm{nm}$ baseado em processadores ARM como estudo de caso. Os resultados experimentais demonstram que, durante a execução da aplicação estudada à altitude de aeronave militar, a probabilidade de executar a carga de trabalho de uma missão de dois anos sem falhas é aumentada em 5.85% se as caches L1 são habilitadas (deste modo, aumentado a área sensível à radiação), quando comparada com nenhum nível de cache habilitado. Entretanto, se ambos níveis L1 e L2 são habilitados a probabilidade é diminuída em 31.59%.

Palavras-chave: confiabilidade, desempenho, sistemas embarcados, cache, radiação.

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS AND ACRONYMS

CMOS	Complementary Metal-Oxide-Semiconductor
COTS	Commercial-Off-The-Shelf
FIT	Failures In Time
GCR	Galactic Cosmic Ray
IC	Integrated Circuit
ITAR	International Traffic in Arms Regulation
MBU	Multiple Bit Upset
MCU	Multiple Cell Upset
MEBF	Mean Executions Between Failures
MTBF	Mean Time Between Failures
MTTF	Mean Time To Failure
MTTR	Mean Time To Repair
MWBF	Mean Workload Between Failures
OS	Operating System
PCR	Primary Cosmic Radiation
RadHard	Radiation Hardened
SBU	Single Bit Upset
SCR	Secondary Cosmic Radiation
SEB	Single Event Burn-out
SEE	Single Event Effect
SEFI	Single Event Functional Interruption
SEGR	Single Event Gate Rupture

SEL	Single Event Latch-up
SEP	Solar Energetic Particle
SET	Single Event Transient
SEU	Single Event Upset
SoC	System on a Chip
SRAM	Static Random-Access Memory
UAV	Unmanned Aerial Vehicle

LIST OF SYMBOLS

λ	Failure Rate
σ	Cross-section
σ_{bit}	Bit Cross-section
σ_{static}	Static Cross-section
Φ	Particle Fluence
ϕ	Particle Flux

CONTENTS

1 INTRODUCTION

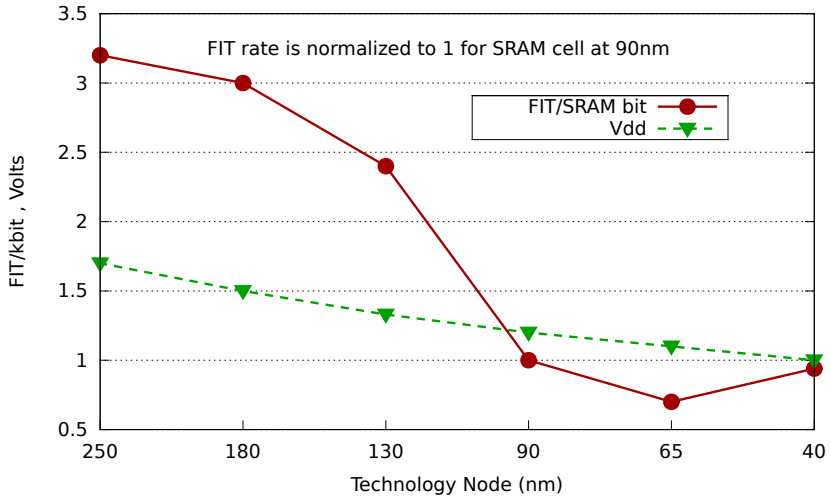
Over the history of computing hardware, the amount of transistors that can be placed within a single silicon die has approximately doubled every two years, as foreseen by Moore's law (MOORE, 1965). This sustained growth in the number of transistors has not only enabled the integration of more complex functionalities in a single chip, but also reduced the costs of already integrated functionalities. As a result, there is an extensive adoption of integrated circuits (ICs) in a wide range of embedded applications. From simple electronic timers superseding mechanical ones in washing machines, allowing a greater flexibility in the wash cycle; to complex fly-by-wire systems replacing manual flight controls, reducing aircraft weight and allowing the deployment of automatic stability systems; ICs are now a widespread and vital component in virtually every modern system.

Unfortunately, the shrinking in transistors' feature sizes that has allowed this increased integration has also increased these devices' susceptibility to radiation-induced failures. Radiation (i.e., energetic particles) may deposit enough charge to invert the state of a logic device - such as an SRAM cell, a latch, or a gate - introducing a fault into the circuit. Smaller transistors are smaller targets, but less energy is required to upset them due to their reduced operating voltage. Because the susceptibility to radiation-induced failures increases exponentially as the operating voltage decreases and decreases linearly as area decreases, the failure rate per logic device is expected to eventually increase (DIXIT; WOOD, 2011). In fact, results from Dixit and Wood (2011) seem to indicate that this is already occurring for $40\,\mathrm{nm}$ static random-access memories (SRAMs), as shown in Figure 1.1a, which shows the evolution of the FIT[1] rate per bit as the technology node shrinks. Additionally, not only the failure rate per bit matters, but it is also necessary to take into account the number of bits integrated into the device, i.e., the transistor density, as the resulting failure rate is a function of both the number of bits in the device and the failure rate per bit. As shown in Figure 1.1, although the per bit FIT has mostly decreased, the overall FIT for the device has usually increased due to the rise in the bit

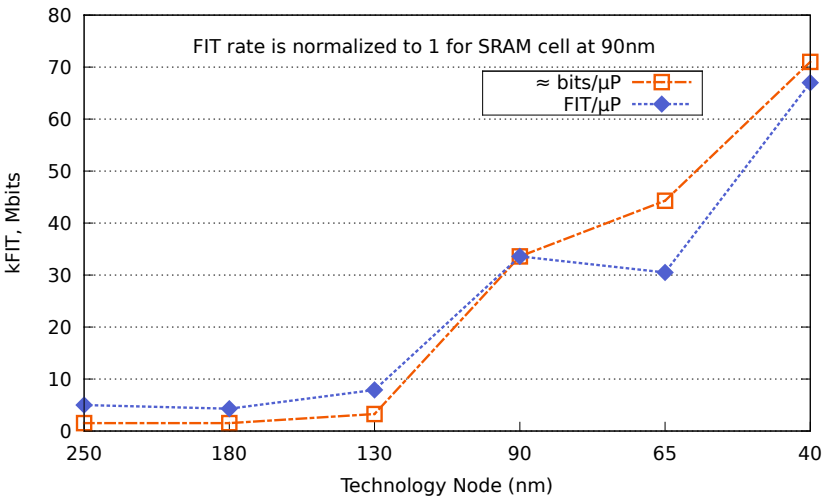
[1] Since failure rates are often very low, it is common to report them as Failures In Time (FIT): the number of expected failures per billion hours of device operation - e.g., failures per 1 device per billion hours, failures per 1000 devices per million hours, etc.

count per microprocessor.

Figure 1.1: Device parameters as function of technology node.



(a) FIT per SRAM bit, and operating voltage.



(b) FIT and number of bits per microprocessor.

Source: author, based on data from Dixit and Wood (2011).

1.1 Motivation

The history of radiation-induced failures starts many years earlier in 1954, when monitoring equipment presented electronic anomalies during above-ground nuclear bomb tests (ZIEGLER et al., 1996). More dramatically, following the Starfish Prime high-altitude nuclear test in 1962, ten known satellites were lost because of radiation damage – some immediately after the explosion (VETTE, 1991). In the same year, Wallmark and Marcus (1962) predicted that, due to upsets caused by terrestrial cosmic rays, the minimum volume of semiconductor devices would be limited to about $10\,\mu\mathrm{m}$ on a side. From then until

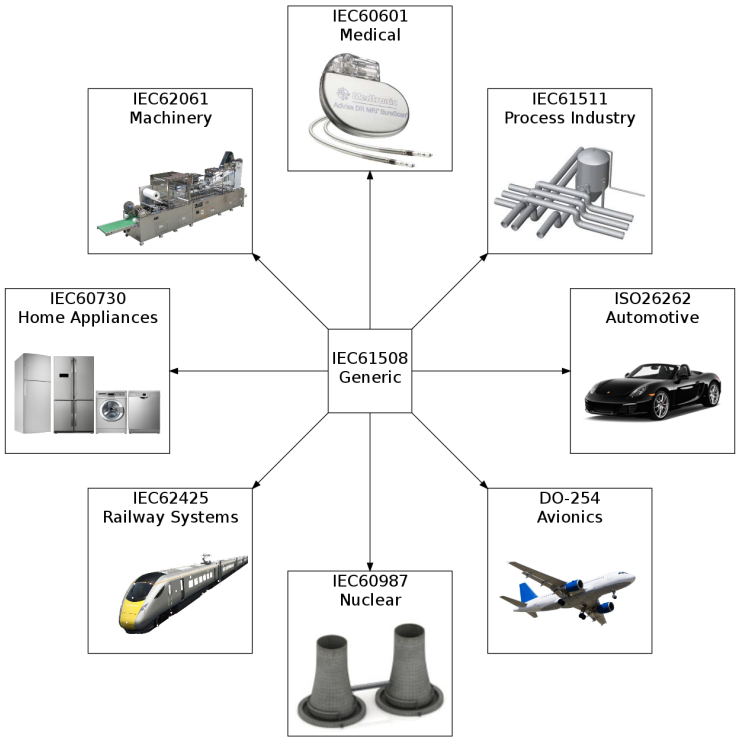
1970, early satellite electronics were found to be unreliable and considerable redundancy had to be applied. Later on, in 1975, came the first confirmed report of cosmic-ray-induced upsets in space: Binder et al. (1975) reported that, in 17 years of a communications satellite operation, four upsets were observed in bipolar JK flip-flops. Soon followed the first reports of upsets in terrestrial microelectronics, induced by alpha-particles in dynamic memories (MAY; WOODS, 1979). In the same year, it was proven that a cosmic ray could induce a latch-up, critical due to its possible destructive nature (KOLASINSKI et al., 1979). The 1980s focused mainly on errors in memory elements, although May et al. (1984) showed the temporal progression of a single upset leading to a fault condition within most parts of the microprocessor, and a few others investigated errors in combinational logic. During this decade, neutrons started being identified as being primarily responsible for radiation-induced upsets in avionics (SILBERBERG; TSAO; LETAW, 1984; DICELLO; PACIOTTI; SCHILLACI, 1989). From then on, more reports of radiation-induced failures followed, specially in large-scale systems such as supercomputers (MICHALAK et al., 2012; MICHALAK, 2004). Recently, a radiation-induced failure remains as the only potential cause not ruled out for an accident that left at least 119 people injured. On October 7, 2008, during an Airbus A330-303 flight at 37.000 feet, one of the plane's inertial reference units started outputting intermittent incorrect values. In response, the aircraft's control computers commanded the aircraft to pitch nose down, throwing unrestrained occupants and objects against the plane's ceiling, resulting in the damage shown in Figure 1.2.

Figure 1.2: Resulting damage to airplane from impact with persons and objects.

Source: (ATSB, 2008).

This accident helps to illustrate the significance of radiation-induced failures in *critical systems*. A critical system is a system whose malfunction could lead to unacceptable consequences. For example, a failure in a plane's flight control system or in a nuclear power plant control system can lead to human injury or damage to the environment (*safety-critical*); a failure in a spaceship's communication system may deem it incommunicable and end the mission prematurely (*mission-critical*); a high-frequency trading system failure may cause millions in capital loss (*business-critical*). To manufacture these systems, it is necessary to follow specific standards that define reliability[2] requirements for each domain, as well as guidelines and examples for development (IEC, 2014). Figure 1.3 shows typical standards and their associated domains for computer-based systems.

Figure 1.3: Typical reliability standards for computer-based systems, and their respective domains.



Source: author.

To achieve some of the reliability requirements imposed by these standards, designers started producing specific Radiation Hardened (RadHard) devices. The development of RadHard devices involves primarily four efforts: 1) choosing a technology process relatively insensitive to the environment where the device is to be deployed, 2) characterizing parts representative of the chosen technology, 3) based on this analysis, implementing the parts which make device response less sensitive to the radiation in an IC design, and

[2]Reliability is defined as the probability that the systems performs correctly during an interval of time, given that it was performing correctly at the beginning of the interval (PRADHAN, 1996).

4) simulating the circuit, identifying problem areas and improving the design (KERNS et al., 1988). Although RadHard ICs are less susceptible to radiation-induced failures, they require unique circuit design and lithography. This makes them expensive due to the non-recurring engineering costs combined with the small demand for such components. Moreover, RadHard ICs tend to be more expensive, larger, slower, and energy-hungrier than their unhardened counterparts. As a consequence, through the 1990s there was a dramatic decrease in the number of manufacturers offering radiation-hardened ICs (DODD; MASSENGILL, 2003). These, among other factors like the difficulty to import these devices due to the International Traffic in Arms Regulation, has led to a increased usage of Commercial-Off-The-Shelf (COTS) components instead. COTS devices are already being used in particular critical applications, for example, in the spacecraft on-board computer in NASA's PhoneSat project – a nanosatellite built around COTS smartphones running the Android operating system (SALAS et al., 2014).

When reliability is a major concern, the use of general purpose devices must be carefully evaluated. As technology scales down, CMOS devices are becoming more susceptible to errors induced by ionizing particles. Nowadays, radiation-induced failures are a concern not only in radiation-harsh environments, such as the space, but also in milder environments, such as at sea level. In fact, Baumann (2005) has shown that high-energy neutrons generated by the interaction of cosmic rays with the terrestrial atmosphere have enough energy to corrupt data stored in SRAM memories or to affect logic computations. If the error rate of a device is found to be unacceptable for the mission requirements, the system reliability may be increased with strategies like Double or Triple Modular Redundancy (D/TMR), Error Correction Codes (ECC), or control flow error detection (MITRA, 2012; MUKHERJEE, 2008; KIM; SOMANI, 1999). Even if effective, all the available hardening techniques introduce significant overheads in terms of area, cost and power consumption.

Particular attention is given to the reliability of cache memories. They occupy about 60% of the on-chip area in today's microprocessors and are considered the most vulnerable parts of modern computing systems (MANOOCHEHRI et al., 2011; MUKHERJEE et al., 2004; LIDEN et al., 1994). In fact, to be fast and efficient, cache memory cells are built as small as possible; thus, their capacitance and critical charge are lowered, hence increasing the probability of having the cell corrupted by ionizing radiation. Protection techniques that may be applied to cache memories, including parity and ECC, incur in extra area and significant additional power consumption and even lead to performance degradation (ASADI et al., 2005). It is worth noticing that the probability of having a Multiple Bit Upset (i.e., one impinging neutron corrupting more than one memory cell belonging to the same word) is increasing with the shrinking of the technology node (IBE et al., 2010; MAIZ et al., 2003). Moreover, caches are compact and dense; thus, the memory cells are close to each other, exacerbating the possibility of having one single

impinging particle interacting with more than one transistor. This results in the ECC design becoming more and more challenging. Although it has been show that Multiple Bit Upsets could be mitigated by the use of memory interleaving (BAEG; WEN; WONG, 2009), there is no guarantee that such hardware support (or even ECC) will be available in COTS embedded systems.

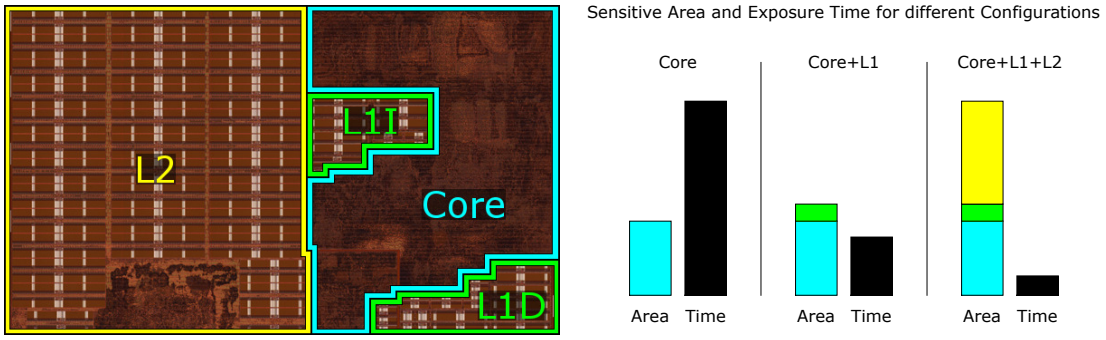
A straightforward solution employed to reduce the occurrence of radiation-induced failures is to minimize the system sensitive area, which is the amount of employed resources that are prone to be corrupted. As mentioned, cache memories are addressed as one of the most critical resources in a microprocessor. These memories are employed to reduce execution time by reducing memory access latency and, thus, are labeled as unnecessary in a critical system design based on COTS components and disabled (REBAUDENGO; SONZA REORDA; VIOLANTE, 2003).

However, this engineering solution works only assuming that disabling caches will improve reliability in spite of the increased execution time. Even if performance may not be the major concern in critical applications, it must be considered that, when radiation is concerned, a slower execution implies a longer exposure time. In other words, to safely disable caches the decrease in sensitive area must compensate the increased exposure time due to slower performance. Moreover, due to the huge difference in improvements between processor performance and memory access time, the gap between logic and memory performance becomes wider at each new device generation (LU et al., 2012). The performance ratio between executing a task with and without caches then increases. At some point, the assumption that the decrease in sensitive area will compensate the increased exposure time no longer holds.

Therefore, we hypothesize that, on modern embedded processors, enabling cache memories may bring benefits to critical systems: the larger sensitive area may be compensated by the shorter exposure time, leading to an overall improved reliability.

Furthermore, the multiple cache levels available in state-of-the-art embedded processors introduce different options for designers. For example, consider the Cortex A9 processor of Figure 1.4a, which has L1 and L2 caches. It is reasonable to assume that there is a given configuration, such as using no caches, only L1 caches, or both cache levels, that optimizes reliability (in terms of exposure time and sensitive area, as shown in Figure 1.4b). The choice of the more reliable configuration is not straightforward. In fact, as will be shown in this work, the reliability of the application will depend not only on the sensitivity to radiation of the chosen cache level, but also on the speed-up that comes from enabling the chosen level.

Figure 1.4: The sensitive area versus exposure time trade-off.



(a) An ARM Cortex A9 with 32 KB Data/Instruction L1 and 512 KB Shared L2.

(b) Caches increase performance, reducing exposure time.

Source: author.

1.2 Goals and Contributions

The main goal of this work is to prove the hypothesis that enabling cache memories may, under certain conditions, increase system reliability for a state-of-the-art COTS microprocessor. We have accomplished this goal and the following are additional contributions derived from this work.

Experimental data to help characterize the device. We have selected a modern device and conducted extensive radiation campaigns to measure the device's sensitivity to neutron radiation, namely its cross-section[3], using three different cache configurations: no cache level enabled, only the L1 cache enabled, and both cache levels enabled. These data complement the information provided by sources that investigate the selected device's radiation sensitivity, such as Quinn (2014) and Lesea (2014).

A novel approach to evaluate system reliability. Typically, the cross-section is used as a measure of reliability and the configuration with the smallest cross-section would be deemed more reliable. In this work, we go a step further in the reliability analysis and propose an intuitive metric and a formal mathematical model to measure device reliability, taking into account not only the sensitive area of the device, but also its performance. We then proceed to prove that an smaller cross-section does not always yield higher reliability, and present an experimental counterexample in which system reliability is improved even though the sensitive area of the device is increased.

[3]By definition the cross-section is the sensitive area of the device and represents the area that, if hit by an impinging particle, generates a failure (BAUMANN, 2005).

1.3 Outline

This dissertation is structured as follows. Chapter 2 discusses the origins of radiation pertinent to this work and its effects on electronic devices. Chapter 3 introduces the taxonomy used throughout this work, existing reliability metrics, and the proposed metric and model. Chapter 4 presents cache memories, focusing on the reliability aspect. Chapter 5 describes the radiation test campaign and reports and analyzes the acquired results. Conclusions drawn from this work are presented in Chapter 6, accompanied by envisioned future works.

2 NEUTRON-INDUCED RADIATION EFFECTS

Radiation is energy that emanates from matter in the form of rays or high-speed particles. The former, known as electromagnetic radiation, is pure energy with no weight such as sunlight, x-rays and radio waves. The latter, particle radiation, is represented by fast moving subatomic particles with both energy and mass such as alpha particles, beta particles and neutrons. Radiation that carries enough energy to free electrons from atoms or molecules is said to be *ionizing* radiation.

Naturally occurring radiation has always been present and can be found around us in many forms. Fortunately, not all radiation is able to upset electronic devices since most lack the required energy. At present, there is a great interest for Single Event Effects (SEEs) induced by high energy ($E > 10\,\mathrm{MeV}$) neutrons, driven by both the avionics industry, concerned with failures at aircraft altitudes, and RAM manufacturers and system vendors, concerned with failures at ground level (NORMAND et al., 2006). In fact, high energy atmospheric neutrons have been shown to be the largest contributor to the error rate of Complementary Metal-Oxide-Semiconductor (CMOS) devices operating at or below aircraft altitude (i.e., 60.000 feet) (FLEETWOOD; SCHRIMPF, 2004). For this reason, throughout this work we focus on high energy neutrons at or below aircraft altitude - although our approach and model are valid for other types of particles as well.

The goal of this chapter is then to elucidate how these particles originate and why they represent a threat to electronic devices. Section 2.1 gives an overview of the origin of high energy atmospheric neutrons and Section 2.2 analyzes their interaction with electronic devices and resulting effects.

2.1 Origin

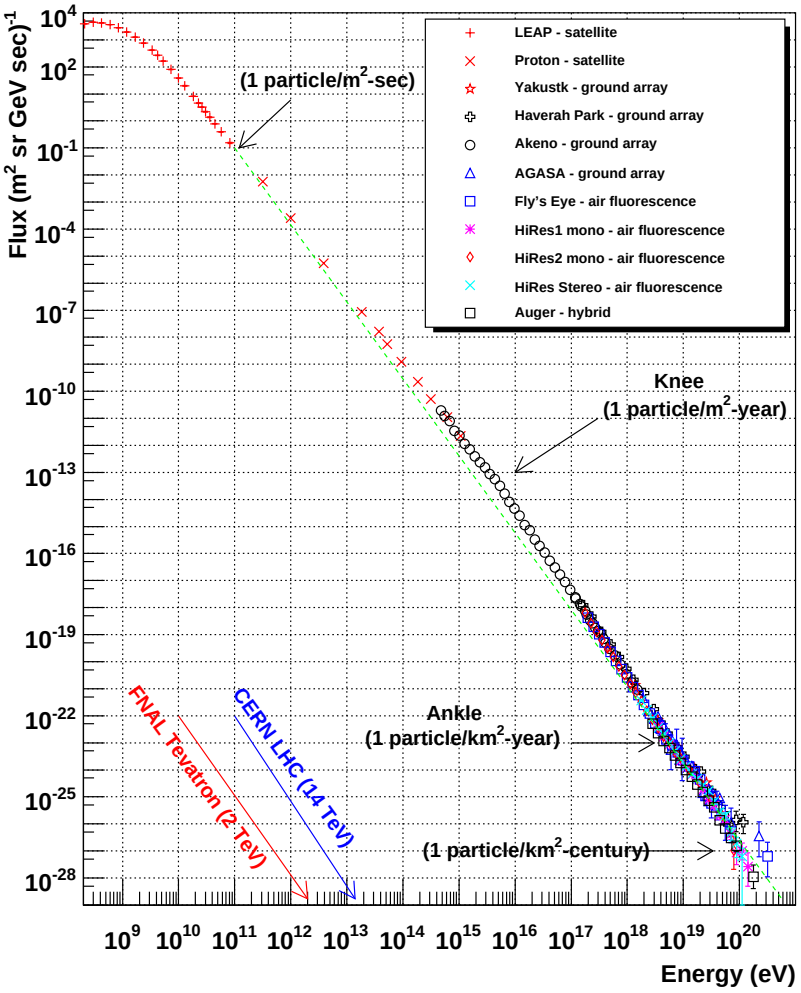
2.1.1 Primary Cosmic Radiation

The Earth is constantly showered by accelerated particles from outer space, commonly referred as Primary Cosmic Radiation (PCR). Most PCRs are completely ionized atoms (i.e., atoms stripped of electrons or protons): about 90% of them are hydrogen nuclei (i.e., protons), 9% helium nuclei (i.e., alpha particles), and the remaining 1% is made up by

heavy ions and other elements (GOLDHAGEN, 2003).

The main sources of PCRs are Solar Energetic Particles (SEPs) and Galactic Cosmic Rays (GCRs). SEPs, collectively called the solar wind, are a stream of particles such as protons and electrons continuously emitted by the sun's corona, reaching energy levels up to $20\,\mathrm{GeV}$. GCRs' origins are very complex to determine since their flight paths are scrambled by different magnetic fields of the galaxy, solar system, and Earth. Nevertheless, supernovas, neutron stars and black holes are some of the sources known to originate GCRs (MURSULA; USOSKIN, 2003). Wandering around the space, these particles gain charge and are accelerated by magnectic fields over a very long time, reaching energies up to $1\,\mathrm{ZeV}$. Figure 2.1 shows how PCRs are distributed in the energy spectrum, as measured by various experiments. Please notice that ultra-high energy particles (i.e., above $10^{19}\,\mathrm{eV}$) are very rare, hitting the Earth about once per square kilometer and century, thus making it very hard to draw any statistical conclusions about their origins (NASA, 2010).

Figure 2.1: Primary Cosmic Radiation energy spectra as measured by various experiments.

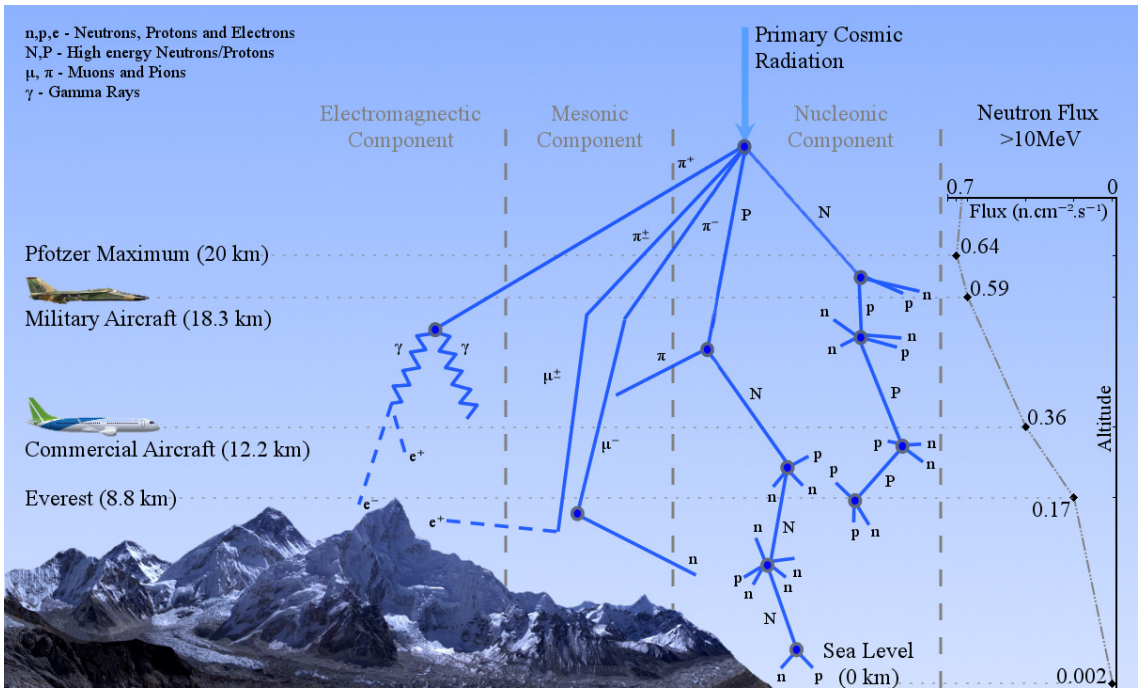


Source: (HANLON, 2008).

2.1.2 Particle Shower

A PCR entering the Earth atmosphere will likely collide with other particles there present, such as nitrogen and oxygen atoms - in fact, less than 1% of the primary radiation reaches sea level (ZIEGLER; LANFORD, 1981). These collisions trigger a process called spallation, in which atoms are divided into a broad spectrum of different particles, both stable and unstable. The resulting particles, dubbed Secondary Cosmic Radiation (SCR), either decay or undergo similar interactions, leading to a chain reaction that produces exponentially more particles until the Pfotzer maximum (PFOTZER, 1936) is reached at about $20\,\mathrm{km}$. Below this point, the particle flux starts decreasing due to energy loss, absorption and decay processes (GRIEDER, 2001). Nevertheless, some high energetic particles are still able to reach the ground. The end result of this process is a *particle shower* and a flux of high energy neutrons in the atmosphere, as depicted in Figure 2.2.

Figure 2.2: Particle shower generated by Primary Cosmic Radiation impinging Earth's atmosphere, and the neutron flux for energies above $10\,\mathrm{MeV}$ at Mount Everest coordinates and a 50% solar modulation - based on the NASA-Langley Model (GORDON et al., 2004).



Source: author.

2.1.3 Terrestrial Neutron Flux

Studies have shown that, at altitudes below $18.3\,\mathrm{km}$, high energy neutrons are the dominant factor in radiation-induced failure rates while, over $21\,\mathrm{km}$, cosmic ray heavy ions begin to dominate these rates (TSAO; SILBERBERG; LETAW, 1984). In fact, Taber and Normand (1993) have shown that memory upset rates measured in flight correspond

with atmospheric neutron flux levels, and, upon reviewing the existing database for large memory banks, Normand (1996) found that upset rates on the ground also match neutron flux levels. Additionally, while other particles that may induce failures such as alpha particles and low energy neutrons may be successfully mitigated, high energy neutrons are so penetrating that there is no practical way to shield critical equipment (BAUMANN, 2005). As such, high energy neutrons are the particles that represent the biggest threat to avionics and ground level electronic equipment.

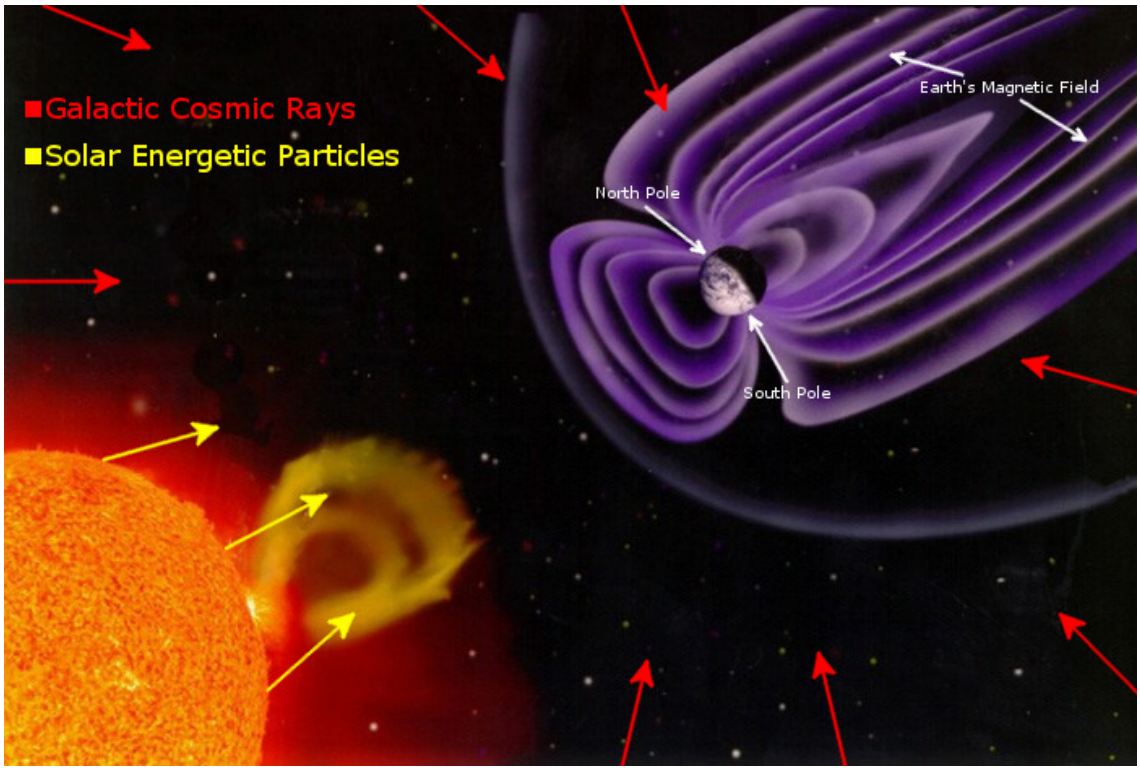
The atmospheric neutron flux is not constant, changing up to several hundred times depending on time and location. The main determinant of the atmospheric neutron flux intensity is the altitude. The balance between produced and absorbed neutrons changes based on the atmospheric atom density (increases as altitude decreases) and the remaining energy in the shower (decreases as altitude decreases), reaching its peak around $20\,\mathrm{km}$. Thus, high in the atmosphere the flux starts increasing until it reaches the Pfotzer Maximum, at which point it will start decreasing again (see the right side of Figure 2.2).

The latitude (and to a lesser extent the longitude) play a role in determining the necessary energy for a PCR to penetrate Earth's magnetic field. The Earth's magnetic field is nearly parallel to Earth's surface near the equator, and almost perpendicular close to the poles (see Figure 2.3). As a result, a charged particle approaching the Earth near the equator needs a higher momentum (i.e., higher energy) to penetrate the magnetic field than one approaching it near the poles (GOLDHAGEN, 2003). Thus, as the latitude increases, i.e., going from the equator to the poles, so does the number of PCRs able to penetrate the magnetic field, increasing the amount of particle showers and, consequently, the neutron flux.

The solar activity also influences the atmospheric neutron flux intensity. The magnetic poles of the Sun exchange places with a period of approximately 11 years. Close to the polarity exchange, the sun reaches maximum solar activity, increasing sunspots, solar flares and coronal mass ejections (NASA, 2011). Unintuitively, an increase in solar activity actually decreases the atmospheric neutron flux due to a phenomenon called the Forbush decrease (LOCKWOOD, 1971). The solar wind carries a strong and convoluted magnetic field with it that deflects GCRs, hence protecting the Earth (GOLDHAGEN, 2003). As a result, the atmospheric neutron flux is inversely correlated to sun activity, as shown in Figure 2.4.

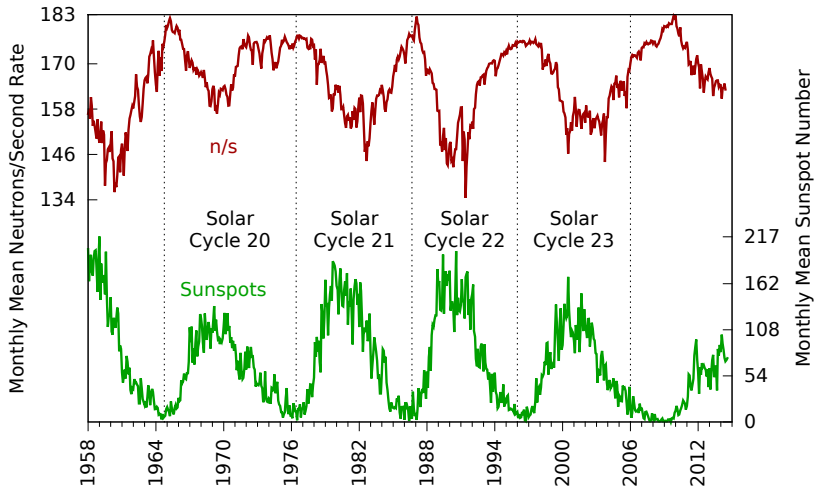
In order to evaluate the reliability of a device, it is then imperative to have a precise estimate of the neutron flux in the environment in which the device will be deployed. While the neutron flux intensity changes significantly based on location and time, its spectrum for high energies remains largely the same, allowing models to be greatly simplified. Currently two models are in use to estimate the atmospheric neutron flux. The Boeing model (NORMAND; BAKER, 1993) is endorsed by the IEC standard and accounts only for altitude and latitude. The NASA-Langley model (GORDON et al., 2004) is endorsed by

Figure 2.3: The Earth is constantly hit by Primary Cosmic Radiation. Particles approaching near the poles find less resistance from Earth's magnetic field than those approaching it near the equator.



Source: NASA (2013), annotated by the author.

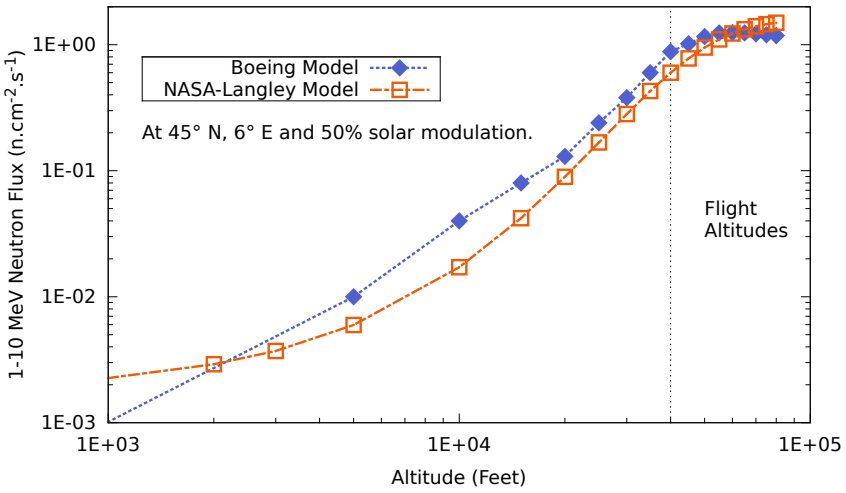
Figure 2.4: Neutron rate and sunspot activity according to the Germany Cosmic Ray Monitor in Kiel (NMDB, 2014) and NOAA's National Geophysical Data Center (NGDC, 2014), respectively.



Source: author.

the JEDEC standard and provides a more precise estimate, accounting for altitude, latitude, longitude, and solar modulation. The Boeing model provides a fair estimate at flight altitudes, but for lower altitudes its usage is not recommended since the estimates starts to deviate from the real neutron flux, for this reason the JEDEC standard moved from the Boeing model to the NASA-Langley model (see Figure 2.5).

Figure 2.5: Neutron Flux estimates based on the simplified Boeing model and NASA-Langley model.



Source: author.

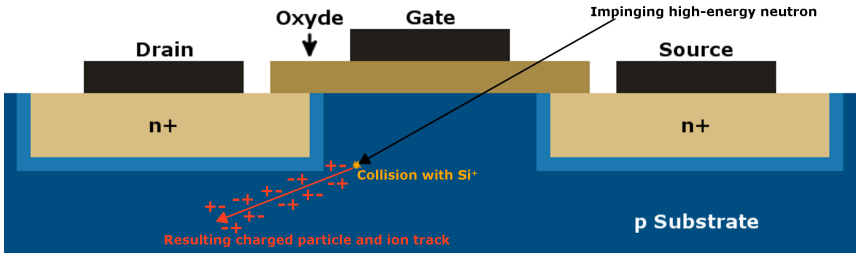
2.2 Interaction with Electronic Devices

Although a high energy neutron impinging a CMOS device is not able to directly ionize the device, it interacts inelastically with chip materials, such as silicon atoms, producing charged ionizing particles. These charged particles then produce a track of electron hole pairs on their wake (see Figure 2.6). Whenever this event happens nearby a depletion region, carriers are quickly collected creating a current pulse at the node. If enough charge is collected, the resulting pulse may then induce an error in the device (BAUMANN, 2005).

2.2.1 Single Event Effects

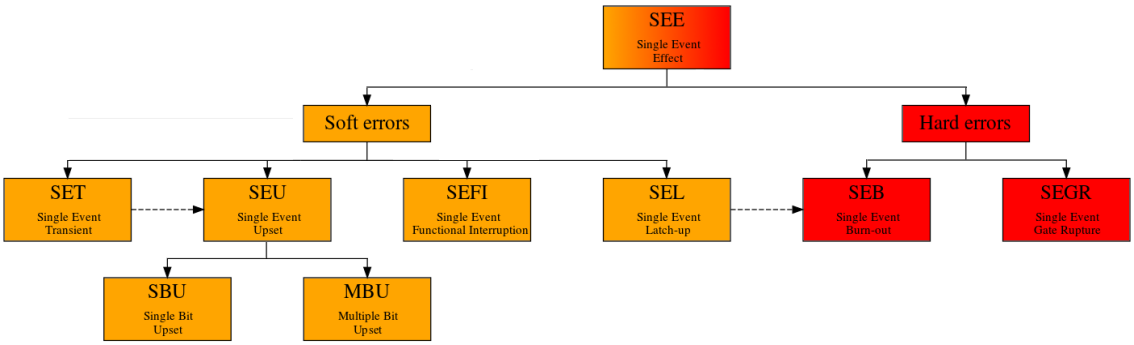
A measurable or observable deviation in performance or state of a microelectronic device arising from a single radiation event is called a Single Event Effect (SEE), and may present itself in various forms depending on the impinging particle's energy and the criticality of the affected node. These deviations (or errors) are divided into two main groups: 1) *soft errors*, when a signal or datum is erroneous but can be corrected by performing one or more normal functions of the device, i.e., the device is not permanently damaged, and 2) *hard errors*, when device operation is irreversibly changed, even after

Figure 2.6: Electron hole pairs track left by an ionizing radiation event originated from a high energy neutron.



Source: author.

Figure 2.7: Single Event Effects taxonomy.



Source: author.

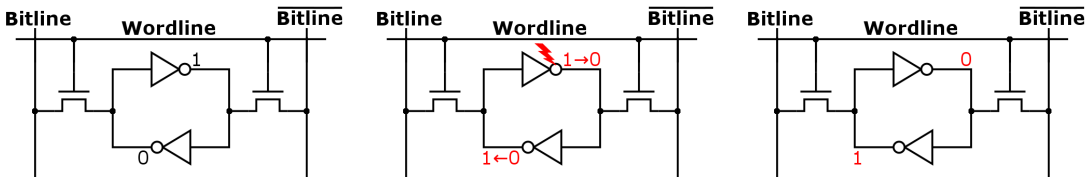
power reset and re-initialization, i.e., the device is permanently damaged. The different types of SEEs and their classifications can be seem in Figure 2.7, and a more detailed explanation of these effects follows.

2.2.1.1 Single Event Upset (SEU)

When a radiation event leads to the corruption of the data state of a memory element the event is called a Single Event Upset (SEU). For example, consider the standard 6-transistor SRAM cell of Figure 2.8a. A particle strike inverts the output of the top inverter from *1* to *0* for long enough for it to drive the bottom inverter (Figure 2.8b). The output of the bottom inverter in turn drives the top inverter, effectively inverting the data state of the cell (Figure 2.8c). Nevertheless, writing the correct value back to the memory cell suffices to restore the cell to its correct state; thus, this is a soft error.

Depending on the number of memory elements affected by the single radiation event, a SEU may be further classified as a Single Bit Upset (SBU), if only a single cell is corrupted, or a Multiple Cell Upset (MCU), if more than one cell is corrupted. Moreover, a MCU in which two or more corruptions occur in the same word is classified as a Multiple Bit Upset (MBU). This division is particularly useful when dealing with error detection

Figure 2.8: A particle strike flipping the data state of a 6-transistor SRAM cell.



(a) (b) (c)

Source: author.

and correction codes.

2.2.1.2 Single Event Transient (SET)

As mentioned earlier, the radiation event may induce a voltage excursion at the affected node. This excursion manifests itself as a transient glitch in the output of the affected logic cell, and is called a Single Event Transient (SET).

A SET disappears by itself unless it is captured by a memory element, in which case the SET evolves to a SEU. For the SET to be captured, it must meet certain conditions. First, the SET must be able to affect the input of the memory cell able to capture it. Second, it must affect the input of the memory element during a latching window, i.e., during a period in which the memory element updates its output value. A SET not captured by a memory element is said to be *masked*.

2.2.1.3 Single Event Functional Interruption (SEFI)

In some cases, a soft error may cause the device to reset, lock-up or simply malfunction in a detectable way, without requiring power cycling to restore correct operability. A case fitting this category is given the name Single Event Functional Interruption (SEFI).

SEFIs are usually associated with SEUs in control bits or registers of the device. Additionally, in complex systems in which an Operating System (OS) is present, it is also associated with kernel data corruption.

2.2.1.4 Single Event Latch-up (SEL)

A single energetic particle passing through sensitive regions of the device may lead to abnormal high-current states similar to a electrical latch-up, for example if a particle induces parasitic bipolar (p-n-p-n) shorting between power and ground. This effect is classified as a Single Event Latch-up (SEL).

2.2.1.5 Single Event Burn-out (SEB)

If the supply current is not limited in some way, a SEL may cause lead to a high-current state causing the device to overheat and become permanently damaged - causing

a Single Event Burn-out. This can also happen immediately, but it is far rarer.

2.2.1.6 Single Event Gate Rupture (SEGR)

An ion may also strike the gate region of a powered insulated gate causing a destructive effect, leading to a Single Event Gate Rupture (SEGR).

3 PROCESSOR RELIABILITY

A couple of aspects must be considered in order to evaluate and compare the reliability of systems operating in a radioactive environment. First, it is essential to establish a consistent taxonomy since different names are given to the same effect in the field of reliable systems. Second, it is necessary to adopt a proper metric that reflects all relevant aspects of a system operating in a radioactive environment.

The goal of this chapter is to satisfy the above-mentioned requirements. Section 3.1 introduces the taxonomy used throughout this work and currently used reliability metrics; this section is based upon the works of Avizienis (2004) and Pradhan (1996), and also on the JEDEC (2006) standard. Section 3.2 justifies the need of new metrics for radioactive environments. Additionally, this section introduces an intuitive metric and a formal model to be used as measures of the operative reliability of a system. Section 3.3 presents a mathematical formulation that demonstrates that a smaller sensitive area does not always yield higher reliability.

3.1 Taxonomy and Existing Metrics

The most basic definition is that of a *system*. A system implements a set of functions, and each function is described by a sequence of states, called the system's *behavior*. The system behavior as perceived by an external user of the function is defined as the *service* provided by the system. Whenever the system service deviates from the expected one, a *service failure* (or simply *failure*) is said to have occurred. For a failure to be experienced, first an error must have taken place. By definition, an *error* is a deviation from the expected system behavior and only results in a failure if it is perceived by the user. Finally, the cause of an error is a *fault*, defined as a physical defect; a fault causes an error only if it leads to a deviation in the system's behavior.

As an example consider a SET caused by an impinging high energy neutron. The particle has caused a physical defect, thus characterizing a fault. If a memory element does not capture the SET, the fault is masked and does not activate an error. Otherwise, the fault activates an error, and the SET becomes a SEU. The resulting SEU may cause

the service delivered by the system to deviate from the service expected from a fault-free system, therefore causing a failure. However, the corrupted memory element may be obsolete or may be overwritten with the correct value; the error is then masked, and no failure is activated.

3.1.1 Failures in Time

Given a population of systems, several related parameters to measure their reliability exist. The most basic parameter is the Mean Time To Failure ($MTTF$), which represents the expected time that a system operates before it fails. Given a population of N_s identical systems and tf_i being the time that the i-th system operates before failing, the expected $MTTF$ is given by

$$MTTF = \sum_{i=1}^{N_s} \frac{tf_i}{N_s} \, . \tag{3.1}$$

For a non-reparable system, the $MTTF$ represents the useful life of the device since, once a failure occurs, the entire system is rendered inoperative. However, reparable systems require additional parameters to completely describe their operation cycle, namely the Mean Time To Repair ($MTTR$) and the Mean Time Between Failures ($MTBF$).

As the name implies, the $MTTR$ represents the expected time required after a failure for the system to be operative again. For a set of N_f failures, the expected $MTTR$ can be estimated as

$$MTTR = \frac{\sum_{i=1}^{N_f} tr_i}{N_f} \tag{3.2}$$

where tr_i is the time required to repair the i-th failure. The $MTBF$ is a measure of the time between two failures, i.e., the time after the system has failed for it to be repaired and fail again. Assuming that the system is perfect once again after each repair, the $MTBF$ is simply evaluated as

$$MTBF = MTTF + MTTR \, . \tag{3.3}$$

Moreover, the system failure rate is inversely proportional to the $MTBF$, and since failure rates are often very low, they are commonly reported as Failures In Time (FIT): the number of expected failures per billion hours of device operation - e.g., failures per 1 device per billion hours, failures per 1000 devices per million hours, etc. Given that the $MTBF$ is expressed in hours, the FIT of a system can then be computed as

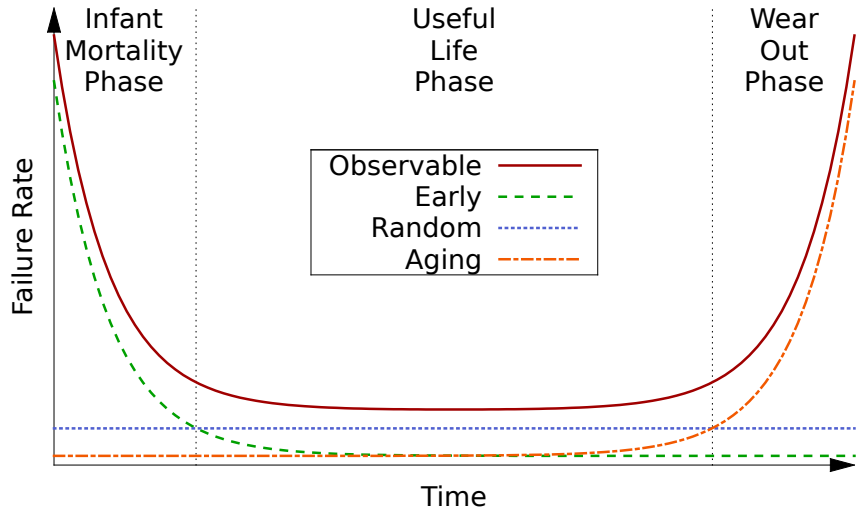
$$FIT = \frac{1}{MTBF} \times 10^9 \, . \tag{3.4}$$

Please notice that the failure rate of a population of systems is not constant. As shown in Figure 3.1, the *observable* failure rate goes through three different phases. During the first phase, called the *infant mortality phase*, the main contributor to the observable

failure rate are *early* failures, i.e., failures that occur due to substandard components and manufacture faults not identified during manufacturing tests. As defective systems are identified and discarded, early failures quickly decrease until the second phase is reached. The second phase is the *useful life phase* of the device, in which the system presents its lowest failure rate. During this phase, the main contributors to the observable failure rate are *random* failures commonly attributed to random effects, such as SEEs. Finally, as components start to degrade, their operating properties change due to aging effects; consequently, leading to an increase in *aging* failures. As aging failures start to be the main contributors to the observable failure rate, the system enters the third and final phase called the *wear-out phase*.

We are mostly interested in the useful life phase of the system since, at this phase, the failure rate is almost constant and the system provides the most predictable service to its users. A burn-in process is typically used to bring the device to this phase before deploying the system; furthermore, the system is normally replaced before it enters the wear-out phase.

Figure 3.1: Observable failure rate during system lifetime and its three components: early, random, and aging failure rates.



Source: author.

3.1.2 Cross-Section

During the useful life phase the main sources of failures are random effects, such as radiation-induced failures. The cross-section is the most widely used metric to evaluate the sensitivity to radiation of a device. By definition the *cross-section*, expressed in cm^2, is the sensitive area of the device – i.e., the area that, if hit by an impinging particle, generates a failure (BAUMANN, 2005). The larger the cross-section, the more likely a radiation-induced failure is. Although the cross-section is expressed in units of area, it

is not a strictly geometrical measure, as it depends not only on the physical size of the employed resources, but also on their sensitivity to radiation and their criticality to the correct execution of the application. As such, the cross-section expresses the probability of having one impinging particle corrupting the processor while executing the given code in such a way that an error appears in the output.

An interesting characteristic of the cross-section is that it represents the intrinsic sensitivity to radiation of the device. Therefore, unlike the $MTBF$, it is not dependent on a particle flux. In fact, given that the system cross-section σ is known, it is possible to evaluate the expected $MTBF$ of the system when exposed to a given particle flux ϕ as

$$MTBF = \frac{1}{\phi\sigma} \, . \tag{3.5}$$

Typically, only the bit cross-section of the device is promptly available, i.e., the probability of a single memory cell being corrupted. A rough estimate of the cross-section of a device with N_{bits} and bit cross-section σ_{bit} is then given by its *static cross-section* σ_{static}, which is evaluated as

$$\sigma_{static} = N_{bits} \times \sigma_{bit} \, . \tag{3.6}$$

The static cross-section is an upper bound of the device error rate and tends to overestimate the real cross-section since it assumes that any corrupted bit will lead to a failure. This assumption is not true since the application may not use the corrupted bit, or the corruption may be masked by logical or temporal factors. In fact, the real cross-section of a system is highly dependent on the algorithm, benchmark or workload that the system is executing.

The cross-section estimation can be further refined by taking into account the Architectural Vulnerability Factor (AVF), i.e., the probability that a fault in the element generates an observable failure (MUKHERJEE et al., 2003). The cross-section can then be approximated as

$$\sigma = \sigma_{static} \times AVF \, . \tag{3.7}$$

Although it is possible to estimate the AVF through fault injection experiments, the estimation requires a deep knowledge of both the circuit description and physical construction of the device, which are not readily available.

In practice, the cross-section is typically evaluated through radiation experiments. Normally, particle accelerators, neutron beams, or radioactive sources are employed to evaluate the device cross-section experimentally, as they provide a controlled and typically high flux, while real tests in the field are very expensive and time-consuming (ZIEGLER et al., 1996). By exposing the system while executing an application as representative as possible of the application that the processor is going to perform when deployed, the cross-section σ can then be obtained experimentally by dividing the num-

ber of observed errors by the total particle fluence Φ (JEDEC, 2007). In other words, the cross-section may be evaluated as

$$\sigma = \frac{\lambda}{\phi} \tag{3.8}$$

where λ is the observed output error rate (i.e., errors per unit time) and ϕ is the particle flux (i.e., number of particles hitting the device per unit area and unit time). The resulting cross-section provides a precise indication of the sensitivity of the processor executing the given code.

3.2 Proposed Metrics

A widely used method to increase radiation reliability is to reduce the processor cross-section by disabling or reducing resources which are not essential for the computation, such as cache memories. Nevertheless, the execution time of the code varies consistently when the processor resources are changed; caches, for instance, were introduced specifically to reduce memory access latency and increase the performance of processors. On a realistic application, the exposure time of the code is modified when the available resources are limited. Naturally, using more resources the code finishes the task at hand faster with respect to the code with limited resources, i.e., from a radiation reliability perspective, using more resources the code is exposed for a much shorter time. As a result, enabling extra resources increases the probability of an impinging particle generating a failure (i.e., the cross-section) but reduces the number of particles that hit the device during computation. The shorter exposure time may then compensate for the larger exposed area, effectively increasing system reliability.

The basic concept explored in this work is that the probability P of executing an application correctly is inversely related not only to the device sensitivity, expressed by its cross-section σ, but also to the total exposure time t, which is determined by the time the device requires to execute the application. Hence, assuming designers can not modify the flux experienced by the system, minimizing both sensitivity and execution time are the two main approaches to increase P. However, these two approaches are often conflicting since the most common mechanisms to improve t have a negative impact on σ.

As an example, a common practice to reduce σ is to disable caches on a processor (LESEA et al., 2014; REBAUDENGO; SONZA REORDA; VIOLANTE, 2003). Nevertheless, memory hierarchies also play a key role in the performance. Thus, even if σ is significantly reduced, t is going to increase, and no assumption on P could be given a-priori.

Parallel processors serve as a second example of the trade-off between σ and t. Exploiting parallelism naturally introduces additional sensitive area due to the very definition of parallel execution. However, the code execution time benefits from parallelism,

potentially increasing P (RECH et al., 2014).

Similarly, increasing the processor frequency may increase its cross-section. In fact, higher clock frequencies increase the sensitivity to SETs since they increase the probability of sampling glitches generated by impinging particles (BAUMANN, 2005). Nonetheless, a higher frequency allows the processor to reduce the execution time t, and, thus, may increase P.

Furthermore, even software implementation choices present a similar trade-off. For example, the usage of lookup tables speeds up execution at the cost of an increase in memory usage. Although t is reduced, a higher memory usage means more memory cells exposed to radiation and, thus, a larger σ.

Thus, in order to compare different solutions in terms of their reliability, an appropriate metric to evaluate their operative reliability taking both the cross-section σ and application execution time t into account is required.

3.2.1 Mean Workload Between Failures

The first proposed metric of this dissertation is the Mean Workload Between Failures ($MWBF$), which is an intuitive metric. As the name implies, the $MWBF$ is the average workload the application elaborates in-between failures.

Start by considering the Mean Time Between Failures $MTBF$ of a processor executing a code, as given by Eq. 3.5. A higher $MTBF$ simply indicates that the processor could work for a longer period before experiencing a radiation-induced error. Nevertheless, no information on the workload computed during that period is given yet.

To evaluate how much useful workload has been correctly processed by the processor during the $MTBF$ window, a new metric called Mean Executions Between Failures ($MEBF$) is introduced. The $MEBF$ is defined as the average number of correct executions of an application that are successfully completed between two radiation-induced failures, which can be easily evaluated by dividing the $MTBF$ by the application execution time t:

$$MEBF = \frac{MTBF}{t} \, . \tag{3.9}$$

Each application execution is characterized by a workload w, i.e. the amount of data that needs to be processed in a single execution. The $MEBF$ can then be further generalized to take the workload computed correctly into account, leading to the concept of Mean Workload Between Failures $MWBF$:

$$MWBF = MEBF \times w \, . \tag{3.10}$$

A higher $MWBF$ actually means that a higher workload was computed correctly before experiencing a failure and, thus, that the operational reliability is higher.

3.2.2 Probability of Executing without Failures

The second proposed metric of this dissertation is the probability of executing without failures. Naturally, a higher probability of executing without failures is equivalent to a higher reliability.

Radiation-induced faults are stochastic independent events, thus yielding a constant failure rate. These faults are also rare when compared to the completion of a program run (i.e., the mean time between failures is much larger than the time it takes to execute the program once). Therefore, it is reasonable to assume radiation-induced errors distributed over time following a Poisson distribution. The probability P of having k failures during the execution time t can then be expressed as

$$P(k, \lambda, t) = \frac{e^{-\lambda t}(\lambda t)^k}{k!} \tag{3.11}$$

where λ is the application failure rate.

In this work, we are interested in finding the probability of executing the code successfully (i.e., without failures). Consequently,

$$k = 0 \ , \tag{3.12}$$

and the probability of executing the application successfully is

$$P(0, \lambda, t) = \frac{e^{-\lambda t}(\lambda t)^0}{0!} = e^{-\lambda t} \ . \tag{3.13}$$

While the execution time t can be directly measured from the running application, the failure rate λ can be inferred from Eq. 3.8 as

$$\lambda = \phi\sigma \ . \tag{3.14}$$

As a result, for any given flux ϕ, the *probability of executing the application without failures* can be evaluated through

$$P(\phi, \sigma, t) = e^{-\phi\sigma t} \ , \tag{3.15}$$

which can then be used as a metric of operative reliability.

Furthermore, please notice that the probability of executing the application without failures is strictly related to $MEBF$. To show this relation, consider Eq. 3.9 rewritten as

$$MEBF = \frac{MTBF}{t} = \frac{1}{\phi\sigma t} \ ; \tag{3.16}$$

thus, Eq. 3.15 can be rewritten as

$$P(\phi, \sigma, t) = e^{-1/MEBF} \ . \tag{3.17}$$

Similarly, it is possible to define the *probability of executing a workload unit without failures* as

$$P(\phi, \sigma, t) = e^{-1/MWBF} \ . \tag{3.18}$$

3.3 Increasing Cross-section and Reliability

The main goal of this work is to confirm the hypothesis that enabling cache memories may increase system reliability for a state-of-the-art COTS microprocessor.

We have divided this task into two parts. First, since the main reason to disable caches is that they increase the sensitive area of the device, in this section we show that it is possible to increase reliability even if the sensitive area of the device is increased, i.e., that a smaller cross-section does not always yield higher reliability. Then, in Chapter 5, we show experimentally that this is indeed feasible for a state-of-the-art COTS microprocessor in practical terms.

Theorem 1. *Enabling extra resources improves the system radiation reliability if the increase in the sensitive area they incur is smaller than the speed-up they provide.*

Proof. Consider two candidate solutions (*a* and *b*) for the same task, where solution *a* increases σ but decreases t, and solution *b* decreases σ but increases t. Let σ_a and t_a denote respectively the cross-section and execution time for solution *a*, and σ_b and t_b the same variables for solution *b*. When comparing these two solutions in terms of reliability, we can evaluate if solution *a* is more reliable than solution *b* when exposed to a given flux ϕ through

$$P(\phi, \sigma_a, t_a) > P(\phi, \sigma_b, t_b) \ . \tag{3.19}$$

This equation can be reduced to

$$e^{-\phi\,\sigma_a t_a} > e^{-\phi\,\sigma_b t_b} \tag{3.20}$$

$$\phi\,\sigma_a t_a < \phi\,\sigma_b t_b \tag{3.21}$$

$$\frac{\sigma_a}{\sigma_b} < \frac{t_b}{t_a} \ . \tag{3.22}$$

Therefore, as shown in Eq. 3.22, solution *a* has a higher operative reliability when the observed increase in the sensitive area brought by *a*, in comparison to *b* (the left-hand side of Eq. 3.22), is lower than the speed-up provided by *a* relative to *b* (the right-hand side). □

4 CACHE MEMORIES

Ideally memory would be large enough to contain any program, and any memory word would be immediately available. Nevertheless, this is unrealistic as the larger the memory, the slower it is to fetch a word from it. As a result, the CPU is forced to wait for the slower memory whenever new data or instructions are needed, slowing down program execution. To cope with this limitation, designers use the concept of cache memories to create the illusion of a large memory that can be accessed as if it were a tiny one.

The aim of this chapter is to examine cache memories from a reliability perspective. Section 4.1 briefly introduces the operation mechanism of cache memories; this section is largely based on the work of Hennessy (2011) and Patterson (2013). Section 4.2 presents related works conducted in the field of unprotected cache memories reliability. Furthermore, this section analyzes cache memories in light of Theorem 1.

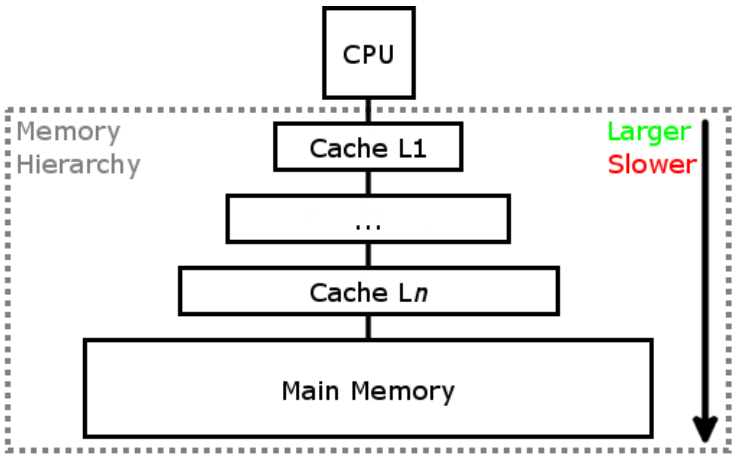
4.1 Cache Operation

Cache memories are buffer memories located between the processor and the main memory, forming a *memory hierarchy*. The memory hierarchy may have any number of cache levels; the upper cache level (L1) is closer to the CPU, and each subsequent level (e.g., L2) increases in capacity and access time, i.e., is *larger* and *slower* (see Figure 4.1).

The minimum unit of information that is transferred between different levels of cache memories is called a *line*, which contains a fixed number of adjacent memory words. The basic mechanism of a cache is as follows: whenever access to a memory address is requested, the cache verifies if the line containing the address is present at this cache level. If the pertinent line is found within this cache level (i.e., a *cache hit* occurred), the access is fulfilled. Otherwise, if the pertinent line is not found (i.e., a *cache miss* occurred), the cache copies the relevant line from the next lower memory level and then fulfill the request. Therefore, a cache miss incurs in a penalty on the access time equal to the access time of the next memory level.

For example, consider a memory hierarchy with two cache levels (L1 and L2), and a

Figure 4.1: Memory hierarchy with *n* cache levels.



Source: author.

main memory that contains the program in its entirety. The CPU accesses the L1 cache. If the L1 cache access hits, the access is fulfilled, and the access time is equal to the L1 cache latency. Otherwise, if the access to the L1 cache misses, the L1 cache accesses the L2 cache incurring in a penalty in access time equal to the L2 latency. Similarly, if the L2 cache access also misses, the L2 cache accesses the main memory, adding a penalty equal to the main memory access time.

Cache memories operate on the principles of temporal and spatial locality. Due to temporal locality, a memory position is likely to be referenced multiple times in a short time window; thus, after the first time the position is accessed, it makes sense to keep the word on a low-latency buffer memory. The spatial locality states that it is highly probable that nearby memory positions are accessed close in time; thus, when accessing a memory position, it is rational to also bring nearby positions (i.e., the cache line) to the low-latency buffer memory.

4.2 Cache Reliability

According to Webster's New World Dictionary of the American Language, a cache is "a ***safe*** place for hiding or storing things". Ironically, the data residing in cache memories may not be safe nowadays. In fact, cache memories are now considered the parts of modern computing systems most vulnerable to SEEs (MANOOCHEHRI et al., 2011; MUKHERJEE et al., 2004; LIDEN et al., 1994). These memories represent around 60% of the on-chip area in current microprocessors (MANOOCHEHRI et al., 2011), and, to be fast and efficient, cache memory cells are built as small as possible; thus, their capacitance and critical charge are lowered, hence increasing the probability of having the cell corrupted by ionizing radiation (BAJURA et al., 2007).

Protection techniques that may be applied to cache memories, including parity and ECC, incur in extra area and significant additional power consumption, and even lead to performance degradation (ASADI et al., 2005). It is worth noticing that the probability of having a Multiple Bit Upset (i.e., one impinging neutron corrupting more than one memory cell belonging to the same word) is increasing with the shrinking of the technology node (IBE et al., 2010; MAIZ et al., 2003). Moreover, caches are compact and dense; thus, the memory cells are close to each other, exacerbating the possibility of having one single impinging particle interacting with more than one transistor. This results in the ECC design becoming more and more challenging. Although it has been shown that Multiple Bit Upsets could be mitigated by the use of memory interleaving (BAEG; WEN; WONG, 2009), there is no guarantee that such a hardware support (or even ECC) will be available in COTS embedded systems. As a result, a classic solution to reduce the occurrence of radiation-induced failures in COTS devices is to disable cache memories (LESEA et al., 2014; REBAUDENGO; SONZA REORDA; VIOLANTE, 2003).

4.2.1 Related Work

The reliability of unprotected cache memories has been investigated in previous works, albeit, to the best of the authors' knowledge, none takes performance into account to evaluate reliability.

Reabaudengo et al. (2003) injected SEUs in the pipeline registers, register file, instruction cache and data cache by instrumenting the HDL code of the Leon core (Gaisler, 2014). The number of faults injected was proportional to the execution time of the application, and the failure rate of four applications was measured. A comparison was made between the Leon core with L1 caches enabled and disabled, showing that enabling the L1 caches increased the failure rate by a factor ranging from 5.46 to 24.70.

Asadi et al. (2005; 2006) have shown that the sensitivity of different cache configurations is very diverse, based on their vulnerability factors, as experimentally obtained with an extended version of the SimpleScalar simulator (BURGER; AUSTIN, 1997), but their relation to the core's vulnerability was not evaluated.

Cai et al. (2006) analyzed cache size impact on time-constrained systems; the authors show the existence of an optimal or Pareto-optimal cache size when optimizing for energy, performance, and reliability. The evaluation was realized on MPARM (BENINI et al., 2005), a cycle-accurate simulator.

4.2.2 Spatio-Temporal Analysis

Based on the new metrics proposed in Section 3.2, it is possible to analyse the reliability of cache memories from both spatial (i.e., sensitive area) and temporal (i.e., performance) perspectives. Theorem 1 can be directly applied to cache memories. Therefore, we are interested in finding out if the speed-up provided by each cache level can be larger

than the increase in sensitive area they incur, which is done empirically in Section 5. Nonetheless, there are additional insights to be discussed beforehand.

The first insight is on the differences to be expected between different cache levels. Consider a system with two cache levels. Typically, the first cache level is smaller and focuses on minimizing hit time, while the secondary cache level tends to be much larger and focuses on reducing miss rates (PATTERSON; HENNESSY, 2013). Furthermore, since the primary cache filters accesses with good locality, the local miss rate of the secondary cache is much higher than the global miss rate. The first implication of this difference between the cache levels is on speed-up provided: intuitively, the speed-up provided by the primary cache (relative to the system with no caches) is likely to be much larger than the speed-up provided by the secondary cache (relative to the system with only a primary cache). The second implication of this difference is on the sensitive area incurred by each level: since smaller caches are less vulnerable than larger caches (ASADI et al., 2005), the increase in sensitive area due to the secondary cache level is larger than the increase in sensitive area due to the primary cache. Therefore, the primary cache level provides a much higher speed-up per increase in sensitive area ratio than the secondary cache level, making it more likely to increase reliability than the secondary cache level.

The second insight relates to the increase in the processor-memory performance gap. With processor performance increasing at roughly 50% per year, while DRAM performance improves at a smaller rate of around 7% per year (LU et al., 2012), the processor-memory performance gap keeps increasing. Although cache memories are a very effective scheme to reduce the average memory access latency based on temporal and spatial locality, the cache-provided speed-up does not increase indefinitely as the processor-memory performance gap increases. As a result, there is an upper limit for the increase in the sensitive area that can be compensated by the cache-provided speed-up even if the processor-memory performance gap continues to increase indefinitely.

Consider a system containing a processor clocked at a frequency F, a primary cache level with separate caches for instructions and data, and a main memory with access time of C_{mem} processor cycles. With caches enabled, an application requiring N instructions to complete, yielding an average of CPI cycles per instruction, an instruction cache miss rate of i_{miss}, a data cache miss rate of d_{miss}, and a percentage β of instructions that access memory, takes t_c units of time to complete, where

$$t_c = \frac{N}{F}.(CPI + i_{miss}.C_{mem} + \beta.d_{miss}.C_{mem}) \ . \tag{4.1}$$

If one disables the cache, the execution time t_{nc} is then given by

$$t_{nc} = \frac{N}{F}.(CPI + C_{mem} + \beta.C_{mem}) \ , \tag{4.2}$$

and, therefore, the cache-provided speed-up S is

$$S = \frac{t_{nc}}{t_c} = \frac{(CPI + C_{mem} + \beta.C_{mem})}{(CPI + i_{miss}.C_{mem} + \beta.d_{miss}.C_{mem})} \tag{4.3}$$

The memory-processor gap increase reflects itself in C_{mem}. For very large values of C_{mem}, the speed-up S is given by

$$\lim_{C_{mem} \to +\infty} S = \frac{1 + \beta}{i_{miss} + \beta.d_{miss}} \tag{4.4}$$

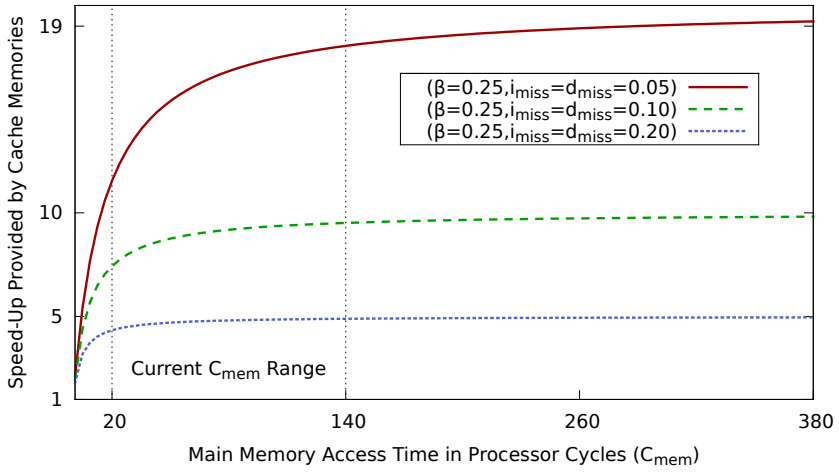
and is limited by the miss rates of both instruction and data caches. Therefore, combining Eq. 3.22 and Eq. 4.4 yields the upper bound

$$\frac{\sigma_c}{\sigma_{nc}} > \frac{1 + \beta}{i_{miss} + \beta.d_{miss}}\,, \tag{4.5}$$

after which the increase in the sensitivity can no longer be compensated, even with the increase in the memory-processor gap.

Nowadays, access time for cache memories (i.e., SRAMs) ranges from 0.5 ns to 2.5 ns, while main memory (i.e., DRAM) access time ranges from 50 ns to 70 ns (PATTERSON; HENNESSY, 2013). Assuming that the processor operates at the same speed as the cache memories, then the expected value of C_{mem} currently lies between 20 and 140. Taking this range into consideration, Figure 4.2 shows how the speed-up for different cache miss rates approximates its limit as C_{mem} increases, hinting at how much one can expect the cache-provided speed-up to increase in the future.

Figure 4.2: C_{mem} impact on application speed-up due to cache memories.



Source: author.

It is worth noting that the sensitivity ratio (σ_c/σ_{nc}) may not remain unaltered as C_{mem}

increases. Although an increase in C_{mem} does not directly affect this ratio, it is a byproduct of the technology shrinking, which does affect the way logic devices react to radiation. Nonetheless, there is no clear pattern on how this shrinking affects the sensitivity ratio. For instance, if the sensitivity of logic elements keeps increasing more than the sensitivity of memory cells, as predicted by Baumann (2005), while cache size remains static, σ_{nc} is likely to increase more than σ_c, thus reducing the sensitivity ratio; however, if the cache size is greatly increased, the opposite is true - σ_c is likely to increase more than σ_{nc}, thus increasing the sensitivity ratio.

5 CASE STUDY

In Section 3.3 we have demonstrated that, in theory, it is possible to increase reliability even if the sensitive area of the device is increased.

The objective of this chapter is to show that, in practice, it is feasible for the cache-provided speed-up to overcome the increase in the sensitive area they incur for a state-of-the-art COTS microprocessor, therefore increasing system reliability. To fulfill this objective, we performed an extensive neutron test campaign on embedded ARM processors according to the guidelines provided by the JEDEC (2006) standard for beam accelerated soft error rate testing. The following sections detail relevant aspects of the test campaign, culminating with the presentation and analysis of the obtained results.

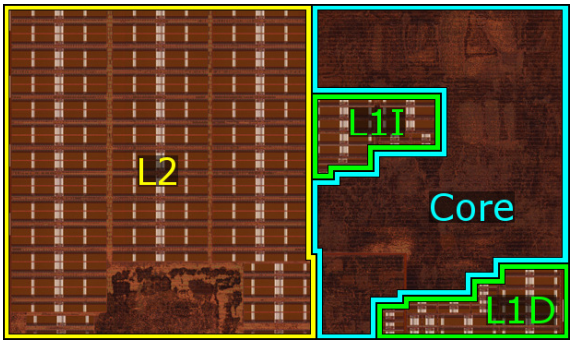
5.1 Device Under Test

The Device Under Test (DUT) is the Xilinx XC7020-1CLG484C Zynq™-7000 AP SoC implemented in a $28\,\mathrm{nm}$ CMOS technology. The DUT is mounted on the Zedboard development kit and disposes of 2 ARM®Cortex™-A9 cores with a maximum frequency of $667\,\mathrm{MHz}$. Each core has $32\,\mathrm{KB}$ Level 1 4-way set-associative instruction and data caches, and they share a $512\,\mathrm{KB}$ 8-way set-associative Level 2 cache (Digilent, 2014a). Both cache levels have parity support, which was disabled during our experiments; assuming single-bit corruptions and that detected faults are observable errors, enabling parity support would increase the observable error rate due to the detection of benign faults (i.e., faults that would otherwise not impact application execution would raise exceptions if parity was enabled). During experiments, only the caches and one of the two available cores are used; the resulting device under test is similar to that of Figure 5.1.

5.2 Tested Configurations

The multiple cache levels available in state-of-the-art embedded processors introduce different options for designers. For the DUT, which has L1 and L2 caches, it is reasonable to assume that there is a given configuration, such as using no caches, only L1 caches, or both cache levels, that optimizes reliability (in terms of exposure time and sensitive area).

Figure 5.1: Approximate area of essential resources for computation (Core) and extra resources used to speed-up execution (L1 and L2 Caches) marked on a polysilicon die photo of a Cortex-A9 with 32 KB Instruction/Data L1 Caches and 512 KB L2 Cache.



Source: author.

It is not obvious, however, which is the optimum choice. The optimal configuration, as defined by Eq. 3.22, depends on the sensitive area introduced by each enabled level and on the speed-up provided by each level as well, which are functions of both application and platform properties.

To experimentally measure how the DUT reliability varies as extra resources are enabled, we generated the same application in three versions: 1) all cache levels disabled, 2) only L1 caches enabled, and 3) both L1 and L2 caches enabled. Only one of the two available cores is allowed to run to avoid cache access conflicts, since the L2 cache is shared. This core then disables parity at the L2 cache, sets a dummy function as the L1 parity interruption handler, and sets the cache hierarchy configuration chosen at compilation time. After this initial setup, a banner is printed, and the application executes repetitively. The banner is used to detect reboots and make sure that the correct cache configuration is being used.

5.3 Application Under Test

The identification of a single specific instance of execution of the DUT in which reliability is improved even if the sensitive area is increased leads to two conclusions. First, a single instance suffices as a counterexample to refute the hypothesis that reducing the cross-section always increases reliability. Second, this instance shows that, for a state-of-the-art COTS microprocessor, cache memories may speed-up the execution enough as to compensate for the increase in the sensitive area they incur. Furthermore, no requirement is necessary on the complexity of the application. Thus, based on these criteria and the fact that neutron beam time is limited, we opted for focusing on a single application during our experiments, since this allows us to minimize the statistical error of our results.

Matrix multiplication was selected as the benchmark since it is a common workload in

critical embedded systems used, for example, in control and filter operations. The detection of incorrect answers is achieved by comparing the computed results of multiplying input matrices A and B with a golden copy G. This verification is done in the DUT itself since this is faster than moving the resulting matrix to be verified elsewhere. Please note that in both cases (during the verification or the transmission of the data) the device is still exposed to radiation and failures may occur during this period. The resulting errors are no different from errors during the computation part of the application and, thus, are taken into account as well. Nevertheless, the verification against the golden copy represents only a small part of the application execution time ($\approx 3\%$) and its contribution to the error rate can be considered negligible.

To increase all caches' utilization, we designed a code in which 200 multiplications of 25×25 integer matrices ($25 \times 25 \times 4 = 2500$ Bytes/Matrix) are executed. Each multiplication has a private set of input matrices A, B and golden copy G ($2500 \times 3 = 7500$ Bytes/Set), amounting to at least $200 \times 7500 \approx 1.43$ MB of data. As shown in Algorithm 1, the application starts after DUT initialization: all matrices are defined as local variables in the stack, and for each of the 200 sets of input matrices A_i is multiplied by B_i; the resulting matrix is then compared to the expected result G_i and, if they differ, a failure flag is raised. After the 200 matrix multiplications are completed, errors are reported, and a new application execution is triggered.

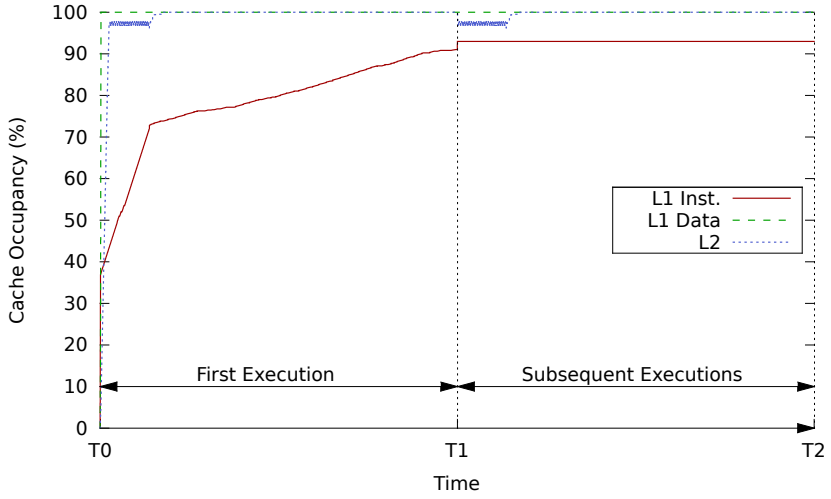
Algorithm 1: Application under test.

```
setup_caches();
print_banner();
while True do
    // Application Start
    Fail ← False;
    for i ← 1 to 200 do // Unrolled
        init_in_stack(A_i, B_i, G_i);
    end
    for i ← 1 to 200 do // Unrolled
        C ← A_i * B_i;
        if C ≠ G_i then
            Fail ← True;
        end
    end
    print(Fail);
    reset_stack_pointer();
    // Application End
end
```

To evaluate the approximate cache occupancy (i.e., number of valid blocks per total number of blocks) during application execution we used a modified version of the *gem5* simulator (BINKERT et al., 2011) using a configuration similar to the DUT. Figure 5.2

shows the resulting instantaneous cache occupancy for the L1 instruction, L1 data, and L2 caches. After a reboot, the cache occupancy is zero due to the cold start. During the first execution, the L1 data cache is quickly filled up due to data initialization and remains with 100% occupancy for the rest of the time. The L2 cache reaches close to 98% occupancy during data initialization, reaching 100% occupancy quickly afterwards. The L1 instruction cache is filled during the whole period of application execution, signaling that different instructions are used throughout the execution, saturating around 92.97% occupancy. These cold executions represent only an insignificant amount ($< 0.03\%$) of the total executions. During subsequent executions, the L1 data and instruction caches remain saturated around 100% and 93%, respectively, while the L2 cache has a few lines invalidated at the beginning of the execution and quickly reaches 100% occupancy again.

Figure 5.2: Instantaneous cache occupancy for the first and subsequent executions.



Source: author.

Table 5.1 shows the average cache occupancy and the time it takes to execute the proposed benchmark once for each of the three cache hierarchy configurations. As expected, when both L1 and L2 caches are enabled, the performance of the DUT is higher, being almost eight times faster than the version in which all caches are disabled. When only the L2 cache is disabled, but the L1 caches remain enabled, the code latency is increased by just 10%. It is worth noticing that in this particular case the application speed-up is not achieved by increasing the processor frequency, which would increase SET occurrences (BAUMANN, 2005). The frequency of operation of the DUT is constant, and the longer execution time observed when L2 or both L1 and L2 are disabled is caused only by memory access latency. Moreover, having a longer execution time increases the exposure time of data stored in internal registers; while data are fetched from main memory, registers remain exposed to radiation, and the data held within those registers are likely to be used

for further computations. Thus, a failure in the register holding data while the processor is waiting because of memory latency is likely to propagate to the output, generating an output error. Remaining resources such as portions of the speculative state may also vary, although in a less predictable manner, due to the execution of more instructions in parallel.

Table 5.1: Application performance with different cache configurations.

Version	**Execution**	**Average Occupancy (%)**		
	Time (s)	L1 Inst.	L1 Data	L2
No Cache	5.04×10^{-1}	N/A	N/A	N/A
L1	7.10×10^{-2}	92.97	100	N/A
L1 + L2	6.40×10^{-2}	92.97	100	99.57

Source: author.

5.4 Facility and Neutron Source

Radiation experiments were performed at Los Alamos National Laboratory (LANL) Los Alamos Neutron Science Center (LANSCE) Weapons Neutron Research (WNR) Facility Target 4 Flight Path 30R, called Irradiation of Chips and Electronics (ICE II).

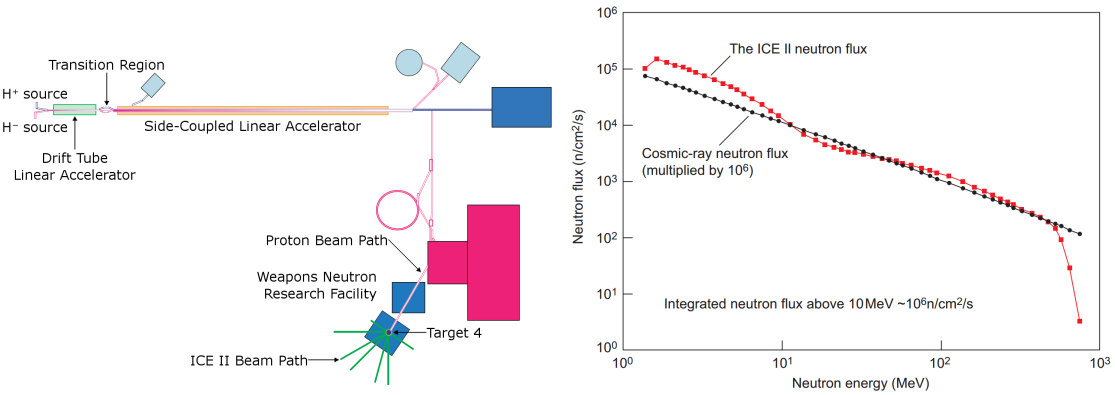
The heart of LANSCE is an $800\,\mathrm{MeV}$ half-mile-long linear accelerator, which produces a pulsed proton beam. At the WNR facility, the proton beam hits an unmoderated tungsten target (*Target 4*), producing six neutron beamlines via spallation reactions. The ICE II facility is located on the 30° right beamline relative to the proton beam (see Figure 5.3a). As shown in Figure 5.3b, the ICE II beamline produces a neutron flux with an energy spectrum very similar to the atmospheric neutron flux produced by cosmic-rays and yields a neutron flux of approximately $1\times10^{6}\,\mathrm{n/(cm^2s)}$ for energies above $10\,\mathrm{MeV}$.

5.5 Experimental Setup

Figure 5.4 shows the position of board components relative to the beam. The beam was focused on a spot with a diameter of 2 inches, which provided uniform irradiation of the SoC without directly affecting nearby board power control circuitry. The DRAM chips were located outside of the beam; thus, we assume no errors are generated within them.

The overall SoC silicon die was then irradiated, thereby taking into account the sensitivities of the core and caches. Furthermore, the individual sensitivities of different components of the system are automatically and implicitly taken into account by the experiment. Since the memory cells used in register files, pipelines and caches are likely

Figure 5.3: ICE II information.

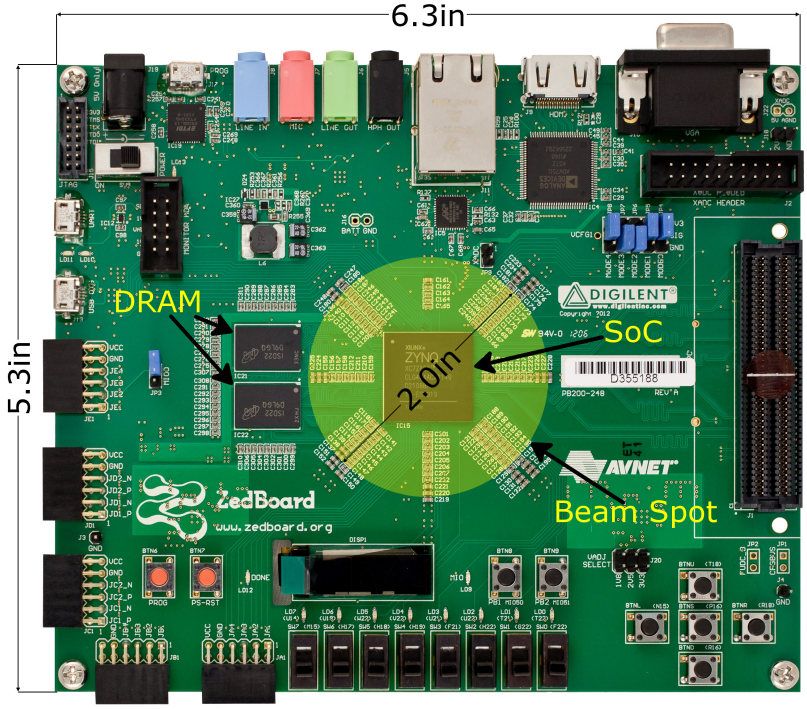


(a) ICE II location within the LANSCE.

(b) Similarity between the energy spectra from the ICE II and atmospheric neutron fluxes.

Source: (LANL, 2014), edited by the author.

Figure 5.4: Beam spot position.

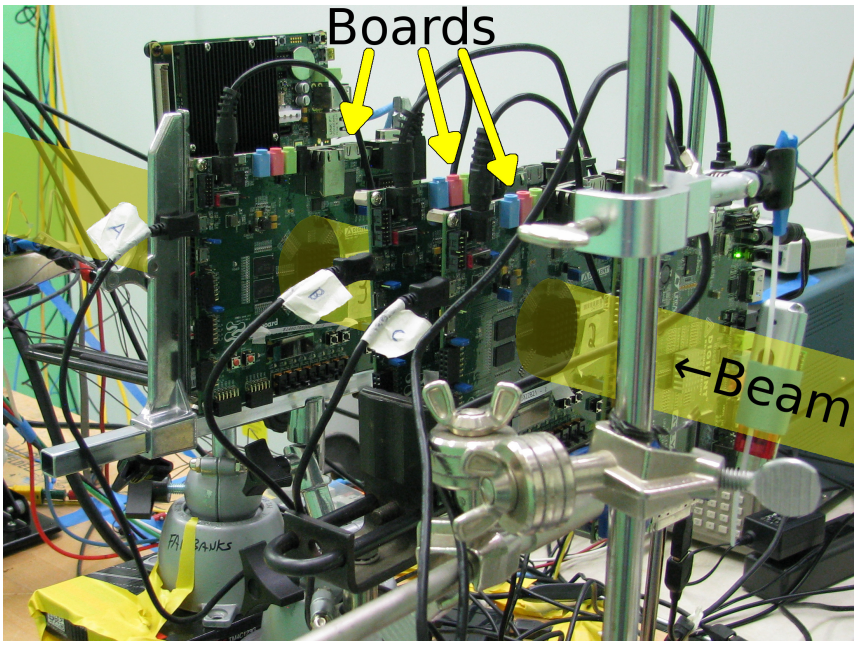


Source: (Digilent, 2014b), annotated by the author.

to have different dimensioning, their cross-section is unlikely to be the same. The conducted experiments accurately capture these subtle differences to provide reliability measurements.

To reduce the uncertainty in the experimental results, we irradiated in parallel three DUTs (see Figure 5.5), running in each the same application but with a specific cache hierarchy configuration: 1) all caches disabled, 2) only L1 enabled, and 3) both L1 and L2 enabled.

Figure 5.5: Experimental setup mounted at ICE II.



Source: author.

Irradiation was performed with normal incidence; experiments were conducted at a constant room temperature, and the boards were powered with a nominal voltage of $12\,\mathrm{V}$. Three boards with the same hardware revision (namely, Rev. C) were aligned with the beam and placed at 61, 62, and 65 inches from the source, respectively. A beam flux de-rating factor was calculated for each board so as to take the beam degradation due to distance from the source into account. To minimize the statistical error and to avoid experimental results bias on the selected board and distance de-rating factor, the codes with the three different cache hierarchy configurations running the aforementioned application were executed alternatively in all three devices.

Each version was executed for more than 80 hours under the beam, receiving a total fluence of $2\times10^{11}\,\mathrm{n/(cm^2)}$ for energies above $10\,\mathrm{MeV}$. As at sea level the natural neutron flux is estimated to be about $13\,\mathrm{n/(cm^2h)}$ (JEDEC, 2006), during our experiments the boards received the equivalent to 1.7×10^6 years of exposure in the natural environment. It is worth noticing that even if the flux of neutrons in LANSCE is several orders of magnitude higher than the natural one, the application workload and run time were tuned

to make negligible the probability of having more than one neutron generating a failure in one single code execution (estimated to be no higher than 1.41×10^{-7} through (Eq. 3.11) with $k > 1$). This small probability allows the scaling of experimental data in the natural radioactive environment without introducing artificial behaviors.

During experiments, a test manager application was responsible for collecting and time-stamping incoming logs from the boards through UART connections. This application was also responsible for detecting otherwise irrecoverable failure situations and rebooting the boards. Whenever such situations happened, they were time-stamped and logged as well. Irrecoverable situations are considered when the board exceeds a time-out much larger than the application execution time without sending successful execution logs.

5.6 Experimental Results

Results were extracted from the logs, and each application execution was classified as follows:

Correct: the application produced the expected output of a fault-free environment. No error was detected when comparing to the golden copy.

Silent Error: the application produced a different output than that of a fault-free environment. This includes errors detected by comparing the output to the golden copy, irrecoverable situations, and whenever garbage is found in the output (i.e., the UART or its flow control were corrupted).

Functional Interruption: a reboot was detected, without an explicit reboot order from the host PC.

The total number of executions is given by the sum of the classified executions. Table 5.2 summarizes the obtained experimental results, indicating the number of correct executions, silent errors, functional interruptions, and the duration of the experiment. The durations of the experiments for each version differ because up to two boards were temporarily used for a different experiment. We attributed the same weight to silent errors and functional interruptions when calculating the overall DUT cross-section. Table 5.3 shows the resulting cross-section for silent errors, functional interruptions, and the overall DUT cross-section when executing the benchmark for each cache hierarchy configuration. The values are shown with relative intervals to account for the neutron count uncertainty. The cross-section is evaluated by dividing the experimentally observed error rate by the flux, representing then the probability for an impinging neutron to generate an observable failure (i.e., the cross-section indicates the sensitive area for each configuration).

As shown in Figure 5.6, if just the cross-section is used as a metric to evaluate the code to be used in a safety-critical application, the version without caches is pronounced

Table 5.2: Result summary.

Version	Correct Executions	Silent Errors	Functional Interruptions	Experiment Duration (h)
No Cache	7.25×10^5	35	13	102.96
L1	5.40×10^6	221	35	108.00
L1 + L2	4.51×10^6	1092	38	85.44

Source: author.

Table 5.3: Cross-sections for each configuration (in cm^2).

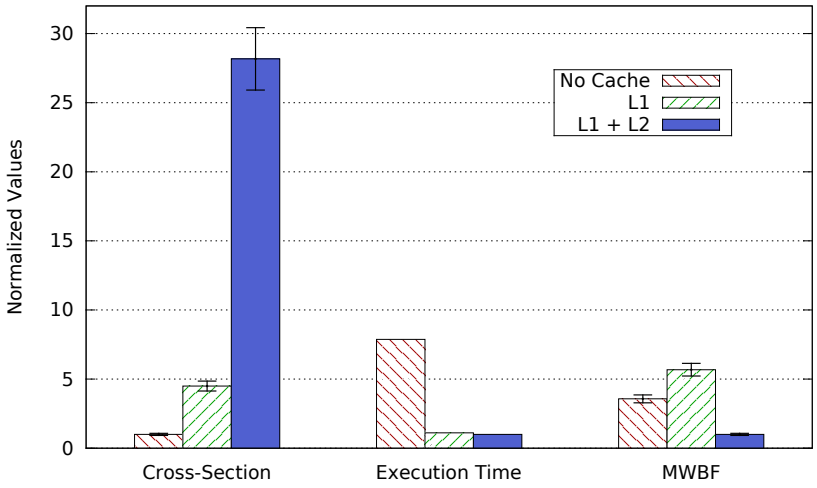
Version	Silent Error	Func. Interruption	Overall
No Cache	$(2.22 \pm 0.18)\times10^{-10}$	$(7.22 \pm 0.58)\times10^{-11}$	$(2.94 \pm 0.24)\times10^{-10}$
L1	$(1.15 \pm 0.09)\times10^{-9}$	$(1.82 \pm 0.15)\times10^{-10}$	$(1.33 \pm 0.11)\times10^{-9}$
L1 + L2	$(8.03 \pm 0.64)\times10^{-9}$	$(2.76 \pm 0.22)\times10^{-10}$	$(8.31 \pm 0.66)\times10^{-9}$

Source: author.

the more reliable, since it has a smaller cross-section. Nevertheless, the execution time of the codes must be taken into account as a longer execution time increases the number of particles hitting the device, potentially undermining the benefit in terms of reliability that a reduced cross-section brings. The $MWBF$, as defined in Section 3.2, conveys this idea and can be used to predict which version is most reliable. In this case study, the $MWBF$ shows that, in fact, enabling only the L1 caches optimizes reliability.

Moreover, embedded applications seldom execute a single time. Figure 5.7 shows the probability of executing without failures according to Eq. 3.15, in terms of consecutive executions when exposed to the approximate neutron flux at military aircraft altitude (Altitude $\approx 60\,000\,\mathrm{ft}$; Flux $\approx 4680\,\mathrm{n}/(\mathrm{cm}^2\mathrm{h})$ for energies above $10\,\mathrm{MeV}$ (QUINN; GRAHAM, 2005)). As a reference, we consider DARPA's Vulture program, whose goal is to enable an Unmanned Aerial Vehicle (UAV) capable of operating uninterrupted for more than five years at altitudes higher than $60\,000\,\mathrm{ft}$ (DARPA, 2014). A small fleet of 10 such UAVs during a two-year mission translates to a workload of roughly 1.25 billion executions of the studied application with caches disabled. The results from our experiments show that enabling only the L1 caches increases the probability of completing the assigned workload without failures by 5.85% when compared to the configuration in which all caches are disabled (from 0.831 to 0.889), while enabling both L1 and L2 caches actually reduces it by 31.59% (from 0.831 to 0.515). As predicted by the $MWBF$, the highest reliability is achieved when L1 is enabled, which is not the configuration with the smallest cross-section. It is worth noting that as the number of consecutive executions increases so does the reliability gap between the considered configurations, and the configuration with only the L1 cache enabled becomes increasingly more reliable with respect to the other

Figure 5.6: Measured impact of different cache hierarchies on the cross-section, execution time, and $MWBF$ of the application under test.



Source: author.

two configurations.

Figure 5.7: Estimated probability of multiple executions without failures at military aircraft altitude ($\phi = 4680\,\mathrm{n}/(\mathrm{cm}^2\mathrm{h})$).

Source: author.

Furthermore, one may claim that if the application were executed during a fixed period of time (e.g., for two years), the version without caches would yield fewer failures. This is correct since that is the version with the smallest cross-section and, thus, smallest failure rate. Nevertheless, this is an unfair comparison reliability-wise as the workload executed by this version is much smaller than the others. In this case, one could take advantage of the faster execution time to apply fault tolerance techniques, such as a time modular redundancy scheme, to reduce the application failure rate. For instance, in this

case study one could multiplex up to seven application executions with only the L1 cache enabled during the time it takes to execute a single time the version with caches disabled. A software implemented voter could then use the output of these seven executions to generate a single output - significantly reducing the amount of silent errors. The amount of functional interruptions would most likely remain the same as the system would become unresponsive before even reaching the voter, but these are typically a minority of the observable errors as only very specific corruptions lead to them.

Table 5.4 allows one to compare the different configurations easily using Theorem 1. The second column shows the increase observed in the cross-section, and the third column shows the speed-up provided by the second configuration of each table row. The fifth row indicates the reliability conclusion based on Eq. 3.22.

Table 5.4: Comparison table.

a vs b	σ_b/σ_a	t_a/t_b	**Smaller Cross-Section**	**More Reliable Cfg.**
No Cache vs L1	4.50	7.15	No Cache	Only L1 Enabled
No Cache vs L1+L2	28.17	7.88	No Cache	No Cache
L1 vs L1+L2	6.28	1.10	Only L1 Enabled	Only L1 Enabled

Source: author.

The basic reading is that, as described in Section 3.3, whenever the speed-up (t_a/t_b) is higher than the increase in the cross-section (σ_b/σ_a), configuration b has a higher probability of completing a given workload correctly. Thus, it can be concluded that, in this case, enabling the L1 cache not only improves performance but also increases reliability to radiation-induced errors, contrary to a conclusion based solely on the cross-sections. In cases where the speed-up is not higher than the increase in the cross-section, the latter becomes the dominant factor and, thus, the cross-section adequately predicts the more reliable version (e.g., the last two rows of Table 5.4).

It is worth noting that enabling both cache levels does not result in the same effect. As previously discussed in Subsection 4.2.2, the L2 cache is much less likely to improve reliability since it provides a small speed-up while significantly increasing the cross-section. This shows that, for a state-of-the-art processor, enabling the L2 cache requires a huge speed-up, which may not be attainable, to achieve the same reliability benefit obtained by enabling the L1 caches; in this case study, the speed-up should be higher than 6.28 times.

6 FINAL REMARKS

6.1 Conclusion

In this dissertation, we have presented a new metric and model to evaluate the realistic reliability of a computing system. The proposed metric and model take into account both cross-section and exposure time and can be used to evaluate the best configuration in terms of reliability for an application given a set of solutions, exploring the trade-off between sensitive area and performance.

Furthermore, a case study with a state-of-the-art embedded processor has been presented and evaluated, contributing to the characterization of the Zynq SoC. Moreover, our results show that, in this particular case, the best choice in terms of reliability for the chosen application is to enable only the L1 caches. The case study serves as a clear counterexample to the assumption that the reduction of the radiation-sensitive area always increases reliability. The performed evaluation demonstrates that the cross-section alone does not yield a proper measure of reliability for an embedded application and that it is fundamental to also take the exposure time into account to have a precise indication of the system reliability. Moreover, we have showed on the basis of this counterexample that there is no go-to solution when it comes to reliability, such as always disabling or always enabling caches. Tuning on an application or application profile basis is required when reliability is at stake. This result confirms our initial hypothesis that enabling cache memories may, under certain conditions, increase system reliability for a state-of-the-art COTS microprocessor.

6.2 Future Work

Future work includes modeling and analyzing common cases in which this trade-off is found, such as those mentioned in Section 3.2, to provide valuable information and guidelines for designers. Additionally, we are also interested in applying our model to evaluate Software Implemented Fault Tolerance (SWIFT) techniques, such as EDDA (OH; SHIRVANI; MCCLUSKEY, 2002) and ACCE (VEMU; GURUMURTHY; ABRAHAM, 2007), from a radiation reliability perspective.

Finally, after having showed that cache memories should not always be disabled, we intend to investigate the impact of cache conflicts on system reliability. We are especially interested in investigating how applications running concurrently on the same processor affect each other's reliability.

REFERENCES

ASADI, G.-H. et al. Balancing Performance and Reliability in the Memory Hierarchy. In: IEEE INTERNATIONAL SYMPOSIUM ON PERFORMANCE ANALYSIS OF SYSTEMS AND SOFTWARE, 2005, 2005, Washington, DC, USA. **Proceedings...** IEEE Computer Society, 2005. p.269–279. (ISPASS '05).

ASADI, G.-H. et al. Vulnerability analysis of L2 cache elements to single event upsets. In: DESIGN, AUTOMATION AND TEST IN EUROPE: PROCEEDINGS, 2006, 3001 Leuven, Belgium, Belgium. **Proceedings...** European Design and Automation Association, 2006. p.1276–1281. (DATE '06).

ATSB. **In-fligh upset 154km west of Learmonth, WA**. [S.l.]: Australian Transport Safety Bureau, 2008. (AO-2008-070).

AVIZIENIS, A. et al. Basic concepts and taxonomy of dependable and secure computing. **Dependable and Secure Computing, IEEE Transactions on**, [S.l.], v.1, n.1, p.11–33, 2004.

BAEG, S.; WEN, S.; WONG, R. SRAM Interleaving Distance Selection With a Soft Error Failure Model. **Nuclear Science, IEEE Transactions on**, [S.l.], v.56, n.4, p.2111–2118, 2009.

BAJURA, M. A. et al. Models and algorithmic limits for an ECC-based approach to hardening sub-100-nm SRAMs. **Nuclear Science, IEEE Transactions on**, [S.l.], v.54, n.4, p.935–945, 2007.

BAUMANN, R. Soft errors in advanced computer systems. **Design Test of Computers, IEEE**, [S.l.], v.22, n.3, p.258–266, 2005.

BENINI, L. et al. Mparm: exploring the multi-processor soc design space with systemc. **Journal of VLSI signal processing systems for signal, image and video technology**, [S.l.], v.41, n.2, p.169–182, 2005.

BINDER, D.; SMITH, E.; HOLMAN, A. B. Satellite Anomalies from Galactic Cosmic Rays. **Nuclear Science, IEEE Transactions on**, [S.l.], v.22, n.6, p.2675–2680, 1975.

BINKERT et al. The Gem5 Simulator. **SIGARCH Comput. Archit. News**, New York, NY, USA, v.39, n.2, p.1–7, Aug. 2011.

BURGER, D.; AUSTIN, T. M. The SimpleScalar tool set, version 2.0. **SIGARCH Comput. Archit. News**, New York, NY, USA, v.25, n.3, p.13–25, June 1997.

CAI, Y.; SCHMITZ, M. et al. Cache size selection for performance, energy and reliability of time-constrained systems. In: DESIGN AUTOMATION, 2006. ASIA AND SOUTH PACIFIC CONFERENCE ON, 2006. **Proceedings...** [S.l.: s.n.], 2006. p.6 pp.–.

DARPA. **Vulture Program**. Available at `http://www.darpa.mil/Our_Work/TTO/Programs/Vulture.aspx`. Accessed 2014-11-11.

DICELLO, J.; PACIOTTI, M.; SCHILLACI, M. An estimate of error rates in integrated circuits at aircraft altitudes and at sea level. **Nuclear Instruments and Methods in Physics Research Section B: Beam Interactions with Materials and Atoms**, [S.l.], v.40, p.1295–1299, 1989.

Digilent. **Zedboard**. Available at `http://www.zedboard.org`. Accessed 2014-11-11.

Digilent. **Zedboard Product Specification**. Available at `http://zedboard.org/sites/default/files/product_spec_images/ZedBoard_RevA_sideA.jpg`. Accessed 2014-11-11.

DIXIT, A.; WOOD, A. The impact of new technology on soft error rates. In: RELIABILITY PHYSICS SYMPOSIUM (IRPS), 2011 IEEE INTERNATIONAL, 2011. **Proceedings...** [S.l.: s.n.], 2011. p.5B.4.1–5B.4.7.

DODD, P.; MASSENGILL, L. Basic mechanisms and modeling of single-event upset in digital microelectronics. **IEEE Transactions on Nuclear Science**, [S.l.], v.50, n.3, p.583– 602, 2003.

FLEETWOOD, D. M.; SCHRIMPF, R. D. **Radiation effects and soft errors in integrated circuits and electronic devices**. [S.l.]: World Scientific, 2004. v.34.

Gaisler. **Leon**. Available at `www.gaisler.com`. Accessed 2014-11-11.

GOLDHAGEN, P. Cosmic-ray neutrons on the ground and in the atmosphere. **MRS bulletin**, [S.l.], v.28, n.02, p.131–135, 2003.

GORDON, M. et al. Measurement of the flux and energy spectrum of cosmic-ray induced neutrons on the ground. **Nuclear Science, IEEE Transactions on**, [S.l.], v.51, n.6, p.3427–3434, 2004.

GRIEDER, P. **Cosmic Rays at Earth**: researcher's reference manual and data book. [S.l.]: Elsevier Science Limited, 2001.

HANLON, W. F. **The energy spectrum of ultra high energy cosmic rays measured by the High Resolution Fly's Eye observatory in stereoscopic mode**. [S.l.]: ProQuest, 2008.

HENNESSY, J. L.; PATTERSON, D. A. **Computer Architecture, Fifth Edition**: a quantitative approach. 5th.ed. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 2011.

IBE, E. et al. Impact of Scaling on Neutron-Induced Soft Error in SRAMs From a 250 nm to a 22 nm Design Rule. **Electron Devices, IEEE Transactions on**, [S.l.], v.57, n.7, p.1527–1538, 2010.

IEC. **Functional Safety and IEC 61508**. Available at `http://www.iec.ch/functionalsafety`. Accessed 2014-11-11.

JEDEC. **Measurement and Reporting of Alpha Particle and Terrestrial Cosmic Ray Induced Soft Errors in Semiconductor Devices**. Available at `http://www.jedec.org/sites/default/files/docs/jesd89a.pdf`. Accessed 2014-11-11.

JEDEC. **Test Method for Beam Accelerated Soft Error Rate**. Available at `http://www.jedec.org/sites/default/files/docs/jesd89-3a.pdf`. Accessed 2014-11-11.

KERNS, S. et al. The design of radiation-hardened ICs for space: a compendium of approaches. **Proceedings of the IEEE**, [S.l.], v.76, n.11, p.1470–1509, 1988.

KIM, S.; SOMANI, A. K. Area efficient architectures for information integrity in cache memories. **SIGARCH Comput. Archit. News**, New York, NY, USA, v.27, n.2, p.246–255, May 1999.

KOLASINSKI, W. et al. Simulation of Cosmic-Ray Induced Soft Errors and Latchup in Integrated-Circuit Computer Memories. **Nuclear Science, IEEE Transactions on**, [S.l.], v.26, n.6, p.5087–5091, 1979.

LANL. **Neutron and Nuclear Science Facility at LANSCE**. Available at `http://wnr.lanl.gov`. Accessed 2014-11-11.

LESEA, A. et al. Soft Error Study of ARM SoC at 28 Nanometers. **Proceedings of the IEEE Workshop on Silicon Errors in Logic - System Effects, 2014**, [S.l.], 2014.

LIDEN, P. et al. On latching probability of particle induced transients in combinational networks. In: FAULT-TOLERANT COMPUTING, 1994. FTCS-24. DIGEST OF PAPERS., TWENTY-FOURTH INTERNATIONAL SYMPOSIUM ON, 1994. **Proceedings...** [S.l.: s.n.], 1994. p.340–349.

LOCKWOOD, J. A. Forbush decreases in the cosmic radiation. **Space Science Reviews**, [S.l.], v.12, n.5, p.658–715, 1971.

LU, S.-L. et al. Scaling the Memory Wall : designer track. In: COMPUTER-AIDED DESIGN (ICCAD), 2012 IEEE/ACM INTERNATIONAL CONFERENCE ON, 2012. **Proceedings...** [S.l.: s.n.], 2012. p.271–272.

MAIZ, J. et al. Characterization of multi-bit soft error events in advanced SRAMs. In: ELECTRON DEVICES MEETING, 2003. IEDM '03 TECHNICAL DIGEST. IEEE INTERNATIONAL, 2003. **Proceedings...** [S.l.: s.n.], 2003. p.21.4.1–21.4.4.

MANOOCHEHRI, M. et al. CPPC: correctable parity protected cache. In: ANNUAL INTERNATIONAL SYMPOSIUM ON COMPUTER ARCHITECTURE, 38., 2011, New York, NY, USA. **Proceedings...** ACM, 2011. p.223–234. (ISCA '11).

MAY, T. et al. Dynamic Fault Imaging of VLSI Random Logic Devices. In: RELIABILITY PHYSICS SYMPOSIUM, 1984. 22ND ANNUAL, 1984. **Proceedings...** [S.l.: s.n.], 1984. p.95–108.

MAY, T.; WOODS, M. H. Alpha-particle-induced soft errors in dynamic memories. **Electron Devices, IEEE Transactions on**, [S.l.], v.26, n.1, p.2–9, 1979.

MICHALAK, S. **Estimation of the expected weekly number of soft errors in QA or QB**. [S.l.]: Los Alamos National Laboratory, 2004. (LA-UR-04-5162).

MICHALAK, S. E. et al. Assessment of the impact of cosmic-ray-induced neutrons on hardware in the Roadrunner supercomputer. **Device and Materials Reliability, IEEE Transactions on**, [S.l.], v.12, n.2, p.445–454, 2012.

MITRA, S. **System-Level Single-Event Effects**. [S.l.]: NSREC 2012 Short Course, Miami, USA, 2012.

MOORE, G. E. Cramming more components onto integrated circuits. **Electronics**, [S.l.], v.38, n.8, April 1965.

MUKHERJEE, S. **Architecture Design for Soft Errors**. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 2008.

MUKHERJEE, S. S. et al. A Systematic Methodology to Compute the Architectural Vulnerability Factors for a High-Performance Microprocessor. In: IEEE/ACM INTERNATIONAL SYMPOSIUM ON MICROARCHITECTURE, 36., 2003, Washington, DC, USA. **Proceedings...** IEEE Computer Society, 2003. p.29–. (MICRO 36).

MUKHERJEE, S. S. et al. Cache Scrubbing in Microprocessors: myth or necessity? In: IEEE PACIFIC RIM INTERNATIONAL SYMPOSIUM ON DEPENDABLE COMPUTING (PRDC'04), 10., 2004, Washington, DC, USA. **Proceedings...** IEEE Computer Society, 2004. p.37–42. (PRDC '04).

MURSULA, K.; USOSKIN, I. Heliospheric physics and cosmic rays. **Lecture Notes, American Geophysics Union, Washington DC**, [S.l.], p.37–66, 2003.

NASA. **Cosmic Rays - Introduction**. Available at `http://imagine.gsfc.nasa.gov/docs/science/know_l1/cosmic_rays.html`. Accessed 2014-11-11.

NASA. **Solar Cycle Primer**. Available at `http://www.nasa.gov/mission_pages/sunearth/news/solarcycle-primer_prt.html`. Accessed 2014-11-11.

NASA. **Geospace Storms**. Available at `http://www.nasa.gov/mission_pages/rbsp/multimedia/rbsp-coronalmass.html`. Accessed 2014-11-11.

NGDC. **International Sunspot Numbers**. Available at `http://www.ngdc.noaa.gov/stp/space-weather/solar-data/solar-indices/sunspot-numbers/international/tables/table_international-sunspot-numbers_monthly.txt`. Accessed 2014-11-11.

NMDB. **Real-time Neutron Monitor Database**. Available at `http://www.nmdb.eu/nest`. Accessed 2014-11-11.

NORMAND, E. Single event upset at ground level. **IEEE transactions on Nuclear Science**, [S.l.], v.43, n.6, p.2742–2750, 1996.

NORMAND, E.; BAKER, T. Altitude and latitude variations in avionics SEU and atmospheric neutron flux. **Nuclear Science, IEEE Transactions on**, [S.l.], v.40, n.6, p.1484–1490, 1993.

NORMAND, E. et al. Quantifying the double-sided neutron SEU threat, from low energy (thermal) and high energy ($>$ 10 MeV) neutrons. **IEEE transactions on nuclear science**, [S.l.], v.53, n.6, p.3587–3595, 2006.

OH, N.; SHIRVANI, P.; MCCLUSKEY, E. Error detection by duplicated instructions in super-scalar processors. **Reliability, IEEE Transactions on**, [S.l.], v.51, n.1, p.63–75, 2002.

PATTERSON, D. A.; HENNESSY, J. L. **Computer Organization and Design, Fifth Edition**: the hardware/software interface. 5th.ed. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 2013.

PFOTZER, G. Dreifachkoinzidenzen der Ultrastrahlung aus vertikaler Richtung in der Stratosphäre. **Zeitschrift für Physik**, [S.l.], v.102, n.1-2, p.41–58, 1936.

PRADHAN, D. K. (Ed.). **Fault-tolerant Computer System Design**. Upper Saddle River, NJ, USA: Prentice-Hall, Inc., 1996.

QUINN, H. et al. Single-Event Effects in Low-Cost, Low-Power Microprocessors. **IEEE Nuclear and Space Radiation Effects Conference, 2014**, [S.l.], 2014.

QUINN, H.; GRAHAM, P. Terrestrial-based radiation upsets: a cautionary tale. In: FIELD-PROGRAMMABLE CUSTOM COMPUTING MACHINES, 2005. FCCM 2005. 13TH ANNUAL IEEE SYMPOSIUM ON, 2005. **Proceedings...** [S.l.: s.n.], 2005. p.193–202.

REBAUDENGO, M.; SONZA REORDA, M.; VIOLANTE, M. An Accurate Analysis of the Effects of Soft Errors in the Instruction and Data Caches of a Pipelined Microprocessor. In: DESIGN, AUTOMATION AND TEST IN EUROPE - VOLUME 1, 2003, Washington, DC, USA. **Proceedings...** IEEE Computer Society, 2003. p.10602–. (DATE '03).

RECH, P. et al. Impact of GPUs Parallelism Management on Safety-Critical and HPC Applications Reliability. In: DEPENDABLE SYSTEMS AND NETWORKS (DSN), 2014, 2014. **Proceedings...** [S.l.: s.n.], 2014.

SALAS, A. G. et al. PhoneSat In-flight Experience Results. In: SMALL SATELLITES AND SERVICES SYMPOSIUM; 26-30 MAY 2014; PORTO PETRO, MAJORCA ; SPAIN, 2014. **Proceedings...** [S.l.: s.n.], 2014.

SILBERBERG, R.; TSAO, C. H.; LETAW, J. R. Neutron generated single-event upsets in the atmosphere. **Nuclear Science, IEEE Transactions on**, [S.l.], v.31, n.6, p.1183–1185, 1984.

TABER, A.; NORMAND, E. Single event upset in avionics. **IEEE Transactions on Nuclear Science**, [S.l.], v.40, p.120–126, 1993.

TSAO, C. H.; SILBERBERG, R.; LETAW, J. R. Cosmic-ray heavy ions at and above 40,000 feet. **IEEE Transactions on Nuclear Science**, [S.l.], v.31, p.1066–1068, 1984.

VEMU, R.; GURUMURTHY, S.; ABRAHAM, J. ACCE: automatic correction of control-flow errors. In: TEST CONFERENCE, 2007. ITC 2007. IEEE INTERNATIONAL, 2007. **Proceedings...** [S.l.: s.n.], 2007. p.1–10.

VETTE, J. I. **The NASA/National Space Science Data Center trapped radiation environment model program, 1964-1991**. [S.l.]: National Aeronautics and Space Administration, Greenbelt, MD (United States). Goddard Space Flight Center, 1991.

WALLMARK, J.; MARCUS, S. M. Minimum Size and Maximum Packing Density of Nonredundant Semiconductor Devices. **Proceedings of the IRE**, [S.l.], v.50, n.3, p.286–298, 1962.

ZIEGLER, J. F. et al. IBM experiments in soft fails in computer electronics (1978 - 1994). **IBM J. Res. Dev.**, Riverton, NJ, USA, v.40, n.1, p.3–18, Jan. 1996.

ZIEGLER, J. F.; LANFORD, W. A. The effect of sea level cosmic rays on electronic devices. **Journal of applied physics**, [S.l.], v.52, n.6, p.4305–4312, 1981.