POLITECNICO DI TORINO

Faculty of Electonic Engineering

Master degree course in Electronic Engineering

Master Degree Thesis

Based on cv32e40p core of OpenHW Group organization

Advisors
Stefano Di Carlo

Correlatore:
Alessandro Savino, Maurizio Martina, Guido Masera

Candidate:
Elia Ribaldone

Marzo 2021

kile

To my family and Anna.

Acknowledgements

Abstract

The miniaturization of the microelectronic components together with the use of integrated circuits in more and more application leads to an increasing use of FT (Fault Folerant) architecture. This thesis investigate the use of FT techniques in a stage of the cv32e40p open source core. We used fault injection simulation to divide our stage in three blocks with increasing level of

Summary

A **FT (fault tolerant) system** continues to work properly even if some of the internal components are broken; this feature is necessary when a failure may cause damage to people, dangerous destruction, military upset or loss of data. FT systems are essential in aerospace, transport, medical and utility industries and they are usually composed by a power source, an hardware system and a software system, each of these parts are fault tolerant depending on application.

This work concerns the hardware system and in particular the chip architecture design. In this context the faults are **transient**, **intermittent** or **permanent** and they are generated by manufacturing defects, system degradation or particle strikes. **Manufacturing** defects generate permanent faults and they reduce the yield with an increase in the cost per piece since the affected chips are discarded during quality control process. **System degradation** produces permanent faults and it is a life-limiting phenomenon that brings the chip to wearout phase. Finally, **particle strikes** produce both transient or permanent faults. These problems rely on the application: system degradation generally depends on chip temperature, clock speed and the workload, otherwise particle strikes rely on sources of alpha particles or neutrons, which are generated by the cosmic rays or radioactive materials.

For these reasons an ideal fault tolerant system should be protected against transient faults caused by particle strikes and it should manage permanent faults in order to increase the yield and chip life. A complete protection against faults creates drawbacks in speed, area and power budgets. This is the reason why we create a configurable architecture where faults coverage can be changed according to the specific application and the project constrains.

In this Master Thesis the **Instruction Fetch of cv32e40p** core is converted in a **configurable fault tolerant stage** in order to reduce failures in fetching instructions. The core used was designed by the "OpenHW Group" organization and it can be integrated in PULPissimo platform in order to create a complete microcontroller architecture.

Before the design of the architecture we study the cv32e40p core and then we proceed with the creation of the simulation environment, building the script used for all simulations. These tools born in the cv32e40p *core-v-verif* repository with te purpose of automatize compilation of testbenches and their simulation using QuestaSim and Modelsim.

The most important feature of the tool is the **optimized fault injection method** used to simulate faults in the architecture. We use a worst case approach injecting faults only in sequential parts (FF and memory), in this way we consider that all faults injected in the combinatory path (for example after a particle strike) reach a FF and are sampled. This is not always true since some bits can be logical masked in the following cases: if they are don't care bits when fault occurs, if they can be electrical masked due to attenuation

before latch or if the fault don't have the time to reach a FF (latch-widows). All this masks can be clustered in AVF (Architecture Vulnerability Factor) which is 1 if every fault generate a failure, otherwise it is lower. Knowing this we can assert that our tool works in the worst-case scenario since the AVF of each combinatory path (excluding inputs) is consider equal to one.

Apart from this first considerations the tool optimizes simulation times also takes advantage of *vcdstim* feature in the case of single stage simulations. Indeed the whole core is initially simulated using a benchmark firmware, meanwhile the input and output data of a specific stage are saved into .vcd and .wlf files, finally a stage-specific simulation started using .vcd as input (*vcdstim* feature). In this way only one stage is simulated and we reduce working times. **Stage-specific simulation** can be repeated a specified number of times using fault injection and the output of the stage is finally compared with .wlf output file in order to find failures.

Using our tool we first simulate fault injection in the reference IF stage of cv32e40p core and we find a fault tolerance equal to 30%. Later we use simulation results to understand the fault masking for each signals and then we divide **IF stage** in **three main blocks**. *Each block have increasing level of intrinsic fault tolerance in original architecture* and knowing this we apply FT techniques to each block. Anyway during simulations and synthesis we can enable FT for one, two or all blocks, depending on these settings we can manage area, speed, power and FT trade-off.

In the design of architecture we use a FT technique that is able to detect and correct transient faults using **TMR** (Triple Modular Redundant), this methods works only if each of the three identical blocks compared don't have permanent faults, and if we assume to neglect multiple particle strikes. Although multiple strikes is improbable, permanent faults is already present after manufacturing process and increase during time, for this reason we implement a technique to detect and correct this faults using some backup stages. Summarizing *we use TMR to manage transient faults and additional logic to protect against permanent faults.* In this way if all FT architecture of the IF stage are enabled during syntesis the final result is an higher yeld, a longer life and a protection against noise and particle strikes. The last important feature of the design is the saving of *permanent faults information* in the **CSR** (Common and Status Registers), in this way we could restore permanent faults settings after a reboot.

Contents

List of Figures

List of Tables

Listings

CHAPTER

1 Introduction

1.1 General context

1.2 Objectives

1.3 Thesis structure

CHAPTER

2 Technical Background

2.1 Safety critical application system

2.1.1 Dependability Model

Dependability is the ability of a system to provide a predetermined level of service to the user [1]. This capacity depends on the system application, for example a wrong use or high workload make the level of service offered go down. From the designer's point of view, the dependability of a system must be verified through tests and simulations , in order to verify the correct functioning of the system in various environment. For system that works in critical applications, in addition to the functional tests must be made tests that verify the level of service required despite environment conditions. For example in satellites it is not possible to do maintenance and the correct behavior of on-board systems is necessary to avoid the fall of the asset, so when the Dependability required to the system is high, many stress tests must be done to have a complete technical testing. For these reasons to guarantee the dependability in a given application the main factors are how the system is designed and which kind of tests is performed on it.

Dependability in characterized by: Metrics, Attributes, Impairments and Means. These four categories allow us to completely define the dependability in a system and they are explained below:

Dependability Metrics

Dependability metrics are used to measure the dependability of a system and they are used to verify Dependability Attributes. The Metrics are experimentally measured or estimated through various techniques. These are the main metrics used:

- ***TTF*** **:** Time To Failure is the time to a error in a specific system [2]. For example a device with TTF equal to 1 year will have an error after one year of correct work.
- ***MTTF*** **:** Mean Time To Failure is the mean time between two failure in a system. Under certain condition (e.g. formula **2.6 on the following page**) we can combine the MTTF of various parts to find the MTTF of overall system, to do this we should use the following formula:

$$MTTF_{system} = \frac{1}{MTTF_{part1}^{-1} + MTTF_{part2}^{-1}} = \frac{1}{\sum_{i=0}^{n_{parts}} \frac{1}{MTTF_i}} \tag{2.1}$$

- ***FIT*** **:** Failure In Time is the number of errors in a billion of hours. The relation between MTTF (expressed in year) and FIT is:

$$FIT = \frac{10^9}{MTTF_{year} \cdot 365\,days \cdot 24\,hour} \tag{2.2}$$

The FIT metric is used instead of MTTF because it makes calculation easier, in fact system FIT can be easly calculated in this way:

$$FIT_{system} = \sum_{i=0}^{n_{parts}} FIT_i \tag{2.3}$$

- ***MTTR :*** The Mean Time To Recover is the time needed to a system to repair an error once it is detected [2].
- ***MTBF :*** The Mean Time Between Failure is the mean time between the start/restart and an error detection, for this reason we have:

$$MTBF = MTTF + MTTR \tag{2.4}$$

Dependability Attributes

Attributes are the properties which are expected from a system that experiencing faults to be dependable [1]. These attributes are evaluated from Dependability Metrics according to a fault model. The most used Attributes are Reliability, Safety and Availability, they are defined below:

- ***Reliability :*** it is the probability that a system will operate without failures in a given time interval. This type of Attribute is widely used for example in space applications, where it is necessary to guarantee operation for certain period. At the integrated circuit level many techniques have been adopted over time to increase reliability by improving production processes, usually are used old processes experiences to predict the reliability of a new product, this is done on all ICs but especially on memories [3]. Reliability can be expressed according to *exponential failure law* :

$$R(t) = e^{-h(t)\ t} \simeq e^{-\lambda\ t} \tag{2.5}$$

  Where $h(t)$ is the *Instantaneous Error Rate* considered as the probability that the system has an error in a certain interval Δt which start at instant t, so it is the probability of error in the time interval $(t,\ t\ +\ \Delta t)$. To simplify calculation $h(t)$ is usually approximated with the constant error rate λ, that is equal to $1/MTTF = FIT$ [2]. For these consideration when we have the FIT of each part of a system we can use formula **Figure 2.5** to find total reliability, in this case we consider to have n independent parts each with a certain failure rate h_i:

$$R(t)_{system} = \prod_{i=0}^{n-1} R_i(t) = e^{-\left(\sum_{i=0}^{n-1} h_i\right)} \tag{2.6}$$

  This model is valid if we consider the failure rate constant. From formula **Figure 2.6** we can states that the FIT of a system is equal to the sum of the FIT of each part.

- ***Availability :*** It is the percentage of time the system remains active and it can be used. This Attribute is employed a lot in the IT field, for example to characterize servers or a communication network [4] [5]. It is therefore required in areas where it is expected that the system may not work for some periods, so in this case we are interested to know how long it will actually work properly. Availability is usually expressed as a percentage or by the downtime at a certain instant. For example, a system with Availability of 99.999% will have a downtime of 5 minutes over a year. The common expression for Availability is:

$$Availability = \frac{MTTF}{MTTF + MTTR} = \frac{MTTF}{MTBF} \tag{2.7}$$

- ***Safety* :** For this attribute, two types of failures are considered : *fail-safe* if the fail does not cause danger or damage, while *fail-unsafe* if the fail causes safety problems. A simple example is a RADAR that detects airplanes, if an airplane that doesn't exist is detected there is no serious damage and therefore we consider this failure as fail-safe, instead if an airplane is not detected we have a fail-unsafe failure. The safety of a system is the probability that it remains fail-safe over a certain period of time. It is used in critical sensing, safety and control systems.

Dependability Impairments

Dependability Impairments are used to communicate that something in the system has gone wrong [1]. There are three types of Impairments and each indicates a problem at a different level:

- ***Faults* :** They indicate a problem at the physical level. For example in a PCB circuit a fault can occur when a component desoldered due to incorrect manufacturing process. In the field of integrated circuits a fault is usually due to a bit flip caused by external particles, by a manufacturing defect or a bug in the microcode or software. Any failure of a system always starts with a fault, this fault may or may not cause a problem depending on how the design was done. In integrated circuits faults can be masked by certain architectural design techniques and their number can be limited by special layouts and processes. However, they cannot be eliminated entirely.

- ***Errors* :** They indicate a problem at computational level caused by a Fault. Errors are caused by Faults that are not masked by the system, for example if there is a bit flip in an input register of the ALU, there will be an Error in the output register because the operation has a wrong result.

- ***Failures* :** They indicate system failure due to an Error. The failure of the system is an Impairments that you never want to have in a critical application since the behavior of the circuit is unpredictable and so unsafe.

To summarize a Fault can cause an Error and this in turn can cause a Failure. For these reason the designers of a critical application system should have the ability to mask Fault and Errors in order to avoid Failure.

Dependability Means

Dependability Means are that set of techniques and methods needed to create a Dependable system[1]. Fault Tolerance is the method that is used in this thesis but it is normally followed by other techniques, these are the most important ones:

- ***Fault Tolerance (FT)* :** Fault Tolerant systems continue to work even in the presence of Faults, this result is achieved through redundancy and a set of processes: The first is called Fault Masking and consists in avoiding the propagation of a fault by correcting the values in the system. In fact Fault Masking consists both in the reduction of errors and in their masking to avoid failures. Common examples of Fault Masking techniques are TMR (Triple Modular Redundancy) and ECC (Error Correcting Code)

that allow to reduce Errors in memories and circuits. The second process is the Fault Detection that allows to recognize the presence of an error in the system, for example using the TMR in order to detect a Fault we can just verify that there is a module with different results from the others. This technique is also used in systems without redundancy where you want to understand if the system is working properly.

When a fault is detected in a FT system, you can decide to correct it and continue with the execution, or you can disable the system part from which the fault started, in the case of permanent fault. This mode of performances decay of a system is called Graceful Degradation.

- ***Fault Prevention (FP)*** **:** FP is a very broad field because it is the set of processes that allow to reduce the introduction of faults in the system. This goal is achieved by controlling all processes from specification to manufacturing.
- ***Fault Forecasting*** **:** Fault Forecasting is the set of techniques that allow to predict the trend of the number of Faults and their effects in a system.
- ***Fault Removal*** **:** Fault Removal is the set of techniques used to eliminate errors already present in the system. This is done through verification of circuit operation and maintenance.

We have seen the basic vocabulary used in dependable system design and maintenance, in figure **Figure 2.1 on the next page** are summarized all concept explained in order to give a graphical overview of the design of a Dependable System.

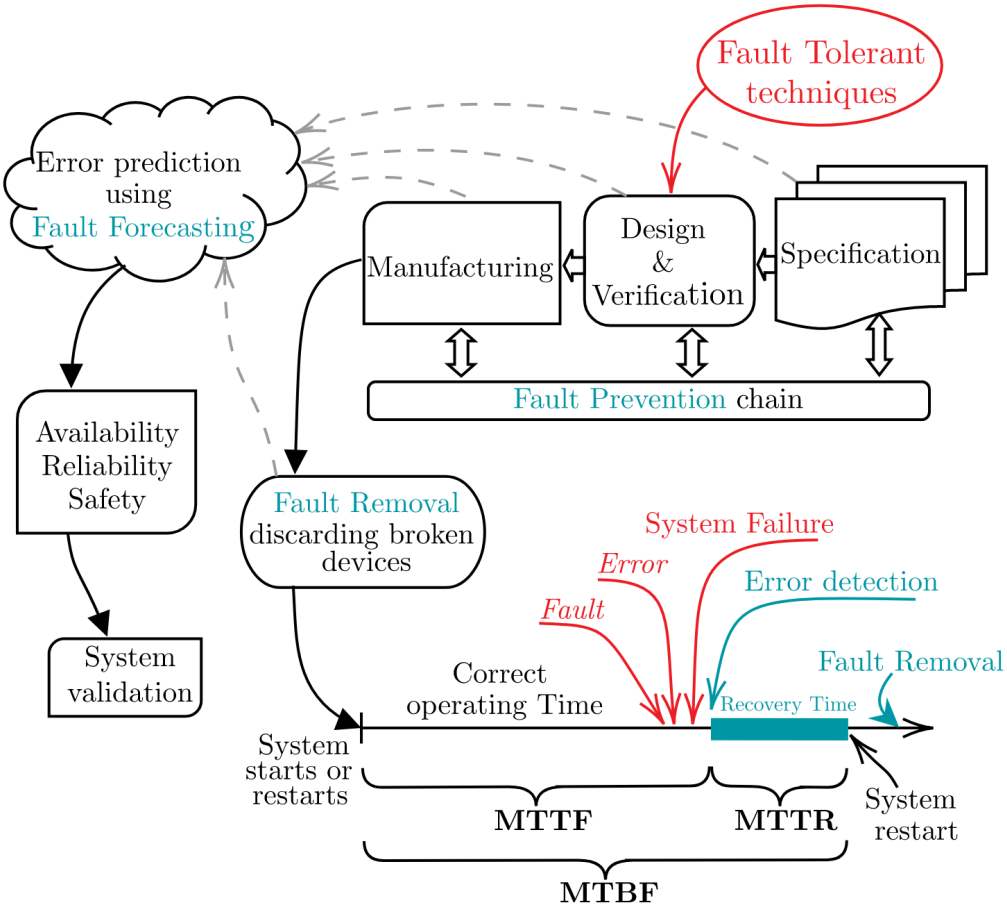

Figure 2.1: **Design and life of a Dependable System**

The block diagram in Figure **Figure 2.1** start with the specification of the system, then the designer use Fault tolerant techniques to design and verify the system, finally the product is manufactured, in these tree steps is applied Fault Prevention in order to reduce unwanted errors. After manufacture, the manufacturer apply a selection in order to discard broken devices and finally the systems is sold and it begins to be used. Meanwhile we gather data from all production chain in order to use Fault Forecasting to predict MTTF,MTTR and MTBF. Then using predicted data are evaluated required Dependability Attributes and finally system is validated and can be sold.

When the system begins to be used there are some periods of correct operations (estimated as MTTF), then at a certain instant a fault occur, this fault can propagate in an Error and this can became a System Failure. If the Failure is detected the system begins the Recovery Time (estimated as the MTTR) in which the failure is fixed. In the diagram we select a time interval in which fault is propagated but in a Dependable system this should happen rarely. It is also indicated the removal of defected parts using Fault Removal, this techniques can be also applied during Recovery time.

In the next section we contextualize this thesis work analyzing the parts of a critical electronic system.

2.1.2 Electronic system parts

This section describe how this Thesis is positioned in a complete dependable electronic system. In figure **Figure 2.2** we give an example of electronic system, it receives information from *sensors* and it controls some *actuators* according to their specification. The circuit is powered by a battery or by power network and this energy should be converted inside the board to be used. For this reason there is a part of the PCB dedicated to *voltage conversion*, this block is composed by analogue and digital components that together create the Power Conversion and Distribution system.

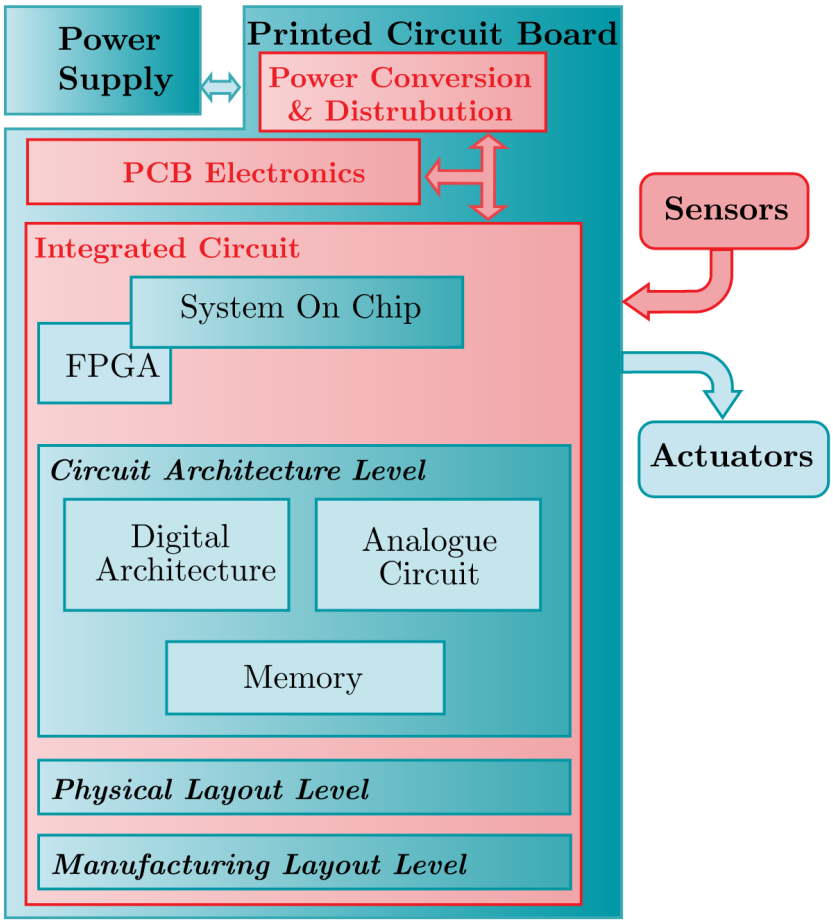

Figure 2.2: **Example of Electronic System**

The elaboration part instead is composed by integrated circuits that analyze the data received from analog and digital sensors and they use this data to decide how to control the actuators. This elaboration is done by a microcontroller or an FPGA and the design of these ICs have four main design level [6] as you can see in Figure **Figure 2.2**:

- **Manufacturing Process Level** (lev. 4) : This is the level of manufacturing processes, in this step are defined all technique to create the die from a silicon wafer. In the

case of hardened chip the manufacturer apply fault tolerant and fault prevention techniques in order to improve system dependability.

- **Physical Layout Level** (lev. 3) : It is the set of techniques used to place transistors properly. In the case of robust systems the layout is improved in order to decrease the sensitivity of the circuit to radiation.
- **Circuit Architecture Level** (lev. 2) : At this level circuits design is carried out at the RTL level; the circuits may be digital, analogue or a mixed signal. Generally to make this level robust are used fault tolerance redundancy and error correction techniques.
- **Electronic System Level** (lev. 1) : In this case we can still work at the RTL level using components previously created at the architectural level, or at the unit level (e.g. cluser computers). In the case of robust systems is used processor redundancy (e.g. lockstep technique) or redundancy of computers.

As we have seen, an electronic system is made up of many parts which must all be dependable in order to have a dependable system. *This Master Thesis will deal with the second design level, which is the architectural one.* In order to be able to use the proposed rtl project correctly, it is necessary to use hardening techniques in all the lower and higher levels. In fact what is important for the final application is the dependability of the system, so it would be almost useless to use a hardened processor in a device where the power supply part is not dependable. Anyway this consideration should be done case by case by designers.

2.1.3 IEC61508 Standard

2.2 Dependability of Integrated Circuits

Faults in integrated circuits are due to both bit flip or electrical problems such as broken interconnects. The origins of these problems are due both to the aging of integrated transistors and their susceptibility to charge injection by external particles, such as cosmic rays.

These two phenomena are influenced by the field of use of the IC and by the working conditions. For example, aging is accelerated by high temperatures and high workloads, which wear out the interconnections. On the other hand the influence of external particles increases in space applications due to the increased cosmic ray flux, as well as in nuclear power plants or where some radioactive materials are present.

The understanding of these phenomena is essential to improve fault tolerance techniques applied to integrated circuits also at RTL level in different application, therefore the causes and mechanisms of faults are now investigated by dividing them into *internal factors* (graceful degradation) and *external factors* (e.g. particle flux).

2.2.1 Internal Factors of Faults

As already mentioned, the internal factors of faults are due to intrinsic electrical problems of transistors, which can be caused either by the *breakage of the interconnections* or by problems related to the *gate oxide failure*.

As far as *interconnections* are concerned, there are two origins of failure:

- **Electromigration** (EM) : EM is a phenomenon known since 1966 [7], whereby the electrons generating the electric current in the interconnections impart a momentum to the atoms of metal. This momentum transfer can create void in the very small interconnections of ICs. The phenomenon is directly proportional to the square of the charge density (j_e; $[A/cm^2]$) and depends exponentially on the *activation energy* of the material (E_a; $[eV]$) and on the temperature (T $[K]$). These relationship are condensed in the Median Time To Failure calculated according to the Black's formula [2]:

$$MeTTF_{system} = \frac{A_0}{j_e^2} e^{\frac{E_a}{kT}} \tag{2.8}$$

  Where A_0 is a technology dependent constant and k is the Boltzmann constant.

  The opposite effect to EM is due to mechanical stress which tends to compensate for the displacement of metal atoms, this principle is the basis of the Blech effect for which below a certain length (called the Blech length) EM has no effect because the two forces are balanced. Normally the length of the interconnections is greater than the Blech length and for this reason EM should be reduced by various techniques. Two of these techniques are the use of metal alloys (Al+Cu, Al+Pd) or the creation of *Bamboo Structures* that reduce the number of metal grains. In fact, the creation of a void in a connection starts at the interface between two or more grains of metal. Here the mobility of the atoms is greater respect to normal mobility, for this reason metal atoms are able to move and they leads to an avalanche effect which creates the final voids. Electromigration create both permanent or intermittent faults and leads

the chip in the wear-out phase. as we have seen this phenomena is related to current density that normally depends on workload, hence architecture and system fault tolerant strategy for EM reduction lead with resource multiplexing and oversizing.

- **Metal Stress Voiding** (MSV) : The MSV is due to the difference in expansion ratios between the metal of the interconnection and the surrounding material. The phenomenon is closely related to temperature and the formula **Figure 2.9** gives a quantitative evaluation in terms of MTTF [2]:

$$MTTF_{system} = \frac{B_0}{(T_0 - T)^n} e^{\frac{E_b}{kT}} \tag{2.9}$$

  Where: B_0, n and E_b are material dependent constants, k is the Boltzmann constant and T is the temperature in Kelvin. According to the equation the larger the temperature the lower the MTTF, this is a further reason why heat dissipation is important for system dependability. Another important methods to reduce the influence of this phenomenon is the use of stronger metals, with expansion constants similar to the interfaces.

  MSV related faults are very similar to those caused by EM and can be either intermittent or permanent.

As far as *Gate Oxide Failure* is concerned, there are three main physical mechanisms that cause faults:

- **Negative Bias Time Instability** (NBTI) : NBTI is the process that causes short-channel pMOS (hence the term Negative Bias) subjected to high temperature or negative gate voltages, to degrade the maximum frequency of the circuit and to create faults. These phenomena is due to charges being trapped under the gate of the pMOS [2] [8]. These charges slow down the switching process, decreasing the speed of the circuit and creating Timing Faults. Timing Faults happen when the propagation time of the critical paths no longer respects the sampling conditions according to the circuit's clock.

  The physical effect related to this phenomenon is the decrease in mobility under the gate due to the bombardment of charges during normal operations. This causes the pMOS threshold voltage to increase (hence the term instability) and the maximum current to decrease, leading the logic gates (which use pMOS) to slow down and fault [2].

  To reduce the contribution of this effect are used Dynamic Voltage Scaling and the power gating [8].

- **Hot Carrier Injection** (HCI) : HCI leads to a reduction of the f_{max} of the circuit but in this case this is due to the charges trapped in the gate. In fact, during the acceleration along the channel, the ionization effect produces electron-hole pairs, if these charges have sufficient energy they can inject themselves in the gate and get trapped [2]. This creates a variation of the threshold voltage that lead to faults as in the case of NBTI.

  Unlike the other effects, HCI get worse at lower temperatures due to the increase in charge mobility in the material. The first consequence of HCI is the degradation of

the threshold voltage that decreases the maximum saturation current, this lead to a reduction of the maximum frequency from 1% to 10% [2].

Again, duty cycle reduction is a way to reduce the effect of HCI. Despite technological advances, HCI is still present in recent Tri-Gate Nanowire [9], FLASH memories [10] and general CMOS electronics.

- **Time Dependent Dielectric Breakdown** (TDDB) : Continuously applied voltages in the transistors create defects in the gate material, which can lead to the creation of conductive paths between the channel and the gate, knocking out the transistors. In thicker gate this effect is more pronounced.

  To reduce this phenomenon, attempts are made to reduce the gate voltage and to use stronger gate materials.

All these effects added to the manufacturing defects lead to: an infant mortality phase of the components which are discarded before being sold, a life phase with a certain fixed value of failure rate and finally a wear-out phase which causes the final failure of the integrated circuit. This variation of the failure rate over time is shown in the figure **Figure 2.3**.

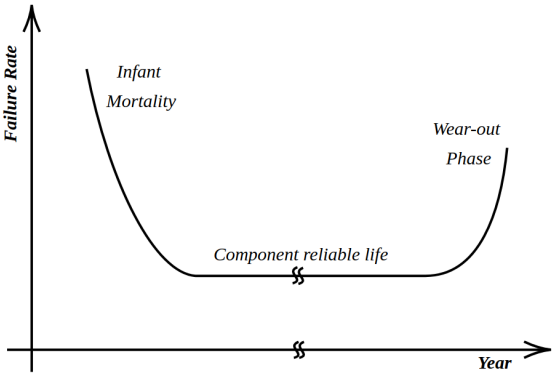

Figure 2.3: **Example of Electronic System**

2.2.2 External Factors of Faults - Radiations

For External Factors of Faults we mean all those external factors that can cause ICs to malfunction. In the next paragraphs we first analyze the different sources of radiations, and then the radiation effects on ICs.

2.2.2.0 Radiation Levels and Sources

There are essentially four sources of radiation: supernovae and celestial explosions that create Galactic Cosmic Rays, the Sun that generates Solar Cosmic Rays, terrestrial radioactive materials (e.g. ^{238}U), and finally nuclear weapons and reactors. The characteristics and radiation levels of these sources are described in the following paragraphs.

Galactic Cosmic Rays (GCRs)

In order to understand how GCRs arrive on earth, we need to know the structure of the heliosphere. As described in **Figure 2.4 on the next page**, the Sun emits particles in all directions, mainly protons and alpha particles that form the Solar Wind at $400-700km/s$. The Solar System moves through the local interstellar medium (LISM) composed mainly

of helium and rarefied hydrogen. For this reason the solar wind collides at supersonic speed with interstellar dust (at a relative velocity of about $26 km/s$ respect to the Sun) and is slowed down to subsonic speeds at the so-called 'Termination Shock' (75-100 AU from the Sun). After the Termination Shock, moving away from the Sun there is a zone where the LISM and solar rays are compressed to form plasma, this zone is called Heliosheath (pink filled at the right of the sun in **Figure 2.4**). At the end of the Heliosheath there is the limit beyond which the solar rays cannot go, this is called the Heliopause ($\simeq$ 121-150AU from the Sun). Beyond the Heliopause there is probably the Bow Wave, the shock wave of the LISM with the heliosphere, such as water does on the bow of a ship.

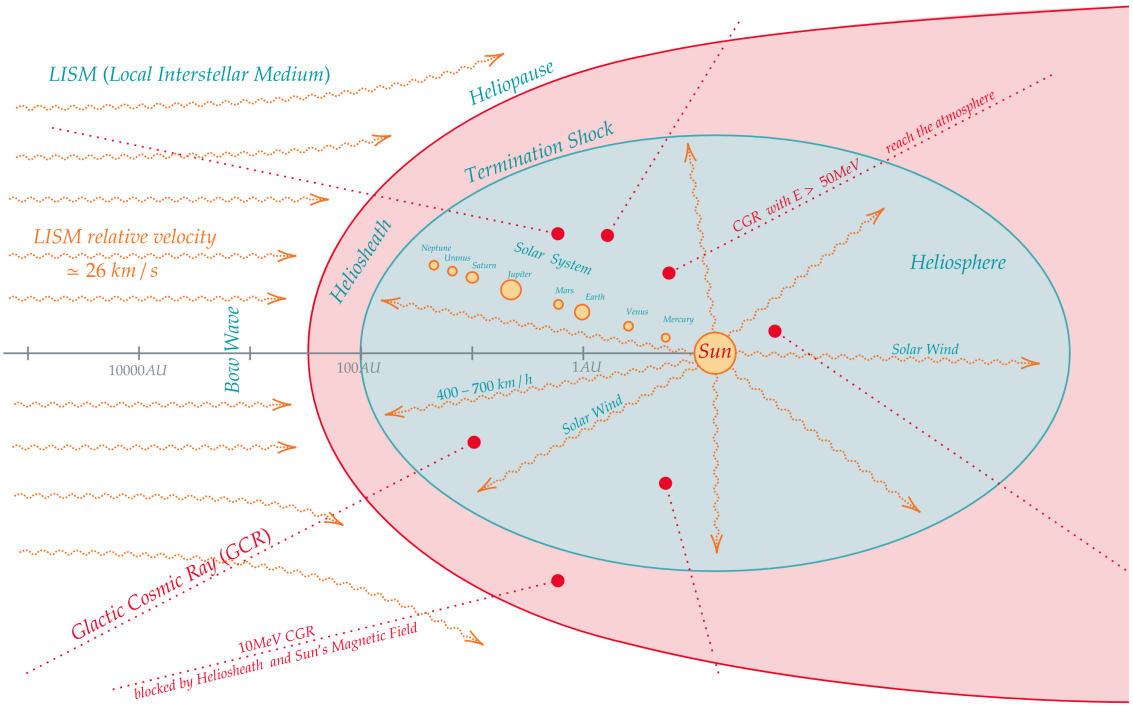

Figure 2.4: **Solar system moving within the LISM while it is hit by GCRs, the planets are not to scale and the distance is logarithmic.**

.

In this environment, the Galactic Cosmic Rays are the isotropic flow of energetic particles from outside the solar system that try to pass through the solar wind and magnetic field shields into the Earth's atmosphere as shown in **Figure 2.4**. GCRs are created by stellar explosions such as supernovae and gamma-ray bursts, active galaxies or quasars, they reach the Earth isotropically and so they hit it uniformly in more or less all directions. In fact, unlike the LISM, these rays have an energy that can reach $100\,000\,TeV = 10^{20}\,eV$. Anyway considering that GCRs need to have an energy of at least $50MeV$ to pass the Termination Shock, only 35% of them reach the Earth's atmosphere.

GCRs are composed for 89% of protons (p^+), 9% of alpha particles (He^+) and 2 % of heavy ions (mainly Lithium, Beryllium and Boron). Their effect on the Terrestrial Cosmic Rays varies according to the variation of the Earth's magnetic field, when the Sun's peak occurs the Earth's magnetic field is maximum, consequently there is a minimum in the radiation induced by the GCRs. On the contrary, when the Sun is at a minimum, there is a maximum of radiation on the Earth.

The flux of cosmic rays depends on their energy, as can be seen in **Figure 2.5** the flux is measured in $\frac{particles}{m^2\, sr\, GeV\, sec}$, where steradians refer to the centre of the earth while m^2 is the distance of the area to be measured from the centre of the earth.

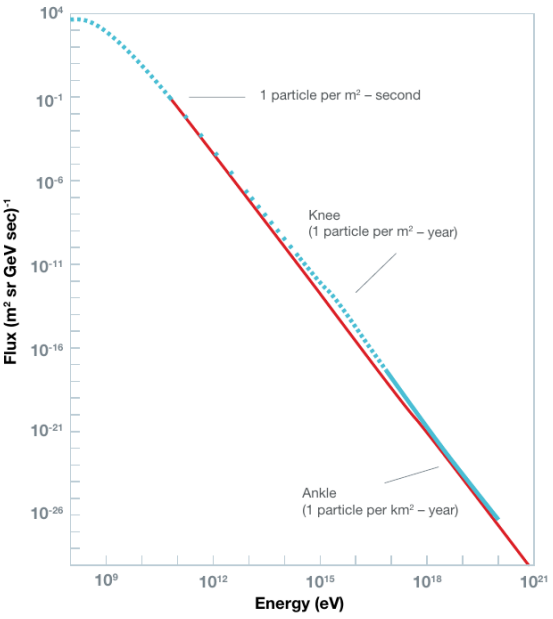

Figure 2.5: **Spectrum of Galactic Cosmic Rays, from Radiation Handbook of Electronics [11]**

For these reasons, the value $m^2 * sr$ corresponds to the area over which we want to calculate the number of particles. Therefore to calculate the flux of 1GeV particles in a $cm^2 = 0.0001\, m^2$ using **Figure 2.5**, we have:

$$Flux_{part} = 10^3 \frac{particle}{m^2\, sr\, GeV\, sec} \cdot 1GeV \cdot 0.0001 m^2 = 0.1 \frac{particle}{cm^2\, sec} = 6 \frac{particle}{cm^2\, min} \quad (2.10)$$

Solar Cosmic Rays

The Sun is a star that continuously converts hydrogen into helium through nuclear fusion, ejecting more than $60 MW/m^2$. Externally it is composed of a visible proton emitting photosphere and a corona composed by plasma. The solar magnetic field is manifested by sunspots, relatively cold spots where there is a concentration of magnetic field, unlike the Earth, the Sun has multiple magnetic poles **Figure 2.6 on the following page** . The appearance of new sunspots is a prelude to a period of high solar activity leading to Coronal Mass Injection (CMEs), solar flares, prominences and coronal rings. These activities in turn depend on the sun's 11-year cycle; during the first 4 years we have an inactive sun with a minimum number of sunspots and in the remaining 7 years we have an increase of the activity with many sunspots.

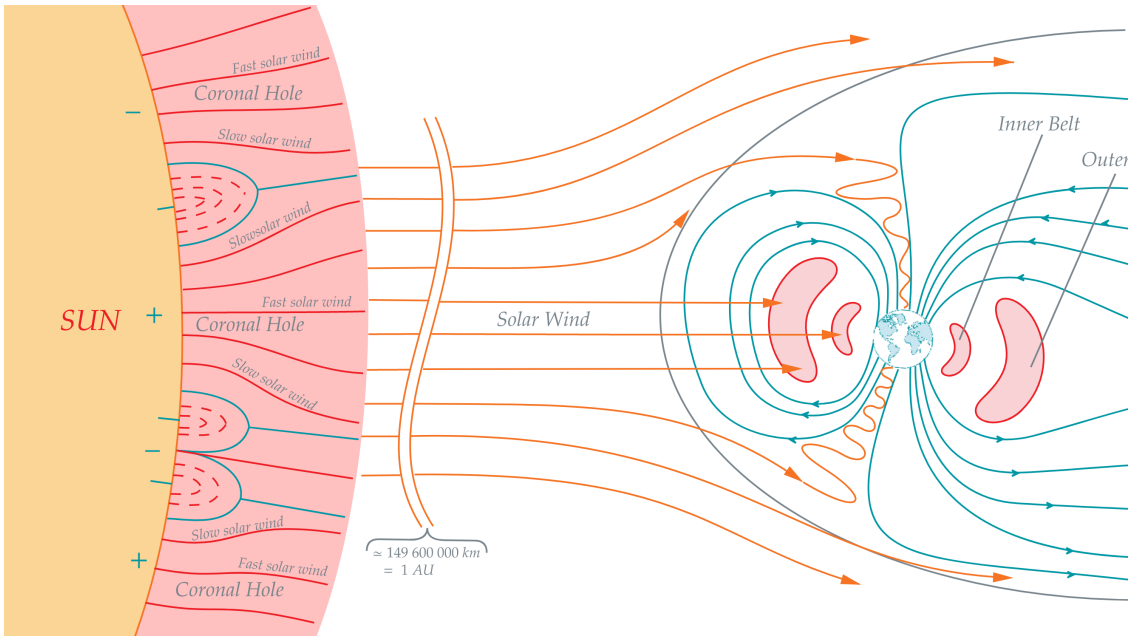

Figure 2.6: **The figure shows the variation of the Earth's magnetic field due to the solar wind. The Van Allen Belts are also highlighted in light red on either side of the Earth. The Corona of the sun show how magnetic field changes particle ejection.**

The emitted particles are mainly photons, protons, electrons, alpha particles and a small number of heavy ions, all of which are energized and ejected at about 400-700km/s out of the sun. The ejection is due to the high temperatures (6000K) and the inability of the Sun's gravitational force to hold the particles because it is too weak at that distance from the nucleus. The solar wind at a distance of 1AU from the Sun ($149\,597\,900\,km$) strikes the Earth with $500 \cdot 10^6$ $particles/(cm^2\,sec)$ at a speed of 300-450 km/s, the average kinetic energy of protons is about $1keV$, while for electrons it is $10eV$ [12]. Because of the low kinetic energy of the solar wind, it is normally trapped in the Van Allen Belts or deflected by the Earth's magnetic field. But when phenomena such as solar flares, CMEs and prominences occur the energy of the ejected particles is higher and particles with $E > 1GeV$ can be detected on the ground, these energized particles are called Solar Energetic Particles (SEPs) and have energy of $1\,MeV$ to $1\,GeV$. The main problem with SEPs is that over a period of a few hours or days they have a very high flux of up to an excess of $500\,000/(cm^2\,sec)$, so solar activity can create serious problems for space mission electronics.

When the solar wind reaches the Earth, it changes the Earth's magnetosphere as shown in **Figure 2.6**. The Earth's magnetic field then deflects much of the solar wind, and the particles that manage to enter the atmosphere (the SEPs and GCRs) collide with hydrogen and oxygen to form a cascade of particles that make up the secondary cosmic rays. As they descend into the atmosphere, Secondary Cosmic Rays continue to collide with nuclei in the air, until they arrive attenuated on earth as Terrestrial Cosmic Rays. The same thing happens to GCRs that normally have higher energies and flux.

Manufacture and Package materials Radiations

The materials used to build the die and package have radioactive impurities that release alpha particles due to natural decay to more stable atoms. For example, ^{232}Th decays by emitting 6 alpha particles from 4MeV to 8MeV, while ^{238}U releases 8 alpha particles with similar energy. For example, in the solder bumps there are some isotopes that create a flux from 7 to 0.002 $alpha\ particles/(cm^2\,hr)$ [11]. For these reasons, the primary source of alpha particles for a circuit is the package; in fact, any particle emitted by radioactive impurities can be the source of a SEE, since it is ionized and cannot be shielded.

Today the limit reached by ULA (Ultra Low Alpha) materials is $\sim 0.001\,\alpha/(cm^2\,hr)$, if each alpha particle generated a SEE we would have about one million FITs, but in reality we have only from 1000 to 100 FITs in common chips since not all alpha particles generate a SEE. This is why in terrestrial applications the main cause of error is due to the isotopes impurities of the package since neutrons are few and rarely create SEE. As the altitude rises, the effects are reversed because the neutrons are increasingly energetic and they cause more SEEs.

Medical Radiation

Radiation in medicine is used in exams (X-rays) and sterilization (X-rays, gamma rays, e-Beams), normally the maximum observable dose in an examination is 20mSv (millisievert) equal to $2\,rad_{Si}$, this dose is normally harmless even for commercial electronics. On the other hand for sterilization the radiation is much higher ($\sim\ 5\,Mrad$) making it impossible for even military electronics to survive, so normally if you have electronics in a device to be sterilized, you either use other techniques or switch it off in order to reduce the damage. In fact the TID depends very much on the electric field, which is absent if the circuit is switched off [11].

Nuclear Power Plants

In nuclear power plants and industrial environments, there are sources of X-rays, gamma rays, e-beams and neutrons. TID effects are the main effects on electronics, although in particular applications (such as measuring the temperature of the cooling ponds of nuclear reactors) the electronics are subject to too much radiation (even for hardened circuits) and must therefore be replaced periodically to prevent deterioration.

Nuclear Weapon

The effects of a nuclear explosion depend on the location of detonation and on the power of the bomb. Many nuclear bomb experiments are carried out in the air, the Hiroshima bomb itself detonated at an altitude of 580m, and some explosions have occurred in water and soil.

For an air blasting, immediately after the explosion is formed a fireball filled with strong radiation and temperatures of $10\,000°C$, it expands over a 1km radius for Megaton. The fireball in turn creates a pressure wave that reaches 5 to 10 psi and speeds up to $1000km/h$, reaching a distance of 5-7km for Megaton. Thus about 50% of the bomb's

energy is converted into the explosion, while 3% becomes thermal energy which heats the explosion site and can explode fuel reserves up to 10km away per Megaton. The initial radiation in the fireball makes up about 5 % of the bomb's energy and is composed of gamma particles (at the speed of light (c)), X-rays and neutrons (at 15 % of speed of light with $12.14MeV$). After the initial explosion, 35% of the energy is converted into Fallout, a residual radiation composed of secondary fission products and neutron-activated products that fall out of the atmosphere for weeks after the explosion [11].

Another very important effect for the circuits is the EMP generated immediately after the explosion, in fact the radioactive emission reacts with the atmosphere, the ionosphere and the magnetic field in three different phases: in the first phase there is a short pulse of a few nanoseconds caused by the hydrogen and oxygen ionized by the Gamma particles, followed immediately by the second phase with a pulse of about 1sec produced by the reflected Gamma rays and by the reactions of the neutrons with the atmospheric nuclei in the air. Finally, the last pulse is formed by the radiation ionizing the upper ionosphere and distorting the magnetic field. This variation in the magnetic field couples with the energy transmission lines, creating strong pulses in the distribution network, which destroys devices and transformers and causes extensive damage [11].

The circuits involved in a nuclear explosion, depending on the distance of the epicentre, may suffer all or some of the above effects.

2.2.2.0 Radiation Effects on ICs

As far as *nuclear radiation* and *Cosmic Rays* are concerned, there is a bombardment of the IC with Alpha or Neutron particles, which penetrate the material and release energy in the form of electron-hole pairs. Depending on the energy of the colliding particle and the sensitivity of the circuit the generated charges can cause a bit flip or soft error.

The sensitivity of the circuit is expressed in *Critical Charge*, which is the charge required in a circuit to create a bit flip. The energy of the particle is referred to as *Stopping Power* that is the energy lost per unit length by the trace left in the material by an Alpha Particle, it is measured in $eV/\mu m$.

The *interaction mechanism of Alpha particles and Neutrons* is different: Alpha particles directly generate electron-hole pairs (this is why the SP refers to Alpha particles), while Neutrons interact with the atoms of the material in an elastic or anaelastic mode. The most dangerous interaction is the anaelastic one, because Neutrons decay into other particles (Alphas, Pions, Muons, Neutrons, Deuterons and Tritons) which in turn generate charges in the material. Normally the particles generated by Neutrons have a higher Stopping Power than Alpha particles and lower penetration ranges. Because of this, Neutrons generate a high charge for a short time (hence high current pulses) while Alpha particles create a charge streak that lasts longer (creating low but prolonged currents)[2].

In the case of Neutrons impact, an example of how the Soft Error Rate can be modelled is the following [13]:

$$SER_{circuit} \;=\; K\,\phi_{Neutrons}\,A\,e^{\frac{Q_{crit}}{Q_{coll}}} \tag{2.11}$$

Where K is a constant depending on the technological processes, $\phi_{Neutrons}$ is the Neutron flux, A is the area of the IC involved, Q_{crit} is the Critical Charge and $Q_{coll} = collected\ \ charge\ /\ generated\ \ charge$ (the ratio between the collected and generated charge

per unit volume). From the formula **Figure 2.11 on the preceding page** it can be seen that as the critical charge decreases, the SER of the circuit increases; there is also a linear dependence with the area and the neutron flux.

The effects of radiation on the components concern the various types of problems that generate the physical mechanisms explained above. They can be divided into Cumulative Effects and Single Event Effects, the former is caused by continuous exposure to energized particles and the latter is due to the effects of a single particle collision. The effects of each group are described in detail below.

Cumulative Effects CEs

The cumulative effects of radiation cause progressive degradation of the components, in fact the exposure to primary and secondary cosmic rays generates long-term changes in the ICs, these defects lead initially to component degradation and subsequently to faults.

There are three main cumulative effects:

- **Surface Charging Damage Effect** (SCDE) :The charges generated by an energy particle can accumulate inside an insulating material in the IC and if the phenomenon continues, they generate electrostatic discharge (ESD). Normally an ESD create noise, bit-flip, latch-up and false signals [14]. The more energized the particles, the more frequent this phenomenon occur.

- **Total Ionizing Dose** (TID) : In this case the charges created by the particles are deposited in the bulk or other active parts of the IC such as the gate, these charges lead to degradation of the V_{th}, Leakage currents and timing skew. The TID is expressed in Gray (Gy) or rad ($100rad = 1Gy$) where $1Gy = 1j/kg$, normally in space or avionics missions the typical received TID varies from 1 to 100 $krad_{Si}$ [6], it usually depends on the orbit, shielding and many other factors that vary the incident radiation on the chip.

- **Total Non Ionizing Dose** (TNID) : TNID is that portion of particles that do not create electron-hole pairs but instead directly apply a momentum to the semiconductor material. This energy applied to the lattice crystal is transformed into defects and variations from the crystal shape. In turn the degradation of the crystal structure leads to degradation in the parameters of the component, especially in optoelectronic systems [6].

These effects occur mainly in the avionics and space environment where the particles are more energetic and their flux is orders of magnitude higher than on earth.

Single Event Effects

Single Event Effects are due to the charges deposited by the particles, SEEs can be either temporary or permanent effects. They are divided into Destructive SEEs that generate permanent damage in the circuit and Non Destructive SEEs that cause damage reparable with fault tolerance mechanisms or by a system reboot.

There are four principal Non Destructive SEEs:

- **Single Event Transient** (SET) : This event is a temporary voltage change in a node of an integrated circuit, it is caused by a single particle releasing charges as it penetrates the material. SEUs, SEFIs and other spurious phenomena can be generated by a SET.
- **Single Event Upset** (SEU) : The SEU is an event that corresponds to a bit-flip of a memory element: a latch, a Flip-Flop or e.g. the cell of a flash . If the corrupted memory is not used or is corrected by ECC, it is called a Silent SEU. The probability of a SEU depends very much on the critical circuit charge, the supply voltages and the size of the transistors.
- **Multiple Cell/Bit Upset** (MCU, MBU) :Both MCU and MBU are caused by the corruption of the value of two or more adjacent cells by a particle. The difference is that MCU occurs between cells of different words while MBU occurs between cells of the same word. This difference is substantial because in a memory with ECC that can correct only one bit, MCUs are correct while MBUs can't be correct since they cause two or more errors in the same word.

  These phenomena are increasing in new generations of memories since the proximity between cells continues to grow [6].
- **Single Event Failure Interrupt** (SEFI) : This event is defined as the soft error that causes a reset or stall of a circuit component or the whole system [6]. It is usually caused by corruption of control memory or program memory, by communications disturbances and internal control signals [15].

  There are also three different types of SEFI, some can be repaired with a software reset, other need power cycling due to a stall and some need partial reprogramming due to corrupted program data.

Instead destructive SEEs are more technology dependent and they is divided in [6]:

- **Single Event Latchup** (SEL) : This event occur when the parasitic PNPN or NPNP thyristor of the CMOS structures are turned on. When this happens and the power supply is on, the component can be destroyed by thermal effects. This mechanism don't exists in SOI systems because there are no parasitic thyristor.
- **Single Event Snap Back** (SESB) : This event occurs when NPN or PNP parasitic bipolar structures in CMOS circuits are activated. These parasitic transistors can self-sustain a current that can be destructive. SOI technology also suffers from this effect because parasitic transistors are present in these systems.
- **Single Event Hard Error** (ESHE also Stuck-bit) : The ESHE or Stuck-bit is a permanent or intermittent modification of a memory element. This applies to both memories and digital circuits. It differs from an ESHE because it is permanent, in the sense that that memory cell can no longer be used by the system after the event.
- **Single Event Gate/Dielectric Rupture** (SEGR, SEDR) : This event indicates the breakdown of a gate oxide or dielectric by a single particle. SEGR and SEDR are dangerous events because they have much faster dynamics than SEL, SESB and SEHE. For this reason, there is no protective circuitry against these events. In any

case they are rarer events and occur mainly in the space environment where there are very energetic particles.

2.2.3 Soft Errors

All the possible transient errors analyzed in the previous chapters are Soft Errors, these errors that remain in the memory elements (e.g. flip-flops, latch) only until a new value is written. When fault detection and correction systems are applied to a system, two categories of errors are created at system level:

- **SDC (Silent Data Corruption) :** a faulty bit without detection is read and it modifies the final result of the program.
- **DUE (Data Unrecoverable Error) :** is when a faulty bit with only error detection is read. At this point if the bit changes the final result is True DUE, otherwise it is a False DUE.

SDC errors are dangerous because they occur on bits for which errors cannot be detected or corrected, these errors can lead to a system crash and must be transformed into DUE errors by error detection or corrected. The advantage of converting SDC errors into DUE is that DUE are detectable and they lead the system in fail-stop mode. In fact once a DUE is detected, the system stops and evaluates how to continue execution. At the operating system level, if the error is inside a process we can kill only that one and we talk about process-killer DUE, otherwise we say that DUE is system-killer because the OS has to restart the machine to avoid the propagation of the error.

DUE and SDC errors have different effects on dependability; DUE causes Availability penalties because the system has to recover, while SDC lowers Reliability, Safety and Availability because it can crash the system. For these reasons normally there is a budget of TDC and DUE expressed in FIT, for example 228FIT of SDC (500 years of MTTF) and 57000FIT of DUE (2 years of MTTF) by specification.

Time Vulnerability Factor (TVF)

The TVF is the fraction of time in which the circuit is vulnerable to errors. It is calculated using the window of vulnerability (WOV), which is the time within the clock period in which the circuit can be subject to SEE, for example in edge-triggered flip-flops the WOV is equal to half the clock because only in that interval the FF change state if it is struck by a particle (only in the high/low phase is the data sampled and held). The TVF is therefore the ratio between the vulnerability window and the clock period, so for an edge-triggered FF the TVF is equal to 50%.

Actually, the calculation of the TVF is more complicated because the propagation delay of the circuit has to be taken into account, assuming in fact a period of 1ns and an average combinatorial delay of 700ps, in this case the TVF will be lower than 50% since some faults injected in the first 500ns can be masked by the logic delay.

CHAPTER

3 Travulog and HTravulog

To create a configurable FT architecture it is necessary to apply fault tolerant techniques to all the blocks of the IF Stage and then allow to apply or not techniques in each block by means of parameters. It was therefore decided to create a metalanguage within the SystemVerilog (SV) that allows the creation of the new architecture starting from a template and a base module, in this way we can define a template and apply it to each block of the IF stage. As you can see in figure **Figure 3.1** this transformation is done by a Python object linked to the template, when you pass the base module to the object the new module is created. The new FT architecture is created on the basis of the Travulog template and the base module, it is therefore an interface layer that makes the old module Fault Tolerant. To allow the conversion through the Travulog object it is necessary to have an object that contains all the data of the associated SystemVerilog base module, we have therefore created a new Python class that parses a SV file containing a module, this class has been called moddata. As shown in the figure **Figure 3.1** the file containing the basic module must be transformed into the corresponding moddata object and then passed to the Travulog object which generates the new architecture.

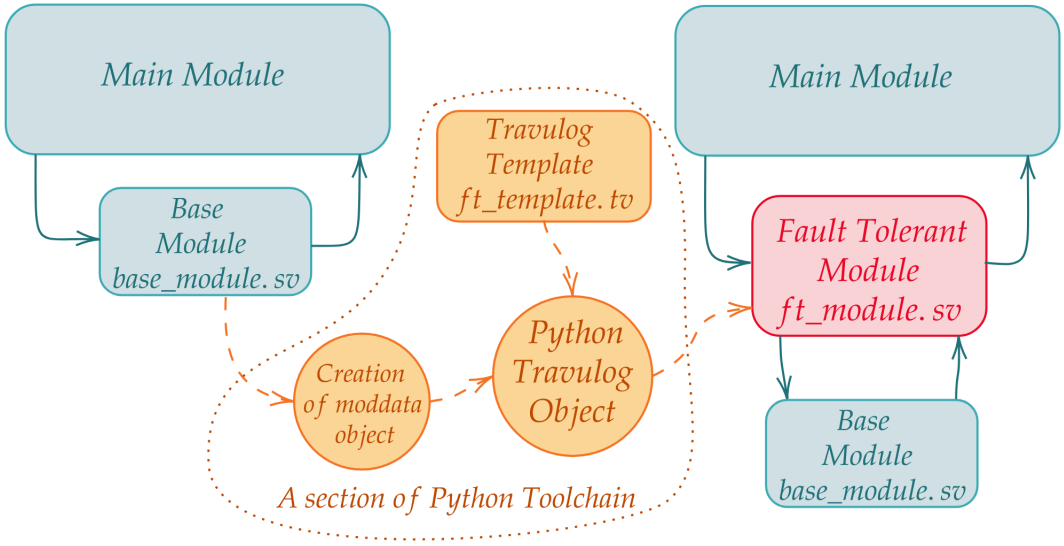

Figure 3.1: **Flow diagram of architecture transformation using Travulog template**

To create the new language we started from the fault tolerant cv32e40p_compress_decoder_ft and we created a series of commands that allow us to create the FT compress decoder from the basic one. We will now analyze the various Travulog commands referring to the compress decoder, comparing pieces of code from the Travulog template and its conversion in

SVerilog.

3.0.1 Declaration of ports

In the listing 3.1 there is a part of the Travulog template, this piece of code allows to generate the System Verilog of the listing 3.2. the commands in this part of the template are these:

PARAMETER_DECLARATION: The command PARAMETER_DECLARATION copy the parameter declaration from the BLOCK module in the new System Verilog module. BLOCK is an identifier used in the Travulog object that should be linked to a moddata object. Note that you can have multiple ID since ids are managed as a dictionary, e.g if you give "BLOCK":moddata_obj1, "BLOCK2":moddata_obj2 to Travulog object you can use both BLOCK and BLOCK2 identifiers in the Travulog code.

DECLARATION_FOREACH: This command cycle on the given signals and it substitute: INOUT with "input" or "output", BITINIT with de bits definition and SIGNAME with the name of the signal. The first argument is the module id, the second one is the type of signal: IN for input port of the module, OUT for output, IN_OUT for both input and output and INTERN for internal signals of the module. You can also indicate some signals to exclude by the list using "NOT sig1 sig2 ... " as you can see at line 7, indeed in the example clk and rst_n signals are excluded since they should not be triplicated. This command can also be used for the declaration of internal signals and for assign statement as we see later.

MODULE_NAME: This is a parameter that is substituted with the name given to the Travulog object, it is the name of the new module.

```
1  module MODULE_NAME
2
3    PARAMETER_DECLARATION BLOCK
4
5  (
6    // compressed decoder input output
7    DECLARATION_FOREACH BLOCK IN_OUT NOT clk rst_n
8      INOUT logic [2:0]BITINIT SIGNAME,
9    END_DECLARATION_FOREACH
10
11
12   input logic clk,
13   input logic rst_n,
14
15   // fault tolerant state
16   input logic [2:0] set_broken_i,
17   output logic [2:0] is_broken_o,
18   output logic err_detected_o,
19   output logic err_corrected_o
20 );
21
```

Listing 3.1: Travulog Code

```
module cv32e40p_compressed_decoder_ft
#(
  parameter FPU = 0
)
(
  // compressed decoder input output
  input  logic [2:0] [31:0] instr_i,
  output logic [2:0] [31:0] instr_o,
  output logic [2:0] is_compressed_o,
  output logic [2:0] illegal_instr_o,

  input logic clk,
  input logic rst_n,

  // fault tolerant state
  input logic [2:0] set_broken_i,
  output logic [2:0] is_broken_o,
  output logic err_detected_o,
  output logic err_corrected_o
);
```

Listing 3.2: SVerilog code derived

3.0.2 Internal signals and assign

In the following listings you can see the continuation of the previous ports definition, the first "declaration_foreach" create the signals that connect the three block outputs to the voter while the second creates block error signals. In the last two lines there is the compound parameter "SIG_NUM-BLOCK-OUT" inside the square bracket, this parameter is substituted in the right listing with the number (SIG_NUM) of output ports (OUT) of the module (BLOCK) minus one, anyway instead of OUT you can use IN, PARAM, INTERN or IN_OUT in order to have the correct signals number.

```
1  // Signals out to each compressed
2  // decoder block to be voted
3  DECLARATION_FOREACH BLOCK OUT
4  logic [2:0]BITINIT SIGNAME_to_vote ;
5  END_DECLARATION_FOREACH
6
7  // Error signals
8  DECLARATION_FOREACH BLOCK OUT
9  logic [2:0] SIGNAME_block_err ;
10 END_DECLARATION_FOREACH
11
12 // Signals that use error signal to
13 // find if there is one error on each
14 // block, it is the or of previous signals
15 logic [2:0] block_err_detected;
16 logic [SIG_NUM-BLOCK-OUT:0] err_detected;
17 logic [SIG_NUM-BLOCK-OUT:0] err_corrected;
18
```

Listing 3.3: Travulog Code

```
// Signals out to each compressed
// decoder block to be voted
logic [2:0] [31:0] instr_o_to_vote ;
logic [2:0] is_compressed_o_to_vote ;
logic [2:0] illegal_instr_o_to_vote ;

// Error signals
logic [2:0] instr_o_block_err ;
logic [2:0] is_compressed_o_block_err ;
logic [2:0] illegal_instr_o_block_err ;

// Signals that use error signal to
// find if there is one error on each
// block, it is the or of previous signals
logic [2:0] block_err_detected;
logic [2:0] err_detected;
logic [2:0] err_corrected;
```

Listing 3.4: SVerilog code derived

3.0.3 Instance

3.1 Travulog

3.2 Hidden Travulog

3.3 V

3.4 CV32E40P core in Pulpissimo

CHAPTER

4 Conclusion

Bibliography

[1] Elena Dubrova. "Fault Tolerant Design : An Introduction". In: *Ece.Nus.Edu.Sg* X.X (2013).

[2] Shubu Mukherjee. *Architecture Design for Soft Errors*. 2008. DOI: 10.1016/B978-0-12-369529-1.X5001-0.

[3] W. K. Chien and F. Hao. "An Extended Building-In Reliability Methodology on Evaluating SRAM Reliability by Wafer-Level Reliability Systems". In: *IEEE Transactions on Device and Materials Reliability* 20.1 (2020), pp. 106–118. DOI: 10.1109/TDMR.2020.2964999.

[4] Hairong Sun, J. J. Han, and H. Levendel. "Availability requirement for a fault-management server in high-availability communication systems". In: *IEEE Transactions on Reliability* 52.2 (2003), pp. 238–244. DOI: 10.1109/TR.2003.812624.

[5] L. Valcarenghi, M. Kantor, P. Cholda, et al. "Guaranteeing High Availability to Client-Server Communications". In: *2008 10th Anniversary International Conference on Transparent Optical Networks*. Vol. 3. 2008, pp. 34–37. DOI: 10.1109/ICTON.2008.4598649.

[6] ECSS. "Space Product Assurance - Techniques for radiation effects mitigation in ASICs and FPGAs handbook". In: *Structure* April (2016).

[7] Paul S. Ho and Thomas Kwok. "Electromigration in metals". In: *Reports on Progress in Physics* 52.3 (1989). ISSN: 00344885. DOI: 10.1088/0034-4885/52/3/002.

[8] *Fault-Tolerant Systems*. 2021. DOI: 10.1016/c2018-0-02160-x.

[9] K. Ota, R. Ichihara, M. Suzuki, et al. "Random Telegraph Noise after Hot Carrier Injection in Tri-gate Nanowire Transistor". In: *2019 Electron Devices Technology and Manufacturing Conference (EDTM)*. 2019, pp. 169–171.

[10] W. Lin, W. Tsai, C. C. Cheng, et al. "Hot-Carrier Injection-Induced Disturb and Improvement Methods in 3D NAND Flash Memory". In: *2019 International Symposium on VLSI Technology, Systems and Application (VLSI-TSA)*. 2019, pp. 1–2. DOI: 10.1109/VLSI-TSA.2019.8804652.

[11] Kirby Kruckmeyer Robert Baumann. *Radiation Handbook for Electronics*. 2020.

[12] "COSMIC-RAY PICTURE OF THE HELIOSPHERE." In: *Johns Hopkins APL Technical Digest (Applied Physics Laboratory)* 6.1 (1985). ISSN: 02705214.

[13] P. Hazucha and C. Svensson. "Impact of CMOS technology scaling on the atmospheric neutron soft error rate". In: *IEEE Transactions on Nuclear Science* 47.6 (2000), pp. 2586–2594. DOI: 10.1109/23.903813.

[14] Mengfei Yang, Gengxin Hua, Yanjun Feng, et al. *Fault-Tolerance Techniques for Spacecraft Control Computers*. 2017. DOI: 10.1002/9781119107392.

[15] P. V. Nekrasov, A. B. Karakozov, D. V. Bobrovskyi, et al. "Investigation of Single Event Functional Interrupts in Microcontoller with PIC17 Architecture". In: *2015 15th European Conference on Radiation and Its Effects on Components and Systems (RADECS)*. 2015, pp. 1–4. DOI: 10.1109/RADECS.2015.7365625.

[16] V. Lari, A. Tanase, J. Teich, et al. "A co-design approach for fault-tolerant loop execution on Coarse-Grained Reconfigurable Arrays". In: *2015 NASA/ESA Conference on Adaptive Hardware and Systems (AHS)*. 2015, pp. 1–8. DOI: 10.1109/AHS.2015.7231157.

[17] B. Vinnakota and N. K. Jha. "A dependence graph-based approach to the design of algorithm-based fault tolerant systems". In: *[1990] Digest of Papers. Fault-Tolerant Computing: 20th International Symposium*. 1990, pp. 122–129. DOI: 10.1109/FTCS.1990.89347.

[18] Kuang-Hua Huang and J. A. Abraham. "Algorithm-Based Fault Tolerance for Matrix Operations". In: *IEEE Transactions on Computers* C-33.6 (1984), pp. 518–528. DOI: 10.1109/TC.1984.1676475.

[19] K. Siozios and D. Soudris. "A Methodology for Alleviating the Performance Degradation of TMR Solutions". In: *IEEE Embedded Systems Letters* 2.4 (2010), pp. 111–114. DOI: 10.1109/LES.2010.2083632.

[20] G. Asadi and M. B. Tahoori. "An analytical approach for soft error rate estimation in digital circuits". In: *2005 IEEE International Symposium on Circuits and Systems*. 2005, 2991–2994 Vol. 3. DOI: 10.1109/ISCAS.2005.1465256.

[21] L. Sterpone and M. Violante. "A new analytical approach to estimate the effects of SEUs in TMR architectures implemented through SRAM-based FPGAs". In: *IEEE Transactions on Nuclear Science* 52.6 (2005), pp. 2217–2223. DOI: 10.1109/TNS.2005.860745.

[22] L. A. C. Benites and F. L. Kastensmidt. "Automated design flow for applying Triple Modular Redundancy (TMR) in complex digital circuits". In: *2018 IEEE 19th Latin-American Test Symposium (LATS)*. 2018, pp. 1–4. DOI: 10.1109/LATW.2018.8349668.

[23] A. Namazi, M. Nourami, and M. Saquib. "A voterless strategy for defect-tolerant nano-architectures". In: *2008 IEEE International Symposium on Nanoscale Architectures*. 2008, pp. 38–45. DOI: 10.1109/NANOARCH.2008.4585790.

[24] T. Koal, S. Scharoba, and H. T. Vierhaus. "Combining Correction of Delay Faults and Transient Faults". In: *2015 IEEE 18th International Symposium on Design and Diagnostics of Electronic Circuits Systems*. 2015, pp. 99–102. DOI: 10.1109/DDECS.2015.23.

[25] H. T. Vierhaus. "Combining fault tolerance and self repair in a virtual TMR scheme". In: *2013 Signal Processing: Algorithms, Architectures, Arrangements, and Applications (SPA)*. 2013, pp. 12–18.

[26] A. Dominguez-Oviedo and M. A. Hasan. "Error Detection and Fault Tolerance in ECSM Using Input Randomization". In: *IEEE Transactions on Dependable and Secure Computing* 6.3 (2009), pp. 175–187. DOI: 10.1109/TDSC.2008.21.

[27] L. A. Tambara, F. L. Kastensmidt, J. R. Azambuja, et al. "Evaluating the effectiveness of a diversity TMR scheme under neutrons". In: *2013 14th European Conference on Radiation and Its Effects on Components and Systems (RADECS)*. 2013, pp. 1–5. DOI: 10.1109/RADECS.2013.6937382.

[28] J. -Y. Yang and S. -Y. Huang. "Fault and Soft Error Tolerant Delay-Locked Loop". In: *2020 IEEE 29th Asian Test Symposium (ATS)*. 2020, pp. 1–6. DOI: 10.1109/ATS49688.2020.9301553.

[29] M. M. Hafidhi and E. Boutillon. "Hardware error correction using local syndromes". In: *2017 IEEE International Workshop on Signal Processing Systems (SiPS)*. 2017, pp. 1–6. DOI: 10.1109/SiPS.2017.8109995.

[30] M. Masadeh, A. Aoun, O. Hasan, et al. "Highly-Reliable Approximate Quadruple Modular Redundancy with Approximation-Aware Voting". In: *2020 32nd International Conference on Microelectronics (ICM)*. 2020, pp. 1–4. DOI: 10.1109/ICM50269.2020.9331771.

[31] H. Pham, S. Pillement, and S. J. Piestrak. "Low-overhead fault-tolerance technique for a dynamically reconfigurable softcore processor". In: *IEEE Transactions on Computers* 62.6 (2013), pp. 1179–1192. DOI: 10.1109/TC.2012.55.

[32] N. D. P. Avirneni and A. Somani. "Low Overhead Soft Error Mitigation Techniques for High-Performance and Aggressive Designs". In: *IEEE Transactions on Computers* 61.4 (2012), pp. 488–501. DOI: 10.1109/TC.2011.31.

[33] F. S. Khodadad and M. Jahed. "Optimization of a Cascading TMR system configuration using Genetic Algorithm". In: *IEEE 10th International Conference on Industrial Informatics*. 2012, pp. 470–474. DOI: 10.1109/INDIN.2012.6300853.

[34] M. S. Farias, N. Nedjah, and P. V. R. de Carvalho. "Resilient Hardware Design for Critical Systems". In: *2019 IEEE 10th Latin American Symposium on Circuits Systems (LASCAS)*. 2019, pp. 237–240. DOI: 10.1109/LASCAS.2019.8667549.

[35] C. J. Hescott, D. C. Ness, and D. J. Lilja. "Scaling Analytical Models for Soft Error Rate Estimation Under a Multiple-Fault Environment". In: *10th Euromicro Conference on Digital System Design Architectures, Methods and Tools (DSD 2007)*. 2007, pp. 641–648. DOI: 10.1109/DSD.2007.4341535.

[36] Leander Jehl and Hein Meling. "Towards Byzantine fault tolerant publish/subscribe: A state machine approach". In: *Proceedings of the 9th Workshop on Hot Topics in Dependable Systems, HotDep 2013*. Association for Computing Machinery, 2013. ISBN: 9781450324571. DOI: 10.1145/2524224.2524232.

[37] G. G. Maxwell. "Pacemaker reliability: design to explant". In: *Annual Reliability and Maintainability Symposium 1995 Proceedings*. 1995, pp. 460–464. DOI: 10.1109/RAMS.1995.513285.

[38] H. Kim, M. Jin, H. Sagong, et al. "A systematic study of gate dielectric TDDB in FinFET technology". In: *2018 IEEE International Reliability Physics Symposium (IRPS)*. 2018. DOI: 10.1109/IRPS.2018.8353577.

[39] K. Joshi, S. W. Chang, D. S. Huang, et al. "Study of dynamic TDDB in scaled FinFET technologies". In: *2018 IEEE International Reliability Physics Symposium (IRPS)*. 2018. DOI: 10.1109/IRPS.2018.8353665.

[40] Z. Zhang, R. Wang, Y. Wang, et al. "Impacts of Channel Doping on NBTI Reliability and Variability in Nanoscale FinFETs". In: *2019 IEEE 26th International Symposium on Physical and Failure Analysis of Integrated Circuits (IPFA)*. 2019, pp. 1–4. DOI: 10.1109/IPFA47161.2019.8984834.

[41] S. Das, T. P. Dash, S. Dey, et al. "NBTI Degradation and Recovery in Nanowire FETs". In: *2019 Devices for Integrated Circuit (DevIC)*. 2019, pp. 70–74. DOI: 10.1109/DEVIC.2019.8783566.

[42] N. Seifert and N. Tam. "Timing vulnerability factors of sequentials". In: *IEEE Transactions on Device and Materials Reliability* 4.3 (2004), pp. 516–522. DOI: 10.1109/TDMR.2004.831993.