UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL
INSTITUTO DE INFORMATICA
PROGRAMA DE POS GRADUACAO EM COMPUTACAO

DESIGNING SINGLE EVENT UPSET MITIGATION TECHNIQUES FOR LARGE SRAM-based FPGA DEVICES

Thesis Proposal

by
Fernanda Gusmao de Lima

Advisor:
Ricardo Augusto da Luz Reis

Porto Alegre, February 11th, 2002

ACKNOWLEDGMENT

I am very grateful to all of you that have been with me all my life: my family, my friends and my colleagues. Every person that I know is important to me and each one has something that has made the difference. Everyday when I make decisions, I know that each one has contributed in some how for these choices and I am sure that my life is so wonderful because of that.

First I would like to thank God for the wonderful life that I have. Thank you very much God for keeping the courage and the enthusiasm on me every day. I would like to thank my family that always looks after me. Thank you mom and dad for support me and for giving to me all the best of your heart. For all that I have conquered in my life so far, I need to thank you both. My friends were all presented in my life too giving to me assistance to get through the difficulties. I want to thank you all of you to make my life so happy and fulfilled of emotion. I want to thank you all professors that have helped me since I started in the GME at 1994. I learned a lot with all of them in all stages of the researches. I am sure that my work today has something from all the work that we have done together since the beginning. You are more than my instructors, you are my friends. Some of them I want to thank you more explicitly for the constant support in the thesis in the past few years. I want to thank professor Ricardo Reis, who has been my advisor since I started in the GME, for the constant and infinity support in my studies. Thank you very much for always trust on me and to encourage me every moment, in every new challenge. I want to thank professor Luigi Carro for inviting me to work in the 8051 again. After many years after we first worked together in the 8051 architecture, I had the chance again to perform this investigation in the 8051 architecture protecting against radiation. This work has opened new doors for me that I am very glad. Thank you Luigi for supporting me and for encourage me every moment with the studies. You are always present to help me in my work, to discuss new ideas and to guide me into new investigations. I want also to thank the researcher Raoul Velazco from the laboratory TIMA in Grenoble for the constant support. The internship that I did in his lab for 6 months was very important for my studies and for my personal life too. I was very pleased for that opportunity. During that period I had the chance to work in the state of the art research related to radiation effects, SEU mitigation techniques and fault injection in digital circuit. There I also had a chance to meet great colleagues and friends. I also want to thank all my colleagues from Xilinx in USA, where I had also the opportunity to be an intern for 11 months. There I had the chance to work with the state of the art products and developments related to FPGAs and to meet wonderful friends. I want to give special thanks to Joe Fabula, Rick Padovani and Carl Carmichael for helping me during all these past months and for always giving me confidence at work. This internship was fundamental for my thesis work and it was an immense experience for my personal life too. Thank you all!!!

CONTENTS

LIST OF FIGURES

LIST OF TABLES

ABSTRACT

This thesis proposes the study and development of SEU mitigation techniques in programmable architectures such as Field Programmable Gate Arrays (FPGAs) customizable by SRAM. Spacecraft electronic designers increasingly demand high performance microprocessors and programmable logic, because of their high performance and flexibility. Because SRAM based FPGAs are reprogrammable, they offer the additional benefits of allowing on-orbit design changes. Data can be sent after launch to correct errors or to improve system performance. System including microprocessors and FPGAs covers a wide range of space applications, and consequently, they are the objects of study.

Many proposed SEU mitigation techniques have been proposed in the past years in order to avoid bit flips in the storage cells of microprocessors and FPGAs. In the case of SRAM based FPGAs this problem is more eminent as a large portion of the device is composed of SRAM memory cells (LUTs, customization bits, routing, flip-flops, embedded memory, etc.) A SEU immune circuit may be fulfilled through a variety of mitigation techniques based on redundancy. Redundancy is provided by extra components (hardware redundancy), by extra execution time (time redundancy), or by a combination of both. Each technique has some advantages and drawbacks and there is always a compromise between area overhead, performance and power

The consideration of using FPGA to space applications are fairly recent and there is a lot of work to be done in this area so far. Presently, there is no total efficient solution for SRAM based FPGAs that can assure 100% of reliability in all conditions for SEU. This thesis has the goal to investigate the techniques used nowadays and to propose improvements in order to increase reliability. Because FPGAs are composed of combinational and sequential logic, and more recently embedded processors, previous works considering standard integrated circuits can be adapted to the programmable logic architecture by finding the best tradeoff among area overhead, performance penalties, single and multiple upsets correction, process technology and implementation cost.

The Virtex® family FPGA is the most high density and high performance FPGA available in the market nowadays and consequently it was chosen for the case of study. Virtex® is a very attractive architecture for space applications, because of its high density, high performance and re-programmability feature. The technique used to protect the designs implemented in the Virtex FPGA is the Triple Modular Redundancy (TMR) with voters. Although the TMR solution is very attractive because it does not need changes in the process masks, TMR shows some limitations as area overhead, resources limitations specially in the number of dedicated clock segments and I/O pads, and the need of a dedicated placement in order to be 100% reliable. The TMR technique in the Virtex® family was tested in a fault injection platform and in a ground test facility. Results showed a high level of immunity when an 8051 protected by TMR was implemented in the FPGA. However, there are very few bits that can provoke an error in the TMR design if upset (less then 1%). Because this number is not zero, and because the TMR presents some limitations in the area overhead and resources availabilities, the necessity to research for new techniques is eminent.

The techniques used nowadays to protect digital circuits can not be simple applied to the FPGA architecture because of the impacts that they can bring in area, performance and power. An FPGA is mainly composed of SRAM cells grouped in diverse logic structures with distinct functionality. This work will investigate new CLB architectures focused to the hardened approach as new routing topologies that will avoid upsets and reduce the probability of upset propagation. Each structure will be analyzed separately and after in the global in order to measure the impact of different solutions in the matrix. The new techniques will be tested by simulation and fault injection.

1 Introduction

Radiation effects on semiconductor devices have always been a meaningful matter since upsets were first experienced in space applications several years ago. Since then, the interest on studying fault tolerant techniques in order to keep integrated circuits operational in such hostile environment has increased, driven by all possible applications of radiation tolerant circuits, such as space missions, satellites, advanced weaponry, high-energy physics experiments and others [NAS01]. Spacecraft systems include a large variety of analog and digital components that are potentially sensitive to radiation and must be protected or at least qualified for space operation.

Single Event Effect (SEE) is the main concern in space applications [BART97], with potentially serious consequences for the spacecraft, including lost of information, functional failure, or loss of control. SEE can be destructive or transient, according to the amount of energy deposited by the charged particle and the location of the strike in the device. This work focus on transient single event effects also called single event upsets (SEUs), and the ways to mitigate this effect on digital circuits. SEU occurs when a charge particle hits the silicon transferring enough energy in order to provoke a bit flip in a memory cell. SEU on devices have been constantly magnified, caused by the continuous technology evolution that has led more and more complex architectures, with immense amount of embedded memories, followed by an amazing scaling down process of transistor dimensions (Moore's Law) [MOO75].

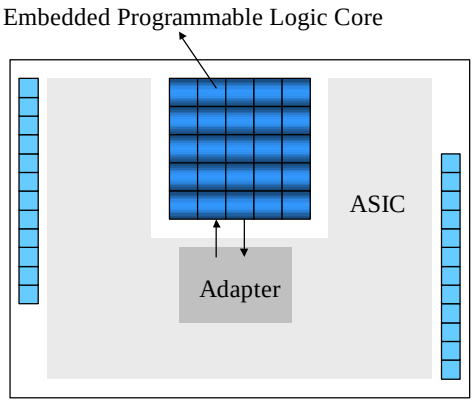
Semiconductor process technology is approaching the ultimate limits of silicon in terms of transistor geometry shrinking, power supply, speed and density [SIA94]. By approaching these limits, the circuits are becoming more and more sensitive to noises coming from magnetic field, signal coupling and radiation field. The necessity to protect them has become more and more eminent [JOH00]. Experiments presented in [NOR96] have indicated that neutrons are capable of producing SEE in avionics. Recent studies also show that memory cells composed of transistors with length smaller than 0.25 ?m and combinational logic composed of transistors with length smaller than 0.13 ?m may be subjected to transient upsets, while operating in the space environment or in atmosphere [BAU01, BOR01]. Terrestrial applications that are determined as critical such as bank servers, telecommunication servers and avionics are more and more considering the use of tolerant techniques to assure reliability. Although many techniques have been developed in the period attempting to avoid SEU, efficient fault tolerant solutions are still a challenge in the future generation semiconductor industry, especially because of the complexity of the new architectures.

The analysis of SEU effects on integrated circuits and the development of SEU mitigation techniques are strongly associated to the target device architecture. For each different circuit, there is a most suitable SEU mitigation solution to be applied. Consequently, in order to suggest a SEU mitigation solution, first it is necessary to investigate the architecture. In the past years the integrated circuit industry has designed complex architectures in order to improve performance and logic density and to reduce cost. Examples of this development are Application Specific Integrated Circuits (ASICs) such as microprocessors and the high-density Field Programmable Gate Arrays (FPGAs). Microprocessors have made a dramatic impact in the way systems are designed, providing a high information process in a single chip. The concept of specific systems including microprocessors covers a wide range of applications, from portable systems to dedicated embedded control devices or computers. FPGAs have also made a major improvement in system designs by adding the reconfigurability feature. More complex structures are constantly being added in the FPGA architecture, supported by substantial increase in logic density and performance in the last few years. Both architectures contain million of transistors located in many distinct logic blocks, making the modeling, test and understanding of such complex architecture very difficult.

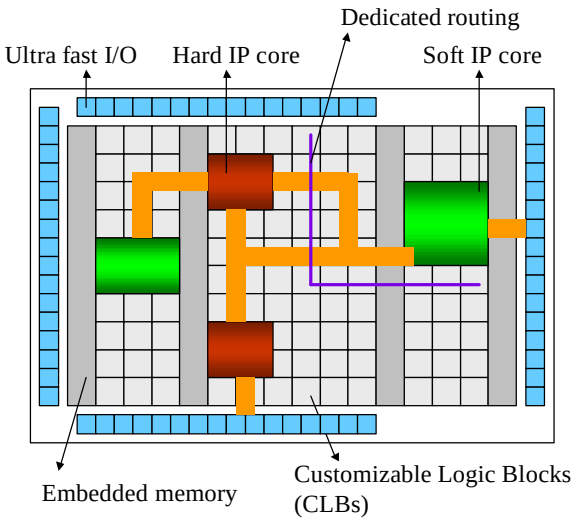
Due to the constant advances in technology in the last few years, Moore's law again, the gap between FPGAs and ASICs in terms of performance has been reduced to a negligible level, for the majority of applications. Consequently, next generation architectures do not claim to reduce that gap anymore, but to merge microprocessors and reconfigurability features in the same device in order to improve performance

and flexibility. However, it is not defined yet which architecture will be embedded in the other one. While some traditional ASIC companies are adding embedded programmable logic cores in their devices, figure 1.1 (a), FPGA companies are adding ASIC IP cores in the FPGA programmable matrix, figure 1.1 (b).

Companies such as LSI Logic [LSI01] and eASIC [EAS01] are proposing programmable cores that can be easily embedded in ASICs in order to increase their flexibility. The idea is to provide to designers the best of both ASIC and programmable technologies: flexibility and some kind of reconfigurability, while delivering low cost, low power, high density, and high performance. On other hand, FPGAs already provide reconfigurability and high performance for many applications, but the necessity of adding either more performance for some applications such as DSP, communication, video and audio, using high-bandwidth and reducing the board space, power and cost, has increased the interest of embedding microprocessors in the programmable matrix. This event had started with the soft cores synthesized to the FPGA architecture in order to get the highest performance and density tradeoff [XIL00a, ALT00]. And it is arriving in a next level of performance and complexity with the new announced Virtex® II generation, which will have hard PowerPC microprocessors core from IBM inside [XIL01a].



(a) Architecture of next generations of ASICs, figure inspired on [LSI01].



(b) Architecture of next generations of FPGAs, figure inspired on [XIL01a].

Figure 1.1 – Next ASIC and FPGA Architecture Generations

In the 2001 Design Automation Conference (DAC) panel, entitled "(When) Will FPGA kill ASICs?", many ASIC and FPGA companies discussed about next generations trends [DAC01]. Which solution is going to be more attractive, the ASIC with embedded programmable logic or FPGAs with embedded soft and hard IP cores? None has given a final answer, but all have agreed that different markets need different solutions, and there is always a price to pay when performance, high density and flexibility are all required together. Both choices force changes in the software design flow. However, it seams that fewer changes must be done in the case of FPGA with IP cores and more advantages can be easily achieved with that.

The space and military market, just as the commercial market, requires high performance devices with low power, low cost, high flexibility and time to market. On the top of that, the space and military applications also request an extra feature: the radiation robustness. Spacecraft electronic designers [SPE00] increasingly demand high performance microprocessors and FPGAs, because of their high performance and flexibility. Because FPGAs are reprogrammable, they offer the additional benefits of allowing on-orbit design changes. Data can be sent after launch to correct errors or to improve system performance. System including microprocessors and FPGAs covers a wide range of space applications, and consequently, they are the objects of study.

This thesis proposes the study and development of SEU mitigation techniques in programmable architectures such as Field Programmable Gate Arrays (FPGAs). The consideration of using FPGA to space applications are fairly recent and there is a lot of work to be done in this area so far. Presently, there is no total efficient solution for SRAM based FPGAs that can assure 100% of reliability in all conditions for SEU. This thesis has the goal to investigate the techniques used nowadays and to propose improvements in order to increase reliability. Because FPGAs are composed of combinational and sequential logic, and more recently embedded processors, previous works considering standard integrated circuits can be adapted to the programmable logic architecture by finding the best tradeoff among area overhead, performance penalties, single and multiple upsets correction, process technology and implementation cost.

In the first phase of the research, available techniques to protect integrated circuits against radiation were studied. The techniques are based on specific process technology such as Silicon on Insulator (SOI) or design modifications such as tolerant memory cells, error detection and correction codes (EDAC) and redundancy. There is a trade-off in each mitigation technique for each type of architecture system, and there is no best unique solution so far. Some of the considered techniques were evaluated in two different architectures: the 8051 micro-controller [INT94] and the Virtex® family FPGA [XIL00a]. These architectures (microprocessor and FPGA) were chosen for their representation in the majority of system requirements in space applications nowadays and to be part of the new generation architectures based on FPGA with embedded hard microprocessor core.

The 8051 micro-controller description used in the experiment was developed at UFRGS [CAR96]. It is composed of Datapath unit, Control unit, State Machine, Instruction unit and internal memory. Although the 8051 micro-controller has a simplified architecture compared to the latest microprocessors, the assumptions made in its architecture can be adapted to any other processor-like circuit. Techniques such as hamming code and radiation tolerant flip-flops were implemented in the 8051 micro-controller [LIM00a, LIM00b]. Fault injection [LIM01a, LIM02a] and simulation was used to analyze the techniques. Results were compared in area and performance.

The Virtex® family FPGA is the most high density and high performance FPGA available in the market nowadays. It is a very attractive architecture for space applications, because of its high density, high performance and re-programmability feature. The Triple Modular Redundancy (TMR) with voters [CAR01a] is used to protect FPGA against SEU because the mitigation can be applied in a high level description language and synthesized in the device without any changes in the mask process. In order to test the TMR in the Virtex® platform, it is necessary to have a design described in VHDL. The same 8051 like micro-controller description was used to test the TMR technique in the programmable platform. There are many advantages of using the same 8051 design such as good description knowledge, importance of micro-controllers IP in FPGA and the possibility of comparison with the previous techniques (Hamming code and SEU hardened memory cells) applied in the same description. The TMR 8051 micro-controller was tested by fault injection [LIM01b] and under proton radiation in a ground facility [LIM02b].

In the second phase of the work, additional SEU mitigation techniques for the Virtex® FPGA architecture are being investigated. The SRAM-based architecture was divided in main blocks classified by functionality such as LUT, flip-flops, customization routing, embedded memory, PLL, etc. New SEU mitigation techniques will be applied and tested for each block. The objective is to show the trade-off of each technique in the Virtex® FPGA in order to improve the results obtained so far. The new techniques are applied in the FPGA design level instead of in the high level description. The solutions require mask changes. This investigation is based on the experience collected in the phase one. This phase also investigate more issues related to the TMR design and solutions that can be applied in order to reduce the error rate. Fault injection and simulation will evaluate the new proposed techniques.

The thesis proposal report is organized as follow. The chapter 2 briefly describes the radiation effects on integrated circuits manufactured using CMOS process and it introduces the techniques available in the literature today, either being commercialized by companies or being studied by researchers, to mitigate the effects of radiation in FPGAs and processor architectures.

Chapter 3 defines the problem of protecting SRAM-based FPGAs against radiation. This chapter shows the architecture analysis of the Virtex® FPGA and all its radiation sensitive area. The Triple Modular Redundancy (TMR) technique for FPGAs is addressed in this chapter. The technique has been analyzed by fault injection and in a radiation ground test facility. These results represent an important base for this work, because they show the limitations of the TMR method on the SRAM-based FPGA, justifying the research of new design techniques for SEU mitigation in SRAM based FPGAs.

Chapter 4 introduces the main proposed techniques that are going to be developed and tested in the next phase of the thesis. The FPGA was divided in main logic blocks by functionality. Each block has different characteristics, and each must be protected in a different way. The goal is to compare diverse SEU mitigation techniques for SRAM based FPGAs and based on the results to design techniques that can improve the results obtained so far in the Virtex® family. The techniques will be compared using the radiation test techniques such as fault injection. Each technique is discussed and final results will be presented in the final document. The conclusions of the thesis proposal are addressed in the chapter 5, followed by the references.

2 State of the Art of SEU Tolerant FPGAs and Microprocessors

Significant research has been done in the last years in order to avoid radiation effects on integrated circuits. Some of these effects such as Total Ionization Dose (TID) effects and single event latch-up (SEL) can be reduced to acceptable levels using some particular CMOS technologies such as the Epitaxial bulk CMOS process. However Single Event Upsets (SEUs) represent radiation induced hazards, which are more difficult to avoid in the space applications especially in high-density sub-micron integrated circuits. The constant evolution of the semiconductor industry, in the effort of reducing transistor sizes and voltage supply, has turned the integrated circuits more sensitive to charged particles interactions. Particles that once were considered negligible are can now be responsible for transient effects on space and on Earth as well.

The SEU phenomenon in memories has been continuously studied and mitigation techniques have been proposed in the past years in order to avoid bit flips in the storage elements. A SEU immune circuit may be fulfilled through a variety of mitigation techniques based on redundancy. Redundancy is provided by extra components (hardware redundancy), by extra execution time (time redundancy), or by a combination of both. However due to the amazing evolution in design architecture where more and more complex circuits are integrated in a single device, the most appropriated SEU mitigation solution has become a challenge in order to combine fast turnaround time, low cost, high performance and high reliability. This chapter presents an overview of the radiation effects on digital circuits and subsequently it shows the state of the art of SEU mitigation techniques for FPGAs and microprocessors.

2.1 Radiation Effects Overview

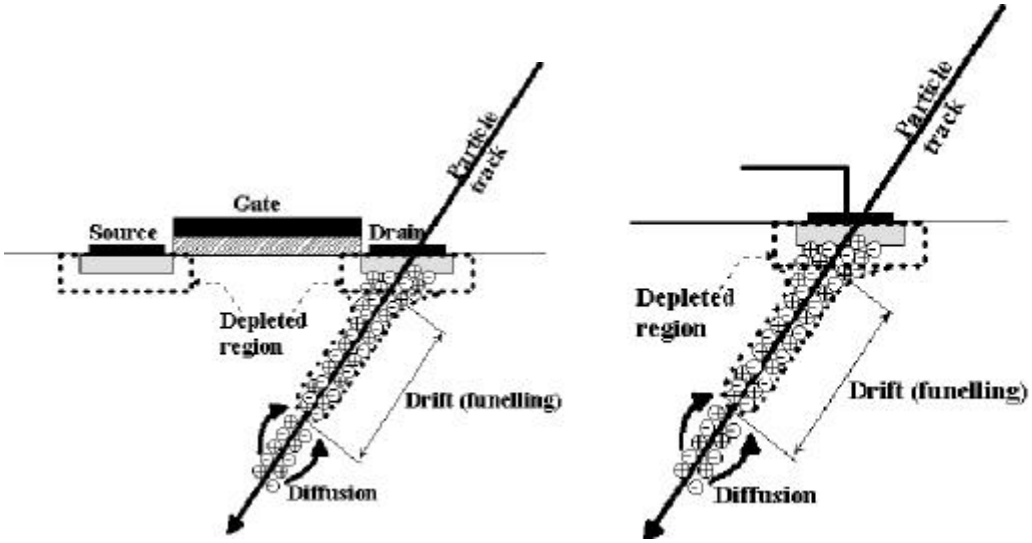
Integrated circuits (ICs) operating in space applications may be upset by different particles located in the radiation environment generated by the Sun activity [BAR97]. The particles can be classified into two major types: (1) energetic particles such as electrons, protons, neutrons, ?-particles and heavy ions, and (2) electromagnetic radiation (photons), which can be x-ray, gamma ray, or ultraviolet light. The photon particles have zero rest mass and are electrical neutral. They interact with target atoms producing energetic free electrons. The neutron particles interact with target atoms by nuclear reaction when the incident neutron is absorbed by the nucleus, which afterwards emits other particles (protons, ??particles, ??photons). The charged particles interact with the silicon atoms causing excitation and ionization of atomic electrons. The main sources of charged particles that contribute to radiation effects on space are trapped particles such as protons, electrons and heavy ions in the Van Allen belts, galactic cosmic ray such as protons and heavy ions and solar flares particles such as proton and heavy ions.

When a single heavy ion strikes the silicon, it loses its energy via the production of electron hole pairs resulting in a dense ionized track in the local region, as illustrated in figure 2.1. Proton and Neutrons can cause nuclear reaction when passing through the material. The recoil also produces ionization. The ionization generates a transient current pulse that can be interpreted as a signal in the circuit.

The influence of a particle in the material is measured by its energy and its flux. The flux is the number of particles passing during one second through one cm^2 of area [$1/s.cm^2$]. Integrating the flux over the time we get the fluence, which is [$1/cm^2$]. The flux of these sources is affected by the activity of the sun. The energy deposited by the charged particle is measured in rad (1 rad = 10^{-2} Js^{-1}) which corresponds roughly to the generation of $4x10^{13}$ electron-hole pairs in one cm^3 of silicon. The rate at which the ion looses energy is called stopping power (dE/dx). The incremental energy dE is usually measured in units of MeV while the material thickness is usually measured as a mass thickness in units of mg/cm^2. The energy transferred to the device is called Linear Energy Transfer (LET) and it is measured by the incremental energy per unit length ($MeV*cm^2/mg$). The minimum LET that can cause an SEU is called LET threshold (LET_{th}) [LAB99], [BRY98].

In other words, there is a minimum charge that must be deposited in the node in order to cause an upset. This minimum charge is called critical charge (Q_{crit}) and it is defined by Q_{crit} =

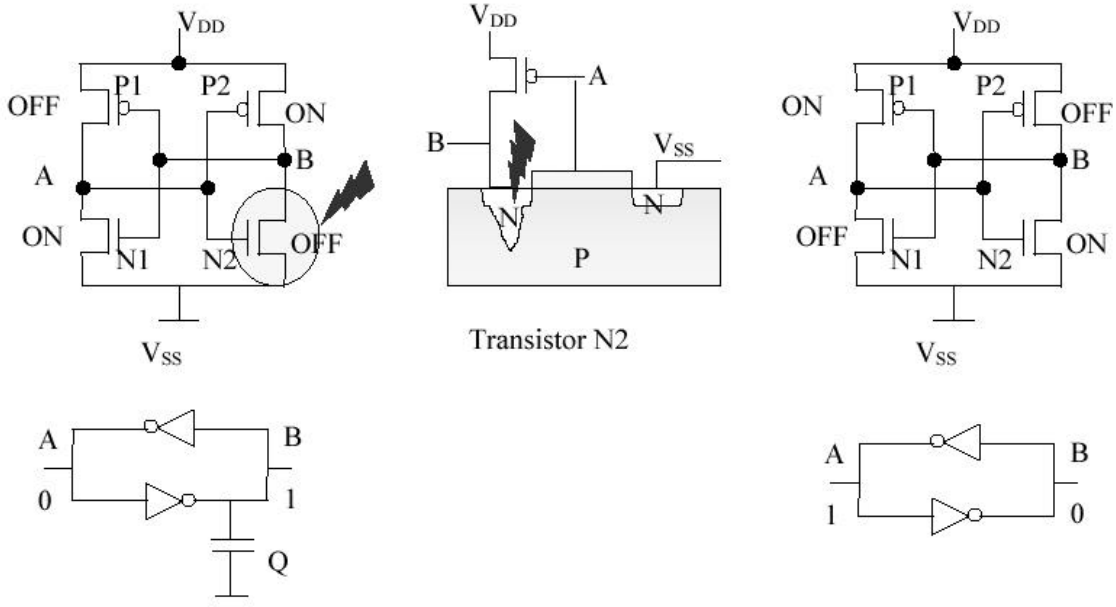
$C_{node}.V_{node}+I_{restore}.T_{flip}$. The critical charge must be bigger than the node collector charge (Q_{coll}) that is based on node parameters such as capacitance and voltage. As it was described previously, high-density devices require smaller feature size, this means less capacitance and hence information is stored with less charge (reduced Q_{crit}). Lower voltage or lower power devices means that less charge or current is required to store information. Each of these effects makes the device more vulnerable to radiation and means that particles with small charge, which were once negligible, are now much more likely to produce upset or damage.



(a) CMOS transistor (b) Capacitor

Figure 2.1 – Charge particle striking a silicon surface and generating a current pulse [BRY98]

A single particle can hit either the combination logic or the sequential logic. When a charged particle strikes one of the sensitive nodes of a memory cell (drains of transistor in off state), the stored value can be flipped. Memory cells have two stable states, one that represents a stored '0' and one that represents a stored '1.' In each state, two transistors are turned on and two are turned off (SEU target drains). A bit-flip in the memory element occurs when an energetic particle causes the state of the transistors in the circuit to reverse, figure 2.2.



(a) Before the particle hit (b) After the particle hit

Figure 2.2 – Single Event Upset (SEU) Effect in a Memory Cell [ANG00]

When a charged particle hits a logic cell sensitive node in the combinational logic, it also generates a transient pulse. This phenomenon is called single transient effect (SET). It may or may not be captured by a memory cell according to path timing, example figure 2.3. In [HAS98, HAS99] the probability of a SET becomes a SEU is discussed. The analysis of SET is very complexity in large circuits composed of many paths. Techniques as timing analysis [GUN00] could be applied.

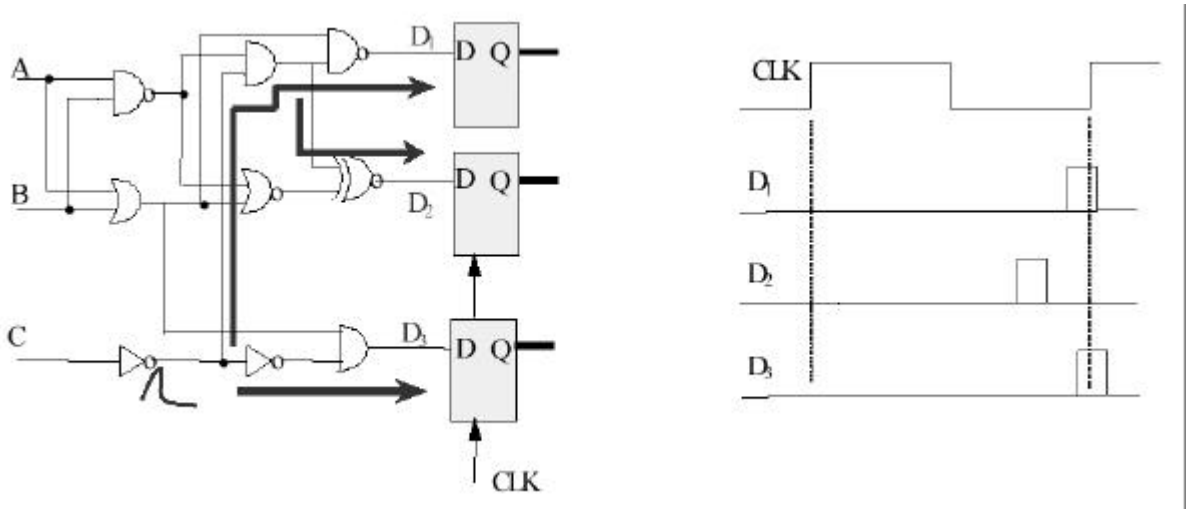


Figure 2.3 – Single Event Transient (SET) Effect in Combinational Logic [ANG00]

SEU can be classified in first, second and third order effects, according to the number of upsets that occur at the same time in the circuit. A single bit upset (SEU) is classified as first order effect, while multiple bit upsets (MBU) are classified as second or third order effect as explained later on in this section. MBU can occur when a single charged particle traveling through the IC at a shallow angle, nearly parallel the surface of the die, simultaneously strikes two sensitive junctions [ZOU89]. Just as SEU, direct ionization or nuclear recoil can induce MBUs, as it is presented in figure 2.4 [VAR01]. In [REE97], experiments in memories under proton and heavy ions fluxes have shown multiple upsets provoked by a single ion. MBUs were observed for all angles of incidence for LET greater than 25 MeV*cm^2/mg. There are three types MBU. The first one is when a single particle hits two adjacent nodes, located in two distinct memory cells. This event is classified as a second-order effect. This type of MBU can be avoided by specific placement.

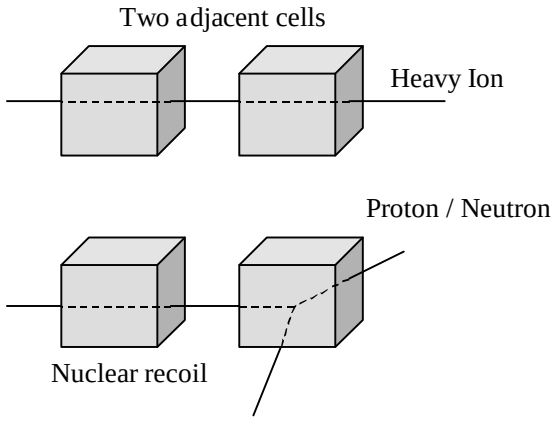


Figure 2.4 – MBU provoked by a single particle

The second type is when a single particle strikes two adjacent nodes, located in the same memory cell. This event is classified as a third-order effect. It can be avoided by layout constrains. In this case, two or more charged particles are responsible for multiple upsets. The probability of this occurrence is related to the placement of the memory cells. The cross section is proportional to the sensitive areas of the

junctions that is normal to the incident cosmic ray and to the solid angle subtended between the these sensitive areas. The probability of such a multiple node strike can be minimized in a circuit design by taking care in the physical layout so as to separate critical node junctions by large distances and to align such junctions so that the area of each, as viewed from the other, is minimized.

The third type of MBU is when multiple bits strike that silicon provoking upsets in multiple nodes. This event can be analyzed as a group of SEU and it will represent the same characteristics of immunity. Based on [REE97], the majority of multiple upsets located in adjacent cells are provoked by a single particle. There is a very low probability of more than one charged particle interacting in adjacent cells, provoking upsets in a period smaller than 1 s.

2.2 SEU Mitigation Techniques

The first SEU mitigation solution that has been used for many years in spacecraft is shielding. It reduces the particle flux but it does not eliminate completely it. One example of shielding is the aluminum that can retain electrons but is inefficient to retain cosmic rains with energies of MeV to GeV. Shielding is also inefficient for neutrons. Extra techniques must be applied to avoid SEU. The primary SEU mitigation techniques used nowadays are based on:

- *Process technology*: the epitaxial bulk CMOS process avoids completely latchup and it can increase the device cross-section. For some applications, the use of "epi" technology can propose enough reliability. Silicon on Insulator (SOI) technology is characterized by the placement of a thin layer of silicon on the top of an insulator during the chip manufacturing process. The transistors are then built on top of this thin layer reducing thus the capacitance [MUS01, COL01]. The main improvement achieved by the use of SOI is the chip performance. SOI technology improves performance over bulk CMOS technology by 25-35%, equivalent to two years of bulk CMOS improvements, exceeding Moore's law. SOI technology also has power consumption advantages of 1.7-3 times. The isolation of each transistor makes SOI technology latchup free. The results present in [COL01] show that SOI with body ties has reduced the error rate by a factor of 50 to 500. In resume, an appropriate use of SOI can increase the reliability of the device in presence of SEU but it does not mitigate completely the effect.
- *Hardened memory cells*: memory cells are the most SEU sensitive elements in the design. For this reason, a solution that can protect each memory cell against bit flips can be very attractive to improve reliability for space applications and military proposes. Different kind of circuits such as microprocessors, memories, ASICs, programmable circuits and others can be SEU protected by replacing each original memory cells for a hardened version. This mitigation technique requires basic three steps: the design of the SEU hardened memory cell according to the technology used in the device, the instantiation of the new memory cells in the design and the timing evaluation constraints. Examples of SEU hardened memory cells are presented in [WEA87, ROC92, VEL94, WIS93, CAN96, WHI91, LIU92, KAT01, CAR01, MAV00]. All of them are based on duplication or triplication using the restore feedback approach. One differs from each other in terms of number of transistors, performance and SEU immunity degrees. The most important characteristic is that almost all of them do not accumulate upsets, because by their design construction the bit flip is completely avoided. The main drawbacks are area overhead and performance penalty.
- *Error detection and correction techniques*: Error Detection and Correction (EDAC) codes [PET80] are used to protect digital data against errors that can occur in storage cells or transmission channels. The basic concept is to have an encoding and a decoding algorithm in order to restore the correct value. The encoding and decoding of data can be done either in hardware, software or a combination of both. There are several types of codes. Hamming code technique is a suitable solution for both circuit design and software level. Using hamming code as a SEU mitigation solution in the design of a circuit, extra logic blocks are needed to code and decode the

stored values such as registers and internal memory. This technique was implemented in a SEU hardened micro-controller [COT00, LIM00a, LIM00b]. Results presented in [LIM01a] showed its efficiency. The main drawback of EDAC techniques are area overhead, performance penalties according to the implementation and the necessity of refreshing in order to avoid accumulation of upsets.

- *Triple Modular Redundancy*: redundancy techniques such as duplication and triplication are commonly used for designing dependable systems to ensure high reliability and data integrity. Triple Modular Redundancy (TMR) [NEU56] uses three identical implementations of the same logic function and the outputs of all the implementations are connected to a majority voter. The voter is usually performed on a bit-by-bit basis [SIE92]. The voter is the single point of failure in the TMR design, but because it is combinational, the chance of an upset being captured by an internal latch or an output is very low. Additionally, the voter logic can be equipped with transistors that are sized large enough to have a high tolerance of environmental conditions. The main drawback is area overhead and power, which is very significant for space applications.

Each one of the techniques mentioned above has some advantages and drawback. The objective is to find the most cost efficient approach analyzing area overhead, performance penalties and SEU tolerance. This section summarizes the developed SEU mitigation techniques that have been used by space agencies, laboratories, universities and companies for the past few years in order to design SEU tolerant FPGAs and microprocessors.

2.2.1 State of the Art of SEU Tolerant FPGAs

Field Programmable Gate Arrays are becoming increasingly popular with spacecraft electronic designers as they fill a critical niche between discrete logic devices and the mask programmed gate arrays. The devices are inherently flexible to meet multiple requirements and offers significant cost and schedule advantages. Since FPGAs are re-programmable, data can be sent after launch to correct errors or improve the performance of spacecraft.

The architecture of a programmable device is based on an array of logic blocks that can be programmable by the interconnections to implement different designs. A FPGA logic block can be as simple as a small logic gate or as complex as clusters composed of many gates. Current commercial FPGAs' logic blocks are composed of one or more of transistor pairs, small gates, multiplexors, Lookup tables, and-or structures. The routing architecture incorporates wire segments of various lengths, which can be interconnected via electrically programmable switches. Several different programming technologies are used to implement the programmable switches. There are three types of such programmable switch technologies currently in use:

?? *SRAM*, where the programmable switch is a pass transistor controlled by the state of a SRAM bit (SRAM based FPGAs)

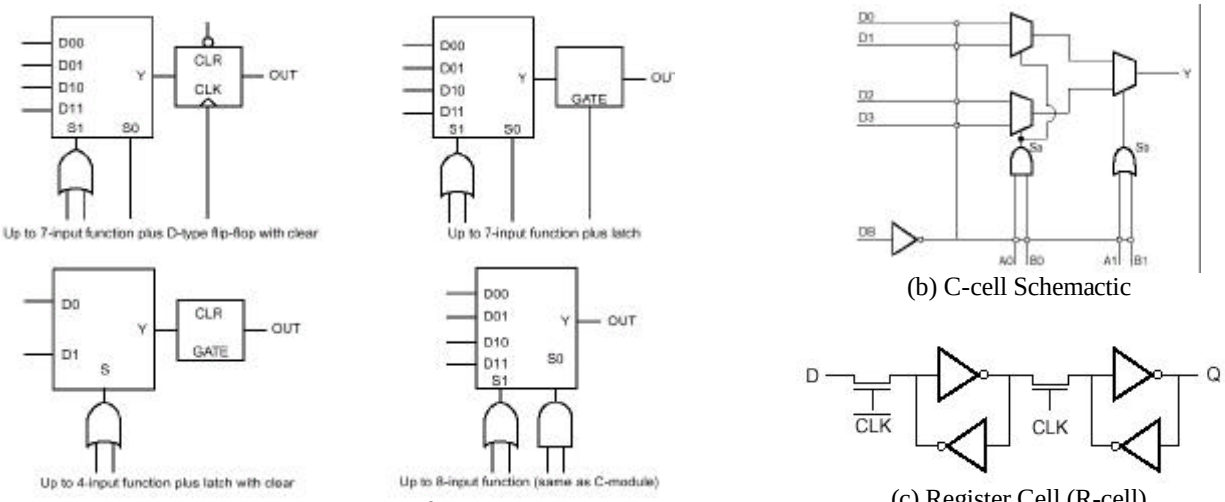
?? *Anti-fuse*, when an electrically programmable switch forms a low resistance path between two metal layers. (Anti-fuses based FPGAs)

?? *EPROM, EEPROM or FLASH cell*, where the switch is a floating gate transistor that can be turned off by injecting charge onto the floating gate. These programmable logic circuits are called EPLDs or EEPLDs.

Both customizations based on SRAM and anti-fuses are volatile. The EPROM and EEPROM customization are non-volatile. Each of them has a particular architecture. Programmable logic companies such as Xilinx and Actel offer radiation tolerant FPGA families. Each one uses different mitigation techniques to better take into account the architecture characteristics. Some companies from the space market are licensed to develop tolerant FPGAs from FPGA companies such as Aeroflex UTMC is licensed to QuickLogic, Honeywell is licensed to Atmel. However, there is no current space product based on the

QuickLogic and Atmel FPGAs. Actel and Xilinx are the mainly FPGA companies to share the market of space FPGAs nowadays as observed in the industry floor of the most important conferences of the area such as Military and Aerospace Applications of Programmable Devices and Technologies (MAPLD), Nuclear and Space Radiation Effect (NSREC) and Radiation Effects on Components and Systems (RADECS). In the next sections an overview of the SEU tolerant FPGA designs is presented and the SEU mitigation techniques for the Virtex® FPGA from Xilinx, the most popular SRAM-based FPGA in the market are discussed in detail.

Actel offers SEU tolerant FPGAs families programmed by anti-fuse called SX [ACT01]. This family architecture is described as a "sea-of-modules" architecture because the entire floor of the device is covered with a grid of logic modules with virtually no chip area lost to interconnect elements or routing. Actel's SX family has been improved in the past years. The first version provided two types of logic modules, identical to the standard Actel family, the register cell (R-cell) and the combinatorial cell (C-cell) exemplified in figure 2.5. Interconnection between these logic modules is achieved using Actel's patented metal-to-metal programmable anti-fuse interconnect elements, which are embedded between the M2 and M3 layers. The anti-fuses are normally open circuit and, when programmed, form a permanent low-impedance connection.



(a) Combinational Logic Blocks Configurations (C-cell)

(b) C-cell Schemactic

(c) Register Cell (R-cell)

Figure 2.5 – Actel Architecture

In this first SEU tolerant FPGA version [ACT00], three proposed techniques for implementing the logic of the sequential elements in order to avoid upsets were presented: CC, TMR, or TMR_CC. The sequential elements are automatically implemented during the synthesis in Symplify tool [SYM01]. The CC technique uses combinatorial cells with feedback instead of flip-flop or latch primitives to implement storage. For example, a DFP1, comprised of two combinational modules, would be used in place of a DF1. This technique can avoid SEU in CMOS technologies larger that 0.23um but it will not be able to avoid SEU in next process technologies where the combinational logic can also be affected by charged particles. TMR is a register implementation technique where each register is implemented by three flip-flops or latches that "vote" to determine the state of the register. TMR_CC is also a triple-module-redundancy technique. Each voting register is composed of combinatorial cells with feedback (instead of flip-flop or latch primitives).

The CC flip-flops (CC-FFs) produce designs that are more resistant to SEU effects than designs that use the standard flip-flop (S-FF). CC-FFs typically use twice the area resources of S-FFs. Triple voting, or triple module redundancy (TMR), produces designs that are most resistant to SEU effects. Instead of a single flip-flop, triple voting uses three flip-flops leading to a majority gate voting circuit. This way, if one

flip-flop is flipped to the wrong state, the other two override it, and the correct value is propagated to the rest of the circuit. Because of the cost (three to four times the area and two times the delay required for S-FF implementations), triple voting is usually implemented using S-FFs. However, one can implement triple voting using only CC-FFs in Synplicity tool.

Actel introduced in 2001 a new version of the space FPGA family SX, composed of special radiation-tolerant flip-flops. These SEU-hardened structures eliminate the need for TMR flip-flop designs implemented in HDL. The family also supports a wider range of I/O voltage standards such LVTTL, TTL, PCI and CMOS.

Figure 2.6 shows an implementation of a D-type flip-flop using TMR [KAT01]. Three D-type flip-flops are connected in parallel to the clock and data inputs. A voter (or majority circuit) is implemented by the top MUX to create a "hardened" output. The outputs of two flip-flops, A and B, go to the selects of the voter MUX. If both A and B read logic zero, MUX input D0 is selected. Since it is tied to GND, the output of the MUX will read logic zero. Similarly, if A and B read logic one, the output of the MUX will read logic one. If A and B disagree due to an SEU (or for other reasons), the MUX will select flip-flop C. We know C agrees with either A or B, and thus the MUX "voted" to produce data agreed on by two of the three flip-flops.

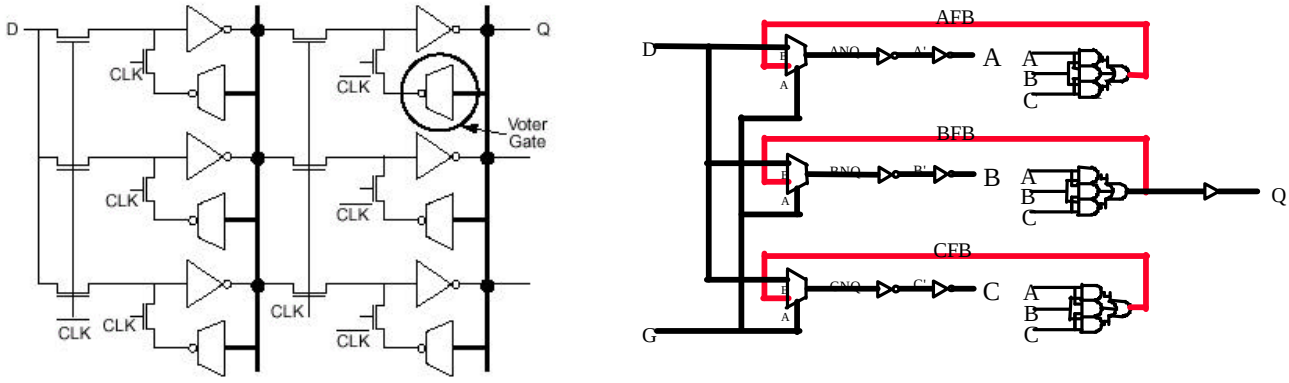
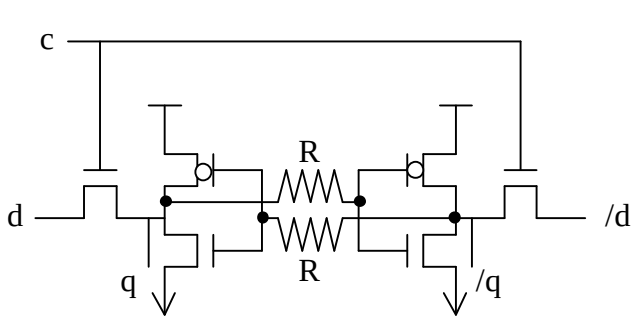
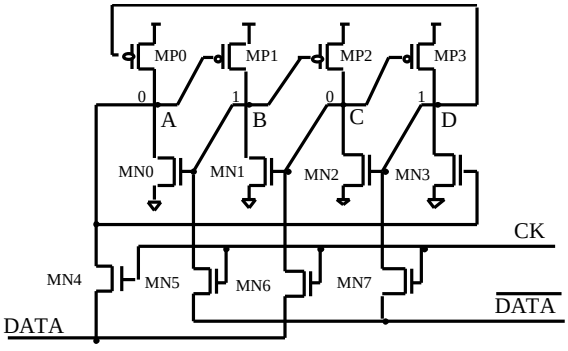


Figure 2.6 – SEU hardened memory cell used by Actel SX family [KAT01]

Actel has also prototyped an SRAM-based FPGA [KAP99]. In this case, the standard SRAM memory cells were replaced by resistor-decoupling memory cells where the effectiveness depends on the resistor value; and DICE memory cells that are practically SEU immune at 0.25um if only one node is hit. Figure 2.7 exemplifies the resistor decoupling memory cell and the DICE cell, respectively. The resistor decoupling memory cell is able to avoid upsets because the resistors inserted in the feedback path work as filters to the transient pulse provoked by the charged particle. The DICE cell can avoid upsets because it stores the data in two distinct parts, where if one part is corrupted the other one is isolated by the cell construction.



(a) Resistor decoupling memory cell



(b) DICE memory cell

Figure 2.7 – SEU hardened memory cells used by the Actel SRAM-based FPGA

Conclusions presented in [KAP99] show that MBU will limit both solutions in the future if the layout does not pay special attention to this issue. The redundancy hardening (DICE memory) is less effective than the resistor solution in two orders of upset rates. Disadvantage of resistor solution is temperature operation range sensitivity and the increase in delay. The DICE also has a disadvantage in area overhead. It has 12 transistors compared to 6 transistors in the standard memory cell. For 0.18 um, the effectiveness of both solutions will be compromised and more ingenious designs will be needed in the circuit level.

Atmel [ATM01] also has published the version of an SRAM-based FPGA (AT6010) using the SOI process. The logic block presented in figure 2.8 was not logically modified. The improvement achieved is limited to the SOI reliability in presence of SEU.

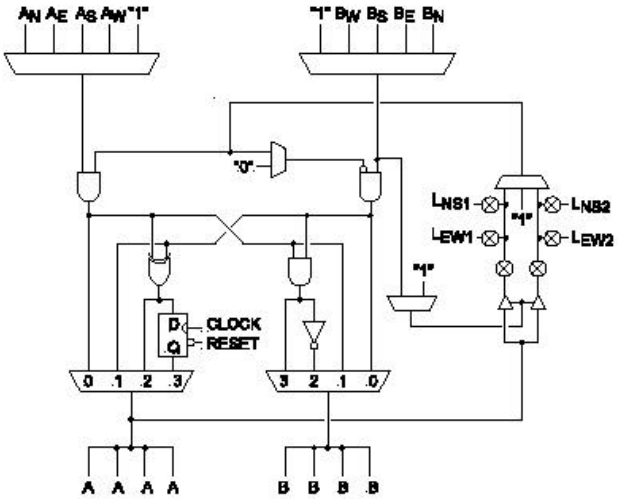
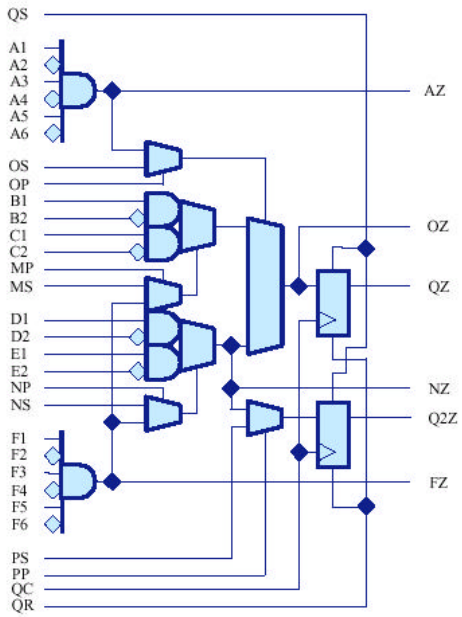
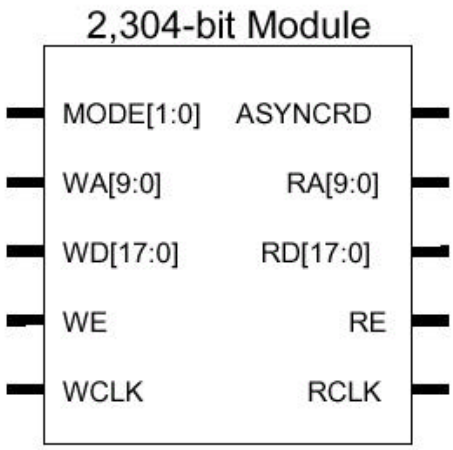


Figure 2.8 – Atmel FPGA logic block

The QuickLogic [QUI01] offers a FPGA fabricated 0.35um epitaxial CMOS process. The SEU sensitive cells are memory cell located in the CLB and the embedded memory. The combinational cells such as multiplexors and the logic cells and the programmable cells (vialink anti-fuse) are not SEU sensitive parts. This kind of FPGA can be protected against SEU by replacing the memory cell by a SEU hardened cell (mask change) or by replacing the memory cell by a TMR memory cell with voter. Figure 2.9 shows the CLB schematic composed of multiplexors, logic gates and a single register cell and the embedded memory cell. But no version with TMR flip-flops or other SEU hardened memory cells are presented in the literature.



(a) CLB Schematic



(b) Embedded Memory Block

Figure 2.9 – QuickLogic FPGA logic block

In order to mitigate Single Event Upsets (SEU) in SRAM-based Virtex® FPGAs [XIL00a], the Triple Modular Redundancy (TMR) with voting technique combining with bitstream scrubbing must be applied [CAR01a]. The TMR mitigation scheme uses three identical logic circuits performing the same task in tandem with corresponding outputs being compared through a majority vote circuit. If an upset occurs in the customized logic, TMR votes out the error. If the upset occurs in the device configuration, TMR votes out the logic leg no longer functioning and reconfiguration (scrubbing) repairs the error before more errors accumulate and overcome the TMR. Bitstream reconfiguration, complete or partial, may occur without interruption of service in the Virtex® device.

The correct implementation of TMR circuitry within the Virtex® architecture depends on the type of data structure to be mitigated. The logic may be grouped into four different structure types: Throughput Logic, State-machine Logic, I/O Logic, and Special Features (Select block RAM, DLLs, etc.). The TMR technique for Virtex® is presented in details in [CAR01a].

The throughput logic is a logic module of any size or functionality, synchronous or asynchronous, where all of the logic paths flow from the inputs to the outputs of the module without ever forming a logic loop. In this case, it is necessary to just triplicate the logic, creating three redundant logic parts (0, 1 and 2). No voters are required, as the FPGA output will be by default voted later.

The state-machine logic is any structure where a registered output, at any register stage within the module, is fed back into any prior stage within the module, forming a registered logic loop. This structure is used in accumulators, counters, or any custom state-machine or state-sequencer where the given state of the internal registers is dependent on its own previous state. In this case, it is necessary to triplicate the logic and to have majority voters in the outputs. The register can not be locked in a wrong value, for this reason there is a voter for each redundant logic part in the feedback path making the system be able to recover by itself. Figure 2.10 shows a general example of this structure.

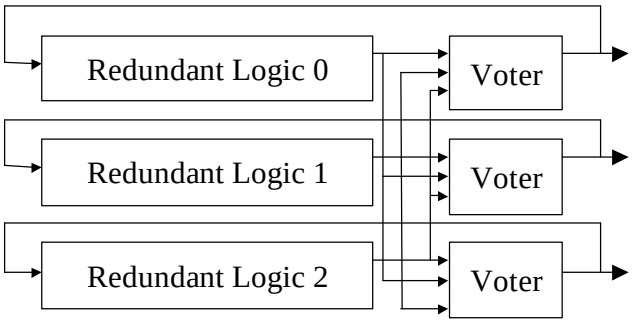
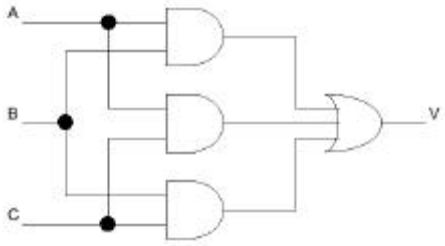
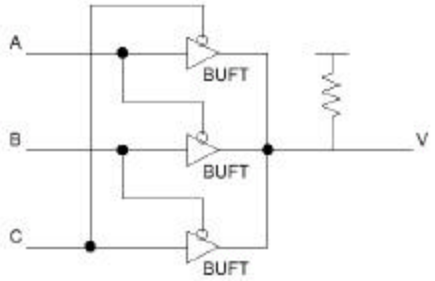


Figure 2.10 - TMR Logic with Voter

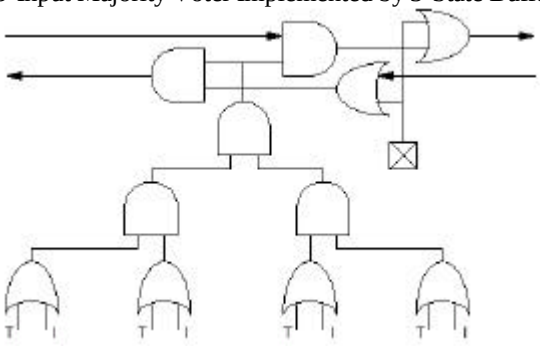
The majority voter, figure 2.11 (a), can be easily implemented by one LUT. For designs constrained by available logic resources, the majority voter can be implemented using the Virtex® 3-state buffers instead of LUTs, Figure 2.11 (b). There are two 3-state buffers per CLB. Figure 2.11 (c) shows the 3-state buffer schematic in the Virtex® matrix.



(a) 3-input Majority Voter Schematic



(b) 3-Input Majority Voter Implemented by 3-State Buffers



(c) Virtex® Bus Logic

Figure 2.11 - Majority Voters [CAR01a]

The primary purpose of using a TMR design methodology is to remove all single points of failure from the design. This begins with the FPGA inputs. If a single input was connected to all three redundant logic legs within the FPGA then a failure at that input would cause these errors to propagate through all the redundancies and thus the error would not be mitigated. Therefore, each redundant leg of the design that uses FPGA inputs should have its own set of inputs. Thus, if one of the inputs suffers a failure, it will only affect one of the redundant logic parts. The outputs are the key to the overall TMR strategy. Since the full

triple module redundancy generates every logic path in triplicate, there must ultimately be a method for bringing these triple logic paths back to a single path that does not create a single point of failure. This can be accomplished with TMR outputs minority voters inside the output logic block, as presented in figure 2.12.

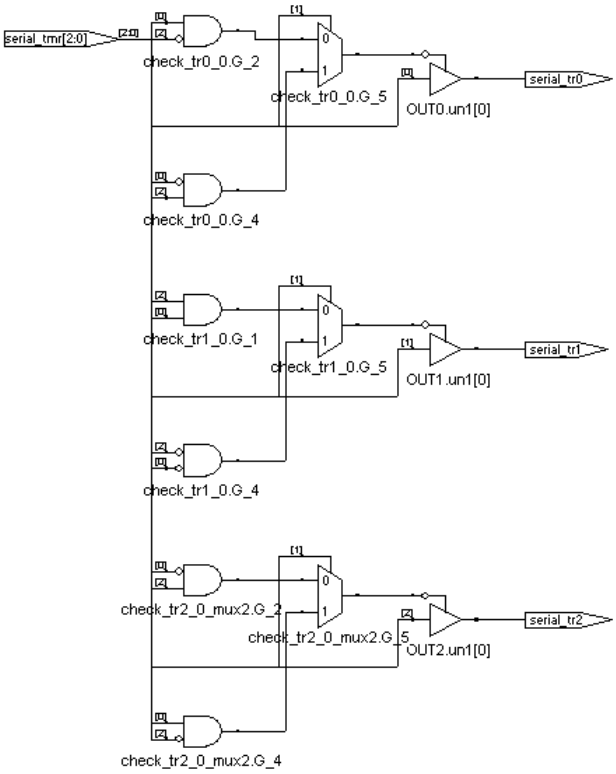


Figure 2.12 - Minority Voter in the Virtex® Output Logic [CAR01a]

The Virtex® architecture provides a number of special features, such as block RAM (BRAM), DLLs, etc, which require specialized methods for implementing effective redundancy. A reliable method to TMR the BRAM is to constantly refresh the BRAM contents, figure 2.13. Since these are dual port memories, one of the ports could be dedicated to error detection and correction. But this also means that the BRAM could only be used as single port memories by the rest of the user logic. To refresh the memory contents, a counter may be used to cycle through the memory addresses incrementing the address once every n clock cycles. The data content of each address is voted at a determined frequency and the majority voter value written back into the cells.

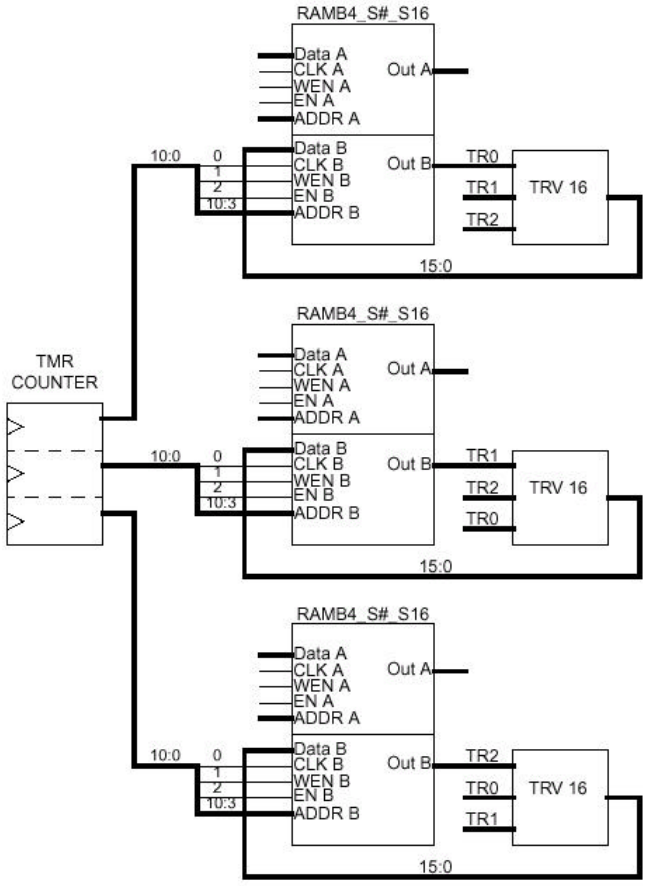


Figure 2.13 - BRAM TMR with Refreshing [CAR01a]

A typical FPGA design will be implemented with signals that were resolved to a logic constant (VCC or GND), but could not be entirely optimized out of the design. When the Place and Route (PAR) tools implement the VCC and GND signals, they are implemented in a way that maximizes device resource utilization. This is accomplished by utilizing "Keeper" circuits that exist at the input pins of all CLBs and I/O blocks (IOBs). Keepers lie in series with a routing channels and logic block input pins. When the routing channel carries an active signal, the keeper is transparent. But when the channel is unused, the keeper will keep its last known value - which was determined when the device was initially powered-up or re-initialized by activating the FPGA input PROG. When a logic element (i.e. flip-flop) inside a logic block (i.e. CLB or IOB) requires a logical constant, such as a VCC or GND, this logical constant may be obtained from the keeper circuit of an unused pin of the logic block. Its polarity may be selected by programmable inversion within the logic block. An SEU may upset, or alter, the state of a keeper circuit either by direct ionization, or indirectly by momentarily connecting an active routing channel to the input of the keeper. In either case, the result is a functional disturbance that cannot be detected by readback nor corrected by partial reconfiguration. Therefore, this type of error is known as a "persistent error". And it can only be corrected by completely re-initializing the FPGA. Schematic designers should be careful to examine the primitive implementation of all library macros that are likely to contain registers, before using them in their design. Even if the macro provides clock enable and reset pins at the top level, the primitive implementation may be different than expected. Similarly, if a VHDL user describes a synchronous process without specifying a clock-enable or initialization function, the synthesis tool will implement this function by using primitives and connecting all unused pins to the correct logical constant, thus creating VCC and GND. In order to avoid persistent errors, user's VCCs, user's GNDs and user clock enables for each redundant logic part must be created in the design as inputs.

A simpler method to SEU correction is to omit readback and detection of SEUs and simply reload the entire CLB Frame segment at a chosen interval. This is called "scrubbing." Scrubbing requires substantially fewer overheads in the system, but does mean that the configuration logic is likely to be in "write mode" for a greater percentage of time. However, the cycle time for a complete scrub can be made relatively short. The SelectMAP interface is capable of operating at a throughput of 400 Mbits/s. Additionally, the chosen interval for scrub cycles should be based on the expected static upset rate for a given application or mission, and may be fairly infrequent. A longer cycle interval (time between scrubs) and shorter cycle time (scrub time) decreases the total percentage of time that the configuration logic is in "write mode."

2.2.2 State of the Art of SEU Tolerant Microprocessors

Many commercial microprocessors from Intel, IBM, Motorola and Sun are available in the market in a radiation tolerant version. These microprocessors are designed and protected by space project companies and research laboratories. Each product offers different levels of radiation immunity for distinct space and military applications. The techniques used to protect the microprocessors are usually based on the process technology or package shielding, TMR, SEU hardened memory cells, EDAC (Hamming code) or a combination of them.

Maxwell [MAX01] has a large range of SEU tolerant microprocessors protected by a patented radiation hardened RAD-PAK® technology that basically is a package shielding. The company offer microprocessors such as Intel 386, 486 and Pentium and SPARC from Sun. This same company also provides the microprocessor PowerPC from Motorola with the CPU protected by TMR and the memory protected by EDAC. The TMR compares the output of each of 3 CPUs on a bit-by-bit basis. In the event of a single upset a simple voting scheme detects and selects the correct value. The advent of a second error would be uncorrectable thus the processor is flushed and synchronized. In addition the components also have the package shielding.

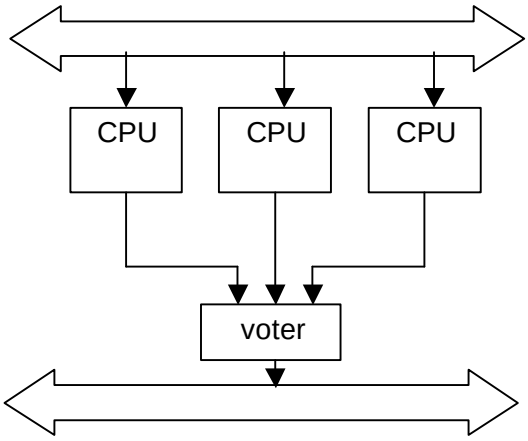


Figure 2.14 – TMR Processor Core with Voter

Honeywell [HON01] also offers SEU tolerant microprocessors based on device redundancy and EDAC techniques. An example is the radiation hardened PowerPC 603 where the data and program memories are protected by SEC-DED Hamming code and redundancy is applied in the internal registers. Aitech Defense Systems Inc. [AIT01] also provides a radiation tolerant of the PowerPC 750 protected by EDAC.

A SEU tolerant PowerPC (G3) has been developed by Lockheed Martin for JPL [JPL01] under the 2000 program. It provides a modular standard product that allows the spacecraft developer excellent flexibility in system configuration. There are over 800,000 storage elements in the PowerPC 750 (G3), all

of which have been replaced with SEU hardened circuitry in the RAD750. The earlier RAD6000 employed resistor decupling memory cells (figure 2.7(a)), requiring a special polysilicon resistor in the manufacturing process. The RAM cells and latches in the RAD750 have been designed using circuit hardening techniques that require no special process steps and optimize performance using the cells referred in [LIU92]. Figure 2.15 exemplifies this memory cell. The RAD750 is expected to achieve SEU hardness levels of 1E-11 upsets/bit-day. The memory and PROM located on the board has been protected by EDAC.

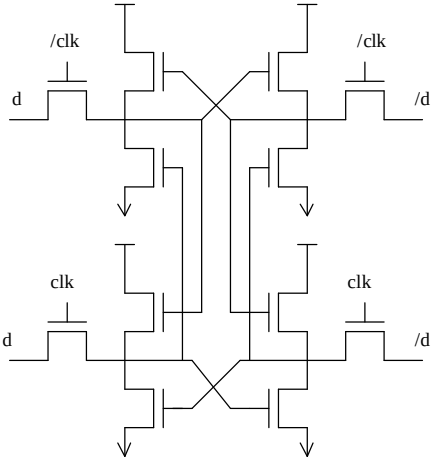


Figure 2.15 – NASA Memory cell [LIU92]

In [GAI98], the Spacelite processor based on the SPARC V8 architecture is presented. The techniques applied to this processor aim to detect and to tolerate one error in any on-chip register, and a double error in two adjacent bits in any on-chip memory structure (caches and tags). The approach to SEU fault-tolerance in the Spacelite processor is to divide all registers into two groups; primary and redundant. A primary register is defined as a register carrying information which is not present any-where else in the system (processor or memory) and where an error in the register contents would cause a malfunction of the system. A redundant register is defined as a register which contains information that is replicated somewhere else in the system, and can be recreated by either reloading the register or performing other recovery actions. An error in a redundant register must also not alter the state or operation of the system in a way that will create a malfunction during the time it contains an erroneous value. To tolerate one random register error, all primary registers are designed fault-tolerant, either by replication or by use of error-correcting codes. The redundant registers need only to be provided with error-detection functions, since they can be recovered from their redundant locations.

Individual fault-tolerant registers are implemented using TMR, for example. three registers in parallel and a voter selecting the majority result. The benefit of such a scheme is that error masking and error-removal is implicit, and than no glitch is produced on the output when an SEU occurs. The register file is provided with a 32-bit single error correction (SEC) and double error detection (DED) EDAC instead of TMR cells to reduce the overhead. Errors in redundant registers are detected through parity generation and checking. Cache memories and tags are protected with two parity bits, one for odd and one for even data bits. This scheme makes it possible to detect a double-error in two adjacent bits. In case of an EDAC error, the corrected register value is written back to the register file when the instruction reaches the write stage, and the instruction is then restarted. An error in the cache memory (instruction or data) will automatically cause a cache miss, and the cache will be updated with the correct data from the main memory.

Atmel provides an 8-bit radiation tolerant micro-controller 80C32E, DSP microprocessor and a SPARC microprocessor for military and space applications [ATM01]. The radiation tolerant DSP microprocessor Radiation from Atmel uses the Hit cell [VEL94] in order to protect the memory cells against radiation. The cell is exemplified in figure 2.16.

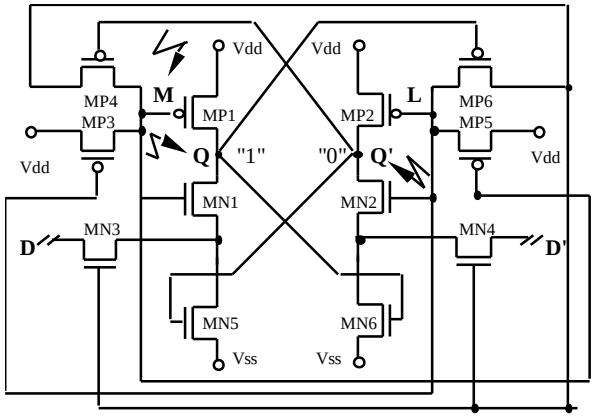


Figure 2.16 – HIT memory cell [VEL94]

The Atmel SPARC microprocessor is protected by EDAC. The Atmel static RAM design separates the cells that represent the different data word bits. This feature virtually eliminates the risk for one impact to provoke dual bit upsets (MBU), leaving only a minute risk for single bit upsets (SEU) that can be corrected by SEC hamming code. The additional processing associated with an EDAC protected solution is the initialization of the check bit RAM and a refresh procedure that performs read-write operations on the protected memory ("scrubbing"). The initialization of the check bit RAM does not introduce an overhead since most space borne applications move their code from ROM to RAM at reset, and automatically initialize the check bit RAM at the same time. The scrubbing performed during processor idle time is necessary to eliminate the risk for two separate impacts to generate a dual bit upset (MBU) in one same data word. However, if a dual bit upset in one same data word should occur it would still be detected and signaled by the EDAC, SEC-DED hamming code. The EDAC implementation uses the "correct always" solution. The "Bus-Watch" system technique is suitable for very fast systems, but implies more overhead in the error handling hardware and software. With respect to the processor speeds used in space borne systems, the propagation delay of flow-through EDAC is fast enough and therefore the "correct-always" solution has been used.

The MSC8051 [INT98] from Intel is an example of a large used microcontroller. A VHDL description of this microcontroller was designed at UFRGS [CAR96, SIL97] and it was re-used to insert SEU radiation fault tolerant structures. The original code is entirely compatible with the INTEL 8051 microprocessor in terms of instruction timing. It contains 24 instructions that are executed in 12 or 24 clock periods. Although the insertion of new instructions is quite easy, only the instructions required by the target application are being used in this description.

The microprocessor description is divided into six main blocks, illustrated in figure 2.17. These units are finite state machine, control unit, instruction unit, datapath and RAM and ROM memories. The Finite State Machine (FSM) block generates the states and number of cycles for each instruction to guide the circuit operation. It has a very simple combinational logic and a set of flip-flops. The control unit generates some control signals for the datapath, and it is basically composed of multiplexors. The instruction unit generates the microcode word for each instruction, and it is also basically composed of multiplexors. The datapath includes an Arithmetic Logic Unit (ALU) and 13 registers (Alu input registers, Alu output register, program counter I, program counter II, stack pointer, accumulator, instruction register, RAM inbus register, RAM output register, RAM address registers, ROM output register).

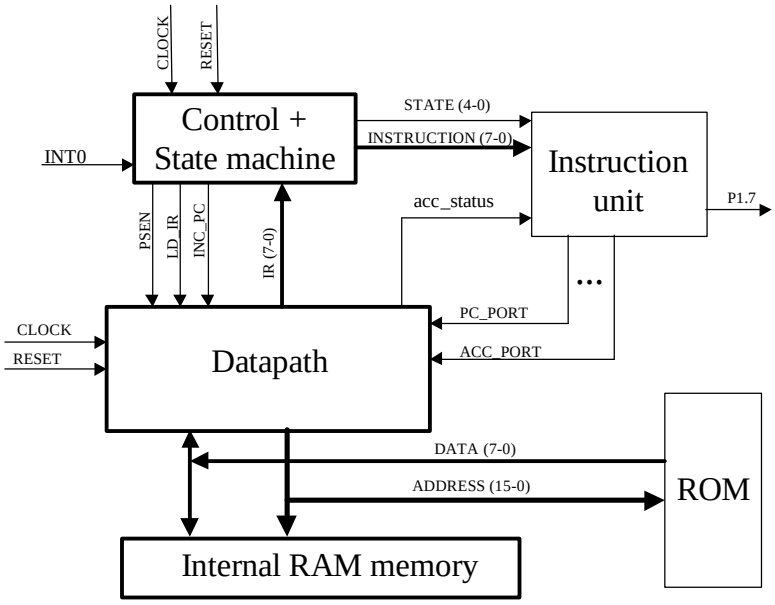
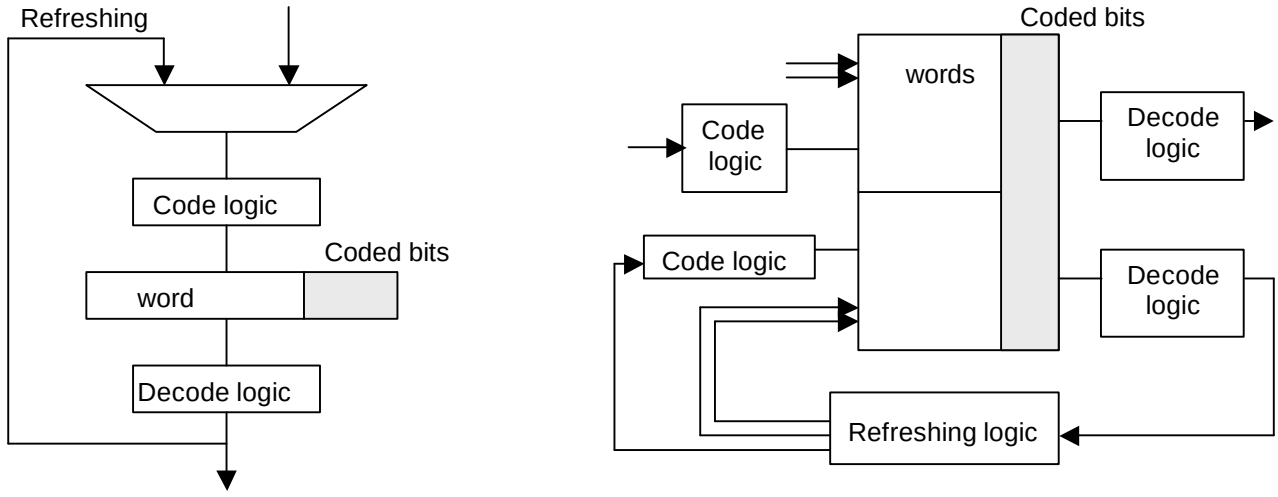


Figure 2.17 – General scheme of the SEU hardened 8051

In [COT00, LIM00] an 8051-like microcontroller protected by single error correction (SEC) Hamming code was presented. The hamming code was implemented in all registers and internal memory as represented in figure 2.18. This technique is innovative because it uses EDAC not only in the memory but also in all registers and single memory cells. This technique has also scrubbing in the memory and like the Atmel EDAC implementation, it corrects always. The efficiency of the SEC hamming code was tested by fault injection [LIM01a]. The results show that no errors were found in the application in presence of SEU. However this technique is not suitable for MBU. In [LIM02a], MBU were injected in the SEU tolerant 8051. The necessity of DEC hamming code and register refreshing in addition of the memory refreshing may be evident in features process technologies.



(a) Hamming Code in Register

(b) Hamming Code in Memory

Figure 2.18 – Hamming code in the 8051-like Micro-controller

3 A Case Study: Analyzing the Xilinx Virtex® FPGAs in Presence of SEU

The Virtex® family [XIL00a] is a high performance SRAM-based FPGA that supports a wide range of configurable gates, from 50k to more than 1M. The Virtex® QPRO family [XIL00b] provides a commercial off-the-shelf system-level solution for aerospace and defense customers. It is fabricated on thin-epitaxial silicon wafers using the commercial mask set and the Xilinx 5-layer-metal 0.22 µm CMOS process. The use of epitaxial CMOS process technology has made Virtex® Single Event latchup immune (LET_{th} >120 MeV*cm^2/mg, TID=100 Krads(si)).

3.1 Virtex® Architecture Overview

Virtex® devices consist of a flexible and regular architecture composed of an array of configurable logic blocks (CLBs) surrounded by programmable input/output blocks (IOBs), all interconnected by a hierarchy of fast and versatile routing resources, fig. 3.1. The CLBs provide the functional elements for constructing logic while the IOBs provide the interface between the package pins and the CLBs. The CLBs are interconnected through a general routing matrix (GRM) that comprises an array of routing switches located at the intersections of horizontal and vertical routing channels. The Virtex® matrix also has dedicated memory blocks called Select block RAMs (BRAMs) of 4096 bits each, clock DLLs for clock-distribution delay compensation and clock domain control, and two 3-State buffers (BUFTs) associated with each CLB.

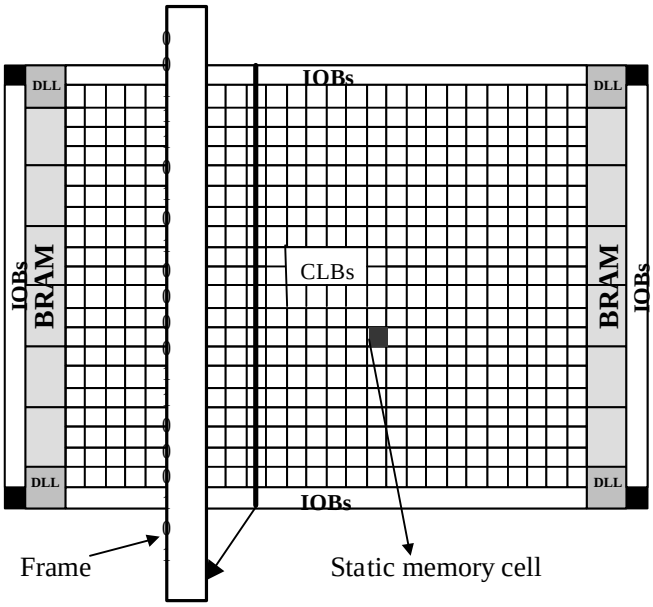


Figure 3.1 - Virtex® Architecture Overview

Each CLB is a complex structure composed of LUTs, flip-flops and routing resources, as it is simplified in fig. 3.2. The block is classified in 4 modules:

- ?? Module A: defines the logic part, responsible to implement the combinational logic and sequential logic described by the user.
- ?? Module B: defines the routing resources.
- ?? Module C: defines the input multiplexors.
- ?? Module D: defines the output multiplexors.

The module A is located inside the CLB, there are two slices of logic (0 and 1), each slice has two 4-input LUT, 2 flip-flops and a set of multiplexors, fig. 3.3. Each CLB slice can implement any two of all 4-input logic functions or some functions up to 9 inputs. In addition to operating as a function generator,

each LUT can provide a 16 x 1-bit synchronous RAM. Furthermore, the two LUTs within a slice can be combined to create a 16 x 2-bit or 32 x 1-bit synchronous RAM, a 16x1-bit dual-port synchronous RAM, or also a 6-bit shift register. The storage elements in the Virtex® slice can be configured either as edge-triggered D-type flip-flops or as level-sensitive latches. The D inputs can be driven either by the function generators within the slice or directly from slice inputs, bypassing the function generators. In addition to Clock and Clock Enable signals, each Slice has synchronous set and reset signals (SR and BY). SR forces a storage element into the initialization state specified for it in the configuration. Signal BY forces it into the opposite state. Alternatively, these signals may be configured to operate asynchronously. All of the control signals are independently invertible, and are shared by the two flip-flops within the slice. Each Virtex® CLB contains two 3-state drivers (BUFTs) that can drive on-chip busses. Each Virtex® BUFT has an independent 3-state control pin and an independent input pin.

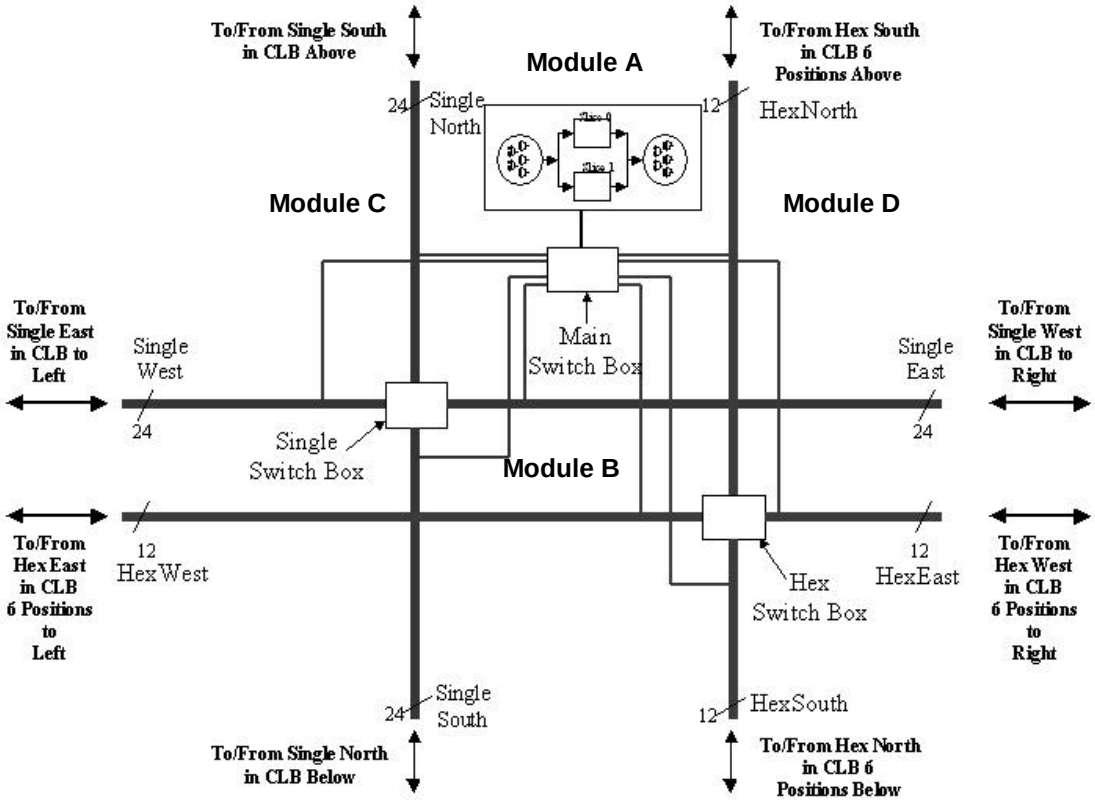


Figure 3.2 – CLB overview

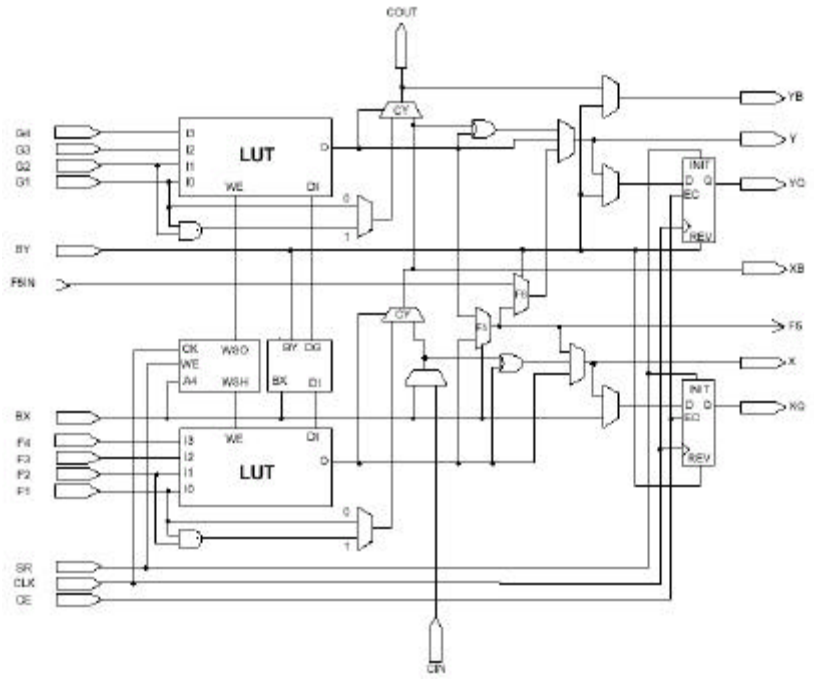


Figure 3.3 – Slice overview

The module B is composed of 3 types of routing resources: Hex lines, Single Lines and Long Lines. The hex lines can connect up to 3 CLBs given an interval of 2 CLBs between each, as shown in fig 3.4. The CLBs marked with a circle indicate that they are connected by the same hex lines. Each CLB has sets of 12 hex lines running in North, South, East and West directions. If the hex line connects to a CLB in the same row or in the same column, it is called HEX_HORIZ or HEX_VERT, respectively. Some hex wires can only drive data into the CLB; these are called unidirectional in. Some hex wires can only drive data out of the CLB; these are called unidirectional out. Some hex wires can drive signal in and out of the CLB; those are called bi-directional.

Figure 3.5 shows the switch boxes and the possibilities of connections between hex lines and single lines. Single lines can just connect 1 CLB. In each CLB, there are sets of 24 single wires in each North, South, East and West directions. Long lines run the length of the chip. There are accesses to 2 vertical and 2 horizontal in each CLB. They connect to other CLBs every 6 CLBs away. There is a twist in them, which changes their name. So for example, if you connect to LongVert[0] in (row, col) you can access the signal from LongVert[1] in (row+6, col), LongVert[0] in (row+12, col), LongVert[1] in (row+18, col), etc. Routing works in a hierarchical manner. Long lines can drive hex lines only; hex lines can drive hex lines and single lines. Also single lines can drive single lines and vertical long lines.

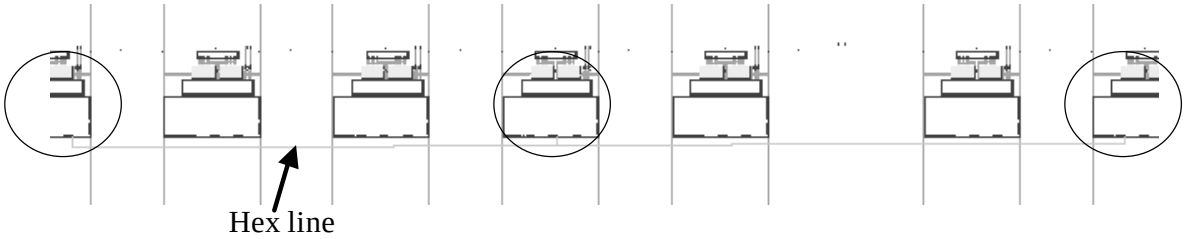


Figure 3.4 – Hex line example in the floorplanning

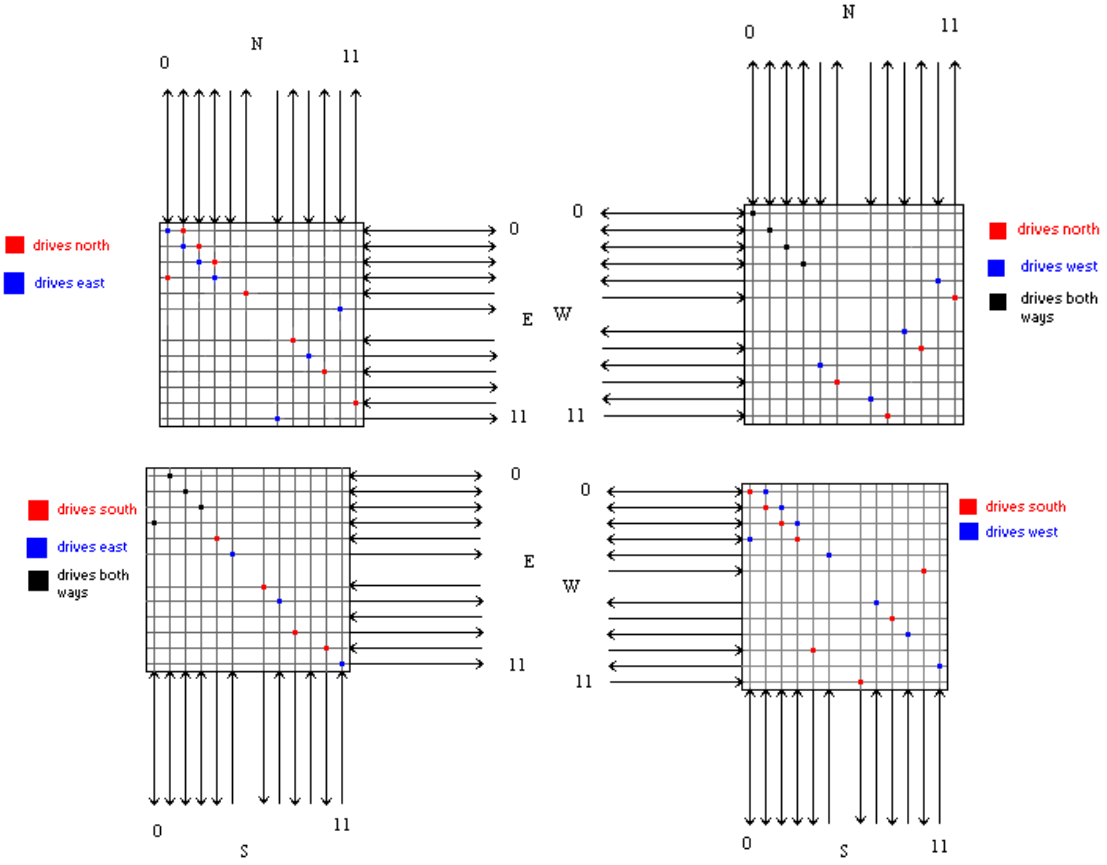


Figure 3.5 – Hex Switch Box

The module C is composed of the input and output multiplexors. There are 13 inputs multiplexors per slice, which includes the F1-F4, G1-G4, CLK, SR, etc. Each input has a multiplexor associated with it that determines which wires drive the inputs, fig. 3.6. There are 8 output multiplexors per CLB. Each output multiplexor can select various slice outputs and drive those signals to the general routing, fig. 3.7.

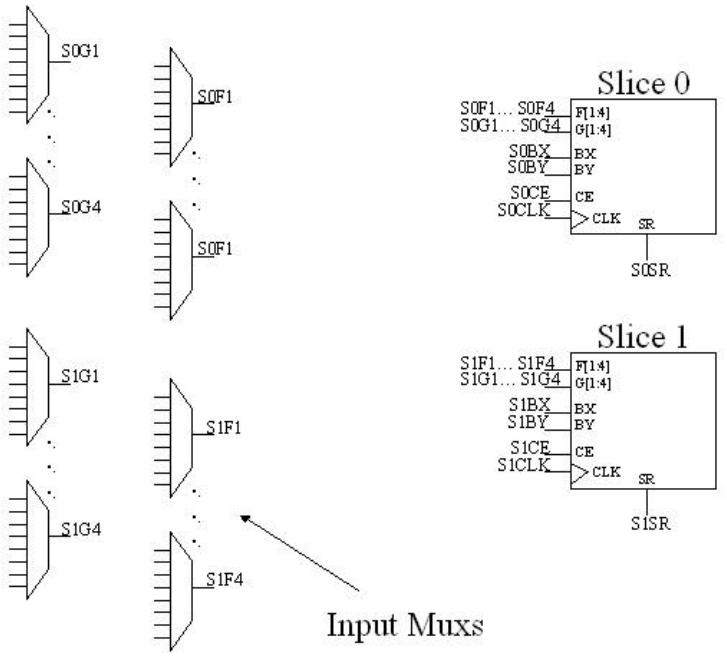


Figure 3.6 – Input Multiplexors

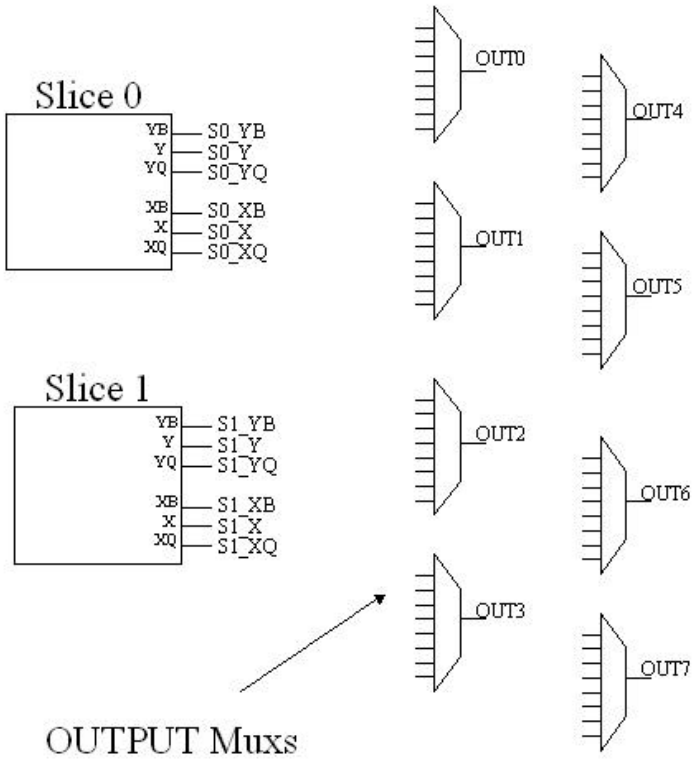


Figure 3.7 – Output Multiplexors

Virtex® family has several large Select block RAM (BRAM) memories, figure 3.8. Each embedded memory can be programmed with up to 4098 bits and a single or dual port mode. These blocks complement the distributed LUT RAMs that provide shallow RAM structures implemented in CLBs. BRAM memories are organized in columns. All Virtex® devices contain two such columns, one along each vertical edge. These columns extend the full height of the chip. Each memory block is four CLBs high, and consequently, a Virtex® device with 64 CLBs high contains 16 memory blocks per column, and a total of 32 blocks. In the Virtex-E there are four BRAM columns in the matrix.

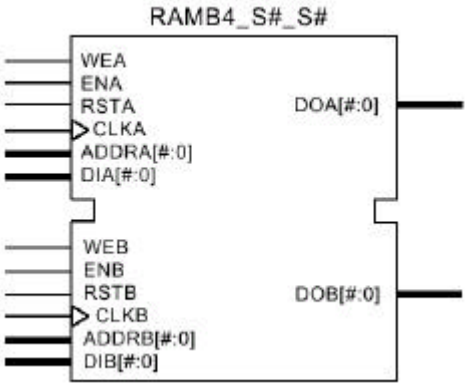


Figure 3.8 – BRAM schematic (single-port or dual-port)

The lately FPGA generation released by Xilinx is called Virtex®2 [XIL01a]. The CLB has the double of logic compared to the Virtex®, this means that there are 4 slices of logic instead of 2, figure 3.9. The LUT has been changed; each memory cell now is reduced to single memory just like the routing. In addition, the logic to access as memory or shift register has been a little modified. Now 128-bit shift registers with addressable access to be implemented in one CLB. Each CLB has 3 tri-state buffers.

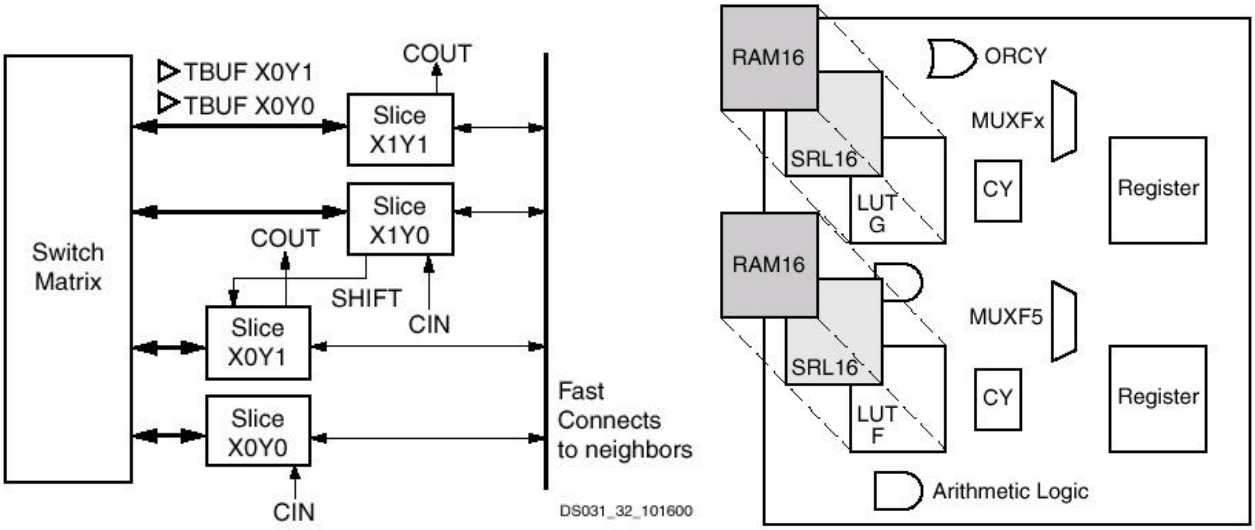


Figure 3.9 – Virtex®2 CLB Schematic

3.2 Virtex® Configuration Overview

Virtex® devices are quickly programmed by loading a configuration bitstream (collection of configuration bits) into the device. The device functionality can be changed at anytime by loading in a new bitstream. The bitstream is divided into frames and it contains all the information to configure the programmable storage elements in the matrix located in the Look-up tables (LUT) and flip-flops, CLBs configuration cells and interconnections, figure 3.10. All these bits are potentially sensitive to SEU and consequently they were our investigation targets.

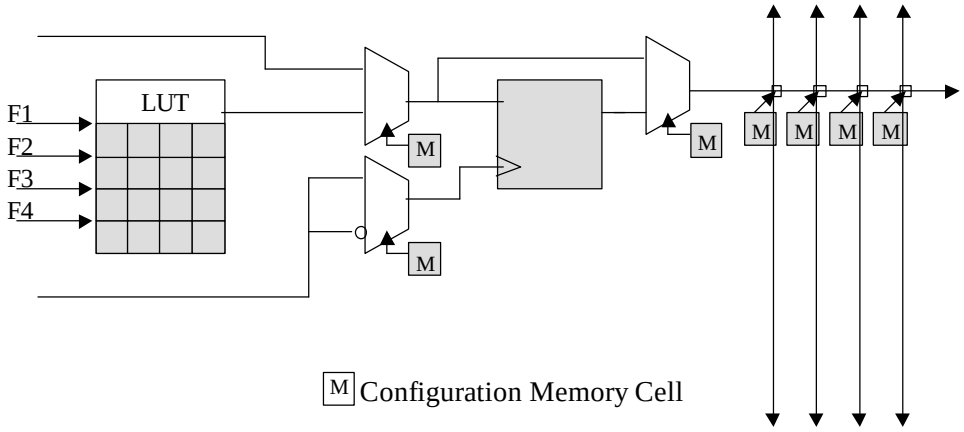


Figure 3.10 - SEU Sensitive bits in the CLB Tile Schematic

The Virtex® configuration memory can be visualized as a rectangular array of bits. The configuration memory array is divided into three separate segments: The "CLB Frames", "BRAM0 Frames" and "BRAM1 Frames. The two BRAM segments contain only the RAM content cells for the Block RAM elements. The BRAM segments are addressed separately from the CLB Array. Therefore, accessing the Block RAM content data requires a separate read or write operation. Read/Write operations to the BRAM segments should be avoided during post-configuration operations, as this may disrupt user operation.

The CLB Frames contain all configuration data for all programmable elements within the FPGA. These include all Lookup Table (LUT) values, CLB, IOB, and BRAM control elements, and all interconnect control. Therefore, every programmable element within the FPGA can be addressed with a

single read or write operation. These entire configuration latches can be accessed without any disruption to the functioning user design, as long as LUTs are not used as distributed RAM components.

While CLB flip-flops do have programmable features that are selected by configuration latches, the flip-flop registers themselves are separate from configuration latches, and cannot be accessed through configuration. Therefore, readback and partial configuration will not affect the data stored in these registers. However, when a LUT is used as either a distributed RAM element, or as a shift register function, the 16 configuration latches that normally only contain the static LUT values are now dynamic design elements in the user design. Therefore, the use of partial reconfiguration on a design that contains either LUT-RAM (i.e., RAM16X1S) or LUT-Shift-register (SRL16) components may have a disruptive effect on the user operation. For this reason the use of these components can not be supported for this type of operation.

However, Select block RAMs (BRAM) may be used in such an application. Since all of the programmable control elements for the BRAM are contained within the CLB Frames and the BRAM content is in separate frame segments, partial reconfiguration may be used without disrupting user operation of the BRAM as design elements.

The configuration memory segments are further divided into columns of data frames. A data frame is the smallest portion of configuration data, which may be read from, or written to, the configuration memory. The bits are grouped into vertical frames that are one-bit wide and extend from the top to the bottom of the array composing a column defined by a major address [XIL00c]. Each matrix column is associated to a major address and to a different number of frames according to the nature of the column, shown in table 3.1.

The frames are read and written sequentially with ascending addresses for each operation. The frame size depends on the number of rows in the device. The number of configuration bits in a frame is 18 x (# of CLB rows +2), and is padded with zeros on the right bottom (LSB) to fit a 32-bit word.

The frame organization differs for each type of column. Each frame is located vertically in the device with the front of the frame at the top. Table 3.2 show the CLB column frame, IOB column frame and BRAM content organization. The frame top is showed on the left.

Table 3.1 - Virtex® Configuration Column Type

Column Type	# of frames	# per device
center	8	1
CLB	48	# CLB columns
IOB	54	2
BRAM interconnect	27	# of blocks SelectRAM columns
BRAM content	64	# of blocks SelectRAM columns

Table 3.2 - Frame Organization

Top 2 IOB	CLB R1	…	CLB Rn	Bottom 2 IOB
18 bits	18 bits	…	18 bits	18 bits

(a) CLB column frame

Top 3 IOB	3 IOBs	…	3 IOBs	Bottom 3 IOB
18 bits	18 bits	…	18 bits	18 bits

(b) IOB column frame

PAD	RAM R0	…	RAM RN	PAD
18 bits	72 bits	…	72 bits	18 bits

(c) Block SelectRAM content column frame

The CLB tile is composed of the CLB logic and the surrounding interconnection placed in a determined row and column in the matrix. There are 864 customization bits per CLB tile distributed in 48 frames with 18 bits each, figure 3.11. The bits can be divided in Look-up table bits (7.4%), CLB configuration bits (6.8%), interconnection bits (84.2%) and 3-state buffer configuration bits (1.6%).

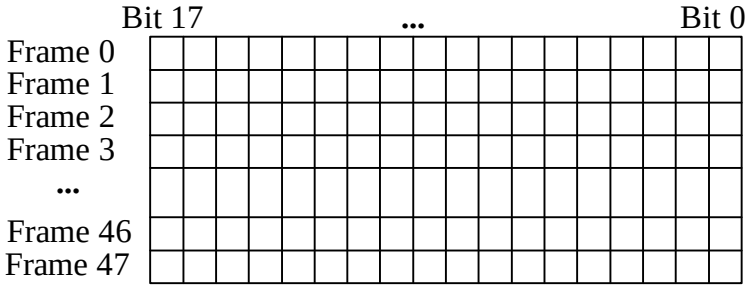


Figure 3.11 - CLB Tile Map

3.3 Fault Injection Analysis in Virtex® protected by TMR

Previous results from the radiation ground testing presented at [CAR01a] showed that using TMR in Virtex® FPGAs, the cross section was reduced by 1,000 times compared to using only the scrubbing technique without TMR. But it was still not zero. The fault injection investigation started with the objective to justify the errors obtained from the ground testing experiments in the Virtex® TMR design. The analysis must explain how a single bit upset in the bitstream of the Virtex® FPGA could cause two errors in distinctly redundant logic parts of the TMR design.

The fault injection in SRAM-based FPGAs is defined as a bit flip in all bits of the configuration bitstream. In this way, it is possible to evaluate the effects of an upset in all sensitive areas of the programmable matrix. Some of these bits are directly related to the user's design combinational and sequential logic, and some of them are related to the FPGA architecture and design implementation.

The fault injection analysis was executed in four main steps. First, the fault injection tool developed by Los Alamos National Laboratory was used to catalogue all the configuration bit locations that caused a dynamic error in the TMR design. Then all the reported bits were identified in the general FPGA matrix in terms of row, column and functionality. Based on this information it was possible to identify those bits in the FPGA IC schematics. The third step identified the correlation between the bit location in the FPGA IC schematic and its location in the design under test in the FPGA editor tool. The last step was the characterization of the error.

3.3.1 Test Design Methodology

The TMR test design methodology used to analyze the SEU in the Virtex® FPGA consists of a TMR counter design replicated in the circuit in order to fill the resources of the device (XQVR300). All of the CLBs were used to implement eight TMR 32-bit counters with pipeline design. The design can be divided in three groups: the redundant logic part 0, redundant logic part 1 and redundant logic part 2. Each redundant group is composed of eight 32-bit counters.

In order to detect an error in one of the 32-bit counters, the eight 32-bit counters located in the same redundant logic group are compared against each other. There is one comparator for each group. Comparators 0, 1 and 2 report an error in the redundant part 0, 1 and 2 respectively.

The three redundant logic groups are finally compared in the minority voter located in the output logic block. The error flag, a result of the minority voter, reports if there is an error in two or more redundant parts. A schematic of this approach is illustrated in figure 3.12.

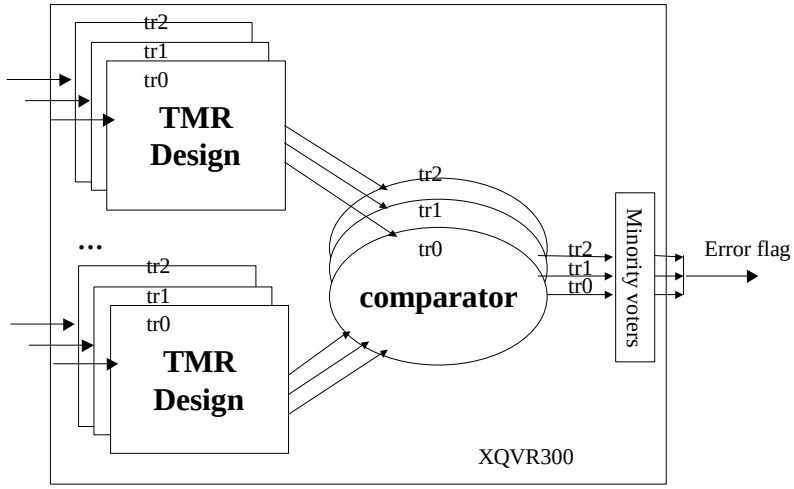


Figure 3.12 - TMR Design Methodology

3.3.2 Fault Injection Tool

The fault injection tool developed in Los Alamos National Laboratory specific for Xilinx needs is able to corrupt all the bits of Virtex® bitstream in a sequential way, or individually by choosing a specific bit location. The objective of this tool is to analyze the effect of a single bit upset in a TMR design implemented in the Virtex® architecture. All single bit upsets able to cause an error in the TMR design were cataloged for investigation. In this text, this tool will be named as Virtex fault injection tool.

In principle, no single bit upset in the bitstream should cause an error in the TMR design if a single upset error affects only one redundant part of the design. By TMR definition, if one redundant part is corrupted by an upset, the majority voters continue voting the correct value from the two other redundant uncorrupted parts.

The Virtex fault injection tool can upset a single bit in the bitstream sequentially starting from a user defined major address and frame, or it can upset one specific bit when the user defines the major address, frame, frame byte and bit. Fault injection is performed in three steps, presented in table 3.3. The three-step method guarantees no double upsets for any short period of time.

Table 3.3 - VIRTEX® Configuration Column Type

Fault injection steps	Bitstream example
Read the bitstream:	… 0110010101010…
Corrupt one bit and load the bitstream:	… **1**110010101010…
Correct the previous bit and load the bitstream:	… **0**110010101010…
Reset the flip-flops	… 0110010101010…

Each time an error is reported by the test design comparator, the fault injection tool shows the location of the upset bit that caused the error. The tool reports the major address, frame, frame byte and bit location of the bit. Using this information it is possible to know exactly the location of the bit in the bitstream, and as consequence in the FPGA matrix.

The fault injection test platform, shown in figure 3.13, is made from two AFX V300PQ240-100 daughter cards, a MultiLINX™ cable used as an interface to a host PC, and a control panel. The system can operate stand-alone or in conjunction with a host PC and test software. The control panel communicates directly with the control chip to specify the mode of operation. Configuration of the DUT may be controlled by either the control chip or the test software via the MultiLINX Cable. The control chip also controls the dynamic operation of the DUT and dynamic error detection.

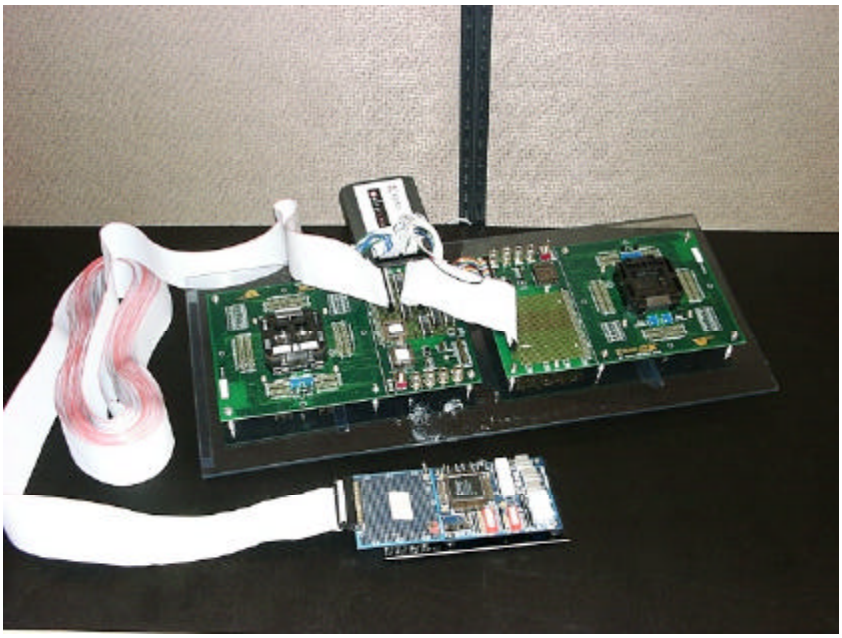


Figure 3.13 - SEU Test platform

3.3.3 Association between a bit upset in the bitstream and its location in the design floorplanning

For a given bit located in the bitstream in the CLB frames, there is a unique address location that is defined by the *Major Address*, *Frame*, *Frame Byte* and *Bit* position. In order to know the location of this bit in the FPGA matrix and consequently its purpose in the user design, it is necessary to follow the next steps:

1) Bit column location in the matrix

The CLB address space begins with ’0’ for the center column and alternates between the right and left halves of the device for all the CLB columns, then IOB columns, and BRAM interconnect columns, as illustrated in figure 3.14. Analyzing the figure schematic, if the major address is 23, for example, it means that the column is 24.

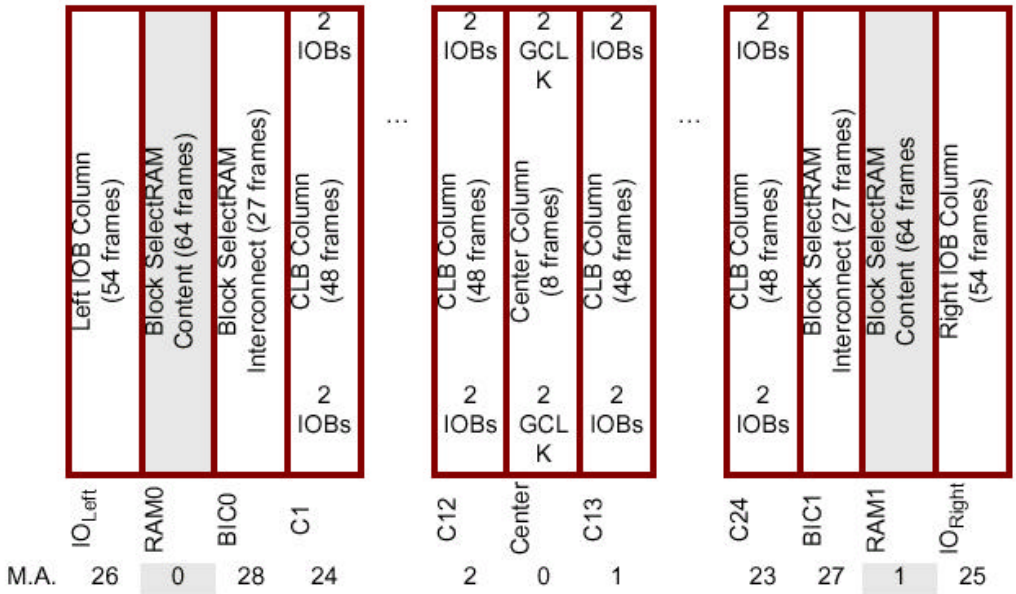


Figure 3.14 - Example of frame organization in Virtex® family

2) Bit row location in the matrix

Each bit column starts from the top I/O block, pass through all CLB blocks and it ends in the bottom I/O block. Each row has 18 bits, including the I/O block. The equation 3.1 provides the row position.

```
Row = (Byte Frame x 8 + bit) / 18                                    3.1
```

The frame, frame byte and bit data are used to obtain the exact bit location in the CLB tile by using the equation 3.2.

```
CLB tile bit = 17 - [Byte Frame x 8 - Floor((Byte Frame x 8 +        3.2
Bit) / 18), 18) + Bit]
```

3) Bit location in the CLB

Each bit of the CLB tile has been identified in the FPGA IC schematic and therefore in the design floorplanning by using some internal Xilinx tools. In this way it was possible to build a design flow from the upset bit information coming from the fault injection tool (major address, frame, frame byte and bit) and the final design floorplanning bit location.

4) Bit Classification

The CLB map has the general bit classification (LUT, flip-flop, customization or routing) in terms of frame number and the bit location in the CLB. This map gives just a general view. In order to be able to find the specific location of the bit in the CLB architecture a table containing all the bit names is used. Table 3.4 shows a portion of the used CLB map table containing just the frame address 0.

Table 3.4 – Bit Classification in the CLB

Frame	Bit	Function
0	0	I_c_hexes.I67.hex_mux_s3
0	1	I_c_hexes.I55.hex_mux_s3
0	2	I_c_singles.I621.I377
0	3	I_c_singles.I625.I377
0	4	I_c_singles.I611.I377
0	5	I_c_singles.Iw22o7.I377
0	6	I_c_singles.Is23n23.I377
0	7	I_c_singles.I101.I377
0	8	I_c_singles.Iw0n23.I377
0	9	I_c_imux.s1ce_16to1.I16to4_2
0	10	I_c_imux.s0ce_16to1.I16to4_4
0	11	I_c_imux.s1ce_16to1.I2to1
0	12	I_c_imux.s0ce_16to1.I16to4_1
0	13	I_c_cle.pblk1.pibce.cfgmem
0	14	I_c_cle.slice1.luts.flut.lm15.I74
0	15	I_c_cle.slice1.luts.glut.lm15.I74
0	16	Ic_lng_tbf.Itblk.I57.Its12
0	17	I_c_omux.Iom6.om2

Analyzing the table 3.4, there are some names that can be easily associated to a routing position, for example I_c_singles.Iw22o7.I377 that shows the connection between the single wire West 22 and single wire out 7. However, the majority of the names do not make any sense without the use of the FPGA schematics.

An intense search was done in the Virtex schematic in order to associate each name to each structure. But still this was not enough to be able to associate the bits in the FPGA architecture to the signals in the user design. The software package called XDL [XIL01b] must be used to see all the

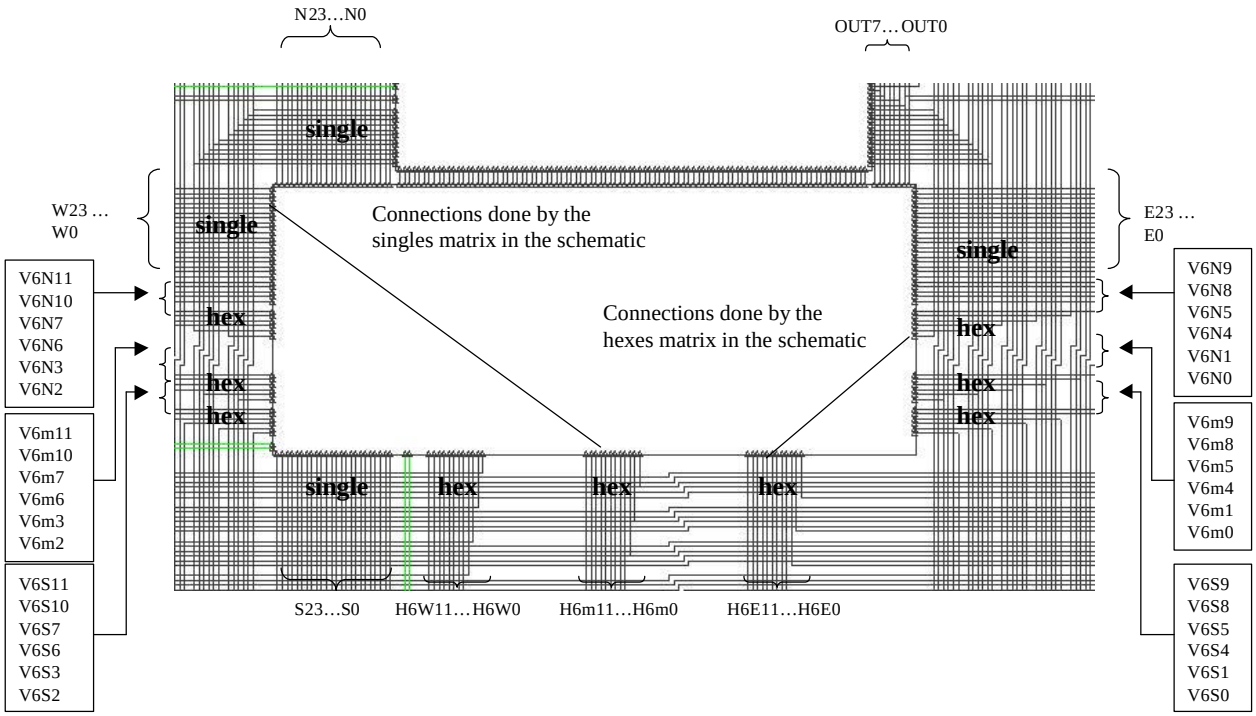
connections and instantiations in the user design. XDL is a fully feature design language that provides direct read and write access to Xilinx proprietary Native Circuit Description (ncd), figure 3.15. It is a single tool with 3 fundamental modes: report device resource information, convert NCD to XDL (ncd2xdl) and convert XDL to NCD (xdl2ncd).

```
Command example: xdl -ncd2xdl design.ncd design.xdl
```

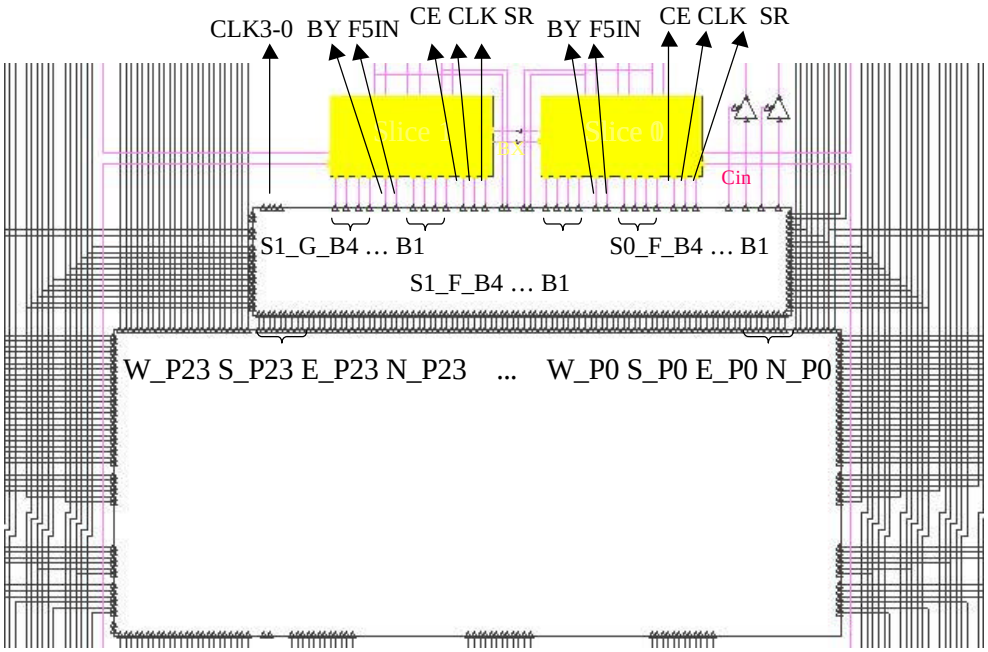
```
inst "counter6/I$2/pipe0/N$3(3)" "SLICE" , placed R31C7 CLB_R31C7.S1 ,
 cfg "CKINV::1 DYMUX::1 DXMUX::1
      F:counter6/I$2/pipe0/stage2/pipe3/I$8:#LUT:D=(~A2*A3)
      G:counter6/I$2/pipe0/stage2/pipe2/I$8:#LUT:D=(~A2*A3)
      CEMUX::CE_B SRMUX::SR_B GYMUX::G FXMUX::F SYNC_ATTR::SYNC
      SRFFMUX::0 INITY::LOW
      FFX:counter6/I$2/pipe0/stage2/pipe3/I$3:#FF
      FFY:counter6/I$2/pipe0/stage2/pipe2/I$3:#FF INITX::LOW
      _PINMAP:24:0,1,2,3,4,5,8,6,7,9,10,11,14,12,13,15,16,17,18,19,20,21,22,23"
 ;

net "count_data_tr2_0(1)" ,
 outpin "count_data_tr2_0(0)" YQ,
 inpin  "mux2/L1.L1.1_mux/G_22_73" G4,
 pip R32C46 S0_YQ -> OUT0 ,
 pip R32C46 OUT0 -> N0 ,
 pip R31C46 S0 == E22 ,
 pip R31C47 W22 -> W_P22 ,
 pip R31C47 W_P22 -> S1_G_B4 ,
 # net "count_data_tr2_0(1)" loads=1 drivers=1 pips=5 rtpips=0
 ;
net "L0.L0.3_C/Pipe2/pipeline_5(0)" ,
 outpin "L0.L0.3_C/Pipe2/pipeline_5(1)" YQ,
 inpin  "count_data_tr2_3(0)" F2,
 pip R16C46 S1_YQ -> OUT5 ,
 pip R16C46 OUT5 -> E15 ,
 pip R16C46 E15 -> E_P15 ,
 pip R16C46 E_P15 -> S0_F_B2 ,
 # net "L0.L0.3_C/Pipe2/pipeline_5(0)" loads=1 drivers=1 pips=4 rtpips=0
 ;
```

Figure 3.15 – Example of design connection file (.ncd)

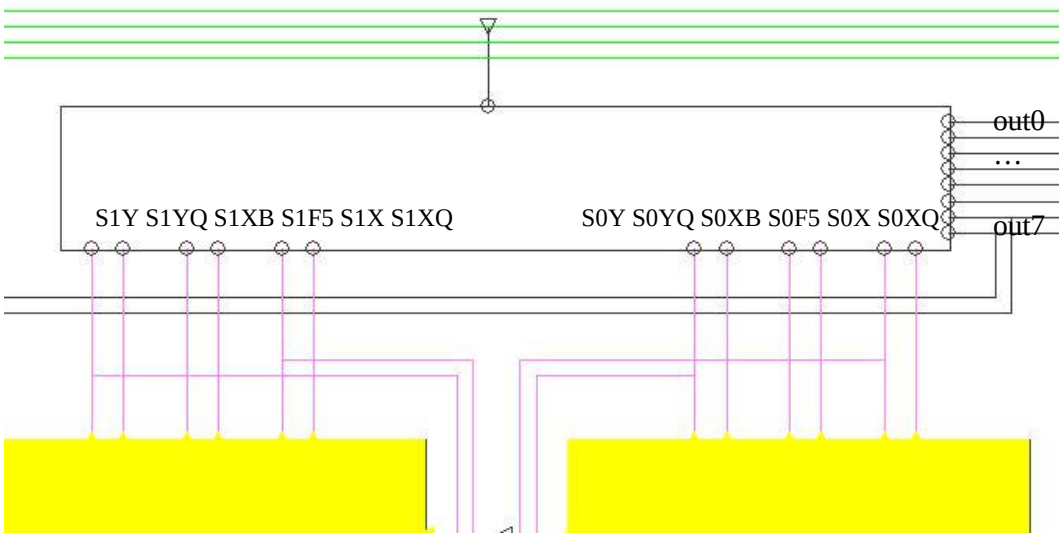
Based on the information of the Virtex® Architecture and the equations presented in this section is possible to make a correlation between the programmable bit in the bitstream and its location in the design floorplanning that can be observed in the Foundation or Alliance software. Figure 3.16 (a) shows the single and hex routing segments in the design floorplanning and the correspondent segments in the FPGA schematic. Figure 3.16 (b) shows the CLB slices signals and input multiplexor connections in the design floorplanning and the correspondent names in the CLB FPGA schematic. Figure 3.16 (c) shows the output multiplexor signals in the design floorplanning and the correspondent names in the CLB schematic. These next three figures help to correlate the signals from the design and the signals in the FPGA schematic, giving the location and placement of both.



(a) Single and Hex Routing



(b) CLB Slices and Input Multiplexors



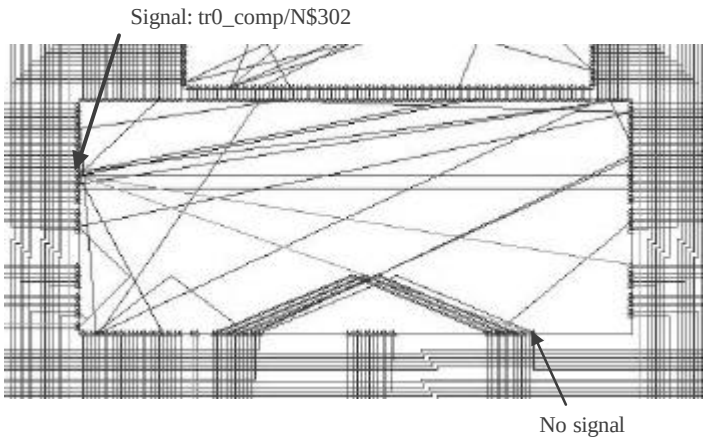
(c) Output Multiplexors

Figure 3.16 – CLB Tile representation in the Foundation Floorplanning Tool

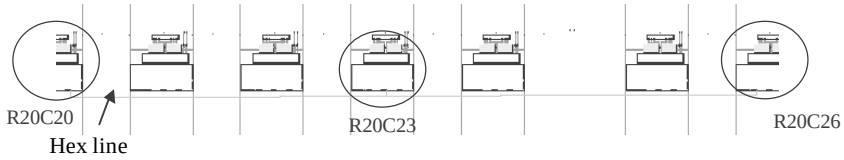
3.4 Fault Injection Results

The fault injection was performed in the TMR test design running at 10 and 20 MHz. The report showed that 224 upset bits of 1,663,200 bits in the XQVR300 bitstream had caused an error in the TMR design application execution.

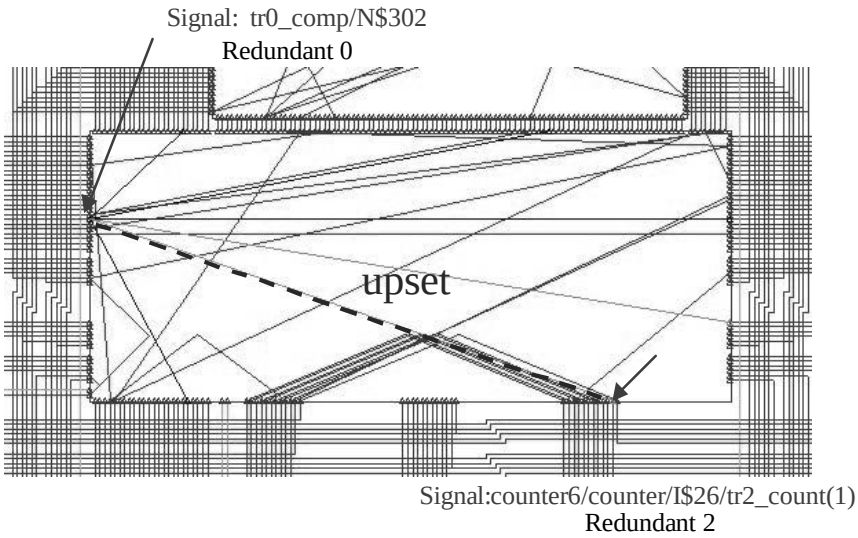
Analyzing the upset bits in the design floorplanning, we observed that a single upset in the routing matrix (GRM) could provoke an undesirable connection between two different signals placed in distinct parts of the FPGA. An example of upset in the GRM that was able to cause an error in the output of the TMR design is located in the major address: 10, frame: 35, frame byte: 46, bit: 5 of the bitstream. Using the equation 3.1 and 3.2, the upset bit in the floorplanning is placed at CLB row 20, column 20 (R20C20) and CLB bit tile: 4. The upset was identified in the IC schematic as I_c_singles.Iw6he1.I377 that means a connection between the single line west 6 and hex line 1, represented in figure 3.17 (a). Apparently this upset can not generate an error because it connects a signal from the comparator of the redundant part 0 to “no” signal. However, the hex line connects the CLB R20C20 to two others CLBs as displayed in figure 3.17 (b) marked by circles. Analyzing the CLB R20C23, for example, we noticed that actually there is a signal connected to this hex line. The signal is from one of the counters of the redundant part 2, as shown in figure 3.17 (c).



(a) Upset in CLB R20C20



(b) CLBs connected by the hex line



(c) Undesirable connection detected in CLB R20C20

Figure 3.17 - SEU example in the GRM user's design floorplanning

The analyzed upset bit was characterized by an undesired connection from one bit of the 32-bit counter in a redundant module to a signal from the comparator logic of another redundant module. In this case, both comparators 0 and 2 are going to report an error producing "one" in the error flag. This kind of error would have never occurred if the comparators were placed out of the chip.

In order to avoid upset connections between the test design and the comparator test circuitry, a new TMR design based on the "golden" chip approach was implemented in the Virtex® device, where the DUT output signal is compared to the golden design placed outside the chip, figure 3.18. In this case, if a single bit upset in the DUT routing matrix provokes an undesirable connection between two signals from different redundant parts of the design, the TMR will always vote the correct signal to the storage elements and to the output. A bit flip in the customization logic will only be able to generate an error if it upsets the exact same bit in two distinct redundant logic parts, which has an extremely low probability to occur. Moreover, this type of error can be totally avoided with a structured floorplanning of the design placement.

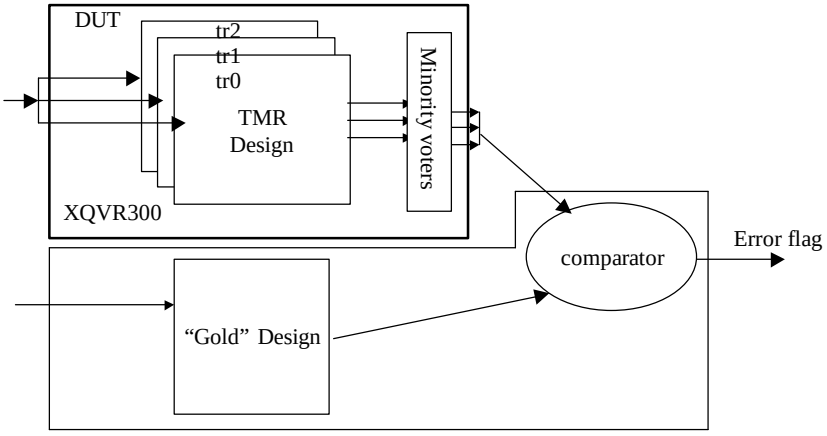


Figure 3.18 - "Golden" Chip Method

The fault injection experiment using the “golden” chip method was performed in the TMR design running at 25 MHz. The tool has reported “no errors” for all the bits in the bitstream. The result has finally confirmed the efficacy of the TMR structure to recover an error in the FPGA architecture. The radiation characterization results [CAR01b] performed at the proton facility in UC Davis show that the *Virtex*® FPGA has presented the same reliability achieved by the fault injection experiment.

3.5 Designing and Testing a TMR Micro-controller on Virtex® FPGA

Micro-controllers implemented in programmable logic platforms are becoming more and more advantageous in order to integrate systems on a single chip (SOC) improving performance, flexibility and time to market. When a micro-controller is implemented in a SRAM based FPGA, not only the registers and memories are sensitive to SEU but also all the programmable logic defined by the FPGA architecture such as the Lookup Tables, routing switches, flip-flops and memories. The previous experiment has shown that the TMR can protect designs against SEU in SRAM-based FPGAs platforms, for this reason it has been applied in the micro-controller architecture too. In addition, a fast time-to-market using commercial off-the-shelf micro-controller architecture for space applications can be achieved by protecting the micro-controller core description and implementing in Virtex® QPRO FPGA.

In this direction a micro-controller VHDL description developed at UFRGS and presented at [CAR96, SIL97] was re-used to implement the SEU hardened micro-controller into Virtex® XQVR300 FPGA using the TMR techniques proposed in [CAR01a]. The 8051-like VHDL description is divided into six main blocks as illustrated in figure 3.19. The Finite State Machine (FSM) block implements a counter that generates 24 clock cycles to guide the instruction execution. The Control unit generates all the enable signals for the registers and Arithmetic unit located in the datapath. The Instruction unit generates the microcode word for each instruction. The datapath includes an Arithmetic Logic Unit (ALU) and many registers. There are two 256 bytes internal memories, one for the data and the other for the application program.

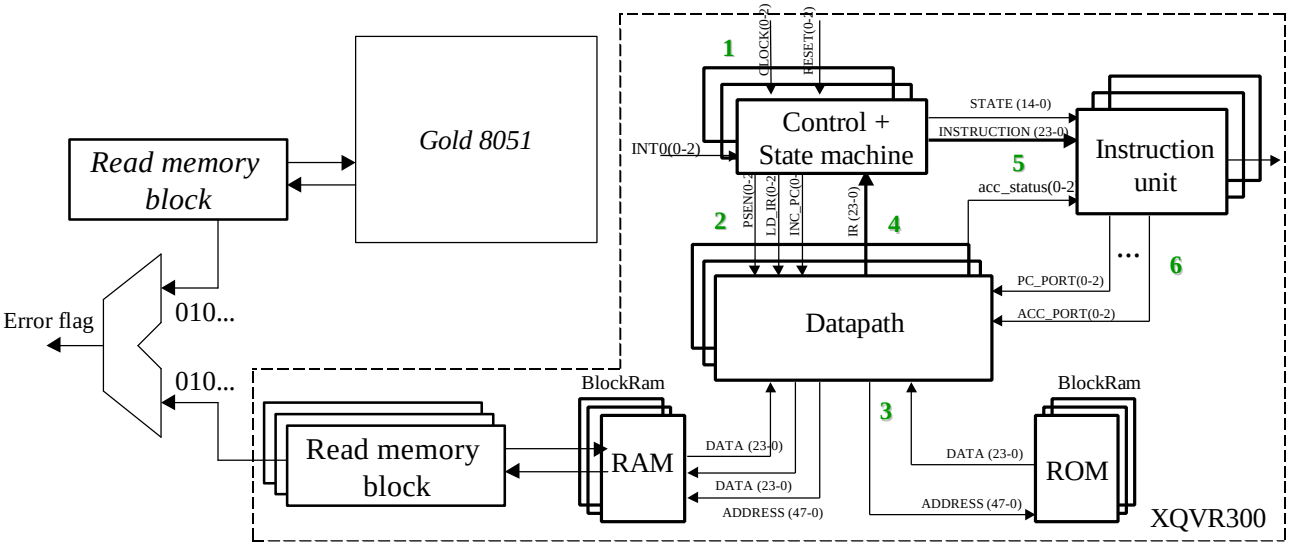


Figure 3.19 - TMR 8051 Design Methodology

The 8051 micro-controller runs an application based on two 6x6 matrix multiplication at a frequency of 10 MHz. This application performs the multiplication by shifter register and addition. This allows an intensive use of the available memory and internal registers since the operators are read and written many times and both operators and result are stored in the internal memory.

An extra logic circuit was designed to be able to analyze the results of the 8051 after a bit is upset. This block is able to read all the memory data and to send serially the data to an output pin. This output

data is compared to the “golden” chip located in a distinct FPGA device. If the data does not match to each other, the comparator circuit sends a flag error to the fault injection tool. Each corrupted bit able to cause an error in the TMR design is reported in a file.

In order to protect the VHDL description against SEU, each logic block has been triplicated and voters were inserted in all register loops. The datapath, control and instruction logic blocks are mainly throughput logic and consequently they were just triplicated. The registers in these blocks are constantly been written avoiding being locked in a wrong state. An example of a TMR datapath register with its surround logic in VHDL code is presented in figure 3.20. The vector signals were replaced by an array of 3 vectors (0, 1 and 2) representing the vector signal for each redundant logic part.

```
L0: For K in 0 to 2 generate
process (OP_ACU(K), reg_alu_out(K), data_rom(K), reg_PC_low(K), data_rd_ram(K))
begin
CASE OP_ACU(K) IS
WHEN "000" => reg_accu_mux(K) <= "00000000";
WHEN "001" => reg_accu_mux(K) <= reg_alu_out(K);
WHEN "010" => reg_accu_mux(K) <= data_rom(K);
WHEN "101" => reg_accu_mux(K) <= reg_PC_low(K);
WHEN others => reg_accu_mux(K) <= data_rd_ram(K);
END CASE; end process;

process (reset_micro(K), clock(K))
begin
if (reset_micro(K)='0') then
    reg_accu(K) <= "00000000";
elsif (clock(K)'event and clock(K)='1') THEN
   if (GCE(K)='1') then
   if (accu_port(K)='1') then
        reg_accu(K) <= reg_accu_mux(K);
   end if; end if;
end if; end process; end generate;
```

Figure 3.20 - Example of TMR VHDL code

All persistent errors (caused by ‘weak-keepers’) were avoided by using user ground input and user global clock enable. The registered loops located in the state machine and in the counters were protected by TMR with a major voter in each redundant feedback path. All voters were implemented using LUTs. The internal memories were replaced by the TMR BRAM component presented previously in figure 3.8. In the DATA memory there is a circuitry able to detect write conflicts in the memory when refreshing. The micro-controller always has the write priority. In the program memory there is no conflict because it is a read only memory from the micro-controller point of view.

Table 3.5 shows a summary of the TMR design logic overhead in the 8051-like micro-controller. The number of flip-flops in the TMR design has increased in 3.6 times. The ratio exceeds 3 because of the extra counters located in the BRAM scrubbing logic. The TMR design contains 3 times the number of BRAM and each one of them has an extra logic of flip-flops and LUTs for voters, counters and logic analyzes. The number of LUTs in the TMR design is approximately 3.6 times bigger than in the standard design. This proportional also exceeds 3 because of the voters and the scrubbing logic. Three of the four available global clock buffers in device are being used for the system clock.

Table 3.5 - TMR Logic Overhead in the 8051 (XQVR300)

Item	Standard 8051	TMR 8051
FDCE	127	459
BRAM	2 of 16	6 of 16
TMR BRAM extra logic	-	36 FDCE, 87 LUTs
Inputs	2	12
Outputs	1	3
BUFG	1 of 4	3 of 4

LUTs	757 (12%)	2778 (45%)

The fault injection experiment was performed using the test board presented in figure 3.13 at 10 MHz. Bit flips were inserted in all 1,663,200 bits of the XQVR300 bitstream. Each fault has remained in the bitstream enough time to run many cycles through the application in the micro-controller. The fault injection tool has reported "no errors" by the "golden" system method output in a presence of single upsets in the TMR bitstream. The radiation ground testing was performed at Davis University in California, USA.

This work has validated a test methodology to evaluate the effects of a single bit upset in the Virtex® architecture. Using the steps mentioned in the paper, it is possible to find the specific location of an upset bit in the FPGA IC schematic and consequently in the user's design floorplanning. This methodology was useful to invalidate the previous test design methodology and to show the new results from the "golden" chip approach.

The fault injection analysis results show that the "golden" TMR method has presented "no errors" in presence of single bit upset in the Virtex® bitstream. This report demonstrates the efficiency of the TMR SEU mitigation technique for SRAM-based FPGAs. Radiation ground testing performed in the "golden" chip TMR methodology in the counter design confirmed the results achieved by the fault injection experiment. The fault injection results in the TMR 8051-like micro-controller were meaningful to continue the validation of this technique and it has confirmed the reliability of the Virtex® TMR design techniques.

3.6 TMR 8051 Micro-controller Radiation Ground Test Results and Conclusions

The fault injection results showed that less than 1 % of the bit upsets could provoke an error in the output of the TMR design, representing a very small cross-section. The test was performed at Crocker Nuclear Laboratory at UC Davis. The proton energy and fluxes were measured as incident on the DUT package. All tests were performed at room temperature. More details about the test can be found in [LIM02b].

The test platform is composed of two AFX V300PQ240-100 daughter cards, a MultiLINX™ cable used as an interface to a host PC, and a control panel. The system can operate stand-alone or in conjunction with a host PC and test software. The control panel communicates directly with the control chip to specify the mode of operation. Either the control chip or the test software, via the MultiLINX™ cable, may control the DUT configuration. The control chip also controls the dynamic operation of the DUT and dynamic error detection.

The beam energy was set to 63.3 MeV. The proton flux was varied from 8.54E+08 to 1.70E+09 protons/sec-cm^2, in order to ensure a scrubbing rate higher than the error rate. The TMR 8051 design was tested in the dynamic mode and compared to the non-protected design. The tested part was XQVR300 (0.22um, 2.5V). The cumulative limit of TID achieved in this test was 116 krads(Si).

The fluence to upset was measured in the no-TMR 8051 design and in the TMR version. The first experiment used the test software to readback the bitstream in order to analyze the nature of the dynamic errors. When an error was detected, a readback of the bitstream was initiated and the number of bitstream errors noted alongside the total fluence to functional error. The second experiment compared the TMR design with and without scrubbing, no readback was performed. A logical reset of the flip-flops used in the design would then demonstrate whether the functional error was from configuration/user upsets or the architecture ones. The fluence to upset was measured while the PROM was continually scrubbing the configuration bits.

Experiment 2 tested 3 different approaches in order to demonstrate the benefits of TMR combined with scrubbing, see figure 3.21. Each test measured the fluence to failure. The no-TMR and TMR designs were tested with and without scrubbing. Table 3.6 presents the TMR 8051 cross-section average for the observed fluence to upset collected in the second experiment.

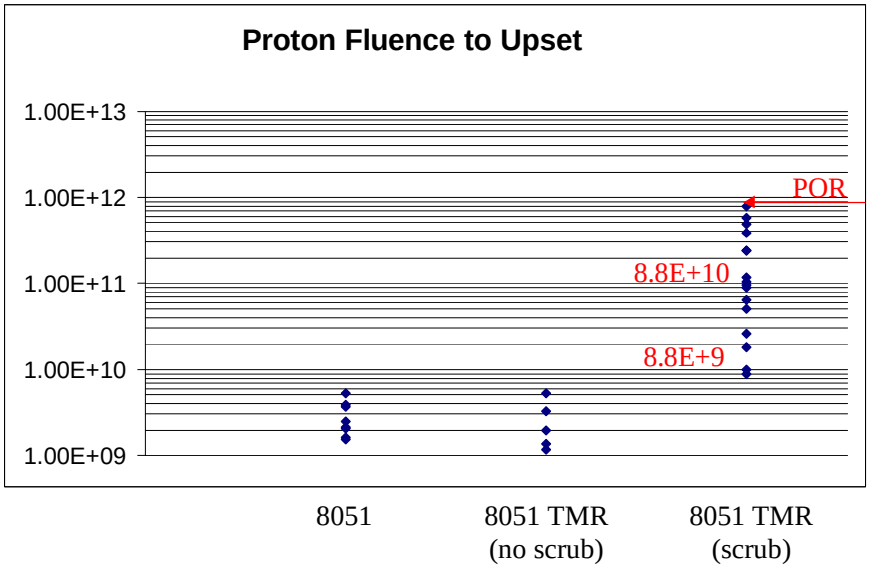


Figure 3.21 - Testing Platform

Table 3.6 - Virtex[7] Dynamic Cross-section of TMR 8051

Upset	Hit	Cross-section (cm^2)
Bit	18	6.93E-12
Persistent	2	1.91E-10
POR	0	
Average		2.54E-11

The experiment frequency was set at 9 MHz. The same clock was provided to the scrubbing PROM. The BRAM refreshing performed inside the DUT used the same clock divided by 8. It takes 4 ms to entirely run the two 6x6 matrix multiplications and the internal memory read. The scrubbing takes 22 ms to refresh the whole matrix. And the BRAM refreshing takes 0.2275 ms to refresh all addresses.

In summary, the application runs 4 times during a scrubbing cycle and the BRAM is refreshed 17 times per application cycle as it is illustrated in figure 3.22. The application re-starts with a reset in the micro-controller coming from the read memory logic. In general, bits form the BRAM and the CLB flip-flops (user logic upsets) have the highest refresh rate. The LUTs, customization and routing bits (configuration upsets) are refreshed by the scrubbing rate.

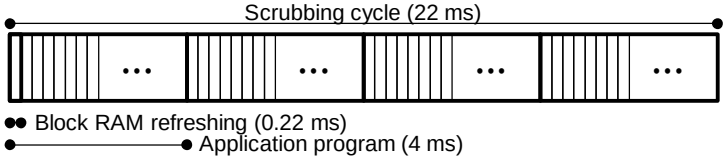


Figure 3.22 - Scrubbing and Refreshing Times

An error can just occur in the design functionality if the number of accumulated upsets is enough to overcome the TMR. For example, if an upset in the routing occurs in the first application execution time of the scrubbing cycle, 2 out of 3 legs in the TMR should be able to vote the correct value. However this undesirable connection or disconnection may affect different parts of the design generating upsets that can be stored in different redundant parts. All of these upset cells must be refreshed with their original values. If the refreshing rate is such that one can not avoid the accumulation of upsets, errors are going to be observed in the output. It is important to analyze how the upsets can propagate inside the architecture. In the next application cycle, the CLB flip-flops are going to be reset, however the BRAM are never reset,

they have always been refreshed by voting their own values. If the refreshing in the BRAM is not fast enough to avoid accumulation of upsets, a failure can be observed in the output.

The average of TMR 8051 error rate = 17 bits/upset (6e-2 upsets/bit/s = 190 upsets/bit/day) and the average of scrub rate is 45. This means that in average 0.4 upset per bits scrubbing. This rate could be insufficient, besides the point that the flux is not always constant (2 or more upsets can occur during a scrubbing cycle) and the upsets can propagate in the architecture generating more upsets. In real applications the scrubbing rate should be at least 2 orders of magnitude higher than the error rate.

In order to improve the results, two solutions can be tried. The first option is to set the proton flux in the radiation facility one order of magnitude or more lower in order to be sure that there is only one upset per scrubbing cycle. In space the flux is much, much lower than the test 99% of the time. However a very low flux would take a long time to observe each error. The other solution is to speed up the scrubbing frequency. However the PROM used can only achieve up to 16 MHz with reliable performance.

3.7 Final Considerations

The TMR technique for SRAM FPGAs was tested in Virtex family using two designs. The first design was a 32-bit counter and the other design was an 8051 micro-controller. The results observed in both designs have proved that the reliability of the TMR is strongly related to the placement of the design in the FPGA matrix. In the first design, the results achieved presented that no errors could be observed in presence of upsets. The main reason is because of the counter design is a simple architecture that does not use embedded memory (BRAM). Consequently the scrubbing issue is not so evident. In addition, the presented result was based on that specific placement. There is a probability that if another placement is performed, the results can change.

When a more complex design that uses embedded memory such as the 8051 micro-controller was tested, the probability of upsets in the routing provoke an error in the application results was more eminent because of its complexity and the scrubbing issue in the embedded memories. Many placements were performed in the TMR 8051 and each one has show different results in terms of upset bits that could provoke an error in the application results. However, in each case the difference was manly in the routing bit locations that could provoke an error and not in the number of bits that was always around 1% of the bits of the bitstream.

Based on the analyzed results, the TMR technique in SRAM based FPGAs can improve substantially the reliability of the design but there is a probability that around 1% of bits in the bitstream related to the routing can provoke an error in the TMR. This result is correlated with the placement. A dedicated placement can completely avoid the chance of errors in the TMR. However, a dedicated placement is very costly, there is no automatic tool for that right now, and can generate a worse area usage in the matrix.

4 Thesis Proposal: Designing and Improving SEU mitigation techniques for SRAM-based FPGAs

SRAM-based FPGAs have many components in their internal architecture such as complex logic blocks with lookup tables, multiplexors and flip-flops, embedded memories, PLLs and dedicated routing segments. In addition, next generation of FPGAs will have not only the possibility of soft cores insertion but there will be also embedded microprocessors hard cores inside the chip in order to improve the data processing and performance (figure 4.1).

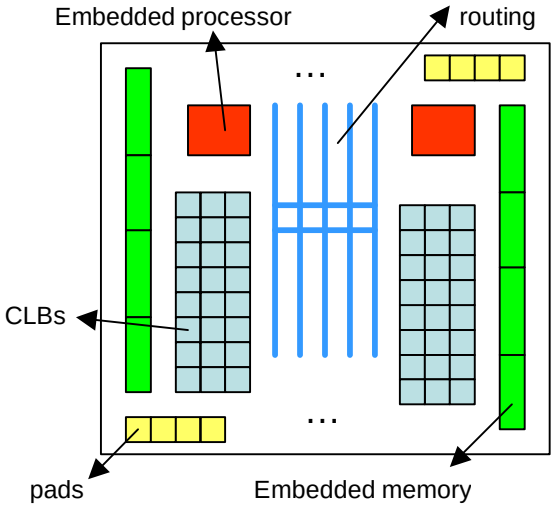


Figure 4.1 – A case of Study: Hypothetical FPGA architecture

The SEU mitigation problem in the next FPGA families with embedded hard processors can be analyzed as two parts: the microprocessor and the programmable logic. And consequently, each part can be sub divided in small logic blocks according to the functionality and some special features. Studies about the protection of a microprocessor have been done in the 8051 like micro-controller developed at UFRGS [CAR96]. All registers and the internal memory were protected by hamming code [LIM00a, LIM00b, COT00]. The results have shown the reliability of this method and the necessity of refreshing in some parts of the circuit, mainly in the memory, in order to avoid accumulation of upsets. A fault injection system was developed [LIM01b] in order to test the standard and the full SEU tolerant 8051 in presence of single and multiple upsets [LIM02a]. Based on the references presented in the chapter 2 and the studies previously done, the problem of protecting microprocessors against SEU is relatively well understood and the available techniques presented in the literature can be applied in order to achieve reliability.

However, the problem of protecting SRAM-based FPGAs against SEU is not well solved yet and much more studies are required to improve the limitation of the methods currently used as presented in the previous chapters [LIM01b]. This work proposes the investigation and development of SEU mitigation techniques for SRAM-based FPGAs that can be applied for FPGAs with or without embedded processors. The SRAM based FPGAs were chosen because their high applicability in space. Different than FPGAs programmed by anti-fuse that can be programmed just once, SRAM based FPGAs can be reprogrammed by the user as many times as necessary in a very short period. So, applications can be updated and corrected after launch. This feature is essential for space applications because it can reduce the cost in update missions or even save missions that were launched with design problems.

Xilinx is the leader of the market in SRAM-based FPGAs with families such as Virtex®, Virtex®E and Virtex®2. And in this direction, Xilinx recently has signed an agreement with IBM to manufacture the PowerPC embedded in the Virtex®2 FPGA core [XIL01a]. The new family called Virtex®2p can have up to 4 PowerPC microprocessors inside with 1-gigabit I/O pins. Because of the large usage of FPGAs from Xilinx, the Virtex family was select to be studied.

The technique used nowadays in the Xilinx FPGAs to protect the design against SEU is the TMR, as presented in chapter 3. The TMR is an attractive solution for SRAM based FPGAs because it only changes the high level design description. It does not need changes at mask level. On the other hand, because it does not change the FPGA design by itself, it presents some limitations that were presented in the previous chapter. Many applications can accept these limitations but some can not. The main limitations are:

- ?? the number of I/O pads that is reduced by three,
- ?? the number of dedicated clock resources segments for the routing that is also reduced by three,
- ?? the size of the design logic that is multiplied by three,
- ?? the embedded memory that also needs to be triplicated and refreshed using an extra logic,
- ?? the delay inserted by the voters,
- ?? the easily propagation of upsets through the TMR routing on the FPGA architecture.

Almost all parts of the SRAM-based FPGA are sensitive to radiation, but some of them are more critical them others as analyzed in chapter 3. The main issues of the SRAM based FPGAs is the upsets in the routing. By the perspective of the TMR approach, there are three blocks, the redundant part 0, redundant part 1 and redundant part 2. All these parts are voted in order to assure reliability. The upsets in the FPGA can be isolated inside a unique redundant part or they can propagate and affect two or more redundant part according to the upset location.

The upsets in the LUT are isolated because they can generate an error only in the combinational logic defined by that LUT restricted to the redundant part affected by the upset. An upset in the flip-flop located in the CLB is also isolated because an error in this location can not propagate to other redundant parts. An upset in some latches of the CLB customization are relative isolated because if two distinct redundant parts are implemented in the same CLB tile using different slices, an upset in one of the customization latches may provoke an error in two redundant parts at the same time. This event characterizes an error in the TMR caused by a single event upset, which is extremely undesirable.

An upset in the routing is definitely not isolated in only one redundant part. There is always a probability that an upset in the routing located in the switch matrixes can connect two different signals from distinct redundant parts that can provoke an error on them. In this case the TMR by definition will not be 100% reliable. The routing upsets effects are strongly related to the scrubbing rate of the matrix. However the scrubbing rate will never be fast enough in order to avoid completely the probability of an upset propagation. Upsets in the embedded memory are isolated in the memory but according to the scrubbing rate of the memory, it is possible to accumulate upsets that can also provoke an error in the TMR design.

All this events were analyzed by fault injection and by ground radiation test in the TMR 8051 micro-controller in the Virtex® FPGA. The number of bits that can be upset and that can generate an error is very low. Around 0.1% of the bits in the bitstream are able to provoke an error in the output. This

number is extremely low but it is not zero, so it brings the necessity to research new SEU mitigation techniques for FPGA that could bring improvements. The solution can be in the improvement of the TMR such as using a fault injection tool combined with a dedicated placement in order to achieve 100% of reliability. However TMR limitations such as all listed above will not be solved unless new techniques are proposed.

In order to better analyze the FPGA architecture, the programmable logic part of the FPGA can be divided in the CLB logic, routing, embedded memory and special features. The CLB can be sub divided in the Lookup Table (LUT), configuration memory cells, combinational logic and flip-flops. The routing can also be sub divided in switch matrix that connects the single and hex segments, input multiplexors, output multiplexors and special segments (clock, high performance connections, etc.). The embedded memory is defined as a configurable block composed of memory cells that can be implemented as single or dual port memories with a large flexibility in the number of addresses and the word size. The special features vary among FPGA architectures, examples are PLL, JTAG, POR, etc. Table 4.1 describes each part of the CLB tile from the Virtex® 300.

Table 4.1 – Evaluation of the Sensitive Cells in the Virtex® 300 CLB

CLB Block (1536 per chip)	# latches				Total
	LUT (combinational logic)	Flip-flop (sequential logic)	Customization	Routing	
Omux: eight mux 12:1				7 latches x 8	56 (6.65 %)
CLE: 2 logic blocks, each one has 2 LUT (16:1), 2 flip-flops and multiplexors	LUTFslice0: 16 latches LUTGslice0: 16 latches LUTFslice1: 16 latches LUTGslice1: 16 latches	Flip-flopyslice0 Flip-flopxslice0 Flip-flopyslice1 Flip-flopxslice1	LUTctrl: 7 latches x 2 Ffctrl: 3 latches x 2 Ffconfig: 3 latches x 2 x 2 Config: 5 latches x 2 2 unused latches		110 + 2 (13.30%)
Long-tbfs			2 + 2 + 4 Buffer: 5 1 unused latch		13 + 1 (1.66%)
Imux: sixteen mux 28:1, twelve mux 16:1, four mux 8:1				9 latches x 16 6 latches x 12 3 latches x 4	228 (27.07%)
singles				24 + 80 latches 96 latches 24 + 80 latches	304 (36.10%)
Hexes: sixteen mux 7:2				8 latches x 16	128 (15.20%)
CLB block TOTAL: 864 bits (22 unused)					842

Observing the numbers in the table 4.1, there are 1,327,104 customization bits associated to a latch or to a flip-flop. It is important to remind that the flip-flops located in the CLB are not accessible by the bitstream, but they are also sensitive to SEU and consequently must be protected. The XCV 300 device has 48 column and 32 rows given a total of 1536 CLB blocks, which is equivalent to the 1,327,104 latches (bits) from the CLB blocks. Analyzing the percentage of each type of memory cell in the whole

set of memory elements in the CLBs, the LUTs represent 7.4%, the flip-flops represent 0.46%, the customization bits in the CLB represent 6.36% and the general routing represents 82.9%. Based on these results, it looks that the major area overhead will come from the solution adopted to protect the routing and customization (~90%), which are implemented by standard 6 transistors latches.

The first solution that can be studied is to replace some or all of the latches in the FPGA by SEU hardened flip-flops. Many hardened memory cells were designed in the last years. However each one has different characteristics that can show more efficiency for some applications. Let's start the analysis by a standard memory cell composed of 6 transistors (figure 4.2). When a memory cell holds a value, it has two transistors in "on" state and two transistors in "off" state, consequently there are always two SEU sensitive nodes in the cell. When a particle strikes one of these nodes, the energy transferred by the particle can provoke a transistor to switch "on". This event will flip the value stored in the memory. If a resistor is inserted between the output of one of the inverters and the input of the other one, the signal can be delayed for such time avoiding the bit flip. The SEU tolerant memory cell protected by resistors [WEA87] was the first proposed solution in this matter, fig. 4.3. The decoupling resistor slows the regenerative feedback response of the cell. So the cell can discriminate between an upset caused by a voltage transient pulse and a real write signal. It provides a high silicon density, for example, the gate resistor can be built using two levels of polysilicon. The main drawbacks are temperature sensitive, performance vulnerability in low temperatures, and extra mask in the fabrication process for the gate resistor. However, a transistor controlled by the bulk can also implement the resistor avoiding the extra mask in the fabrication process. In this case, the gate resistor layout has a small impact in the circuit density.

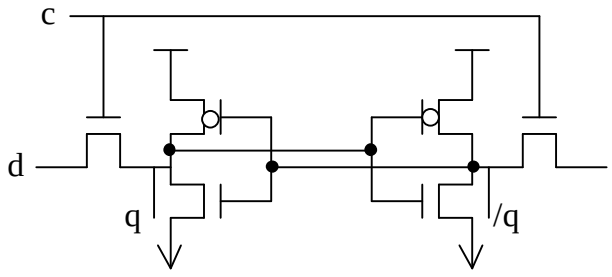


Figure 4.2 – Standard Memory Cell

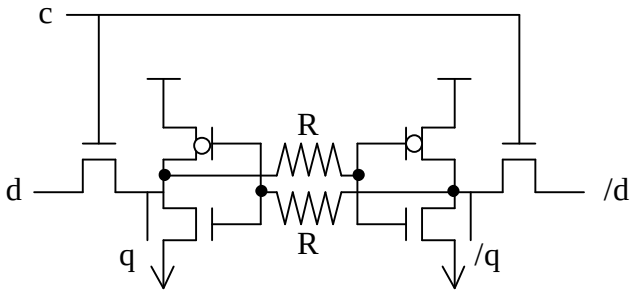


Figure 4.3 – Resistor Hardened Memory Cell

Memory cells can also be protected by an appropriate feedback devoted to restore the data when it is corrupted by an ion hit. The main problems are the placement of the extra transistors in the feedback in order to restore the upset and the influence of the new sensitive nodes. Examples of this method are IBM hardened memory cell [ROC92] in fig 4.4, HIT cell [VEL94] in fig. 4.5 and CANARIS memory cell [WIS93] in fig. 4.6. The main advantages of this method are temperature, voltage supply and technology process independence and good SEU immunity. The main drawback is silicon area overhead that is due to the extra transistors and their extra size. The IBM cell has 6 extra transistors, PA and PB are called data state control transistor, PC and PD are pass-transistors and PE and PF are cross-coupled transistors. The sensitive nodes are A, B, and C. The HIT cell has also 6 extra transistors placed in a feedback around the main storage cell. SEU testing presented in [VEL94] shows that the hardened HIT1 cell design is less sensitive at least by a factor of 10 than unhardened cell design. The CANARIS approach consists of a memory cell built with AND-NOR and OR-NAND gates that are SEU immune. Each logic gate has two outputs, one for the N-channel transistor and other for the P-channel transistors. Transistor M1 is sized to

be weak compared to the p-channel array and transistor M2 is sized to be weak compared to the n-channel array in such way that it can be restored the original value in the output if particle hit the sensitive nodes. The interesting aspect of this solution is that it can be applied even for the combinational and sequential logic when memory cells are implemented using the SEU immune combinational gates. Using this approach all the combinational part of the circuit can be grouped in complex logic functions where each one of these functions has two extras transistors dividing their outputs. For large complex logic gates, two extra transistors may not represent a high addition of area. However, due to the duplication of outputs the number of internal connections can increase according to the implementation architecture. The main drawback of CANARIS hardened memory cell is the long recovery time after upset.

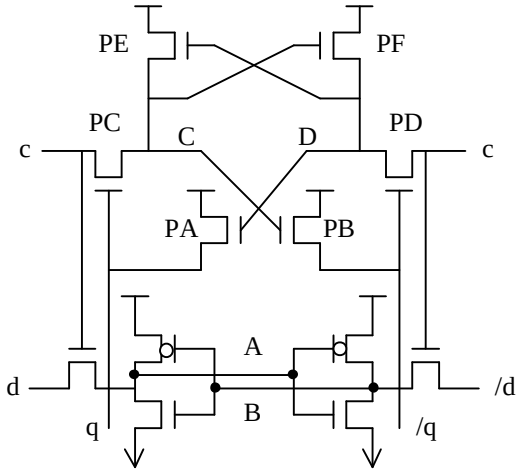


Figure 4.4 – IBM Hardened Memory Cell

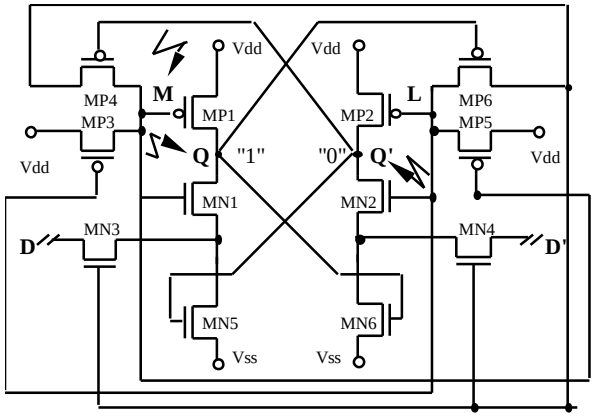
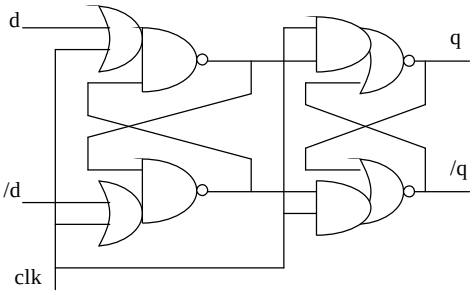
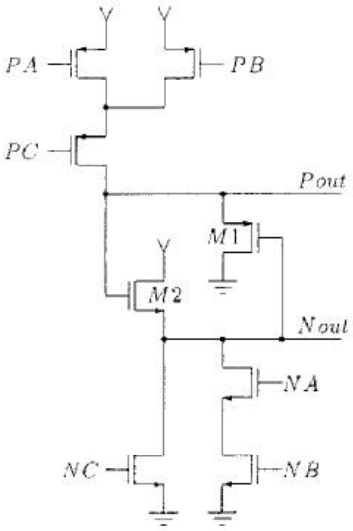


Figure 4.5 – HIT Hardened Memory Cell





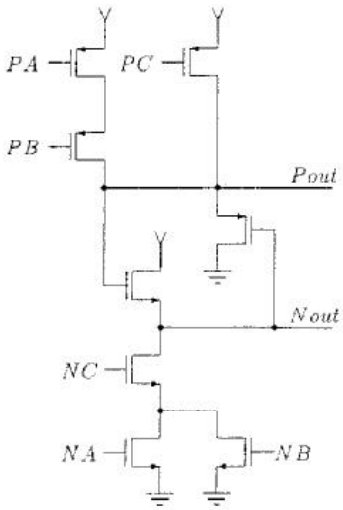


Figure 4.6 – Canaris Hardened Memory Cell

Another mitigation principle is to store the data in two different locations in the cell in such way that the corrupted part can be restored. Examples of this technique are DICE cell [CAN96] in fig. 4.7,

NASA cell [WHI91, LIU92] in fig. 4.8 and 4.9 respectively. The main advantages of this method are also temperature, voltage supply and technology process independence, good SEU immunity and high performance (read/write time). Dice cell consists in a symmetric structure of four CMOS inverters, where each inverter has the n-channel transistor and the p-channel transistor separately controlled by two adjacent nodes storing the same state. The 4 nodes of the DICE cell form a pair of latches in two alternate ways, depending on the stored logic value. One of the adjacent nodes controls the conduction state of the transistor connecting the current node to a power supply line, and the other node blocks on the complementary transistor of the inverter, isolating it from the opposite supply line. The NASA cells also store the information in two different places. This provides a redundancy and maintains a source of uncorrupted data after a SEU. The recover path is based on the use of weak and strong transistors. The weak transistor size is approximately 1/3 of the normal transistor size. The size of the weak feedback transistors is responsible for the recovery time.

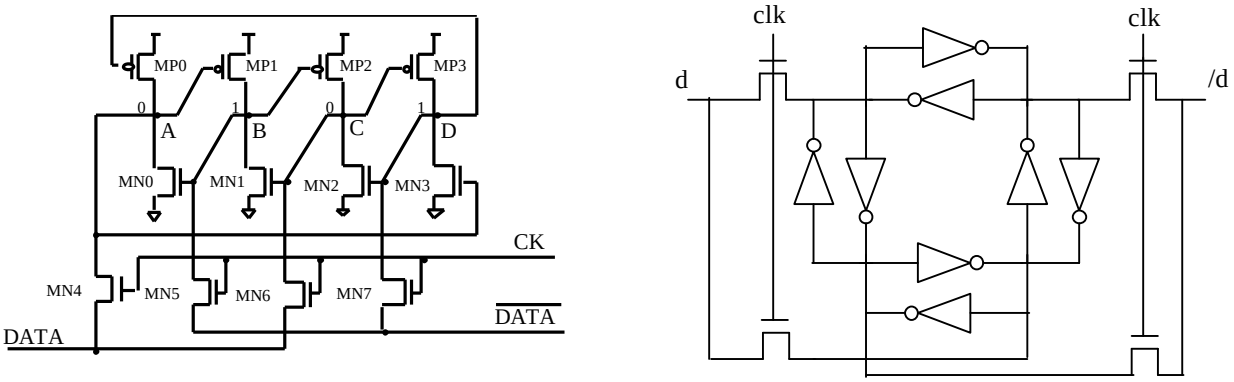


Figure 4.7 – DICE Hardened Memory Cell

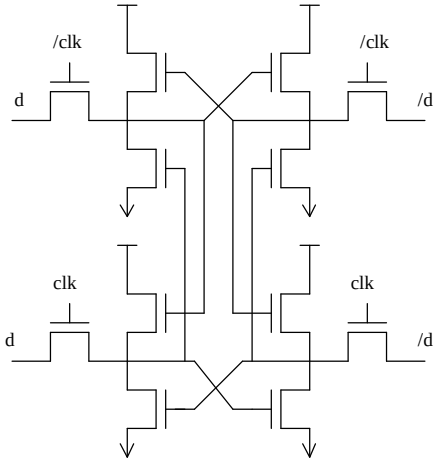


Figure 4.8 – NASA I Hardened Memory Cell

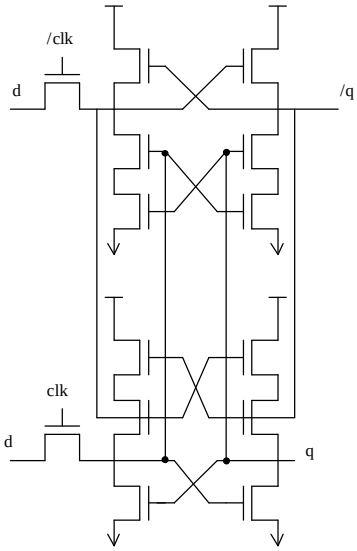


Figure 4.9 – NASA II Hardened Memory Cell

Triple Modular Redundancy (TMR) is also a considered technique to protect memory cells against SEU. The solution consists in the triplication of memory cells and to implement a voter to choose the correct stored value, fig. 4.10. However, this solution does not avoid the upset and an extra mechanism is

necessary to restore the correct value. In addition, it presents a large area overhead. In order to restore the corrected value, a solution using 3 voters with a feedback was proposed [CAR01a], [KAT01]. Fig. 4.11 and fig. 4.12 show two of these solutions.

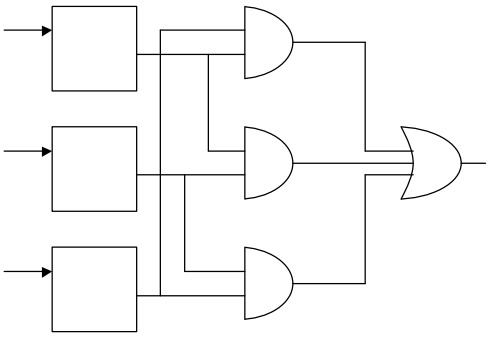


Figure 4.10 TMR Memory Cell with Single Voter

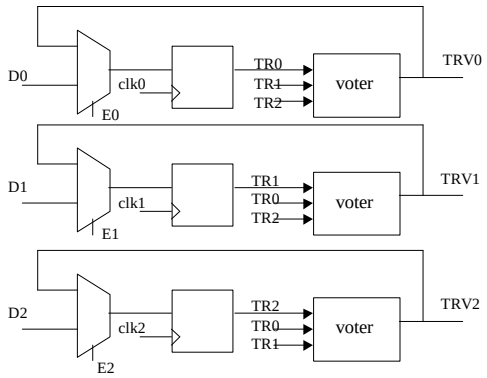


Figure 4.11 TMR Memory Cell with Triple Voters by Xilinx

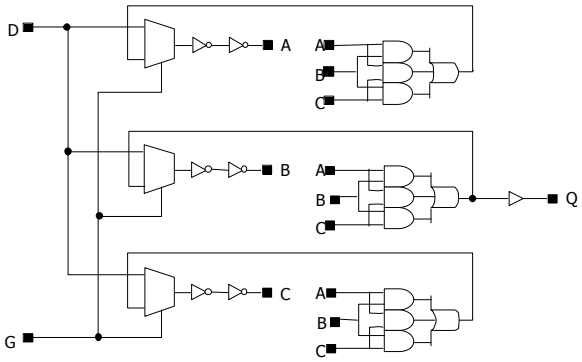


Figure 4.12 TMR Memory Cell with Triple Voter by Actel

Another SEU hardened memory solution is presented in [MAV00], fig. 4.13. The hardened memory cell contains nine level sensitive latches (U1 through U9), one majority gate (U10), and three inverters (U11 through U13). Each level sensitive latch is transparent (sample mode) when its clock input is high and is blocking (hold mode) when its clock input is low. When in sample mode, data appearing at the input D also appears at the output Q. When in hold mode, the data stored within the latch appears at the output Q and any data changes at the input D are blocked. Two level sensitive latches in tandem and clocked by complementary clock signals (such as U1 followed by U2) form an edge triggered D flip-flop. With the clock inversions, the D-Flip-Flops formed by (U1,U2), (U3,U4), and (U5,U6) are triggered on the falling edges of the clocks CLKA, CLKB, and CLKC, respectively. Each of these four clocks operates at a 25% duty factor and each is phased to the master clock. CLKA is high during the first half of cycle one of the master clock. CLKB is high during the second half of cycle one of the master clock. CLKC and CLKD are high during the first and second halves, respectively, of cycle two of the master clock. Thus a full cycle of the A, B, C, and D clocks occupies two cycles of the master clock. These clocks are actually quite easy to generate with simple circuitry presented in a later section. Controlling the fidelity of the four clocks is not a problem since the temporal sampling latch will operate correctly even in the presence of skew or overlaps.

The upset immunity of the circuit in fig. 4.13 is a consequence of two distinct parallelisms: (1) a spatial parallelism resulting from the three parallel circuit branches and (2) a temporal parallelism resulting from the unique clocking scheme. In addition, when implemented using DICE-based latches, the temporal latch can achieve immunity to multiple node cosmic ray strikes and, unlike any other SEE mitigation approach, it is immune to a second and third-order effect. Analyzing the SEU hardened robustness to MBU, the temporal latch, in its simplest form, is clearly immune to upset from any single cosmic ray striking a single circuit node (a first-order effect). This is also true for TMR-based latches and for DICE-based latches. Multiple node strikes (a second-order effect), although having much lower probabilities of occurrence, will surely cause upsets when such latches are fielded in an actual space environment.

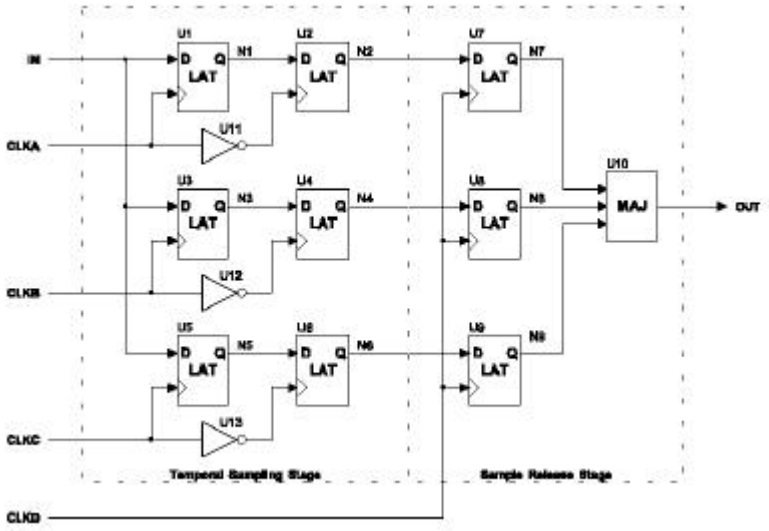


Figure 4.13 Temporal Sampling Latch With Sample and Release Stages.

Table 4.2 shows a summary of a comparison among them. The main characteristics used for the comparison is the number of transistor, the method, the SEU order effect, the ability to accumulate or not upsets and the SET immunity in combinational logic. For example, standard latches have a first order of susceptibility, in other words, they are upsettable by a single node strike. Some of them require multiple node strikes to upset cells such as TMR memory cells, DICE memory cell and simple temporal memory cells. Temporal latches build from DICE cell for example have a second and third order of susceptibility.

Table 4.2 – Summary of Hardened Memory Cells: main Advantages and Drawbacks

Hardened memory cell	Method	# trans.	SEU / MBU Order Effect Immunity			Accumulation of upsets	SET Immunity
			1st	2nd	3rd		
Resistor memory cell	Decoupling resistor	8	Yes	Yes	No	No	No
IBM memory cell	Restore feedback	12	Yes	Yes	No	No	No
NASA I and II memory cell	Physical redundancy	12	Yes	Yes	No	No	No
DICE memory cell	Physical redundancy	12	Yes	Yes	No	No	No
HIT memory cell	Restore feedback	12	Yes	Yes	No	No	No
CANARIS memory cell	Restore feedback	32	Yes	Yes	No	No	No
TMR with one voter without refreshing	Physical redundancy	38	Yes	No	Yes	Yes	Yes*

TMR with three voters with refreshing	Physical redundancy	90	Yes	No	Yes	No	Yes**
Temporal memory cell	Temporal and physical redundancy	80	Yes	No	Yes	Yes	Yes
Temporal memory cell with DICE	Temporal and physical redundancy	134	Yes	Yes	Yes	No	Yes

* It is SET immune if the combinational logic is also TMR.
** It is SET immune if the combinational logic is also TMR. The number of transistor is calculated with the multiplexor implemented by pass transistors.

Although these SEU hardened memory cells are efficient solutions applied nowadays in ASICs, they generate an undesirable impact in area and performance if used for example in the FPGA routing cells. The technique to protect FPGA can not be a simply application of the techniques current available for ICs because of their limitation for the programmable architecture. SEU hardened cells will at least duplicate all the routing area that represents almost 90% of the sensitive part of the chip and will decrease the connection speed, which can be very costly to a FPGA.

Another alternative for SEU mitigation technique is error detection and correction codes (EDAC). An example of EDAC is the Hamming code. In this case it is possible to protect group of memory cells as registers and embedded memories or group of routing customization bits. Hamming Code is an error-detecting and error-correcting binary code that can detect all single- and double-bit errors and correct all single-bit errors. This coding method is recommended for systems with low probabilities of multiple errors in a single data structure (e.g., only a single bit in error in a byte of data). The code satisfies the relation $2^k >= m+k+1$, where $m+k$ is the total number of bits in the coded word, m is the number of information bits in the original word, and k is the number of check bits in the coded word. Following this equation the Hamming code can correct all single-bit errors on n-bit words and detect double-bit errors when an overall parity check bit is used. According to the number of check bits, it is possible to correct more than a single-bit error.

The check bits are placed in the coded word at positions 1, 2, 4, …, $2^{(k-1)}$. For example, for 8-bit data, 4 check bits (p1, p2, p3, p4) are necessary so that the Hamming code is able to correct a single-bit error. Figure 4.14 exemplifies a 12-bit coded word (m=8 and k=4) with the check bits p1, p2, p3 and p4 located at positions 1, 2, 4 and 8 respectively. The check bits are able to inform the position of the error.

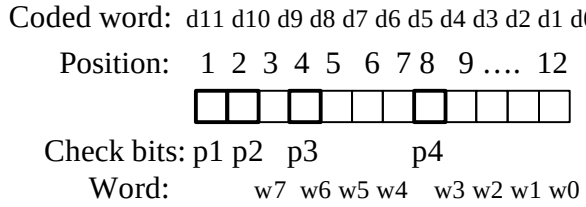


Figure 4.14 – Hamming code 12-bit word and the check bits

The check bit p1 creates even parity for the bit group {1, 3, 5, 7, 9, 11}. The check bit p2 creates even parity for the bit group {2, 3, 6, 7, 10, 11}. Similarly, p3 creates an even parity for the bit group {4, 5, 6, 7, 12}. Finally, the check bit p4 creates even parity for the bit group {8, 9, 10, 11, 12}, as shown in figure 4.15.

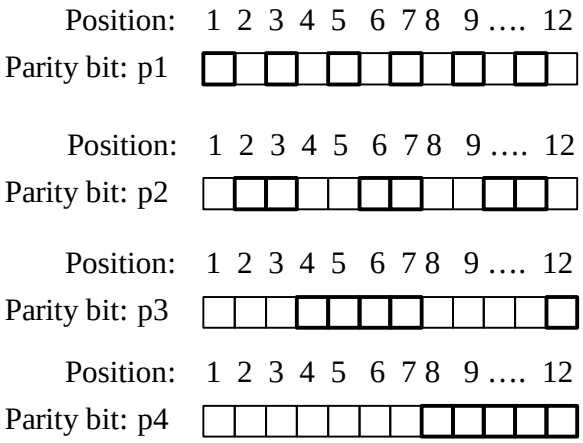


Figure 4.15 –Hamming code check bits generation

Examples of implementations of the Hamming code in registers and in memory were presented in figure 2.18. The hamming code can correct more than one single upset if more coded bits are used. The necessity of using hamming code with the capability of double error correction will be investigated. The overhead in area will be analyzed as the necessity of refreshing.

The application of hamming code in the FPGA structure will be study. The implementation can be directly as a memory implementation or as a register, for example, if implemented in the embedded memory or in the lookup table. More complex implementations will also be studied using the routing architecture. Some bits in the bitstream can be grouped and codified to be used in the routing in special situations to avoid upsets.

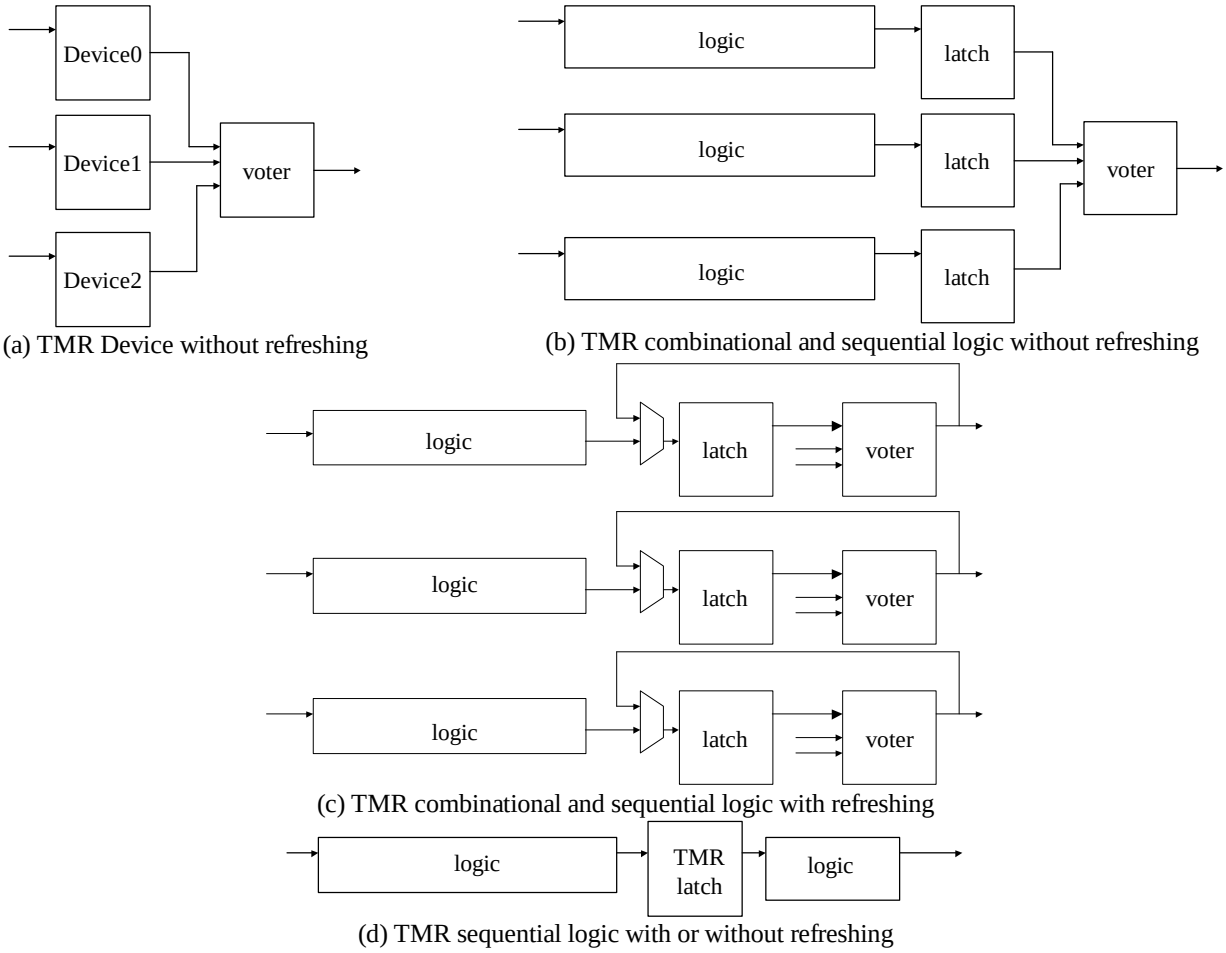
The triple modular redundancy (TMR) is another choice for SEU mitigation. There are many TMR approaches. Each implementation can be restricted in some aspects such as protection of both combination and sequential logic and ability to correct the upset. The system requirements and the architecture must be analyzed in order to correctly choose the most convenient approach.

The first one is a triplication of the entire device, fig. 4.16(a). This approach uses a voter as a fourth component in the board. It needs extra connections and it presents a large area overhead. If an error occurs in one of the three devices, the voter will choose the correct value. It protects both combinational and sequential logic against upsets. However, in order to avoid accumulation of faults in the design, an additional mechanism is necessary to correct the upset in each device before a next one happens. This scheme can be used in ASIC but it can not be used in SRAM-based FPGAs for example.

Another TMR scheme triplicates all the internal paths as described in fig. 3.16(b). A single component implements this methodology and there will be many voters as necessary. This method also protects both the combinational and sequential logic upsets. There is no mechanism of refreshing (upset correction). The upsets in the latches must be corrected by an extra logic in somehow in order to avoid accumulation. An evolution of this approach is presented in fig. 3.16(c). There is an extra logic responsible to load the correct voted value to the latches. The load frequency (refreshing) can be set by the multiplexor control signal.

Finally, the TMR approach can protect just the sequential logic by replacing only the latches of flip-flops by their TMR version, fig. 3.16(d). In this case, the solution is the same of using the TMR latch presented in figure 4.10. Either the TMR cell with refreshing or without refreshing can be used in this solution. Upsets in the combinational logic are not protected in this case and if the transient error is

captured by a TMR latch, the upset will be stored in it until they load a new value from the design. Table 4.3 summarizes the TMR approaches.



(a) TMR Device without refreshing

(b) TMR combinational and sequential logic without refreshing

(c) TMR combinational and sequential logic with refreshing

(d) TMR sequential logic with or without refreshing

Figure 4.16 – Example of TMR structures

Table 4.3 – Summary of TMR approaches: main Advantages and Drawbacks

TMR Approach	SEU / MBU Order Effect Immunity			Accumulation of upsets	SET Immunity
	1st	2nd	3rd		
TMR Device without refreshing	Yes	Yes	Yes	Yes	Yes*
TMR sequential logic without refreshing	See table 3.1, according to the TMR latch			Yes	No
TMR sequential with refreshing	See table 3.1, according to the TMR latch			No	No
TMR combinational and sequential logic without refreshing	Yes	No	Yes	Yes	Yes
TMR combinational and sequential logic with refreshing	Yes	No	Yes	No	Yes

As mentioned before, TMR is a very costly solution. It may be used in very few isolated points but definitely must be avoided in the protection of the whole architecture by itself. Each technique may be

more efficient for one kind of topology than other. The hamming code and the TMR technique were already tested in the 8051 architecture. Results comparing the hamming code and TMR are presented in [HEN02]. In the next step, the SEU hardened memory cells will also be tested in the 8051 architecture. Final conclusions will be presented in the final thesis document. These results will help in the justification of some techniques for the FPGA architecture. Table 4.4 presents a general comparison between these mainly techniques: hardened memory cells, hamming code and TMR.

Table 4.4 - SEU mitigation techniques

SEU Mitigation Technique	SEU Tolerant Memory cells	Hamming Code	TMR
Area	Usually it doubles the area of each memory cell. It is strongly layout and transistor size dependent	It depends on the number of bits to be protected. It has an extra sequential and combinational logic	It presents a little more than 3 times of area overhead because of the voter.
Performance	The performance is not affected if the extra transistors or resistors (path delay) work only when the cell is on hold.	The performance can be affected by the coding and decoding blocks.	The performance is strongly affected. The only source of delay is the voter.
Error correction	It avoids the error by a delay in the memory loop.	Normally it corrects one single upset per word, but in order to refresh the stored value it is necessary an extra path (scrubbing rate)	It does not correct the upsets. The upsets will accumulate if there is no extra logic.
Multiple Upset	Robust to multiple upsets as each cell protects itself.	Not efficient for multiple upsets in the same coded word. But efficient multiple upsets in different parts of the circuit.	It can be robust to multiple upsets in different parts of the circuit but not in the same TMR signal.
Technology	It can use some special features	Completely compatible	Completely compatible
COTS	Require design development	It can be designed in high level languages	It can be designed in high level languages

The main question for the Thesis is: Which is the best SEU mitigation solution for a SRAM-based FPGA? Is it better to protect each part with a different solution or it is better to use the same solution for all memory cells presented in the matrix? The answer for the questions requires design, simulation and fault injection test. The radiation ground test depends on the fabrication of the new FPGA chip. The solution is not a simple application of the methods available in the literature because of the impacts that they bring in area, performance and power consumption. It is important to develop new techniques based on the methods already tested in order to improve results. The SRAM programmable routing cells are the main concern because it takes 80% of the FPGA programmable cells. A solution that can duplicate or triplicate this logic is not very attractive in final area and cost of this FPGA.

The Virtex CLB tile, figure 4.17, is composed of routing (switch matrix, multiplexors and connection segments) and the CLB. In the CLB, the storage cells can be classified at least in 3 categories:

the lookup tables (64 latches), the 4 flip-flops and the customization cells (42 latches). Each one has special characteristics that may limit the choice of the SEU mitigation technique. In the other hand, all must have the same feature that is the re-configurability by the bitstream. The bitstream must be loaded and partially re-loaded in the programmable SRAM cells at any time. So the SEU mitigation solution must agree with all the features possibilities.

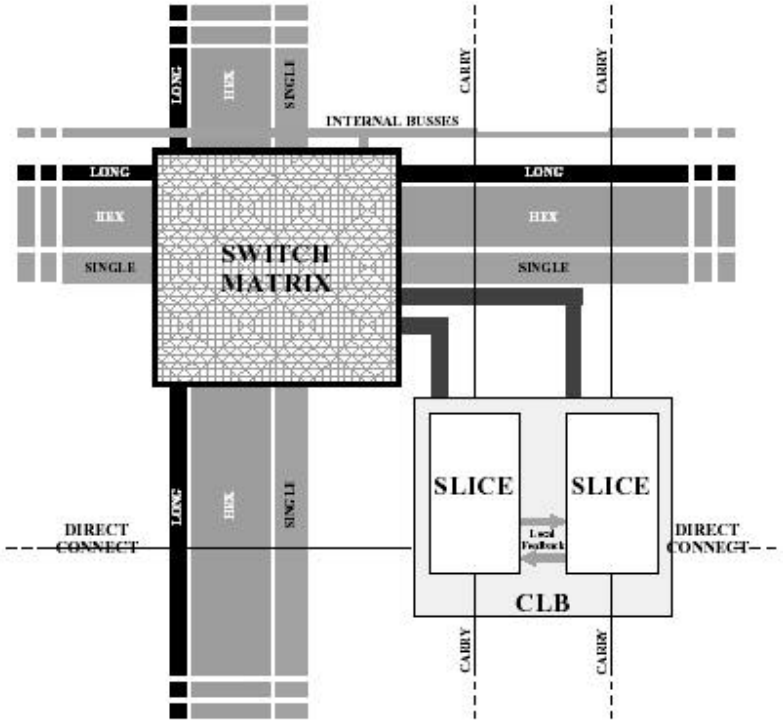


Figure 4.17 – Virtex CLB Tile Schematic

As mentioned before, the routing programmable cells compromise 80% of the CLB tile and consequently is the main concern. The routing is composed of the switch box and input and output multiplexors. The switch block is a programmable interconnect block that is found at the intersection of each horizontal and vertical routing channel. The flexibility of a switch matrix (Fs) is defined as the number of connections to each incoming track to a number of outgoing tracks (figure 4.22). Clearly, the flexibility of each switch block is the key to the overall flexibility and routability of the device. The number of switches required in the matrix is defined as 2xFsxW, where W is the number of connections directions as represented in figure 4.23. Figure 4.24 shows two examples of connection elements, the pass transistor and the tri-state buffer. Since the transistors in the switch block add capacitance loading to each track, the switch block has a significant effect on the speed of each routable connection, and hence the speed of the FPGA as a whole. In addition, since such a large portion of an FPGA is devoted to routing, the chip area required by each switch block will have a large effect on the achievable logic density of the device. Thus, the design of a efficient hardened switch block is of the up-most importance.

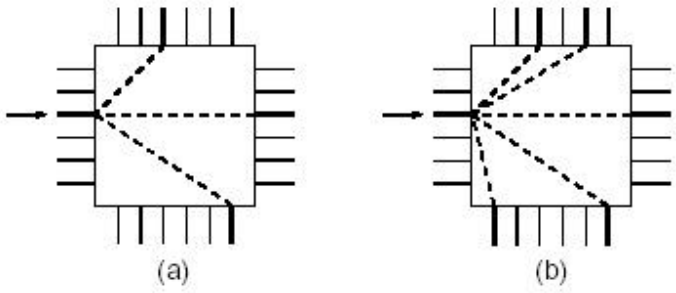


Figure 4.22 – Two examples of switch matrices with a different flexibility (a) Fs=3 (b) Fs=5. [DEP98]

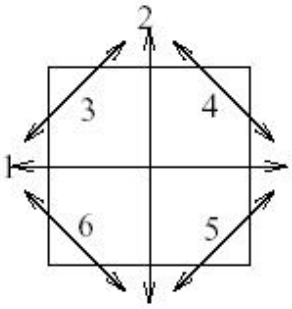
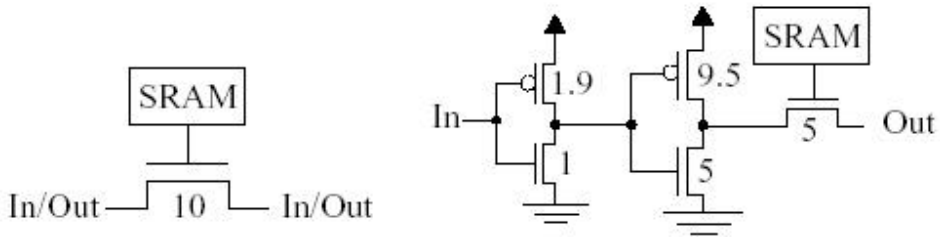


Figure 4.23 – Direction of the Connections in a Switch Matrix (W=6)



(a) Pass transistor (b) Tri-state buffer

Figure 4.24 – Routing Switch Connections [BET99]

The switch matrix connects the single and the hex wires. A representation of this matrix is illustrated in figure 4.25. The hex wires are also connected only with other hex lines by multiplexors, as represented in figure 4.26. The input and output multiplexors located in the CLB tile are responsible by the connections between the incoming wires (switch matrix or hex connections) to the CLB slices and the output signals from the CLB slices to the outgoing wires (switch matrix or hex connections), respectively. The representation is in figure 4.27.

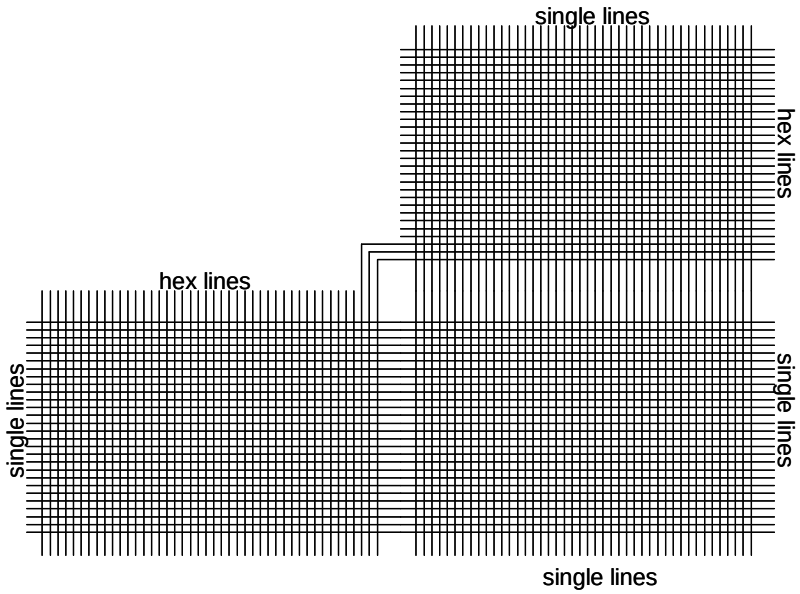


Figure 4.25 – Switch matrix connects the Single and Hex Segments

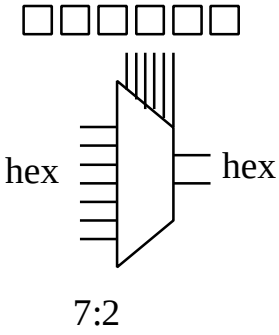


Figure 4.26 – Hex line connections in the routing

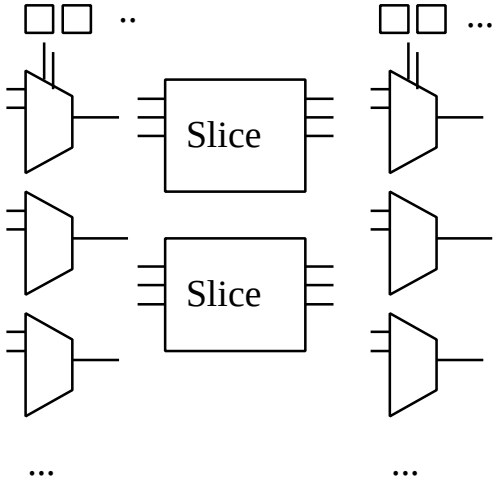


Figure 4.27 – Input and Output multiplexors in the routing

Two problems related to the routing will be address in the Thesis:

?? Investigation of a new routing topology able to avoid propagation of upsets in different redundant parts of the TMR design. This solution must be easily applied by software in the placement and routing, when the TMR technique is used, to avoid undesirable connections between signals from different redundant part. This solution can be in the combinations of connections (change in the switch matrix flexibility) or in the topology. In the second case, it is necessary mask changes. This solution is attractive for the fabricant point of view because it reduces the cost in the design of a completely new FPGA architecture.

?? Design of a new routing topology completely SEU hardened for a SEU hardened FPGA, where the TMR solution in the high level will not be applied. This solution requires changes in the switch matrix design, in the multiplexors connections and in the SRAM programmable cells. One solution, for example, can focus on the bitstream environment. The routing bits in the bitstream can be encoded and decoded as a group of words. In this case the FPGA can be observed as a matrix composed of arrays of bits placed in columns and rows. And it may have a solution that can observe and correct the upset of the FPGA by analyzing, encoding and decoding these arrays. Another solution is to change the routing matrix topologies to turn the arrange SEU immune. The solution will be tested by simulation and fault injection.

In the CLB, the storage cells can be classified at least in 3 categories: the lookup tables (64 latches), the 4 flip-flops and the customization cells (42 latches). Figure 4.18 shows the LUT schematic. LUTs are one of the special cases because their cells can operate as shifter registers, figure 4.19 and memories, figure 4.20. For this reason they must be interpreted separately.

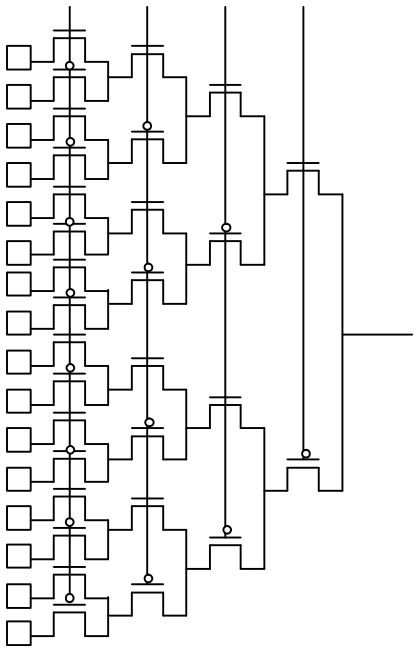


Figure 4.18 – 4-input LUT Schematic

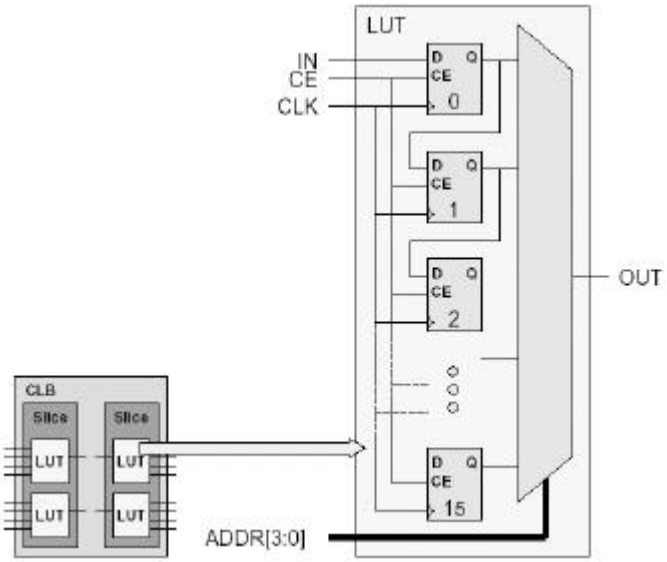


Figure 4.19 – LUT Configured as Shift Register

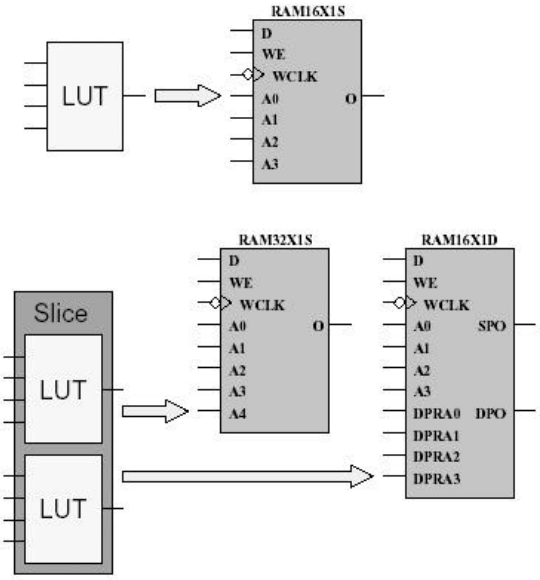


Figure 4.20 – LUT Configured as Memory

The area used by these programmable cells are relatively low compared to the routing programmable cells, consequently SEU mitigation solutions used in ASIC may not have a big impact in the area and performance. Different solutions can be investigated. One of them is to replace each memory cell composed of 6 transistor plus buffer by a SEU hardened cell. Another solution is to protect the full LUT by EDAC code or TMR. In addition, a completely original solution can be proposed based on the topology and redundancy.

Flip-flops can be programmed by many different ways and consequently they can be analyzed in a separately way too. Figure 4.19 show the possibilities of flip-flop configuration in the CLB.

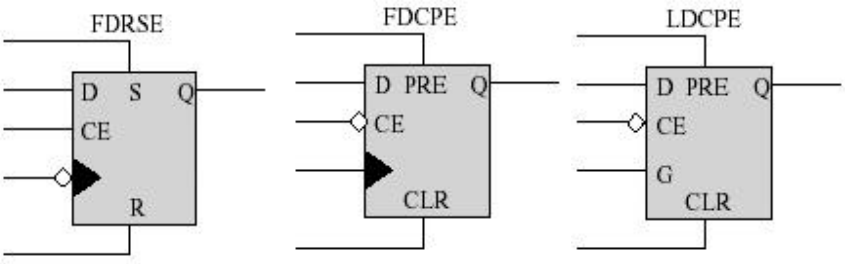
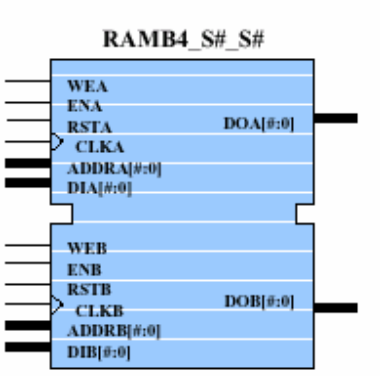


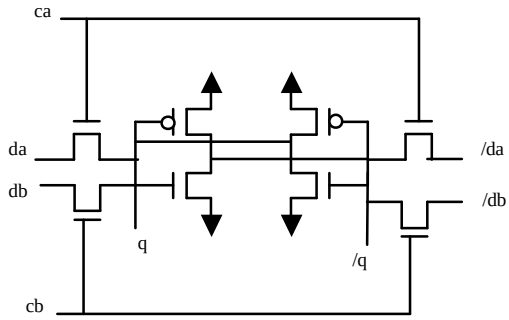
Figure 4.21 – Examples of CLB flip-flop Configuration

The work will investigate new CLB architecture focused to the hardened approach. A SEU hardened CLB can have a combination of techniques such as hardened memory cells and hamming code without change the functionality of the CLB. The solutions will be tested by simulation and fault injection.

The block RAM (BRAM) module can also be studied separately. Figure 4.28 show the SRAM representation and the schematic of each memory cell. Solutions based on EDAC will be studied. But there is always limitation using the available techniques. For example, if Hamming code is used in the BRAM, it is necessary to provide the scrubbing to avoid accumulation of upsets. The scrubbing can reduce the BRAM to a single port (user view). In addition, MBU must be also investigated. If the memory cells are replaced by SEU hardened memory cells, by definition it is not necessary scrubbing, but MBU must also be taken into account due to the small dimensions of the SEU hardened cell.



(a) BRAM Representation



(b) BRAM Memory cell

Figure 4.28 – Embedded Block RAMs (BRAM)

It is evident that the most convenient and attractive solution to SEU mitigate a SRAM based FPGA is not the simple application of the methods available in the market. The trade-offs between area overhead, performance and power must be taken into account in all new projects. In summary, there will be two solutions. One it will be almost the same of the standard FPGA with some small changes in the routing in order to improve the reliability of the TMR in high level description designs implemented in the SRAM-based FPGAs. The other one will be a completely new SEU hardened FPGA architecture with modifications in the routing and in the CLB logic without compromise the functionality.

5 Conclusions

This thesis proposed the study and development of SEU mitigation techniques in programmable architectures such SRAM based FPGAs. The choice of SRAM based FPGAs is due to their high applicability in apace applications. Because they are reprogrammable, designs can be updated or corrected after launch. The consideration of using FPGA to space applications are fairly recent and there is a lot of work to be done in this area so far.

The problem of how to protect SRAM based FPGAs was addressed in this proposal. The Virtex FPGA from Xilinx uses the TMR technique to protect the designs against SEU. Fault injection analysis presented in [LIM01b] showed that there are a few upset bits in the bitstream related to the routing that can provoke an error in the TMR design. This limitation is due to the switch matrix that can connect two signals from different redundant part when a programmable cell is upset. Based on the references presented in chapter 2, there is no total efficient solution for SRAM based FPGAs that can assure 100% of reliability in all conditions for SEU. This thesis has the goal to investigate the techniques used nowadays and to propose improvements in order to increase reliability.

The goal is to adapted techniques that were already developed to ASICS to the programmable logic architecture by finding the best tradeoff among area overhead, performance penalties, single and multiple upsets correction, process technology and implementation cost. But it is evident that the best solution is not a simple application of the actual techniques on the SRAM based FPGA, but a combination of them with topology modifications.

The first structure to be analyzed and modified is the routing because its cells correspond around to 80% of the total sensitive cells of the FPGA. A first solution aims to improve the TMR already proposed by Xilinx. This solution will be almost the same of the standard FPGA with some small changes in the routing in order to improve the reliability of the TMR in high level description designs implemented in the SRAM-based FPGAs avoiding connections between signals located in different redundant parts of the TMR. The second project proposes a completely change in the architecture. It will be a new SEU hardened FPGA architecture with modifications in the routing and in the CLB logic without compromise the functionality.

References

[ACT00] ACTEL. Using Synplify to Design in Actel Radiation-Hardened FPGAs. Application Note. Disponivel por www em http://www.actel.com/appnotes/SynplifyRH.pdf (Nov. 2001)

[ACT01] ACTEL. RT54SX-S Rad-Tolerant FPGAs for Space Applications. Data Sheet. Disponivel por www em http://www.actel.com/docs/datasheets/RT54SX-S.pdf (Nov. 2001)

[ALF98] ALFKE, Peter; PADOVANI, Rick. Radiation Tolerance on High-Density FPGAs. In: http://www.xilinx.com (Oct. 1998).

[ALT01] ALTERA CORPORATION. Data Book 2001. Disponível por WWW em http://www.altera.com (nov. 1998).

[ANG00] ANGHEL, Lorena. LES LIMITES TECHNOLOGIQUES DU SILICIUM ET TOLERANCE AUX FAUTES, These, 2000.

[BAR01] BARTH, Janet. Living with the Start Project from NASA.

[BAR97] BARTH, Janet. Radiation Environment. In: IEEE NSREC Short Course, July 21, 1997. http://flick.gsfc.nasa.gov/radhome/ RPO_slides.htm.

[BAU01] BAUMANN, R. C. Silicon Amnesia - Terrestrial Effects. In : Radecs 2001.

[BES93] BESSOT, Denis. Conception de Deux Points Memoire Statiques CMOS durcis Contre L'effet des Aleas Logiques Provoques par L'environment Radiatif Spatial. These. INPG. Novembre, 1993.

[BET99] BETZ, Vaughn; ROSE, Jonathan Rose, FPGA Routing Architecture: Segmentation and Buffering to Optimize Speed and Density. In FPGA99. Disponivel por www em: http://www.eecg.toronto.edu/~vaughn/papers/fpga99a.pdf

[BOR01] BOREL, J.; GAUTIER, J.; GASIOT, J. Silicon Redemption. In: RADECS 2001.

[BRY98] O'BRYAN, Martha; LABEL, Kenneth A.; REED, Robert A; BARTH, Janet; SEIDLECK, C.; MARSHALL, Paul; MARSHALL, C.; CARTS, Martin. Single Event Effect and Radiation Damage Results For Candidate Spacecraft. In: IEEE NSREC Conference, 1998.

[CAL96a] CALIN, T.; VELAZCO, R.; NICOLAIDIS, M.; MOSS, S; LAMONDIERE, S. D.; TRAN, V. T.; KOGA, R. Topology-Related Upset Mechanisms in Design Hardened Storage Cells. In: NSREC Conference, 1996.

[CAL96b] CALIN, T.; NICOLAIDIS, M.; VELAZCO, R. Upset Hardened Memory Design for Submicron CMOS Technology. In: IEEE Transactions on Nuclear Science. VOL. 43, NO. 6, December 1996.

[CAM99] CARMICHAEL, C.; FULLER, E.; BLAIN, P.; CAFFREY, M. SEU Mitigation Techniques for Virtex® FPGAs in Space Applications. In: http://www.xilinx.com (Sep. 1999).

[CAN95] CANARIS, J.; WHITAKER, S. Circuit techniques for the radiation environment of space,"in IEEE 1995 Custom Integrated Circuits Conference, pp. 5.4.1{5.4.4, 1995.

[CAR00] CARMICHAEL, C., CAFFREY, M., SALAZAR, A., Correcting Single-Event Upsets Through Virtex® Partial Configuration, Xilinx Application Notes 216, v1.0, Jun. 2000.

[CAR01a] CARMICHAEL, C., FULLER, E., FABULA, J., LIMA, F. Proton Testing of SEU Mitigation Methods for the *Virtex*® FPGA. In: MAPLD, 2001.

[CAR01] CARMICHAEL, C., Triple Module Redundancy Design Techniques for Virtex® Series FPGA, In: Xilinx Application Notes 197, v1.0, Mar. 2000

[CAR96] CARRO, L.; PEREIRA, G.; SUZIN, A. Prototyping and Reengineering of Microcontroller-Based Systems. In: IEEE Rapid Systems Prototyping Workshop. Proceedings… June 1996.

[CHI98] CHIP EXPRESS CORPORATION. Data Sheet. Disponível por WWW em http://www.chipexpress.com (nov. 1998).

[COL01] COLINGE, Jean-Pierre. Silicon-on-Insulator Technology: Overview and Device Physics In: NSREC Short Course, 2001.

[COT00] COTA, E.; CARRO, L.; LUBASZEWSKI, M.; VELAZCO, R.; REZGUI, S. Synthesis of 8051-like Microcontroller Tolerant to Transient Faults. In: 1st IEEE Latin America Test Workshop (LATW), Brazil, 2000.

[COT00] COTA, E.; LIMA, F.; REZGUI, S.; CARRO, L.; VELAZCO, R.; LUBASZEWSKI, M.; REIS, R. "Synthesis of an 8051-like Micro-Controller Tolerant to Transient Faults", JETTA, 2000.

[DAC01] LEWIS, B.; TANURHAN, Y.; OR-BACH, Z. GARVERICK, T., When) Will FPGAs kill ASICs? In Design Automation Conference, Las Vegas, 2001.

[DEL96] DELONG, T., JOHNSON, B. PROFETA, J., "A Fault Injection Technique for VHDL Behavioral Models", IEEE Design and Test of Computers, 1996.

[DEN00] Martin Dentan. RADIATION EFFECTS ON ELECTRONIC COMPONENTS AND CIRCUITS. In: CERN Training Course. April 11, 2000. (http://atlas.web.cern.ch/Atlas/GROUPS/FRONTEND /radhard.htm).

[DEP98] J. Depreitere, H. Van Marck and J. Van Campenhout, Evaluation of FPGA Switch Matrices using a Monte Carlo Approach. In: DCE, 1998. Disponivel por www em: http://www.elis.rug.ac.be/~jdp/DCE98.pdf

[DON93] DONG, S. K. et al. Two channel Routing Algorithms for Quick Customized Logic. In: EDAC, 1993. Proceedings… Los Alamitos : IEEE Computer Society, 1993. p. 122-126.

[EAS01] EASIC CORPORATION. Products Data Book. http://www.easic.com/

[FUL00] FULLER, E., CAFFREY, M., SALAZAR, A., CARMICHAEL, C., FABULA, J., "Radiation Testing Update, SEU Mitigation, and Availability Analysis of the *Virtex*® FPGA for Space Re-configurable Computing", NSREC, Jul. 2000.

[FUL99] FULLER, E.; BLAIN, P.; CAFFREY, M.; CARMICHAEL, C. Radiation Test Results of the Virtex® FPGA and ZBT SRAM for Space Based Reconfigurable Computing. In: http://www.xilinx.com (Sep. 1999).

[GAI98] GAISLER, J., A fault-tolerant microprocessor based on the PARC V8 architecture. In: Seventh Swedish Workshop on Computer Systems Architecture. Disponivel por www em http://www.ce.chalmers.se/~dsa98/camera/c1_2.pdf

[GUN00] GUNTZEL, Jose. Functional Timing Analysis of VLSI Circuits Containing Complex Gates Tese de Doutorado. Instituto de Informatica, 2001.

[HAS89] HASS, K. J.; TREECE, R.K.; GIDDINGS, A. E., A Radiation-Hardened 16/32-Bit Microprocessor, IEEE Transactions on Nuclear Science, 36(6):2252-2257, December 1989.

[HAS99] HASS, J., Probabilistic Estimates of Upset Caused by Single Event Transients, In: 8th NASA Symposium on VLSI Design, 1999

[HASS98] HASS, J.; GAMBLES, J.; WALKER, B.' ZAMPAGLIONE, M., Mitigating Single Event Upsets From Combinational Logic, In: 7th NASA Symposium on VLSI Design, 1998

[HEN02] HENTSCHKE, Renato; MARQUES, Felipe. Caracterização de Dois Circuitos VLSI Protegidos Contra Radiação. Relatorio de Pesquisa. Instituto de Informatica, 2002

[HOP99] HOPKIN, Vince. Programmable Device or Gate Array? Disponível por WWW em http://www.isdmag.com (jan., 1999).

[IBM00] IBM. SOI Technology: IBM's Next Advance in Chip Design. In: http://www.ibm.com (Jan. 2000).

[INT94] INTEL. Embedded Micro-controllers, Intel Datasheet, 1994.

[JOH00] JOHNSTON, A. "Scaling and Technology Issues for Soft Error Rates", 4th Annual Research Conference on Reliability, Stanford University, Oct. 2000.

[KAT01] KATZ, R. et al. An SEU-Hard flip-Flop for Antifuse FPGAs, In: MAPLD, 2001.

[KAT94] KATZ, R.; BARTO, R.; McKERRACHER, P.; CARKHUFF, B; KOGA, R. SEU Hardening of Field Programmable Gate Arrays (FPGAs) for Space Application and device characterization. In: NSREC Conference, 1994.

[KAT97] KATZ, R.; LABEL, K.; WANG, J.; CRONQUIST, B.; KOGA, R.; PENZIN, S.; SWIFT, G. Radiation Effects on Current Field Programmable Technologies. In: NSREC Conference, 1997.

[KAT98] KATZ, R.; WANG, J.; LABEL, K.; McCOLLUM, J.; BROWN, R.; REED, R.; CRONQUIST, B.; CRAIN, S.; SCOTT, T.; PAOLINI, W.; SIN, B. Current Radiation Issues for Programmable Elements and Devices. In: NSREC Conference, 1998.

[LAB99] LABEL, K. et al. Commercial Microelectronics Technologies for Applications in the Satellite Radiation Environment. In: http://flick.gsfc.nasa.gov/ radhome.htm (Nov. 1999).

[LEA91] LEAVY, J.; HO_MAN, L. F.; SHOVAN, R. W.; JOHNSON, M. T., Upset due to single particle caused propagated transient in a bulk CMOS microprocessor," IEEE Trans. On Nuclear Science, vol. 38, pp. 1493{1499, Dec. 1991

[LER01] LERAY, J. Earth and Space Single-Events in Present and Future Electronics, *Short Course, RADECS 2001*

[LI 84] K. W. Li, J. R. Armstrong, J. G. Tront, An HDL simulation of the effects of Single Event Upsets on microprocessor program flow, IEEE Trans on Nuclear Science Vol. NS 31, N° 6, pp. 1679-1681, Dec. 1984.

[LIM00a] LIMA, F., REZGUI, S., COTA, E., CARRO, L., LUBASZEWSKI, M., VELAZCO, R., REIS, Ricardo, "Designing and Testing a Radiation Hardened 8051-like Micro-controller", IN: MAPLD Conference, Sept. 2000.

[LIM00b] LIMA, Fernanda G.; COTA, Érika; CARRO, Luigi; LUBASZEWSKI, Marcelo; REIS, Ricardo; VELAZCO, Raoul; REZGUI, Sana., "Designing a Radiation Hardened 8051-like Micro-controller", In: SBCCI Conference, 2000.

[LIM01a] LIMA, F.; REZGUI, S.; CARRO, L.; VELAZCO, R.; REIS, R., "On the use of VHDL Simulation and Emulation to Derive Error Rates", In: Radiation and its Effects on Components and Systems (RADECS), Sept. 2001.

[LIM01b] LIMA, F.; CARMICHAEL, C.; FABULA, J.; PADOVANI, R.; REIS, R., "A Fault Injection Analysis of Virtex® FPGA TMR Design Methodology", In: Radiation and its Effects on Components and Systems (RADECS), Sept. 2001.

[LIM02a] LIMA, F.; CARRO, L.; VELAZCO, R.; REIS, R., "Injecting Multiple Upsets in a SEU tolerant 8051 Micro-controller", IN: LATW, Feb. 2002.

[LIM02b] LIMA, F.; CARMICHAEL, C.; FABULA, J.; PADOVANI, R.; REIS, R., "Proton Testing of a TMR 8051 micro-controller in Virtex? FPGA", In: Submitted to NSERC, Jun. 2002.

[LIU92] LIU, M. N.; WHITAKER, S., Low power SEU immune CMOS memory circuits, IEEE Trans. on Nuclear Science, vol. 39, pp. 1679{1684, Dec. 1992.

[LIU92] LIU, M.; WHITAKER, S. Low Power SEU immune CMOS Memory Circuits. In: NSREC Workshop. 1992.

[LSI01] LSI LOGIC CORPORATION. Product Data Book. Disponível por WWW em http://206.204.107.130/techlib/marketing_docs/asic/liquid_logic_pb.pdf

[LUM98] LUM, G.; MARTIN, L. Single Event Effects Testing of Xilinx FPGAs. In: http://www.xilinx.com (Oct. 1998)

[MAV01] MAVIS, D.; EATON, P., SEU and SET Mitigation Techniques for FPGA Circuit and Configuration Bit Storage Design, MAPLD 2000

[MAX01] MAXWELL TECHNOLOGIES. Datasheet. Disponivel por www em http://www.spaceelectronics.com/documentation/datasheets.html (Nov. 2001)

[MEG01] MEGGYESI, Zoltan; VAN DER BIJ, Erik; MCLAREN, Robert., FPGA Design in the Presence of Single Event Upsets, CERN, 2001.

[MOO75] MOORE, G. E., "Progress in Digital Integrated Electronics", *Technical Digest of the IEEE IEDM 1975*.

[MUS01] MUSSEAU, Olivier; FERLET-CAVROIS, Veronique. Silicon-on-Insulator Technology: Radiation Effects, In: NSREC Short Course, 2001.

[NAS02] NASA. NASA Homepage. Disponivel por www em: http://radhome.gsfc.nasa.gov/top.htm

[NOR96] NORMAND, Eugene. Single Event Upset at Ground Level. In: IEEE Transactions on Nuclear Science. VOL. 43, NO. 6, December 1996.

[OHL97] OHLSSON, M.; DYREKLEV, P.; JOHANSSON, K.; ALFKE, P. Neutron Single Event Upsets in SRAM based FPGAs. In: http://www.xilinx.com (1997).

[PET80] PETERSON, W. Wesley. Error-correcting codes. Ed. 2.ed. Cambridge : The mit Press, 1980. 560 p. ISBN 0262160390.

[PLA00] PLANK, James S. A Tutorial on Reed-Solomon Coding for Fault-Tolerance in RAID-like Systems. Department of Computer Science, University of Tennessee. In: http:// www.cs.utk.edu/~plank/plank/papers/SPE-9-97.html

[RAB96] RABAEY, Jan. Digital Integrated Circuits - A Design Perspective. Upper Saddle River : Prentice Hall, 1996. 702 p.

[REE97] REED, R.; et al, "Heavy Ion and Proton Induced Single Event Multiple Upsets", IEEE Nuclear and Space Radiation Effects Conference (NSREC), July 1997.

[REZ00] REZGUI, S.; VELAZCO, R.; ECOFFET, R.; RODRIGUEZ, S; MINGO, J. R. Estimating Error Rates in Processor-Based Architectures. In: RADECS Workshop, 2000.

[RIT99] RITTER, James. Microelectronics and Photonics Test Bed. In: http://ssdd.nrl.navy.mil/www/mptb/introduction.htlmx (Nov. 1999).

[ROC92] ROCKETT, L. SEU Hardened Scaled CMOS SRAM Cell Design Using Gate Resistors. In: IEEE Transactions on Nuclear Science. October, 1992.

[SIA94] SIA SEMICONDUCTOR INDUSTRY ASSOCIATION. The National Technology Roadmap for Semiconductors. 1994.

[SIL97] SILVA, L.; LIMA, F.; CARRO, L.; REIS, R. Synthesis of the FPGA Version of 8051, UFRGS Microelectronics Seminar (12), June 6-7 : Porto Alegre, BRAZIL, pp. 115-120, 1997.

[SKA96] SKAHILL, Kevin. VHDL for Programmable Logic. [S.l.] : Addison Wesley. 1996, p. 1-23.

[SPE00] SPENCER, Henry. *Faster, Better, but Most Important, Much Much Cheaper. MAPLD 2000.*

[STA88] STASSINOPOULOS, E.; RAYMOND, J. The Space Radiation Environment for Electronics. In: Proceedings of the IEEE, VOL. 76, NO. 11, November 1988.

[VAR01] VARGAS, F.; AMORY, A., "Circuit Modeling and Fault Injection Approach to Predict SEU Rate and MTTF in Complex Circuits", LATW 2001.

[VEL00] VELAZCO, R.; REZGUI, S.; ECOFFET, R. Transient bitflip injection on microprocessor-based digital architectures, presented at Nuclear and Space Radiation Effects Conference, NSREC 2000 , (Reno, USA, 24-29 July 2000).

[VEL01] VELAZCO, R., BRAGAGNINI, A., CALVO, O., "Upset-like fault injection in VHDL descriptions: A method and preliminary results", Proceedings of the IEEE Latin-American Test Workshop, Feb. 2001.

[VEL94] VELAZCO, R.; BESSOT, D.; DUZELLIR, S.; ECOFFET, R.; KOGA, R. Two Memory Cells Suitable for the Design of SEU-Tolerant VLSI Circuits. In: IEEE Transactions on Nuclear Science. VOL. 41, NO. 6, December 1994.

[VEL98] VELAZCO, R; REZGUI, S.; CHEYNET, Ph.; BOFILL, A.; ECOFFET, R. THESIC: A Testbed suitable for the qualification of integrated circuits devoted to operate in harsh environment. In: IEEE European Test Workshop (ETW'98) pp. 89-90, 27-29 Mai 1998, Spain.

[WAN99] WANG, J.; KATZ, R.; CRONQUIST, B.; MCCOLLUM, J.; SPEERS, T.; PLANTS, W. SRAM Based Re-programmable FPGA for Space Application. IEEE Transactions on Nuclear Science, Vol. 46, No. 6, pp. 1728-1735, December 1999.

[WEA87] WEAVER, H.; et al. An SEU Tolerant Memory Cell Derived from Fundamental Studies of SEU Mechanisms in SRAM. In: IEEE Transactions on Nuclear Science. VOL. 34, NO. 6, December 1987.

[WHI91] WHITAKER, S.; CANARIS, J.; LIU, K. SEU Hardened Memory Cells for CCSDS REED Solomon Encoder. In: IEEE Transactions on Nuclear Science. VOL. 38, NO. 6, December, 1991.

[WIS93] WISEMAN, D.; CANARIS, J.; WHITAKER, S.; GAMBLES, J.; ARAVE, K.; ARAVE, L., Test results for SEU and SEL immune memory circuits," in 5th NASA Symposium on VLSI Design, pp. 2.6.1{2.6.10, Nov. 1993.

[WIS93] WISEMAN, D.; CANARIS, J.; WHITAKER, S.; VEMBRUX, J.; CAMERON, K.; ARAVE, K.; ARAVE, L.; LIU, N.; LIU, K. Design and Testing of SEU / SEL Immune Memory and Logic Circuits in a Commercial CMOS Process. In: NSREC Conference, 1993

[XIL00a] XILINX, INC. Virtex®™ 2.5 V Field Programmable Gate Arrays, Xilinx Datasheet DS003, v2.4, Oct. 2000.

[XIL00b] XILINX, INC. QPRO™Virtex®™ 2.5V Radiation Hardened FPGAs, Xilinx Application Notes 151, v1.3, Feb. 2000.

[XIL00c] XILINX, Inc. Virtex® Series Configuration Architecture User Guide, Xilinx Application Notes 151, v1.3, Feb. 2000.

[XIL01a] XILINX, Inc. Virtex-II IP-Immersion™ Technology Enables Next-Generation Platform FPGAs. Xcell Journal Online, Sept. 2001. Disponivel por www em: http://www.xilinx.com/publications/xcellonline/virtex/ipimmersion.htm

[XIL01b] XILINX, Inc. Foundation Softaware Manual. Disponivel por www em: http://www.xilinx.com (Nov. 2001)

[YU 01] YU, Y. A perspective on the State of Research on Fault Injection Techniques, Research Report, May 2001. (www.people.virginia.edu/ ~yy6m/researh.htm)

[ZIE01] ZIEGLER, J. F. Review of Accelerated Testing of SRAMs and DRAMs, Sh*ort Course, RADECS 2001*.

[ZOU89] ZOUTENDYK, L.; EDMONDS, D.; SMITH, S. “Characterization of multiple-bit errors from single ion tracks in integrated circuits, ” IEEE Transactions on Nuclear Science, vol. 36, no. 6, pp. 2267-2274, Dec. 1989.