Design Methodology for highly Reliable Digital ASIC Designs Applied to Network-Centric System Middleware Switch Processor

Von der Fakultät für Mathematik, Naturwissenschaften und Informatik der Brandenburgischen Technischen Universität Cottbus - Senftenberg

zur Erlangung des akademischen Grades

Doktor der Ingenieurwissenschaften

(Dr.-Ing.)

genehmigte Dissertation

vorgelegt von

Diplom Ingenieur

Vladimir Petrovic

Geboren am 13.12.1982. in Pozarevac, Serbien, Jugoslawien

Gutachter: Prof. Dr.-Ing. Rolf Kraemer

Gutachter: Prof. Dr.-Ing. Heinrich Theodor Vierhaus

Gutachter: Prof. Dr. Sergio Montenegro

Tag der mündlichen Prüfung: 12.12.2013.

Acknowledgments

I would like to express my sincerely gratitude to my supervisor Prof. Rolf Kraemer for support and opportunity to conduct my research work at the IHP through last years. While working in IHP I had support from many other colleagues and on this place I would like to thank them all.

Before I express my gratitude to all people which were important during the thesis writing, I would like here specially to thank Prof. Dr. Vanco Litovski for the provided knowledge and support since 2001, when I started seriously to learn electronics. Specially thanks to Gunter Schoof and Ulrich Jagdhold for constructive discussions, interesting ideas and very helpful support during practical tests and thesis development. Dr. Zoran Stamenkovic put a lot of effort during thesis writing and he helped me a lot to bring a scientific form of the work provided in thesis. Therefore, I owe special gratitude to him and also to Dr. Michael Methfessel for support. I would like to thank Dr. Milos Krstic for very important discussions, thesis organization and great support in last years.

The measurements and other tests would be impossible without Horst Frankenfeldt, Irina Matthaei, Silvia Hinrich, Christoph Wolf, Mahdi Khafaji and Florian Teply. I would like to express my sincerely gratitude to all of you.

I would like to give my warmest thanks to my brother, parents, family and all friends, especially from KUD Kostolac for tremendous support all these years.

This work is especially dedicated to my wife Danica. Her love, motivation and support have created the confidence that is needed to complete such a huge and complex task.

Vladimir Petrovic

22.01.2014.

Contents

Abstract

The sensitivity of application-specific integrated circuits (ASICs) to single event effects (SEE) can lead to failures of subsystems which are exposed to increased radiation levels in space and on the ground. The work described in this thesis presents a design methodology for a fully fault-tolerant ASIC that is immune to single event upset effects (SEU) in sequential logic, single event transient effects (SET) in combinatorial logic, and single event latchup effects (SEL). Redundant circuits combined with SEL power switches (SPS) are the basis for a design methodology which achieves this goal. Within the standard ASIC design flow enhancements were made in order to incorporate redundancy and SPS cells and, consequently, enable protection against SEU, SET, and SEL. In order to validate the resulting fault-tolerant circuits a fault-injection environment with carefully designed fault models was developed. The moments of fault occurrence and their durations are modeled according to the real effects in actual hardware. The proposed design methodology was applied to an innovative space craft area network (SCAN) central processor unit, known as middleware switch processor. The measurement results presented in this thesis prove the correct functionality of DMR and SPS circuits, as well as the high fault-tolerance of the implemented ASICs along with moderate overhead with respect to power consumption and occupied silicon area. Irradiation measurements demonstrated the correct design and successful implementation of the SPS cell.

Zusammenfassung

Die Empfindlichkeit von anwendungsspezifischen integrierten Schaltungen (ASICs) zu den einzelnen Ereigniseffekten (SEE), kann zu Ausfällen von Subsystemen führen, die erhöhten Strahlungspegeln im Raum und auf dem Boden ausgesetzt werden. Die Arbeit, die in dieser Thesis beschrieben wird, stellt eine Entwurfsmethodologie vor um fehlertolerante ASICs zu entwerfen, welche die immun gegen singuläre Störung Effekte (SEU) in sequentielle Logik ist, einzelne Ereignis vorübergehende Effekte (SET) in der kombinatorischen Logik und einzelnes Ereignis Latchup-Effekte (SEL). Modulare Redundanz und SEL-schalter (SPS) sind die Basis für eine Design-Methodik, die volle fehlertolerante ASIC liefert. Der Standard ASIC-Designflow ist erweitert worden, um Redundanz mit SPS-schalter zu enthalten und Schutz gegen SEU, SET und SEL zu ermöglichen. Um die fehlertoleranten Stromkreise zu validieren ist eine Fault-Injektion Umgebung mit Fault Modellen entwickelt worden. Die Momente des Auftretens und der Dauer der injizierten Fehler werden entsprechend den realen Effekten in die Hardware modelliert. Die Methodologie des vorgeschlagenen Entwurfs ist an einer innovativen Space Craft Area Network (SCAN) Schaltkreis angewendet worden, bekannt als Middleware Switch Prozessor. Die Messergebnisse, die in dieser These dargestellt werden, haben die korrekte Funktionalität von Redundanz- und SPS-Stromkreisen sowie die hohe Fehler-toleranz der resultierende ASICs zusammen mit mäßigen Unkosten in Bezug auf Leistungsaufnahme und besetzten Silikonfläche nachgewiesen. Die Strahlungsmessungen haben das korrekte Design und die erfolgreiche Umsetzung der SPS-Zelle bewiesen.

1. Introduction

Improper functioning of electronic devices due to single event effects, being an effect of radiation, is observed not only in space and airborne equipment, but also in mainstream applications. Together with the progressing integration and scaling of electronic chips their susceptibility to such errors increases. Current space microelectronics development and high energy physics research require shorter design time and cheaper solutions for radiation and fault tolerant ASIC designs [DRE02], [TAR11], [JUG07]. The main purpose is to provide ASICs capable of correct and reliable functioning in radiation environments using the standard semiconductor technologies and design flow. Therefore, the design of advanced fault-tolerant digital integrated circuits needs scientific research and progress concerning three important aspects:

1. Analysis of irradiation effects and circuit faults.
2. Development of fault models and simulation test benches.
3. Design of fault-tolerant circuits and systems.

The definition and description of basic irradiation effects and fault mechanisms can be found in the literature [HOS02], [NIC11], [INI11]. The thesis referenced in [JAV12] provides an overview of different radiation environments and investigates interaction mechanisms between energetic particles and matter. It also introduces a new semi-empirical model for estimating the electronic stopping force of heavy ions in solids. The most common irradiation effect is a single event effect (SEE) induced by a cosmic particle strike. Three common types of SEE are known: single event upset (SEU), single event transient (SET) and single event latchup (SEL). A single event upset causes the change of state in a storage element. It affects memory cells and sequential logic. A single effect transient causes a short impulse (and wrong logic state) at a combinational logic output. The wrong logic state will propagate in case that it appears during the active clock edge.

On the other hand, a single event latchup causes excessive current flow through an NPNP structure in CMOS circuits. A single event latchup, compared to the SEU and SET, is a potentially destructive state [TRO86], [VOL08], [WES11], [LYE09]. In order to analyze the final impact of single event effects and to provide the high fault coverage at circuit and system levels, it is necessary to develop practical, accurate, and simple simulation fault models. A good fault model should reproduce the correct function of a circuit (NAND, NOR, flip-flop, etc.) in case of no fault and to model the circuit malfunction in case of a fault. Besides the logic function, the most relevant modeling parameters are the effect duration, the minimal time of current discharge, and the current pulse intensity. The easiest way to simulate a fault in a logic circuit is use of a fault injector, which is an additional XOR or XNOR gate. Fault injectors can be added once the logic synthesis has been completed and a design netlist has been generated. Computer parsers are used for automatic insertion of the fault injectors [MCH97], [BEN03], [SHE08], [SHA09].

Simulation of fault injection is the most widely used approach for validation of fault-tolerant systems. The proposed fault models can be used during behavioral and net-list simulation. The models are

usually described in a hardware description language (VHDL or Verilog). We can clearly separate the known SEU, SET, and SEL fault-tolerant techniques into two categories:

- Circuit level techniques [LAC08]
  - Hardened-cell design [BAZ00]
  - Triple modular redundancy (TMR) [LYO62]
  - Double modular redundancy (DMR) [TEI08]
  - Error detection and correction for memories [LAL03]
- Layout level techniques [MAK03]
  - Enclosed layout transistors
  - Guard rings
  - Trench isolation.

There are also other mitigation techniques based on expensive technology changes [MAK03].

If SEU or SET occurred, the most common fault-tolerant techniques are TMR and DMR. A triple modular redundant circuit consists of three identical modules and a 3- input majority voter. The voter's function is to pass through the major input value to the output. In the context of digital circuits, these modules are memory elements such as flip-flops or latches. The main disadvantage of this technique is that the system fails in case of a faulty voter. Therefore, a new triple voting logic was developed to complete the circuit redundancy. Each of the three voters is fed from outputs of all three memory modules. This technique is known in the literature as the full triple modular redundancy. A detailed analysis of the triple modular redundancy was presented in [ROL03]. Higher redundancy provides a higher fault tolerance and circuit reliability. On the other hand, higher redundancy increases the chip area, energy consumption, and costs. Therefore, one goal is to find a trade-off between the redundancy level and the reliability requirements taking into account the prospective application. In order to reduce the high hardware overhead required by TMR while keeping the design reliability high, double modular redundancy with self-voting was developed [TEI08]. A novel double/triple modular redundancy (DTMR) technique for dynamically reconfigurable devices tested on a finite state machine was presented in [SHI08].

The SEL mitigation techniques can be classified into three main groups:

1. The first approach uses current sensors at board level to detect excessive current induced by latchup. The power supply of the affected device is switched off and, after a long enough period of time, reestablished again. This approach suffers from a serious drawback: the circuit state is destroyed and cannot be recovered.

2. The second approach [EST78] is based on introduction of an epitaxial buried silicon layer and reduction of the well resistivity. However, this modification incurs additional costs and may impact circuit performance (breakdown voltage, for example).

3. The third approach [TRO83] uses guard rings (additional N-type and P-type regions) that break the parasitic bipolar transistor structure. This solution is very efficient but can result in excessive circuit area and, therefore, price.

A new SEL mitigation scheme which combines error correcting codes with intelligent power line implementation was presented in [NIC06]. It prevents the circuit damage and corrects errors caused by latchup in a transparent manner. The area cost is low and the scheme can also be used to correct SEUs and reduce power dissipation.

In spite of the huge research efforts and respectable scientific achievements, there are still challenges regarding the use of commercial ASIC technologies in space and safety-critical applications. The most important challenges are:

1. Radiation-hardened technologies are expensive (qualification and quality requirements are severe) and commercially not attractive (small volume production).

2. There is no standard integrated framework of circuit design techniques that provides simultaneous SEU, SET, and SEL fault-tolerance.

3. There is no standard design automation flow for fault-tolerant digital ASICs and SOCs.

This thesis presents a design flow for fault-tolerant ASIC that is based on redundant circuits with latchup protection and additional implementation steps during logic synthesis and layout generation. The proposed approach protects ASICs from the SEU, SET, and SEL faults by combining and integrating different fault-tolerant techniques in a carefully designed procedure. The redundancy provides correction of transient and upset effects. In case that a latchup occurs in a digital cell, the best known protection method is switching off the power supply of affected cell. In redundant digital circuits the voter defines a correct logic value based on outputs of redundant flip-flops. If one of redundant flip-flop is switched off due latchup protection, the other voters receive invalid logic value from the affected flip-flop, what causes the problem for the system with double and triple modular redundancy. This thesis describes how the mentioned problem can be solved using the perceptive flip-flops. Perceptive flip-flops are specially designed components which observe the logic values and power supply status from redundant flip-flops. The perceptive flip-flop provides basis for combined protection against SEU, SET and SEL in redundant circuits.

This thesis includes six additional chapters. Chapter 2 defines the thesis topics and describes the state-of-the-art in the field of fault-tolerant circuits. New fault-tolerant integrated circuit design techniques will be proposed and developed in Chapter 3. Chapter 4 describes the simulation environment and presents a comparative analysis of simulation and calculated results. In Chapter 5, modifications of the standard digital design and implementation flow imposed by newly developed circuit design techniques will be discussed. The proposed approach is proven using several fault-tolerant circuits as test cases. Chapter 6 is a case study that presents design and implementation of a fault-tolerant middleware switch processor. Chapter 7 highlights the main scientific contributions and research results and concludes the thesis.

2. Topic Definition and State-Of-The-Art

Advanced design of the fault-tolerant digital integrated circuits requires scientific research and progress concerning three important aspects:

1. Analysis of irradiation effects and circuit faults.
2. Development of fault models and simulation test benches.
3. Design of fault-tolerant circuits and systems.

Flow-diagram in Figure 2.1 presents interdependences between the research topics (simulation, design, and implementation of the fault-tolerant integrated circuits) addressed in this chapter. Section 2.1 describes the well-known irradiation effects and corresponding circuit faults. Section 2.2 and Section 2.3 deal with the fault models. Section 2.4 is dedicated to the existing design techniques which preserve the space and safety-critical electronic systems from the failure. Section 2.5 describes the standard ASIC design flow. Section 2.6 summarizes the open issues in the field.

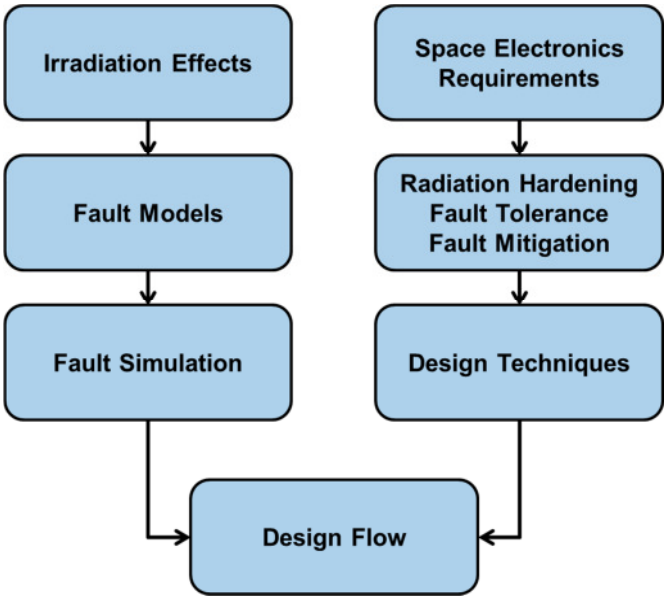


Figure 2.1. Relevant research topics in the field of fault-tolerant circuit design

2.1. Irradiation Effects and Circuit Faults

The unavoidable existence of particle radiation in space and on the ground, combined with continuous scaling of the semiconductor device sizes, requires a thorough investigation of the basic irradiation effects and fault mechanisms. Definition and description of the mentioned effects and mechanisms can be found in literature [HOS02], [NIC11], [INI11]. The thesis referenced in [JAV12] provides an overview of different radiation environments and investigates interaction mechanisms between energetic particles and the matter. It also introduces a new semi-empirical model for estimating the electronic stopping force of heavy ions in solids.

Natural space radiation (high-energy ionizing particles) may induce electrical noise (***single event effect***) in many types of semiconductor technologies and electron devices (Figure 2.2). It can cause the data corruption, transient disturbance, and high-current conditions. Each particle produces an ionization track (and electrical charge flow) which has two components: a prompt component (funneling in high-field regions) and a delayed component (diffusion in low-field regions). Non-destructive irradiation effects are ***single event upset*** (SEU) and ***single event transient*** (SET). The most important destructive irradiation effect in CMOS integrated circuits is ***single event latchup*** (SEL).

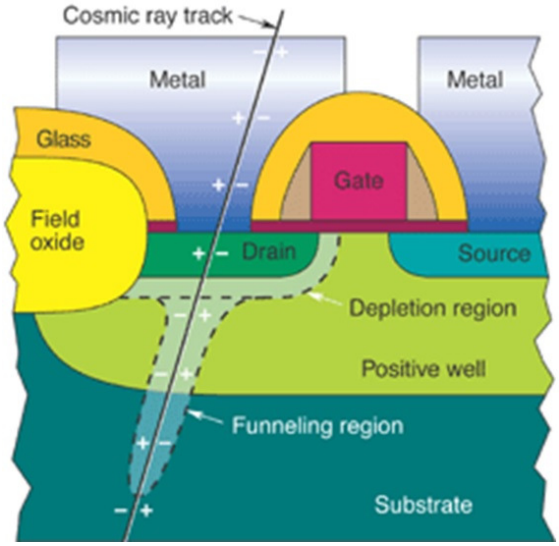


Figure 2.2. Illustration of single event effect in MOS device caused by ionizing particle

A single event upset causes the change of state in a storage element. It affects the memory cells and sequential logic. A single event transient causes a short impulse (and wrong logic state) at the combinational logic output. The wrong logic state will propagate in case that it appears during the active clock edge. On the other hand, a single event latchup causes the excessive current flow through a NPNP structure in CMOS circuits (Figure 2.3).

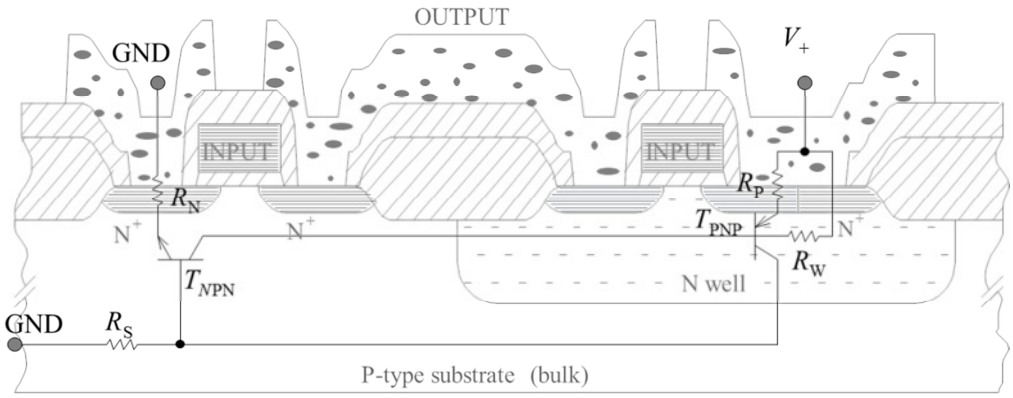


Figure 2.3. Intrinsic NPNP structure and parasitic bipolar transistors in CMOS circuits

In normal operating conditions, the power supply (quiescent) current is very low. In case of ionization and minority carrier injection, the current raises and one of the parasitic bipolar transistors (lateral NPN

transistor and vertical PNP transistor) turns on. If a positive feedback between transistors formed by the N-well (R_W) and P-substrate (R_S) resistances maintains the current increase long enough, the central PN junction of the NPNP structure turns on in forward direction and the latchup occurs. Both bipolar transistors presented in Figure 2.3 are then in saturation. The current quickly increases and, if not limited, causes the circuit break-down [TRO86], [WES11].

2.2. SEU and SET Fault Models and Injection Techniques

In order to analyze the final impact of the single event effects and provide the high fault coverage at circuit and system levels, it is necessary to develop practical, accurate, and simple simulation fault models. A fault model is a model of digital circuit that could operate wrong. From the fault model it is possible to predict the consequences of particular fault. A good fault model should keep the correct logic function of a circuit (NAND, NOR, flip-flop, etc.) if no faults are present. On the other hand, the fault model causes the circuits malfunction if fault is activated. Beside the logic function, the most relevant modeling parameters are the effect duration, the minimal time of current discharge, and the current pulse intensity. The easiest way to simulate a fault in the logic circuit is use of a fault injector: XOR or XNOR gate. The truth-table of XOR and XNOR gates is presented in Table 2.1.

Table 2.1. Truth table of XOR and XNOR gates

A	B	A xor B	A xnor B
0	0	0	1
0	1	1	0
1	0	1	0
1	1	0	1

In case of a XOR fault injector, it is enough to set one of the XOR inputs to high logic value and the logic value of the other input will be inverted at the XOR output. If a XOR gate is connected to flip-flop output, it can simulate a single event upset. On the other hand, if a XOR gate is connected to the combinational logic output, it can generate a single event transient. It is important to take care about the fault duration in order to keep the model accurate as much as possible.

The fault-injection model for a flip-flop and corresponding timing diagram are presented in Figure 2.4. The model consists of a flip-flop and a XOR gate. In case that the SEU input is activated (set to high logic value), the logic value at the fault injector output will be inverted compared to the value of the original flip-flop output. On the other hand, if the SEU input is not active (low logic value), the model is reduced to the original flip-flop.

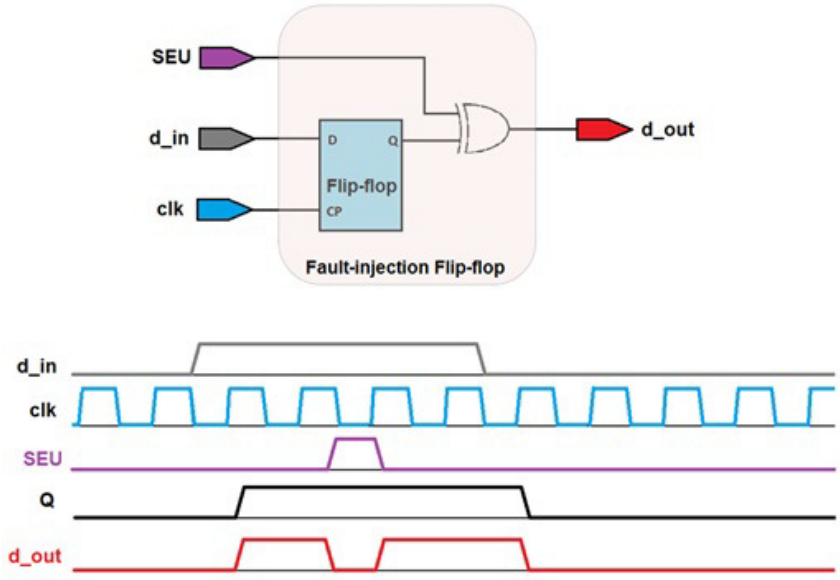


Figure 2.4. Fault injection in a flip-flop

Transient effects in the combinational logic can be treated similar as upsets in the sequential logic. However, SET model in comparison with SEU model (which is independent of clock frequency and disappears after the next active clock edge) is dependent on clock frequency and the SET probability increases with increase of clock frequency. The fault-injection model for combinational logic and corresponding timing diagram are presented in Figure 2.5.

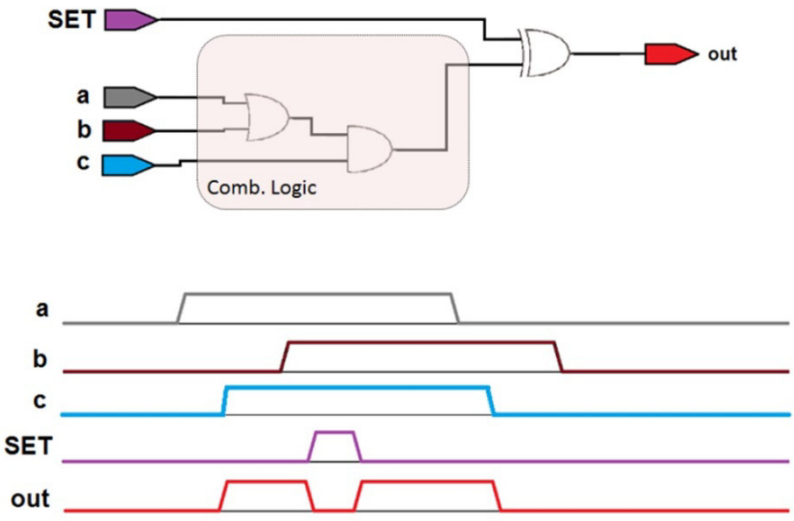


Figure 2.5. Fault injection in combinational logic

Fault injectors can be added once the logic synthesis has been completed and a design netlist has been generated. Computer parsers are used for automatic insertion of the fault injectors [MCH97], [BEN03], [SHE08], [SHA09]. Simulation fault-injection is the most used approach for validation of fault-tolerant systems. The proposed fault models can be used during behavioral and netlist simulation. The models are usually described in hardware description language (VHDL or Verilog). Main advantages of the simulation based fault-injection are:

- Full control of the fault models and fault-injection mechanism,
- Computer automation without investment in extra fault-injection hardware,
- Full controllability and observability,
- Transient and permanent faults are covered,
- Modeling of timing-related faults is possible.

Main disadvantages of the simulation based fault-injection are:

- Model development requires high efforts,
- Simulation tests are time-consuming,
- Model accuracy must be proved,
- High fault-coverage must be achieved.

Detailed classification of simulation fault-injection models and techniques is presented in [BEN03] by Benso and Prinetto. Moreover, an integrated fault-injection framework with fault models described in hardware description languages, fault libraries, fault generators, and test benches is described. The authors also provide a global view on the simulation techniques for setting up a fault-injection experiment.

An automated fault-injection technique for characterization of the soft error sensitivity of a VHDL based design is described in [SHE08]. The technique is based on lexical and syntax analysis. A stratified sampling technique is used to reduce the fault-injection overhead. A fault-injection tool called HSECT (HIT Soft Error Characterization Toolkit) is developed and 3000 soft errors are injected into a simple RISC processor called DP32-processor. The recovery coverage and soft error sensitivity of the processor are further investigated to increase the fault tolerance and system dependability.

Evaluation and possible improvement of the fault tolerance and error detection in safety-critical electronic systems is presented in [SHA09]. The proposed approach is based on simulation-based fault injection and allows analysis of the system behavior when faults occur. The paper describes how a microprocessor board (Motorola MVME162 Controller Board) employed in an automated light-metro control system has been modeled in VHDL and a fault-injection environment has been set up using a commercial simulator. Preliminary results about the effectiveness of the hardware fault-detection mechanisms are also reported.

Beside the fault injection in simulation phase, there are two more fault-injection techniques: the hardware fault-injection and the software fault-injection. Hardware fault-injection represents a type of fault-injection technique where the specially designed test hardware is integrated into a system in order to inject the faults. They are usually of three types: breaking faults, bridging faults, and transient faults. The fault-injection tests are performed after chip fabrication. This fault-injection technique requires additional test equipment and long setup time. On the other hand, compared to the simulation

based fault-injection, the hardware based fault-injection provides a fast and direct way of detecting the incorrect circuit behavior [MCH97]. The main advantages of the hardware fault-injection technique are:

- Testing of the chip locations which are not accessible by other techniques,
- The injected faults cause lower perturbation overhead,
- No need for the model development and validation,
- Testing is in real time.

The main disadvantages of the hardware fault-injection are:

- High risk of the hardware damage,
- High level of device integration,
- Low portability (the implemented hardware can be used in one type of system only),
- Limited number of injection points,
- Limited number of injectable faults,
- Long setup time for experiments.

Software fault-injection is based on modifications of the software code in order to change the system behavior and observation of the system functionality under different and unexpected conditions. The software designer usually understands a system through programmer's model view of system. In order to suppress eventual system terminations, it is important to create a programmer's model of the system to behave same as the real one. In this process the software based fault-injection can provide estimation of the hidden bugs. It is important to note that software errors are today the most important causes of the system failure. All types of faults may be injected – from the register and memory faults, via dropped or replicated network packets to the erroneous error conditions and flags [MCH97]. The main advantages of the software based fault-injection are:

- It targets the applications and operating systems,
- Testing is almost in real time,
- Fault injection is performed on the real hardware,
- No need for extra hardware and, therefore, the implementation costs are low,
- No need for the model development and validation.

The main disadvantages of the software based fault-injection are:

- Limited set of injection instances,
- Impossible to add faults into locations not accessible for software,
- Extra code is required in order to inject faults (the software is different in test and operation modes),

- Limited controllability and observability,
- Permanent faults are difficult to model,
- Disturbances of the task scheduling are possible.

In this thesis, the simulation and hardware fault-injection techniques will be considered. A detailed description of the used simulation framework will be given in Chapter 3. Hardware fault-injection techniques and measurements will be given in Chapter 5.

2.3. SEL Model and Characterization

To model the single event latchup, an equivalent circuit of the CMOS NPNP structure and its current-voltage characteristics are drawn in Figure 2.6.

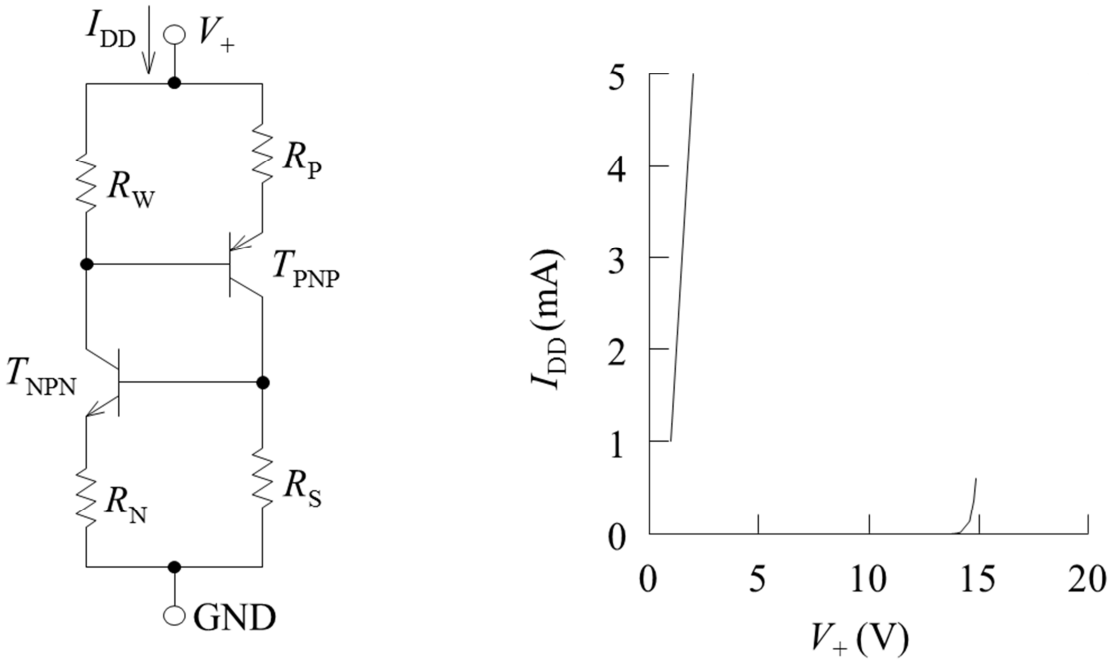


Figure 2.6. Equivalent circuit of the CMOS NPNP structure and I-V characteristics

The NPN and PNP bipolar transistors are connected in a feedback loop. The collector current of one bipolar transistor feeds the base current of the other. Current that flows through one transistor is amplified by the other and in the same time further amplified by the first one. Before explanation of activating and conducting processes, it is important to describe relations between currents in the four-level structure. In order to explain the latchup effect, in the following text is used the equivalent thyristor component. The Equation 2.1 defines the relation between current intensity on the collector junction and the total current intensity through emitter junction. This relation is in the literature known as current amplification coefficient α_c [LIT96], [HAS05]. The current which flows through collector junction (I_{pC}) is the collector current (I_C) reduced for the intensity of reverse saturation current (I_{C0}).

$$\alpha_c = \frac{I_{pC}}{I_E} \tag{2.1}$$

$$I_{pC} = I_C - I_{C0} \tag{2.2}$$

A silicon-controlled rectifier is a four-layer device formed of alternating P and N regions as shown in Figure 2.7 a). If a four-layer device is cut along the dotted line, it can be seen that the silicon-controlled rectifier (SCR) is electrically equivalent to a PNP (P1 - N1 - P2) and an NPN (N1 – P2 - N2) bipolar junction transistors (BJT) wired back-to-back as in Figure 2.7 b). The top three layers form the PNP transistor and the bottom three layers form the NPN transistor [LIT96], [HAS05].

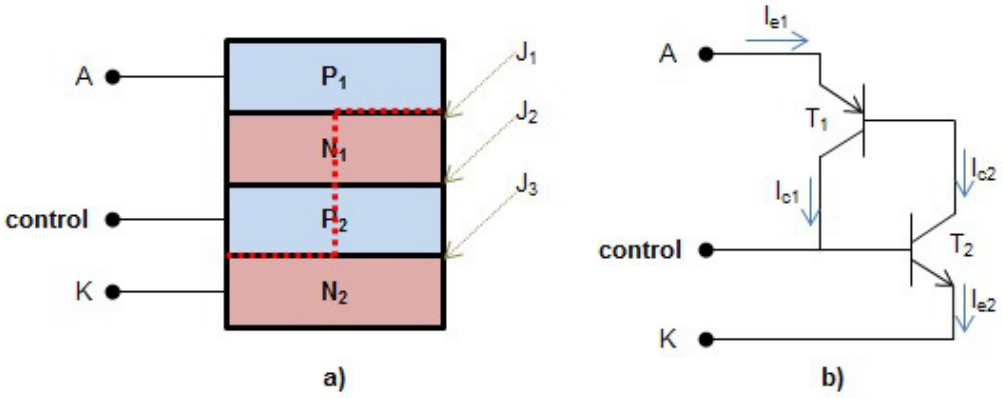


Figure 2.7. Silicon-controlled rectifier: a) four-layer structure and b) equivalent schematic

The current amplification coefficient (α_c) is a little bit less than one (for example - 0.98). Related to the Figure 2.7 b) and Equations 2.1 and 2.2, we can define the currents I_{C1} and I_{C2} [LIT96], [HAS05].

$$I_{C1} = \alpha_1 I_{E1} + \frac{I_{C0}}{2} \tag{2.3}$$

$$I_{C2} = \alpha_2 I_{E2} + \frac{I_{C0}}{2} \tag{2.4}$$

The current amplification coefficient α_1 and α_2 are related to the PNP transistor T_1 and NPN transistor T_2, respectively. The reverse saturation current (I_{C0}) is halved related to the Figure 2.7 a), where we can see that collector-basis junctions of both transistors use same P-N junction. The recombination and carrier flow is almost equally distributed. The input current I_{E1} is equal to the output current I_{E2} of SRC. On the other hand, I_{E2} is composed from currents I_{C1} and I_{C2}. The Equation 2.5 and 2.6 define the SRC output current [LIT96], [HAS05].

$$I_{E2} = I_{E1} = I_{C1} + I_{C2} \tag{2.5}$$

$$I_{E2} = \alpha_1 I_{E1} + \alpha_2 I_{E2} + I_{C0} \Rightarrow I_{E2} = \frac{I_{C0}}{1 - (\alpha_1 + \alpha_2)} \tag{2.6}$$

The thyristor can be activated on two ways – by higher anode voltage or by “control” pin current injection [LIT96], [HAS05].

The activation by higher anode voltage is based on the leakage current amplification. If voltage between anode and cathode increases, raises the leakage current too. The main current component of the four-layer structure is in this case the reverse saturation current. The leakage current of one transistor is the basis current of the other. Therefore, the increase of the anode voltage, induce higher

leakage currents and higher collector currents. This effects provide the increase of current amplification coefficients α_1 and α_2. From the Equation 2.6 it is possible to see that the output current of SCR is stable until sum of coefficients $\alpha_1 + \alpha_2$ is not equal to 1. Then the current intensity is not any more controlled by the SCR component. The current is in this case controlled by the outside circuit. The junctions J_1, J_2 and J_3 are in this case directly polarized. The current stops only if anode potential is equal to the cathode potential [LIT96], [HAS05].

Other way to activate a SCR is to inject small current in to the basis of NPN transistor (T_2). Similar as it is explained above, the current pulse provides a cumulative effect in the base-collector loop of the schematic presented in Figure 2.7 b). The basis current of the NPN transistor is gained by parameter – β_1. The β parameter presents a current gain of the transistor. It is a relation between collector and base current. Equation 2.7 defines a β parameter and Equation 2.8 defines a relation between the current amplification coefficient and the current gain [LIT96], [HAS05].

$$\beta = \frac{I_C}{I_B} \tag{2.7}$$

$$\beta = \frac{\alpha}{1-\alpha} \tag{2.8}$$

$$\alpha = \frac{\beta}{1+\beta} \tag{2.9}$$

Until the sum of coefficients $\alpha_1 + \alpha_2$ is not equal to 1, the SRC conducts just leakage current. Combining the Equations 2.9 and 2.10 it is possible to express the conduction condition using β parameter of both transistors. The Equation 2.14 describes condition of SRC activation, related to the current gain parameters β_1 and β_2.

$$\alpha_1 + \alpha_2 = 1 \tag{2.10}$$

$$\frac{\beta_1}{1+\beta_1} + \frac{\beta_2}{1+\beta_2} = 1 \tag{2.11}$$

$$\frac{\beta_1(1+\beta_2) + \beta_2(1+\beta_1)}{(1+\beta_1)(1+\beta_2)} = 1 \tag{2.12}$$

$$\beta_1 + \beta_1\beta_2 + \beta_2 + \beta_1\beta_2 = 1 + \beta_1 + \beta_2 + \beta_1\beta_2 \tag{2.13}$$

$$\beta_1\beta_2 = 1 \tag{2.14}$$

Therefore, when the product of the current gains $\beta_1\beta_2$ (loop gain) of the two bipolar transistors is greater than one, the current flowing through the SCR grows in an unbounded fashion [LIT96], [HAS05].

In practice however, the loop gain depends on the current through and the potential across the SCR. For instance, at very large currents, the current gain of a typical bipolar transistor drops significantly from its peak value. Furthermore, the emitter current in a bipolar transistor is exponentially related to the base-emitter voltage. As the current in SCR increases, the base-emitter junction voltage of both transistors (NPN and PNP) is higher. It results in a decrease in the collector-emitter voltage of each

bipolar transistor, what represents an effect which can be limited by adding any series resistance in the SCR. Once the collector-emitter voltage drops below the saturation voltage, the current gain of the bipolar transistors drop rapidly, and the SCR current will stabilize.

Since the current gain of a typical bipolar transistor also drops significantly from its peak value at very low currents, the SCR can also stabilize with no current flowing through it, other than leakage. Thus the SCR is typically bi-stable, with a high-current on state, and a zero-current off state.

Once the SCR is activated, it is difficult to return to the off state. In theory, the SCR can be shut off by shunting all the collector current from the upper PNP transistor (T_1) out via the control terminal, but in practice, the SCR current is usually so large that shunting it is impractical. Thus, the only way to shut the SCR off again is to bring the potential across it to zero (or some negative value, what is possible just in the AC SCR application).

The work presented in [LYE09] highlights and investigates the latchup effect at device and circuit levels. A practical analytical model based on the Ebers-Moll formulations and equivalent circuits is presented. The model parameters are experimentally extracted for 0.25 µm CMOS technology. The authors claim for successful prediction of the latchup using the proposed model in SPICE simulations.

More about the existing design techniques for detecting and correcting the SEU, SET, and SEL faults in CMOS integrated circuits comes with the following section.

2.4. Fault-Tolerant Circuits

Although the fault-tolerant SRAM-based FPGA has been increasingly becoming more common in the space-based computing [WIR03], [SAM04], [STE06], [BOL07], the space microelectronics industry and scientific research in the nuclear physics area need in any case high complexity integrated circuits based on ASICs [DRE02], [SZC07], [TAR11], [BRE05].

The main requirement of the space and safety-critical systems is high reliability. In the environments where is hard or even impossible to provide maintenance (space and military applications), it is very important to deploy the circuits and systems which can tolerate faults. Practically, almost all SEU and SET fault-tolerant techniques are based on the redundancy. There are a few types of the redundancy:

1. Hardware redundancy (triple and double modular redundancy),
2. Information redundancy (error detection and correction),
3. Time redundancy, and
4. Software redundancy.

In this thesis, the hardware redundancy will only be discussed. As the technology allows for smaller transistors, the digital cells get smaller and, therefore, the hardware redundancy becomes more popular. The hardware redundancy provides masking of faults and protects the circuit (or system) from failure. Common hardware-redundancy techniques are the triple modular redundancy (TMR) and the double modular redundancy (DMR).

The triple modular redundancy was mentioned for the first time in the literature in 1956 by J. Von Neumann [NEU56]. The redundant circuit consists of three identical modules and a 3-input majority voter (Figure 2.8 a)). The voter's function is to pass through the major input value to the output. As we speak about digital circuits, the modules are memory elements such as flip-flops or latches. The main disadvantage of this technique is that the system fails in case of a faulty voter. Therefore, a new triple voting logic was developed to complete the circuit redundancy (Figure 2.8 b)). Each of the three voters is fed from outputs of all three memory modules. This technique is known in the literature as the full triple modular redundancy. A detailed analysis of the triple modular redundancy is presented in [LYO62].

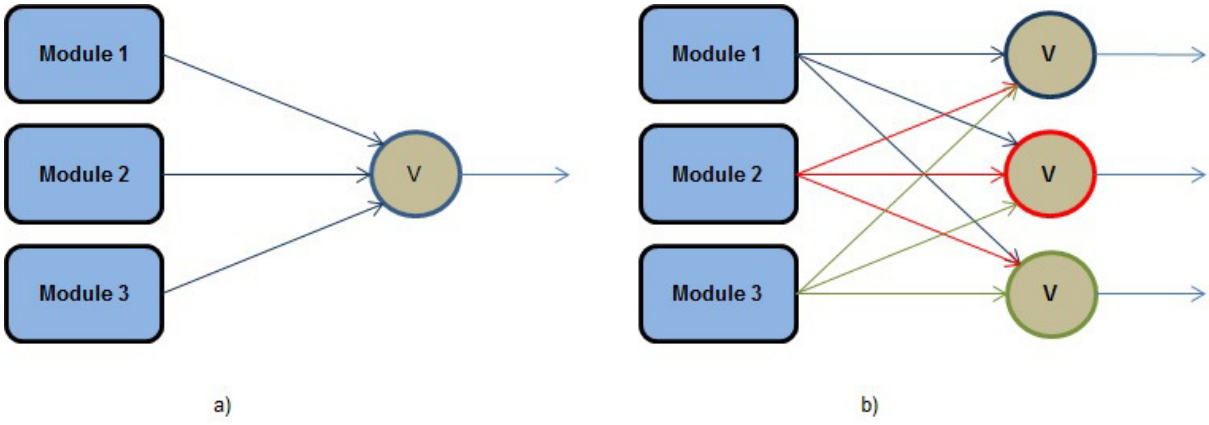


Figure 2.8. Triple modular redundancy: a) standard TMR and b) full TMR

Based on the published work [LYO62] [TKS82] [KOL33], here is presented short background related to the failure-free probability of the standard TMR redundant circuit. The reliability is defined as conditional probability that system will operate correctly during the specified time interval [KHA93].

The failure rate (λ) is a number which represents how often the system fails in a period of time. Instead of the failure rate, it is possible to define the **mean time between failures** (MTFB), which represents the time period between two failures. The MTFB is very important parameter for the fault-tolerant systems and it represents the system ability to operate without failure. It is usually defined in hours. Based on the failure rate, the reliability of the module "M" is calculated using Equation 2.15.

$$R_M = e^{-\lambda t} \qquad (2.15)$$

Using the reliability (R_M) it is possible to calculate the failure rate if it is unknown. The related procedure is described in Equation 2.16, which represents the Equation 2.15 after performing the integration process.

$$\int_0^\infty R_M dt = \int_0^\infty e^{-\lambda t} dt = -\frac{1}{\lambda} e^{-\lambda t}\Big|_0^\infty = 0 - \left(-\frac{1}{\lambda}\right) = \frac{1}{\lambda} = MTFB_M \qquad (2.16)$$

Equation 2.16 defines also the mean time between failures (MTBF). Based on Equations 2.15 and 2.16, can be calculated the TMR system reliability. In Figure 2.9 is presented a TMR system with single-voting structure. It is assumed that the voter has ideal characteristics [LYO62].

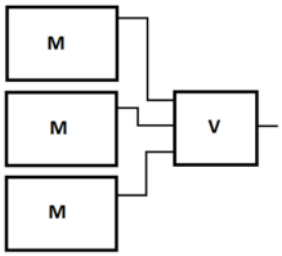


Figure 2.9. TMR system with single voter

In order to calculate the TMR system reliability, the discussion starts with definition of the N-modular system (NMR) [KOR79]. The reliability of the NMR system with ideal voter is defined in Equation 2.17.

$$R_{NMR} = \sum_{i=0}^{n} \binom{N}{i} (1 - R_M)^i R_M^{(N-i)} \qquad (2.17)$$

Related to [KOR79], a NMR system can tolerate up to n module faults and still to keep correct function. Therefore, the relation between redundancy level N and maximal number of modules which are faulty n, defined in the Equation 2.17, can be calculated as $n = (N-1)/2$. The reliability of the TMR system related to Equation 2.17 with $N = 3$ and $n = 1$ is:

$$\begin{aligned} R_{TMR} &= \sum_{i=0}^{1} \binom{3}{i} (1 - R_M)^i R_M^{(3-i)} \\ &= \binom{3}{0} (1 - R_M)^0 R_M^3 + \binom{3}{1} (1 - R_M)^1 R_M^2 \\ &= R_M^3 + 3(1 - R_M) R_M^2 \\ &= R_M^3 + 3R_M^2 - 3R_M^3 \\ &= 3R_M^2 - 2R_M^3 \end{aligned} \qquad (2.18)$$

There is another way to calculate the reliability of TMR system presented in Figure 2.9. If all three modules are functioning correctly than the reliability of the TMR system is calculated as R_M^3. The TMR system still functions correct if one of three modules is faulty. The reliability in this case is calculated as $3R_M^2(1 - R_M)$. The number three is included while there are three cases when one module is faulty and other two are operating correctly. The TMR system reliability is represented as addition of the above calculated reliabilities, because it represents the failure-free probability. Equation 2.19 defines the TMR reliability calculated on the mentioned way.

$$\begin{aligned} R_{TMR} &= R_M^3 + 3R_M^2(1 - R_M) \\ &= R_M^3 + 3R_M^2 - 3R_M^3 \\ &= 3R_M^2 - 2R_M^3 \end{aligned} \qquad (2.19)$$

The Equations 2.18 and 2.19 are providing the same result. In order to provide more clear description of the terms used in this analysis, in the further text for the redundant circuit reliability will be used the term failure-free probability. Compared to the general system reliability, the failure-free probability represents ability of the digital circuit to handle single event effects. On the other hand, instead of the

term module reliability will be used the term module fault-free probability and it is related to the flip-flop for example. Not to the complete redundant circuit. This is done because the reliability describes the ability of correct system function during the time period. The fault-free probability (P), is expressed in percent and it is related to the redundant digital circuits. Therefore, the fault-free probability for the TMR digital circuit is defined by Equation 2.20.

$$P_{TMR} = 3P_M^2 - 2P_M^3 \qquad (2.20)$$

Figure 2.10 presents the dependence between TMR circuit failure-free probability and the fault-free probability of the single sequential element. The voter used in analysis has ideal characteristics.

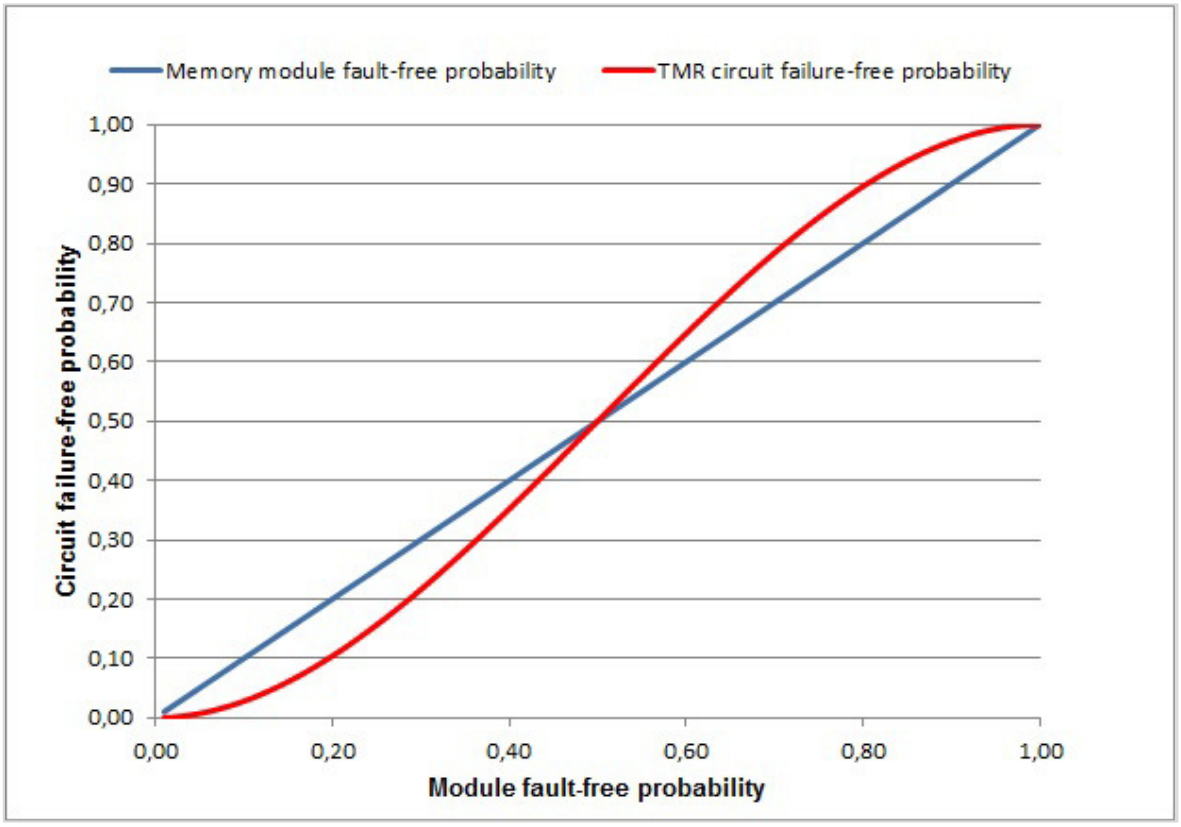


Figure 2.10. Failure-free probabilities for single memory module and TMR circuit

It is important to note that the use of redundancy is not always able to provide the improvement of the failure-free probability. In case that fault-free probability of module "M" (P_M) is less than 50% (0.5), then the TMR circuit has worse characteristics. It is possible to notice this effects and related dependences in Figure 2.10. It represents an example of the general truth that probability of failure-free circuit, even in redundant systems, cannot be obtained if the redundancy is applied at a level where the probability of non-redundant fault-free cell is very low.

The failure-free probability of TMR circuit depends strongly on the voter characteristic. First is analyzed the failure-free probability of TMR circuit with one voter (P_{TMRV}). Before the equation for this case is provided, it is important to note that the failure-free probability of the serially connected modules is calculated as multiplied fault-free probabilities of the modules [RAC12]. Therefore, if two

modules “A” and “B”, with fault-free probabilities P_A and P_B, are sequentially connected, the failure-free probability is represented as $P_{SEQ} = P_A \cdot P_B$. The Equation 2.21 defines the failure-free probability for single-voting TMR circuit, illustrated in Figure 2.9.

$$P_{TMRV} = P_V P_{TMR} = P_V\left(P_M^3 + 3P_M^2(1 - P_M)\right) = 3P_M^2 P_V - 2P_M^3 P_V \qquad (2.21)$$

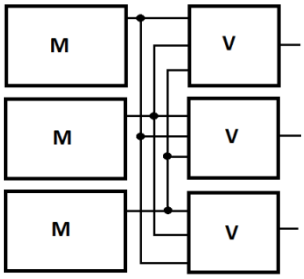


Figure 2.11. TMR structure with voters for each memory element

For the TMR structure with full-voting approach (P_{TMRF}), shown in Figure 2.11, the failure-free probability is expressed in Equation 2.22. The parameter P_V represents the probability of fault-free voter.

$$P_{TMRF} = P_M^3 P_V^3 + 3P_M^2 P_V^2(1 - P_M P_V) = 3P_M^2 P_V^2 - 2P_M^3 P_V^3 \qquad (2.22)$$

It is useful to explain how the Equation 2.22 is generated. If all modules are working without faults, the Equation 2.23 defines the failure-free probability of the full TMR circuit (P_{TMR0}). The index “TMR0” signifies that all modules are working fine.

$$P_{TMR0} = P_M^3 P_V^3 \qquad (2.23)$$

The TMR circuit operates also if one of the memory elements is faulty. The failure-free probability (P_{TMR1M}) is in this case expressed by Equation 2.24. Index “TMR1M” means that one memory module is faulty. There are three same possible cases wherefore is equation multiplied by 3.

$$P_{TMR1M} = 3(1 - P_M)P_M^2 P_V^3 \qquad (2.24)$$

It is also possible that one of three voters is faulty. As there are three cases, the probability should be multiplied by 3. The failure-free probability of the TMR circuit is in this case defined by Equation 2.25, where index “TMR1V” signifies that one voter is faulty

$$P_{TMR1V} = 3(1 - P_V)P_M^3 P_V^2 \qquad (2.25)$$

There is only one more case when the TMR circuit can operate without errors. If one memory module and one voter are simultaneously faulty, other two pairs still provides correct function of the circuit. The Equation 2.26 covers this case, where index “TMR1M1V” signifies the mentioned case. It is also multiplied by 3 because there are three possible situations covered by mentioned Equation 2.26.

$$P_{TMR1M1V} = 3(1 - P_M)(1 - P_V)P_M^2 P_V^2 \qquad (2.26)$$

The failure-free probability of TMR circuit is sum of the above mentioned cases only if the circuit operates without errors. Therefore, the Equation 2.27 provides above mentioned Equation 2.22.

$$P_{TMRF} = P_M^3 P_V^3 + 3(1-P_M)P_M^2 P_V^3 + 3(1-P_V)P_M^3 P_V^2 + 3(1-P_M)(1-P_V)P_M^2 P_V^2$$
$$= P_M^3 P_V^3 + 3P_M^2 P_V^3 - 6P_M^3 P_V^3 + 3P_M^3 P_V^2 + 3P_M^2 P_V^2 - 3P_M^3 P_V^2 - 3P_M^2 P_V^3 + 3P_M^3 P_V^3$$
$$= 3P_M^2 P_V^2 - 2P_M^3 P_V^3 \qquad (2.27)$$

The proposed way of the redundant circuit failure-free probability calculation will be used for all developed redundant circuits in the following text. Here is described the calculation way for known redundant circuit in order to prove the calculation approach.

In order to reduce the high hardware overhead produced by the TMR [BAZ00], [ROL03] and keep the design reliability high, implementation can be performed using the double modular redundancy with self-voting [TEI08]. A DMR circuit can be designed in two versions: single-voter version and double-voter version. Both versions are shown in Figure 2.12. In the literature is possible to find a “C element” as self-voting structure.

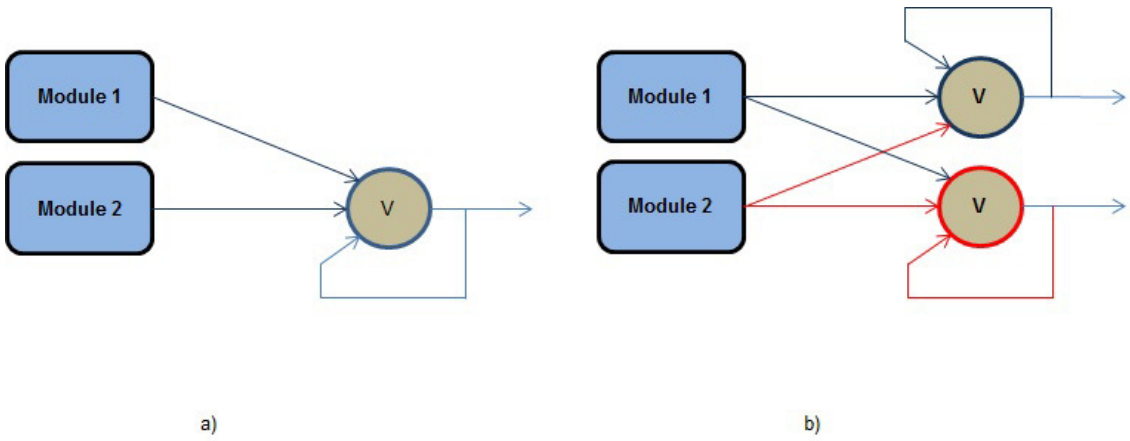


Figure 2.12. Double modular redundancy: a) single self-voting and b) double self-voting

A DMR technique that utilizes the self-voting circuits to mitigate the effects of SETs in digital integrated circuits is described in [TEI08]. The DMR solution requires up to 33 % smaller circuit area than the TMR and on the other hand provides the acceptable level of protection against SET pulses. DMR circuits for flip-flop data-paths, latch data-paths, and interface logic are illustrated. These circuits are implemented with the existing standard-cell library and standard EDA tools.

To justify the use of double modular redundancy (DMR) instead of triple modular redundancy (TMR), it is important to compare the probabilities of the failure-free standard TMR and failure-free self-voting DMR structures. The DMR approach with self-voting is used in order to reduce the hardware overhead of TMR and keeps the design reliability high. Self-voting is based on a 3-input majority voter that is configured to vote on two external inputs taking into account the state of its own output [PET12b] [PET12c].

Related to the previous work in this area [SCH09], the recommendation for accurate voting near active clock edge is to use the logic state on the flip-flop input as third input of the voter. After hold time

margin passes, the voter output can be again used for voting until next active clock edge arrives. The circuit shown in Figure 2.13 illustrates the solution for the mentioned problems.

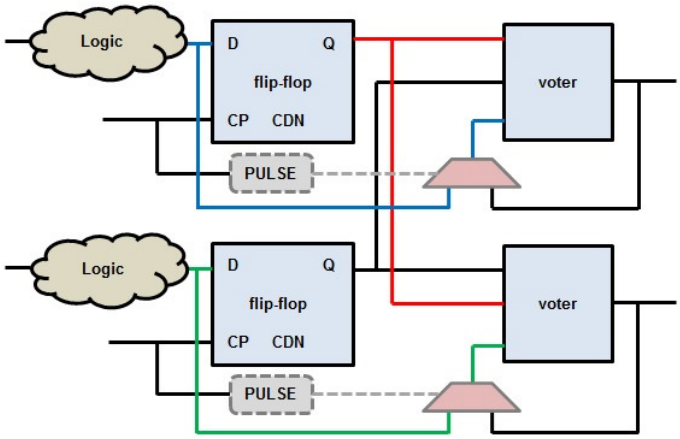


Figure 2.13. DMR redundant circuit

In the previous discussion, the TMR failure-free probability model is completely described. On the other hand, the failure-free probability of the DMR circuit with self-voting approach can be described in a similar way. Differences are caused by the feedback line of the self-voting circuit. The feedback line should provide a stable state for the correct voting but in case that some transient effects occur in the voting logic or in the flip-flop, the output of DMR circuit is not any more stable [SCH09]. The DMR circuit with multiplexed feed-back line, presented in [SCH09], provides better failure-free probability then simple DMR with feed-back line.

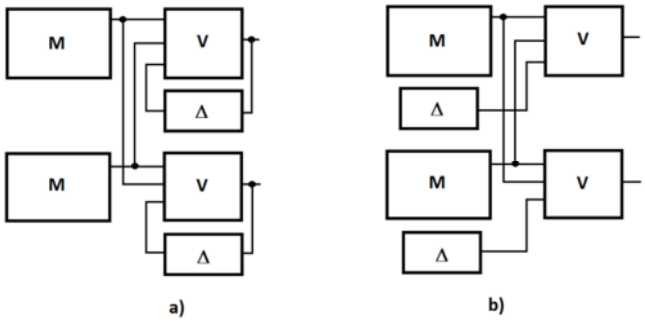


Figure 2.14. a) DMR circuit with delay line (ΔDMR); b) ΔDMR structure for mathematical analysis

In order to provide stability, it is required to implement an extra circuit in the feed-back line (ΔDMR). In real implementation, the mentioned extra circuit is a digital circuit which operates in relation with logic state of the voter output (V) and logic states saved in both memory elements (M). For extra digital circuit is assumed that it brings some delay and therefore is marked as Δ. The mathematical analysis is based on the block diagram presented in Figure 2.14.a. To simplify pretty complex mathematical analysis, the digital blocks in the feed-back lines (Δ) are analyzed as components with independent inputs (Figure 2.14.b). The fault-free probability of the feed-back logic depends on the fault-free probability of block itself, memory, and voter. Compared to the TMR circuit, where all elements are providing the data-flow without any feed-back, for the DMR approach story goes in more complex

feed-back dependences. Fault-free probability of the feed-back digital block itself is assumed to be 100%. The probability that the feed-back digital circuit will have wrong logic state on its output is related to the fault-free probabilities of both memory elements and the fault-free probability of the voter. Therefore, the fault-free probability of the feed-back circuit is given by Equation 2.28.

$$P_{\Delta} = P_M^2 P_V \tag{2.28}$$

For the DMR circuit presented in Figure 2.13.b, we can define the probability that complete circuit operates correctly (P_0).

$$P_0 = P_M^2 P_V^2 P_{\Delta}^2 \tag{2.29}$$

Case that just one of the memory elements (flip-flop) functions incorrectly and the outputs of DMR circuit have correct values (P_{1M}) is described by Equation 2.30.

$$P_{1M} = (1 - P_M) P_M P_V^2 P_{\Delta}^2 \tag{2.30}$$

If just one of the feed-back components is in wrong state, the fault-free probability of DMR circuit function is defined by Equation 2.31 ($P_{1\Delta}$).

$$P_{1\Delta} = (1 - P_{\Delta}) P_{\Delta} P_V^2 P_M^2 \tag{2.31}$$

The Equation 2.32 defines a case of failure-free DMR circuit operation (P_{1V}) when one of voters is in the incorrect logic state.

$$P_{1V} = (1 - P_V) P_V P_{\Delta}^2 P_M^2 \tag{2.32}$$

The probability of failure-free DMR circuit is presented in Equation 2.33 and it is sum of probabilities defined in Equations 2.29, 2.30, 2.31 and 2.32, with integrated Equation 2.28.

$$\begin{aligned} P_{DMR} &= P_0 + 2P_{1M} + 2P_{1\Delta} + 2P_{1V} \\ &= P_M^6 P_V^4 + 2(1 - P_M) P_M^5 P_V^4 + 2(1 - P_M^2 P_V) P_M^4 P_V^3 + 2(1 - P_V) P_M^6 P_V^3 \end{aligned} \tag{2.33}$$

Figure 2.15 shows a comparison diagram of TMR and DMR failure-free probabilities. It is important to notice that the voter reliability represents the most important factor for the redundant circuit reliability.

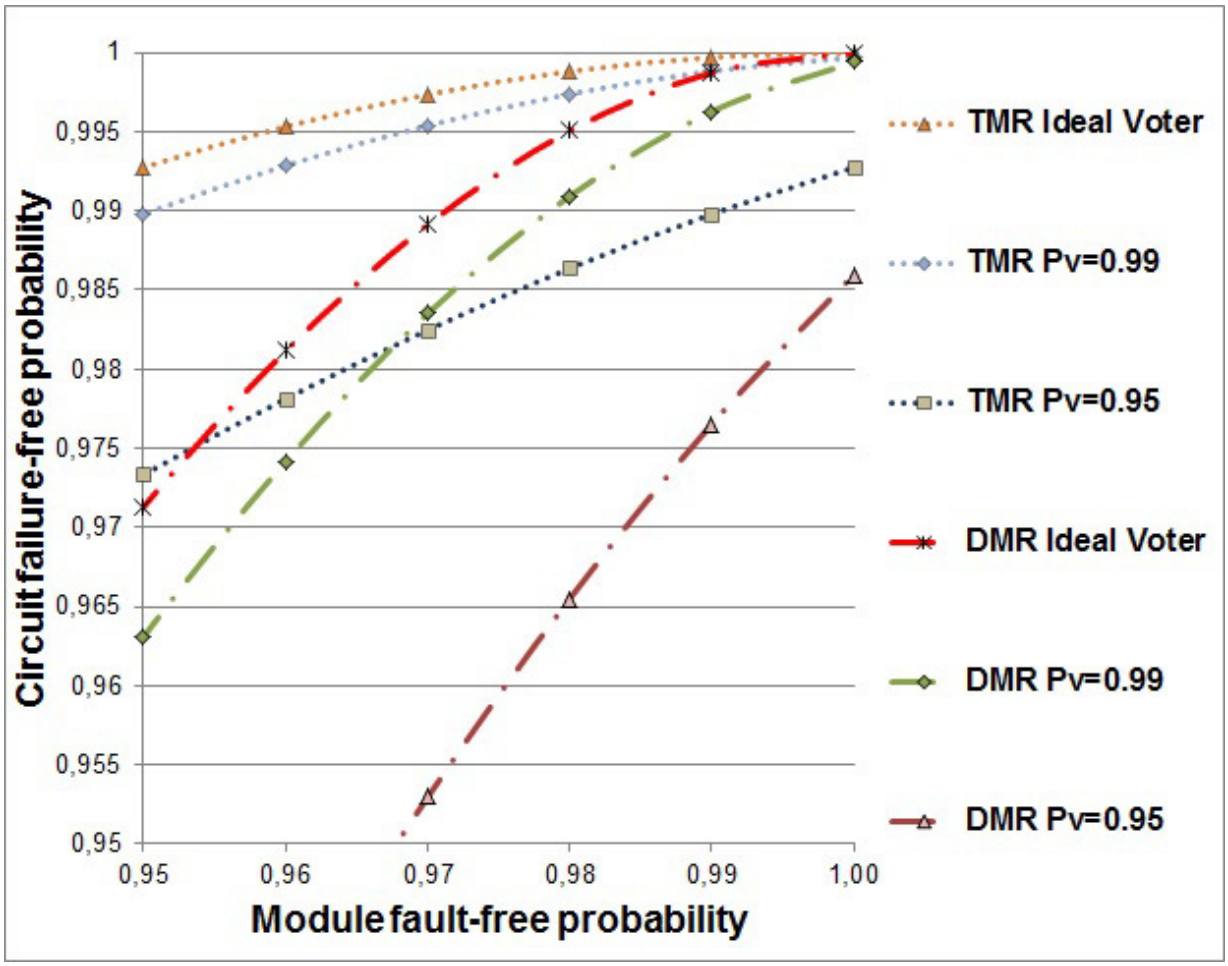


Figure 2.15. Single module, TMR and DMR failure-free probabilities

If we compare the TMR and DMR circuits, where TMR circuit integrates voters with lower fault-free probability and DMR circuit integrates voters with higher fault-free reliability, it is possible that DMR circuit has better reliability characteristics than TMR circuit. This effect is significant for the memory module fault-free probabilities higher than 95% (0.95). In Figure 2.14 this effect can be noticed for the TMR circuit with voter fault-free probability $P_V = 0.9$. The DMR circuits with voter fault-free probability $P_V = 0.99$, has better reliability than the mentioned TMR circuit. It is important to note that voter fault-free probability has high influence on the total failure-free probability. In case that the voter has lower fault-free probability in comparison with memory element, it can decrease the failure-free probability of redundant circuit significantly. This effect can be noticed form Figure 2.15.

It is important to note that the DMR circuit has acceptable failure-free characteristics only for the high reliable memory modules and voters ($P_M > 0.9 \; and \; P_V > 0.9$). In case that memory module has fault-free probability less than 0.9 ($P_M < 0.9$) the DMR circuit can't provide reliability required for the space applications.

The framework presented in [JIA07] is a circuit design technique for SEU immunity using dual-rail delay-insensitive asynchronous logic. This technique incorporates the double modular redundancy to

reduce the circuit area overhead that introduces the TMR. However, there is no information on the design flow used and the implementation results (maximum frequency reduction, and circuit area).

A novel double/triple modular redundancy (DTMR) technique for dynamically reconfigurable devices tested on a finite state machine is presented in [SHI08]. When the occurrence probability of SEUs is nearly equal to zero, the hardware cost can be reduced to nearly 2/3 compared to the TMR.

Regarding latchup effects, all the known techniques for latchup mitigation are classified in three main groups:

1. First group uses the current sensors at the board level to detect the excessive current induced by the latchup. The power supply of the affected device is switched-off and, after a pre-specified (long enough) period of time, reestablished again. This approach suffers from a serious drawback: the circuit state is destroyed and cannot be recovered. In addition, the board protection circuits must be designed with special care and the following requirements have to be met:
   - Proper decoupling of ICs,
   - Clamping of outputs with diodes when driving inductive loads,
   - Clamping of inputs with diodes if the input signal exceeds the power supply voltage,
   - Use of star grounds in high-current applications.
2. Second approach [EST78] is based on introduction of an epitaxial-buried layer process and reduction of the well resistivity. However, this modification incurs additional costs and may impact circuit performance (the breakdown voltage, for example).
3. Third approach [TRO83] uses guard rings (additional N-type and P-type regions) that break the parasitic bipolar transistor structure. This solution is very efficient but can result in excessive circuit area and, therefore, price.

Nicolaidis presents in [NIC06] a new SEL mitigation scheme that combines error correcting codes with intelligent power line implementation. It prevents the circuit damage and corrects errors caused by the latchup in a transparent manner. The area cost is low and the scheme can also be used to correct SEUs and reduce power dissipation. The integrated circuits made in nanometer technologies are more prone to soft-errors and, therefore, low-cost SEL mitigation schemes are especially needed in the case of commercial applications that cannot afford the high cost of radiation hardening. In this context, the proposed scheme represents a significant advantage with respect to other known approaches. However, there is currently no efficient and acceptable for commercial applications latchup mitigation solution that has minor impact on the circuit size, performance, and cost.

2.5. Standard Design Flow

Design flow represents a combination of electronic design automation tools required during the design of an integrated circuit. Moore's law has driven the entire integrated circuit (IC) implementation RTL

(register transfer level) to GDSII design flows from one which uses primarily standalone synthesis, placement, and routing algorithms to an integrated construction and analysis flows for design closure. The challenges of rising interconnect delay led to a new way of thinking about and integrating design closure tools. New scaling challenges such as leakage power, variability, and reliability will keep on challenging the current state of the art in design closure [SCA06].

The RTL to GDSII flow went through significant changes from 1980 through 2005. The continued scaling of CMOS technologies significantly changed the objectives of the various design steps. The lack of good predictors for delay has led to significant changes in recent design flows. Challenges like leakage power, variability, and reliability will continue to require significant changes to the design closure process in the future. Many factors describe what drove the design flow from a set of separate design steps to a fully integrated approach, and what further changes are coming to address the latest challenges. In keynote at the 40th Design Automation Conference entitled The Tides of electronic design automation (EDA), Alberto Sangiovanni-Vincentelli [SAN03] distinguished three periods of EDA:

> The Age of Invention: During the invention era, routing, placement, static timing analysis and logic synthesis were invented.
>
> The Age of Implementation: In the age of implementation, these steps were drastically improved by designing sophisticated data structures and advanced algorithms. This allowed the tools in each of these design steps to move forward with the rapidly increasing design sizes. However, due to the lack of good predictive cost functions, it became impossible to execute a design flow by a set of discrete steps, no matter how efficiently each of the steps was implemented.
>
> The Age of Integration: This led to the age of integration where most of the design steps are performed in an integrated environment, driven by a set of incremental cost analyzers [SCA06].

In the further text are provided details related to the standard design flow, which are used nowadays in the IC design. In order to involve the presented design technique, in this chapter are provided descriptions of extra steps as well as required technology files modifications.

Standard design flow can be divided in three phases. First design flow phase is used for hardware description and achievement of functional requirements. In this phase a designer should also take care about implementation characteristics and therefore it is very important to provide synthesizable hardware description language (HDL) codes. First design flow phase integrates the development of digital blocks as well as development of test benches. Development of digital blocks is based on the HDL language description. Verilog and VHDL are two used HDL languages. Test benches are important for the system functional verification, as well as verification process during the chip design.

Second design flow phase is related to conversion of the “high level” HDL languages to the “lower level” hardware – standard cells. Here are usually included tools for synthesis process and tools for physical design - placement of standard cells and routing.

In the third standard design flow phase are integrated all steps which are related to the important design rule checks (DRC) and match of the connections for the final layout and netlist – layout vs. schematic (LVS). Figure 5.1 illustrates a standard design flow block diagram, described using mentioned design flow phases.

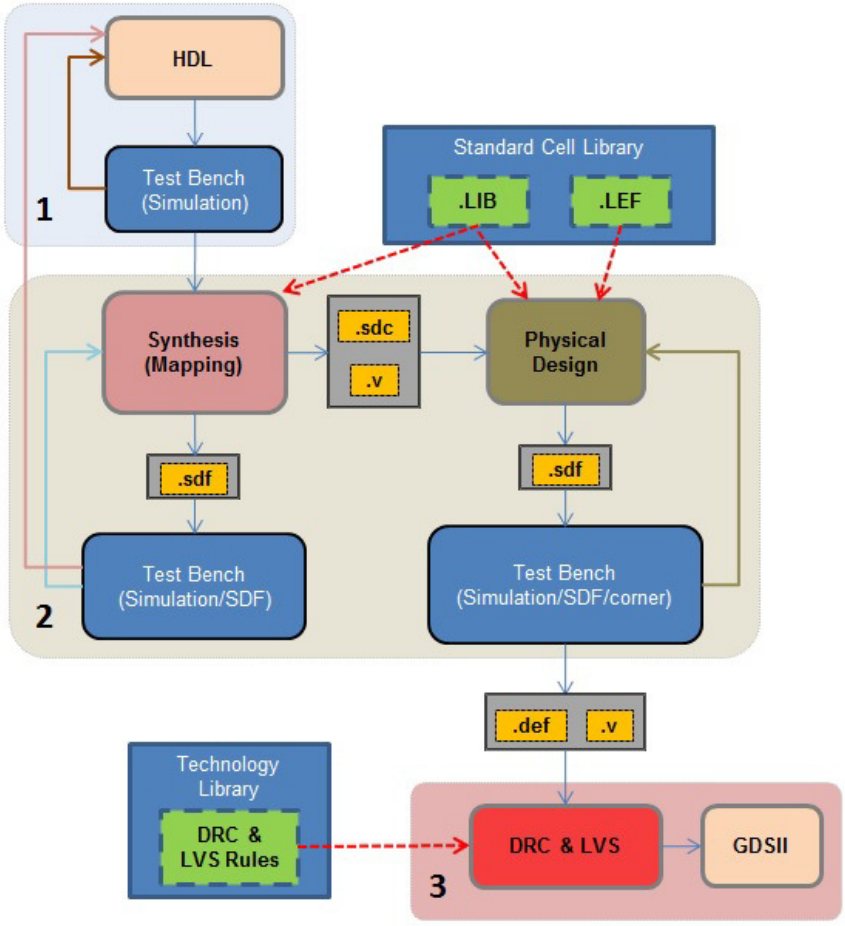


Figure 5.1. Standard design flow

HDL Description

During first design-flow phase, it is usual to iterate between test procedure and HDL editing procedure. This phase is known as HDL debugging process. After successful functional verification of the HDL code, the next standard design flow step requires tools able to parser the HDL code and maps the related hardware – synthesis.

Synthesis

Synthesis is an adaptation process that provides a hardware description using the components from the lower abstraction level. The original hardware description can be provided in either a behavioral view or a structural view, and the resulting description, after synthesis process is a structural view (i.e.,

netlist). The structural view is presented in the lower abstraction level. The synthesis process moves the system from a high-level abstraction to a low-level abstraction. Thus, synthesis either derives a structural implementation from a behavioral description or realizes an upper level description using finer components. As the synthesis process progresses, more details are added. The final result is a gate-level structural representation using the primitive (standard) cells from the chosen device technology. To make the process manageable, synthesis is usually divided into several smaller steps, each performing a specific transformation. The major steps are listed below:

1) **High-level synthesis** – transforms an algorithm into a register transfer (RT) level description, which is specified explicitly in terms of register transfer operations. Due to the complexity of transformation, it can only be applied to relatively simple algorithms in a narrowly defined application domain [CHU06].

2) **Register Transfer (RT) level synthesis** – analyzes a RT-level behavioral description and derives the structural implementation using RT-level components. It may also perform a limited degree of optimization to reduce the number of components [CHU06].

3) **Gate-level synthesis** – similar to RT-level synthesis except that gate-level components are used in structural implementation. After the initial circuit is derived, two-level or multilevel optimization is used to minimize the size of the circuit or to meet the timing constraints. In general, generic components are used in gate-level synthesis, and thus the synthesis process in this step is still independent of device technology [CHU06].

4) **Technology mapping** – Each device technology includes a set of predesigned primitive gate-level components, which can be cells of a standard-cell library. To implement the gate-level circuit in a particular device technology, the generic components have to map into the cells of the chosen technology. The transforming process is known as technology mapping. It is the last step in synthesis, and clearly the process is technology dependent [CHU06].

The functional netlist verification, which is generated during synthesis phase, is performed by the same Test Bench, used during the HDL development phase. Additional information which should be involved during synthetized netlist verification is the timing data file, provided by synthesis.

The timing data file or standard delay format (SDF) file contains the timing data generated by electronic design automation (EDA) tools for use at any stage in the design process [IEE13]. The data in the SDF file is represented in a tool-independent way and can include:

1) *Delays:* module path, device, interconnect, and port
2) *Timing checks:* setup, hold, recovery, removal, skew, width, period, and no-change
3) *Timing constraints:* path, skew, period, sum, and diff
4) *Timing environment*: intended operating timing environment
5) Incremental and absolute delays
6) Conditional and unconditional module path delays and timing checks

7) Design/instance-specific or type/library-specific data

8) Scaling, environmental, and technology parameters

Throughout a design process, several different SDF files are used. For simulation tests after synthesis phase is used the pre-layout timing data (SDF). In order to test the digital netlist with parasitic parameters after layout placement and route phase the post-layout timing data is used.

Physical Design

Physical design includes two important phases. The first phase is related to the refinement process between the structural and physical views, which derives a layout. The second phase involves the analysis and tuning of a circuit's electrical characteristics. The main tasks in physical design include floor-planning, placement and routing and circuit extraction [CHU06].

Floor-planning provides layouts at the processor and register transfer (RT) levels. It partitions the system into large function blocks and places these blocks in proper locations to reduce future routing congestion or to achieve certain timing objectives. Furthermore, floor planning may also provide a global plan for the power and clock distribution schemes. Placement and routing provides a layout at the gate level. The layout involves the detailed placement of cells and the routing of interconnecting wires [CHU06].

After the placement and routing are complete, the exact length and location of each interconnect are known, and the associated parasitic capacitance and resistance can be calculated. This process is known as circuit extraction. The extracted data are used to construct a resistance and capacitance network, which in turn is used to compute the propagation delays [CHU06].

In addition to the foregoing tasks, the physical design also includes design rule checking, derivation of the power grid, derivation of the clock distribution network, estimation of power consumption, and assurance of signal integrity [CHU06].

Here is important to notice that the mentioned SDF file involves more details on digital circuit after completed physical design phase compared to the SDF file generated after synthesis phase.

"Sign-Off" – DRC and LVS Check

After the physical mask layout is created for a circuit and for a specific design process, the layout is measured by a set of geometric constraints, or rules, for that process. The main objective of design rule checking (DRC) is to achieve a high overall yield and reliability for the design. To meet this goal of improving die yields, DRC has evolved from simple measurement and Boolean checks, to more involved rules that modify existing features, insert new features, and check the entire design for process limitations such as layer density. A completed layout consists not only of the geometric representation of the design, but also data that provide support for manufacture of the design [SCE06].

While design rule checks don't validate the functional correctness of the design, they are constructed to verify that the structure meets the process constraints for a given design type and process technology.

The Layout Versus Schematic (LVS) is the class of electronic design automation (EDA) verification software that determines whether a particular integrated circuit layout corresponds to the original schematic or circuit diagram of the design. LVS checking software recognizes the drawn shapes of the layout that represent the electrical components of the circuit, as well as the connections between them. This netlist is compared by the "LVS" software against a netlist generated after placement and routing.

2.6. Open Issues

Radiation effects (total dose, latchup, single event upsets) are one of the main concerns for the space microelectronics. As the radiation tolerance can often be achieved purely 'by design', i.e. by the schematics and layout of the chip, not only by process hardening, the use of commercial wafer fabs appears to be an attractive alternative for space microelectronics [ESA12].

Space applications strongly require more adaptability, because of inherently long design and maintenance period including continuous upgrade with new algorithms and technologies. There is a list of challenges related to the use of commercial ASIC technologies in space applications. The most important of them are:

1. The radiation-hardened technologies are expensive (qualification and quality requirements are severe) and commercially not attractive (small volume production).
2. There is no standard integrated framework of circuit design techniques that provides simultaneous SEU, SET, and SEL fault-tolerance.
3. There is no standard design automation flow of fault-tolerant digital ASICs and SOCs.

This thesis offers potential answers to the aforementioned challenges. The combination of a self-voting DMR technique for SEU and SET prevention and a power switching technique for SEL prevention is proposed as base of a novel fault-tolerant design methodology. This design methodology incorporates a modified design automation flow that easies the implementation of fault-tolerant ASICs and SOCs.

3. Fault-tolerant Circuits for Highly Reliable ASIC Designs

In order to have an automated design flow for the fault-tolerant circuits, it is essential to design the specific components which are not present in standard or radiation hardened design kits. Each component, described in this chapter, provides protection related to the particular effect. A circuit for the latchup protection is described first. After the latchup protection circuit description, details of the redundant circuits with separated power domains are presented.

3.1. Latchup Protection Circuits

Based on the Latchup Protection Technology (LPT) [MAX13], a SEL protection switch (SPS) for power domain control is developed and integrated in ASIC design flow using the Cadence low power flow [CPF13]. The idea is to partition the design in more power domains and to control these power domains with integrated latchup protection circuits instead of using the PCB-featured LPT. The most important advantage is combination of the redundant circuits (used for protection against upsets and transients) and High-Current-Flow protection circuits (used for protection against potential destructive latchup effects). Therefore, the presented approach provides protection against upsets, transients and latchup effects without expensive technology changes. Single event latchup protection switch and its operation mechanism are described in this section.

Figure 3.1 presents a block-diagram of SPS cell. It consists of current-flow sensor/driver, feed-back block, control block and communication interface for a power network controller (PNC).

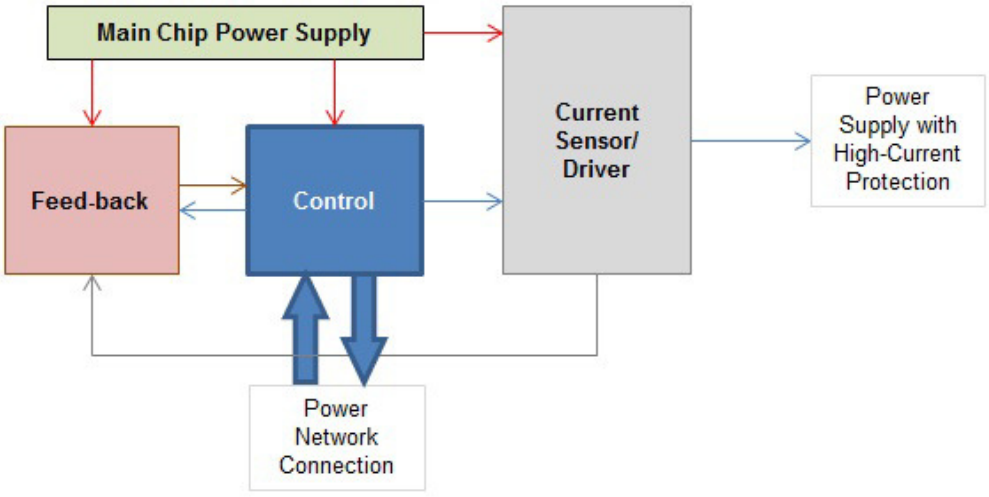


Figure 3.1. Single event latchup protection switch (SPS) block diagram

The most important component in the SPS circuit is a current sensor. It is in the same time a current driver, used to provide the power supply for the logic which needs to be protected against latchup effect. It is basically a PMOS transistor with wide channel, used in the linear (ohmic) region. The minimal MOS transistor channel width, in the SGB25V IHP technology is 330 nm. The current sensor/driver (CSD) transistor is designed in two versions: 5 µm and 10 µm widths. Two versions of

CSD are designed in order to provide enough current during normal operation (as driver) and to survive a potential latchup for different placement densities of standard cells. The feed-back logic, illustrated in Figure 3.1, provides important information about the current-flow status in the controlled circuit (standard cells). In case that sensor (CSD) detects a higher current than usual it will automatically provide related signal and feed-back circuit together with control logic turns-off the power supply of controlled standard cells, where the latchup or high-current flow occurred. The control-logic block communicates with power network. The SPS cell informs a power network about current status of the protection mode – whether a protection is activated due latchup effect. On the other hand, if protection mode was triggered, a power network provides information to the SPS cell when protection mode should be deactivated. More details on the mentioned power network are provided in Chapter 3.3.

Here is important to explain how a PMOS transistor provides enough power to standard cells during operation in the ohmic region and to notice a difference between quality of standard power supply and PMOS based power supply. In Figure 3.2 are presented output characteristics of the PMOS transistor for different voltages applied between gate and source terminals. Figure 3.2 shows the absolute values of currents and voltages. This is done because of the natural direction of currents and polarization voltages, which are in case of PMOS transistor inverted in comparison to the NMOS transistor.

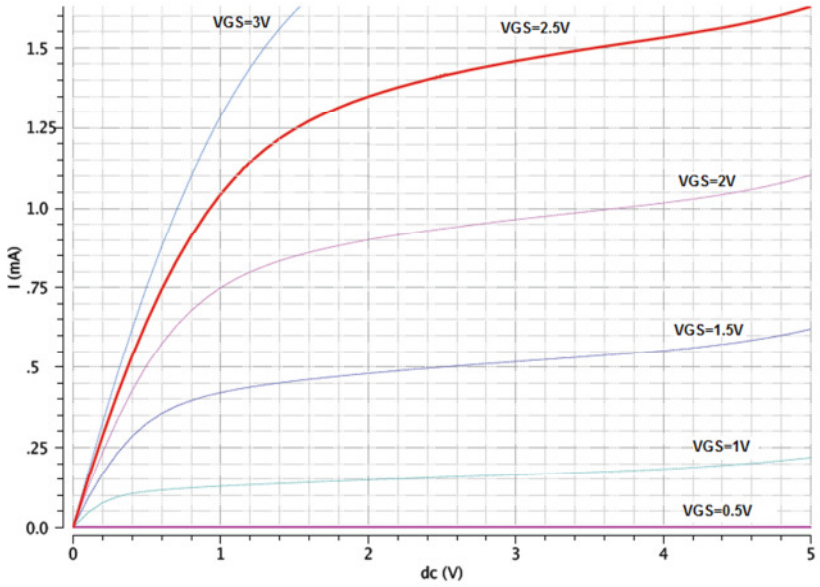


Figure 3.2. Output characteristics of PMOS transistor; Channel width: 5µm (IHP 250 nm)

If PMOS transistor is polarized on the input terminals (gate and source) with enough voltage to form the channel, it is possible to measure drain current for the different voltages on the output terminals (drain and source) [LIT96]. The PMOS transistor, used as sensor/driver, is connected in the SPS circuit regarding Figure 3.3.

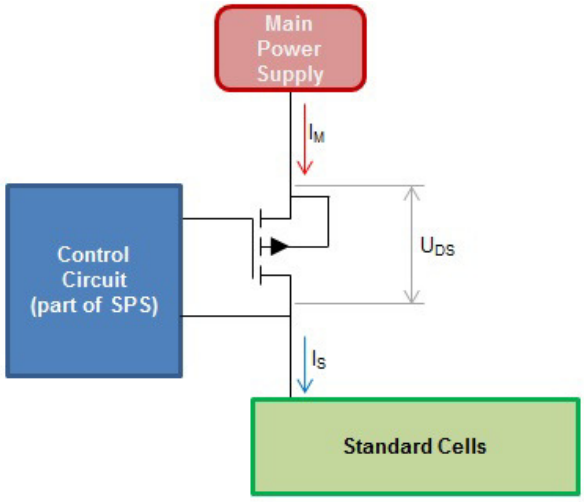


Figure 3.3. Block diagram of the current sensor/driver transistor

U_{DS} voltage is an important parameter which affects the quality of the sensor/driver transistor. For the 250 nm IHP technology, the digital circuits are nominally supplied by 2.5V. The worst case power supply is defined based on the 10% degradation and therefore it is 2.25V. On the other hand, the best case power supply goes up to 2.75V. It was mentioned before that during normal operation, in order to have correct sensitivity of the high current flow, the sensor/driver transistor should operate in the ohmic (linear) region. Following the output characteristic, presented in Figure 3.2 as red line, it is possible to notice that linear region goes up to 0.9V of drain-source voltage and 1mA drain current. If standard cells (the green block in Figure 3.3) require more power, the goal is to provide more current with lower U_{DS} voltage. It is clear that in this case the required scenario is not possible, because the increase of the current increases the U_{DS} voltage, what directly provides lower voltage for the standard cell power supply. Therefore, only way is to find a trade-off between required current intensity for the supplied circuit (standard cells) and the voltage of the sensor/driver transistor.

The maximal current provided by the sensor/driver transistor of 5µm channel width is between 250µA and 750µA. The maximal voltage drop is about 0.6V for operational mode when transistor is used as driver. Therefore, the power provided by this transistor is not more than 450µW, what is enough to supply 2 flip-flops and few combinational cells. Once more, it is important to note that a SPS controls a small area of standard cells. Compared to the PMOS with channel width of 10µm, it is possible to notice better driver performances. For a voltage drop of 0.6V it can provide current of 1.5mA. Therefore, use of stronger driver (channel width of 10µm) provides possibility to drive double more cells than previous one. The second advantage of the driver with higher current capacity is that it has better characteristics for higher speed digital circuits. More details how the SPS is integrated with DMR-based redundant circuit are provided in Chapter 5.

After describing the sensor/driver transistor, it is very important to describe a control circuit (control and feed-back blocks) of the SPS cell (presented Figure 3.3). In further text are provided functional descriptions of two similar SPS circuits. The difference is based on the communication interface to the power network.

3.1.1. SPS Type 1

Before providing a detailed functional description of SPS, it is useful to list its modes of operation with short description.

1) Redundant mode (RM) – SPS enables power supply for redundant digital circuits in order to provide the protection against SEUs and SETs and SELs. In this mode of operation logic states should not be lost during latchup protection phase.
2) Single mode (SM) – Redundancy is deactivated (Single mode) – only one redundant module operates. Single mode doesn't provide protection against SEUs and SETs. Protection against SELs is active but during latchup protection phase the logic states are lost.
3) Deactivated mode (DM) – in this mode all controlled standard cells (circuits) are switched-off. This mode is usually used in case when a controlled section is destroyed due irradiation effects and is not usable any more.

The SPS circuit type 1 schematic-diagram is illustrated in Figure 3.4 (next page). The mentioned schematic-diagram represents a controlled power supply only for one redundancy. For DMR approach two such circuits should be used. On the other hand, for the TMR approach it is required to use three SPS circuits. The functional description of circuit is based on the mechanism how the power supply for standard cells is switched-off in case that high current flow is detected. Communication between sensor and control is provided by feedback line between the output CMOS pair (P5 and N4) and the input control logic.

The output CMOS pair consists of one PMOS transistor P_5 which provides enough power for a SEL protected logic and one NMOS transistor N_4 for discharge of the residual electricity in a SEL controlled circuit after latchup occurrence.

At the moment when latchup occurs, in some of the controlled standard cells, through the driver/sensor PMOS (P_5) transistor flows more current than usual and the voltage between source and drain is higher. If we follow the output characteristic of PMOS transistor, presented in Figure 3.2, it is easy to notice that higher current flow provides higher U_{DS} voltage. Related to Figure 3.3, the voltage distribution is changed. In usual operation environment the voltage of the driver/sensor transistor goes between 0.01V and 0.6V. On the other hand, the voltage provided for the standard cells, in worst case is between 2.5V and 1.9V. In case that a latchup occurs, voltage U_{DS} increases up to 2.5V. That means - the voltage on the drain of the PMOS transistor is going down (VDD_a or a blue line in Figure 3.4). Output of the mentioned CMOS pair (P_5 and N_4) is used as power supply for the standard cells. In the same time, the output of the P_5 and N_4 CMOS pair is connected to control circuit by feed-back line. The feed-back line provides information for the control logic about the actual current-flow status and therefore provides protection against SEL effect.

Lower voltage on the output CMOS pair (for better understanding it is useful to follow the blue line in schematic illustrated in Figure 3.3), provides activation of PMOS transistor P_6. This implies higher current I_0, which flows through resistor R_0 and activates the NMOS transistor N_0. After activating the

NMOS transistor N_0, the output pin "TSTART" should be at low logic state (logic zero). Resistor R_2 is used as a pull-up resistor and in the same time to limit the current I_2 in the moment when transistor N_0 is activated. Pull-up resistor R_2 controls the state on "TSTART" pin to be stable during N_0 transistor is in inactive state (the gate of N_0 transistor is driven by low logic state). The "TSTART" pin is used for the communication with the power network controller and it indicates that the latchup has occurred. After the latchup is indicated, an external programmable timer, which is a part of the power network, starts to count up to defined period. Defined period should be defined related to the technology and the characteristics of the SPS circuit. The most important parameter for the power network is ability to discharge the residual electricity in the group of standard cells or part of the circuit where the latchup was detected.

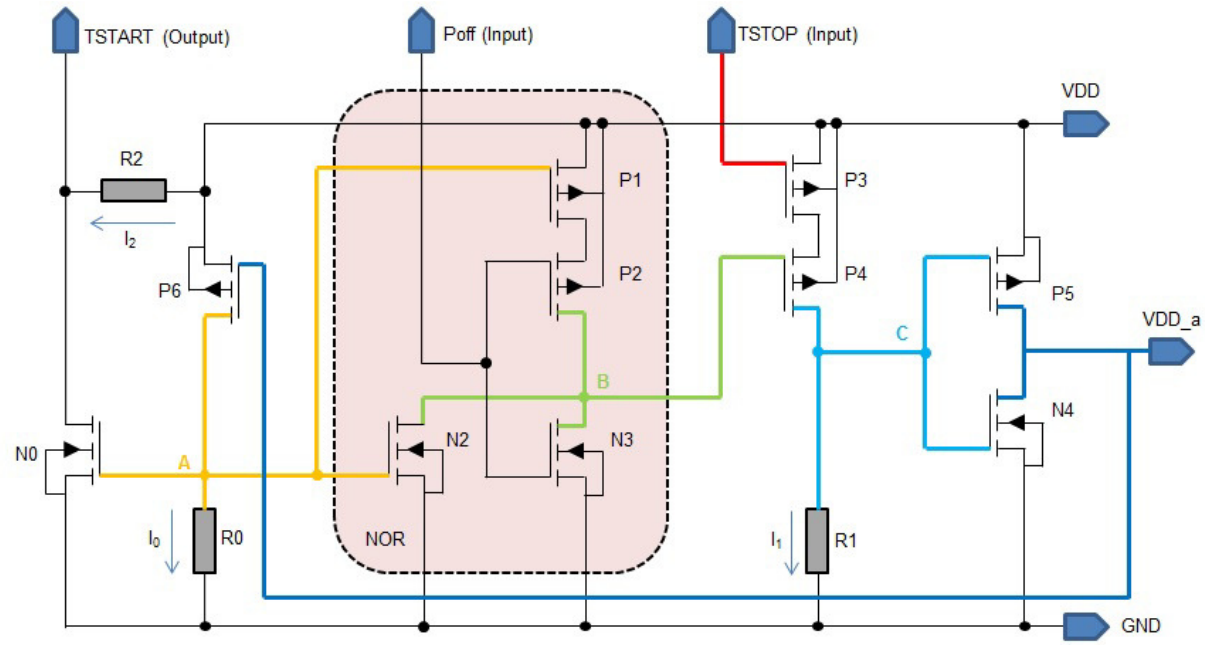


Figure 3.4. Schematic of SPS circuit 1

In order to understand how the SPS circuit operates in the redundant mode, we should follow the logic state of node "A" (orange colored signal). If we carefully look in the circuit, it is possible to notice a segment which consists of transistors P_1, P_2, N_2, and N_3. It is easy to recognize the "NOR" circuit. However, in order to better understand following text a truth table of the "NOR" circuit is provided in Table 3.1. The input signal "Poff" is low active signal used to control power switching function. It is generated by power network controller. Advantage of the proposed latchup protection switch is based on the selective redundancy activation, which can be controlled by "Poff" pin. More details on the different power modes are provided in Chapter 3.3.

Table 3.1. Truth table for the NOR circuit

A	B	NOR
0	**0**	**1**
0	**1**	**0**
1	**0**	**0**
1	**1**	**0**

In the redundant mode, when all circuits are operating correctly, both inputs of the “NOR” circuit are driven by low logic states and output (node “B”) is therefore in a high logic state. The node “C” is also set to low logic level because of pull-down resistor R_1. This causes that PMOS transistor P_5 is active and therefore provides enough current and voltage on the controlled output VDD_a. High logic state of the controlled supply, VDD_a, provides that transistor P_6 is deactivated. In this case, the node “A” is set to the low logic state. As it was mentioned before, during operation without latchup or short-circuit effects, the output of the “NOR” circuit is set to high logic level (node “B”) what stops the transistor P_4. The “TSTOP” signal is high active and it is used to control the duration how long the power should be turned-off after latchup has been detected (related to the activation of the “TSTART”). The “TSTOP” signal is generated in the power network control unit. In the idle state, the signal “TSTOP” is set to low logic state (P_3 active) but the output from “NOR” circuit is at high logic state (P_4 inactive) and the resistor R_1 provide stable gate polarization of transistor P_5 (active) and transistor N_4 (inactive).

In case of higher current flow, than usual, the drain-source voltage of the transistor P_5 increases. As the source terminal of the transistor P_5 is connected to the main VDD supply, the voltage of the drain terminal (VDD_a) decreases in relation to the ground (GND). This process reduces the voltage and therefore the dissipation in the controlled standard cells or circuits. As the VDD_a voltage is going under threshold voltage, the feed-back line provides the activation of the transistor P_6. In this moment, during protection phase, the node “A” goes in high logic state. It activates then the NMOS transistor N_0, which set the output “TSTART” at low logic state and informs a power network controller that latchup occurred in the controlled section.

In Figure 3.5 are showed the timing diagrams of the described protection mechanism. The protection phase starts after a latchup is detected. Protection phase is active until “TSTOP” signal is activated by power network. Colors of the signals in Figure 3.5 are related to the colors of the nodes and terminals illustrated in Figure 3.4.

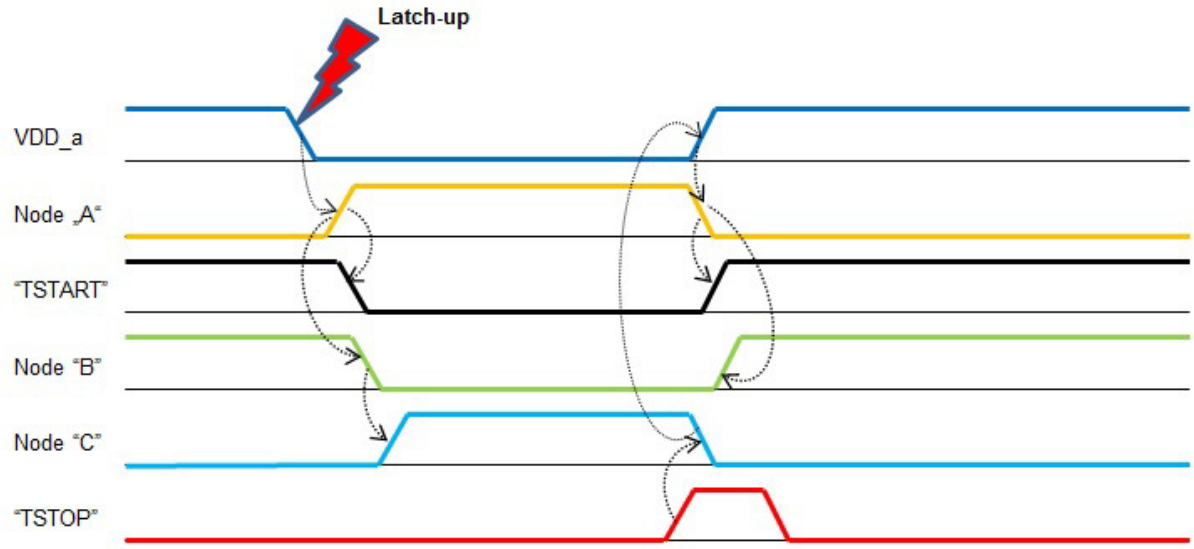


Figure 3.5. Timing diagrams of the SPS circuit 1

On the other hand, the node “B” (output of the “NOR” circuit) is set to low logic state. Above mentioned procedure provides activation of the transistor P_4. It was mentioned before that “TSTOP” input is controlled by power network. During the period of the SPS latchup protection phase, the input “TSTOP” is set to low logic state. Therefore, short after latchup effect has been detected, the node “C” goes to high logic state. That was the most important procedure in order to discharge the residual electricity in the controlled standard cells or circuits, where latchup effect has been detected.

When power network controller sends a pulse on the “TSTOP” input (after programmed time), the PMOS transistor P_3 is deactivated. It stops the current flow required for keeping high logic state of the node “C”. Pull-down resistor R_1 defines low logic state of the node “C” and transistor N_4 is not active any more. As the gate terminal of transistor P_5 is then set to low logic state, transistor P_5 starts to conduct the current. Therefore, on the drains of transistors P_5 and N_4 is again the voltage near to the VDD. Standard cells which were affected by latchup effect have again the power supply as before protection procedure.

After the protection phase is finished, the feedback line is set to high logic state, what deactivates the PMOS transistor P_6. The pull-down resistor R_0 provides low logic state in the node “A”. As effect, NMOS transistor N_0 is also deactivated and low active output “TSART” will be set to high logic state because of the pull-up resistor R_2. The “TSTART” informs a power network controller that this part of circuit has no more any short-circuit (latchup). Following the inputs of the mentioned “NOR” circuit, the node “A” and “Poff”, it is easy to notice that they are, as before protection procedure, both set to the low logic state. This condition forces the output of the “NOR” circuit to the high logic state. Gate terminal of transistor is P_4 is polarized by high logic state, what for sure stops the current flow through transistors P_3, P_4 and resistor R_1. In the proposed circuit, presented in Figure 3.4, the resistor R_1 is used as a pull-down, in order to stabilize the gate terminals polarization of transistors P_5 and N_4 during change of the logic states.

In case that controlled section (standard cells) need to be switched-off for a longer period of time, due the power saving procedure or dangerous hardware damage, the power network controller sets “Poff” input to high logic state. In this case the node “B” is set to low logic level and activates the PMOS transistor P_4. As the PMOS transistor P_3 is activated because of the “TSTOP” signal, the node “C” is then set to the high logic state. This stops the transistor T_5 and activates the transistor N_4. Controlled power supply (VDD_a) is in this case switched-off. The circuit can be easy switched-on again by setting the “Poff” terminal to the low logic state.

In the following text is described second version of SPS circuit. Second version is designed in order to provide better power-off control as well as behave during permanent short-circuit effect in the power controlled standard cells. It is easy to notice that power-off control is dependent on “TSTOP” logic state. In case that “TSTOP” is stack at high logic state, it is impossible to switch-off SPS circuit using “Poff” signal. If this version of SPS circuit is used for latchup protection it is important to take special attention during design of SPS power network controller. It should provide independent “TSTOP” signal generating – “TSTOP” activation and deactivation should be dependent just on the time related

parameters, not on “TSTART” signal. This is important especially in case of permanent short-circuit detection due hardware damage.

3.1.2. SPS Type 2

Schematic of the SPS circuit type 2 (Switch type 2) is presented in Figure 3.6. The functional description of SPS circuit type 2 is similar to the previous described SPS circuit type 1. Controlled power supply is provided by CMOS pair (transistors P_5 and N_4 shown in Figure 3.6). PMOS transistor P_5 provides the power supply for controlled logic (standard cells). The W/L ratio of the driver/sensor transistor is directly proportional to the maximal current density. Communication between sensor/driver transistor and control block of SPS circuit is provided by feedback line between the output CMOS pair (P_5 and N_4) and the feed-back control logic (inverter - P_8 and N_6). The output CMOS pair consists of one PMOS transistor P_5 which provides enough power for protected logic against latchup effect and one NMOS transistor N_4 for discharge of the residual electricity in controlled digital circuit after latchup occurrence. The protection algorithm can be described similar as it is provided for SPS Circuit type 1.

After latchup effect occurrence, the drain current of transistor P_5 increases. Similar as it was described before, the increase of the drain current, directly affect the drain-source voltage of sensor/driver transistor. As the source voltage is fixed (VDD), increase of source-drain voltage causes that the drain voltage decreases in comparison to the referent ground voltage (GND). Therefore, the power supply of digital logic where latchup effect has been detected is going from the high voltage level near to the low logic level. This causes that the circuit logic states are lost. Reduce of the drain voltage, directly reduces the total power dissipation in the controlled standard cells. This procedure provides protection against potential hardware destruction. Following the feedback line (from VDD_a to the gates of CMOS pair N_6 and P_8 presented in Figure 3.6), it is possible to notice that information related to the actual status of VDD_a is also distributed to the SPS circuit control block.

The terminals used for communication between power controller and SPS circuit are: “Poff”, “TSTOP” and “TSTART”. “Poff” is used to control power switch from outside and it is low active signal. Using this pin it is possible to switch-off the controlled section in order to provide power save mode. “TSTOP” is generated by power network controller and it is used to reactivate the power supply for the standard cells where latchup was detected. “TSTART” is low active terminal and it is generated by SPS circuit in order to inform the power network that latchup has been detected in the standard cells controlled by actual SPS circuit. As it is possible to notice in Figure 3.6, the SPS circuit (type 2) uses two “NOR” gates in order to process the “Poff” and the “TSTOP” signals. Transistors N_8 and N_9 are used for the “TSTART” signal generating.

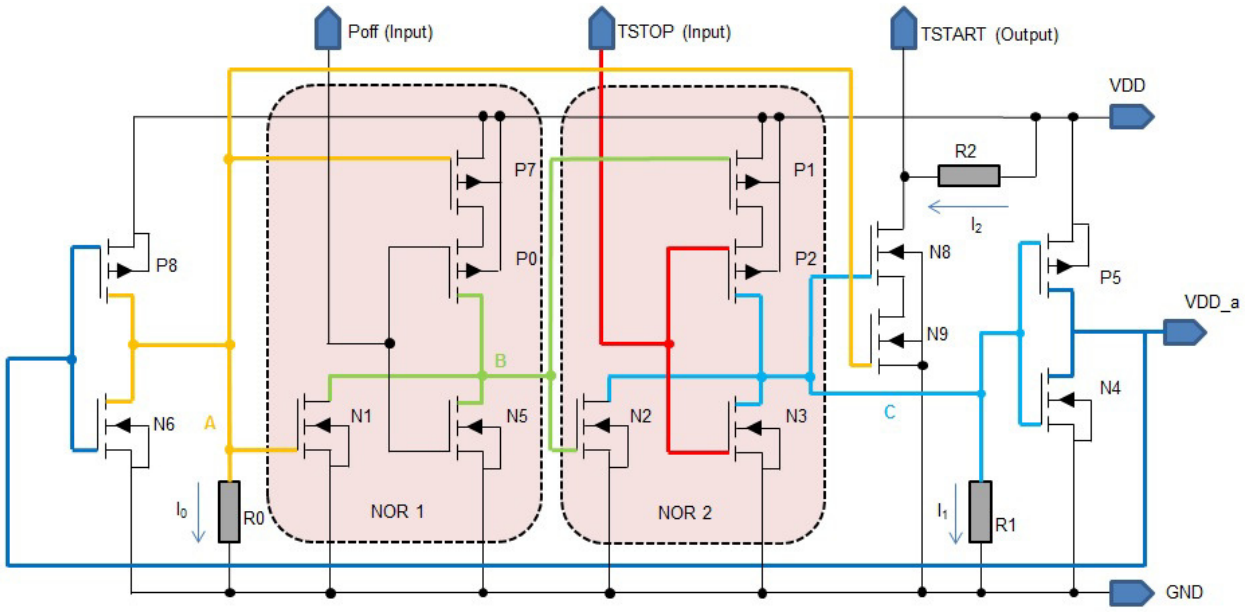


Figure 3.6. Schematic of SPS circuit type 2

The functional description starts from the point when all controlled circuits are operating normally. The VDD_a pin is set at high logic state (about 2.5V). Related to Figure 3.6, the node “A” is set to the low logic state (0V). As the “Poff” is low active control pin, defined by power network controller, it is also set to low logic level in order to provide normal operation of the SPS power circuit type 2. The node “B” is therefore set to high logic level, what directly set the node “C” to low logic level. This provides that the PMOS driver/sensor transistor P_5 is activated.

Similar to the previously described SPS circuit type 1, in Figure 3.7 are shown timing diagrams of the related signals and nodes of SPS circuit type 2.

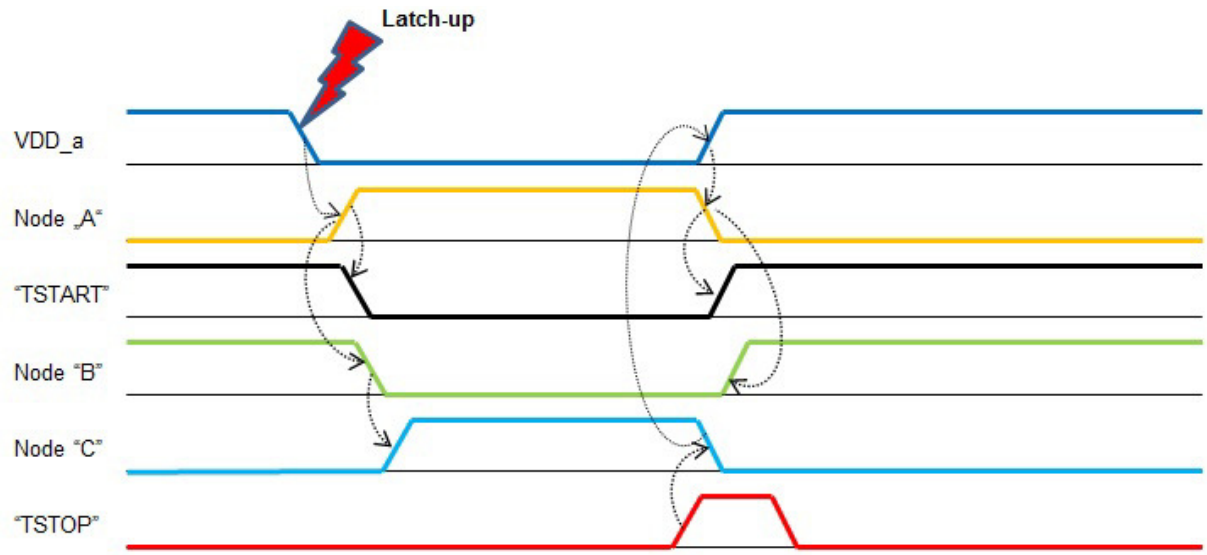


Figure 3.7. Timing diagrams of the SPS circuit type 2

In case that a latchup occurs, through the sensor/driver transistor (P_5) is detected a higher current flow. This provides, as it was mentioned before, that the drain terminal (VDD_a) voltage decreases.

When the voltage on VDD_a pin is less than threshold, the CMOS invertor (composed of the transistors P_8 and N_6) changes the output (node "A") from low to high logic state. The output of the "NOR" circuit, node "B", also changes to low logic state. Therefore, the output of the second "NOR" circuit (in Figure 3.6 output of the NOR2 circuit is represented as node "C") changes to the high logic state and provides that the sensor/driver transistor is deactivated. This process provides that the power supply for the latchup-controlled standard cells is switched-off. On the other hand, the high logic state of the node "C" provides that NMOS transistor N_4 is activated. Transistor N_4 provides fast discharge of the residual electricity, accumulated in the standard cells during latchup. As it was mentioned before, the only way to stop latchup effect is to switch-off power supply. In order to provide a complete switch-off procedure, the NMOS transistor N_4 is obligate. Parallel with described switching procedure, transistors N_8 and N_9 are used for "TSTART" signal generating. As the node "A" is set to the high logic level and the same is with node "C", it is easy to notice that both transistors N_8 and N_9 are activated. Resistor R_2 is used as pull-up resistor in order to provide stable logic state when one or both NMOS transistors are inactive. Therefore, when a latchup effect occurs, described procedure provides that logic state of the "TSTART" output is changed from high logic state to low logic state. This signal informs a power network controller that a latchup is detected in the actual controlled section. Resistors R_0 and R_1 are pull-down resistors and they are used to provide stable logic states during transitions of the nodes "A" and "C", respectively.

After time period defined in the power network controller, "TSTOP" signal is activated. This causes that node "C" changes from the high logic state to the low logic state. Transistor N_8 is deactivated and "TSTART" output is set to high logic state ("TSTART" is inactivated because it is low active signal). Low logic state of node "C" provides that NMOS transistor N_4 is deactivated and PMOS sensor/driver transistor P_5 is again activated. The controlled standard cells are switched-on. The VDD_a output, through feed-back line and CMOS inverter – P_8 and N_6, provides that node "A" is set to low logic state. This automatically stops the NMOS transistor N_9, used for generating of the "TSTART" signal. Output of the "NOR" gate, node "B" is set to high logic state, what for sure provide stable low logic state at node "C".

As it was mentioned before, the advantage of presented SPS circuit type 2 is based on the detection of permanent short-circuit in the controlled logic. After "TSTOP" signal is set, the "TSTART" signal is immediately deactivated. As the VDD_a is in case of the permanent short-circuit impossible to set to high logic state, the logic states of other nodes in the SPS circuit will not be changed. This provides that standard cells are still protected and after "TSTOP" pulse the output "TSTART" will be automatically activated. The power network controller will be again informed that latchup is detected. This procedure can be repeated more times and regarding the power network controller organization, the controlled section can be permanently switched-off. Permanent switch off can be provided by power network controller and "Poff" terminal.

3.2. Redundant Circuits with Latchup Protection

During digital system implementation, instead of using single memory element (flip-flop or latch), it is possible to use different redundant forms. Redundancy is always required for a fault-tolerant design. The higher redundancy, the better protection, but also the larger chip area, power consumption, and cost are. The automated fault-tolerant design, presented in thesis, uses latchup protection in combination with redundant circuits. Therefore, in this section are provided descriptions of TMR and DMR redundant circuits, which support the above mentioned design methodology. This requires significant modifications of the standard TMR and DMR circuit designs.

In the thesis is proposed a combination of the SEL protection switch and the independent redundant power domains as a foundation of the latchup protected redundant circuits. The goal of this research is a fault-tolerant system development using standard design tools with optimal resource usage (power, area and timing degradation). Therefore, it is necessary to trade-off between the redundancy rate and the cost for each and every application.

Compared to the TMR circuit, main disadvantages of the standard DMR circuit with self-voting are:

- High sensitivity on the transient effects near active clock edge.
- High sensitivity on the upset effects.
- Lose of the circuit state during the latchup protection phase.

The mentioned disadvantages disable DMR circuits to be used in the fault-tolerant digital systems. Therefore, the modifications are required. To avoid the error in case when a transient pulse changes the input value of a flip-flop in the voter logic during the active clock edge, some extra logic is added in the voter feedback line. This extra logic also provides normal operation during the upset effects as well as during latchup protection phase.

3.2.1. TMR Circuit with Latchup Protection

Compared to the TMR circuit described in Chapter 2, in this chapter is discussed a modified version of TMR circuit which supports the latchup protection technique. Latchup protection is based on the redundant power supply lines with integrated current intensity control. Related to the standard TMR circuit, this extended version is developed in order to handle destructive latchup effects without losing the required functionality.

The implementation of latchup protection technique in redundant circuits, in fact enables use of more independent power supplies. Therefore, when latchup effect occurs, in one of three power supply domains, other two power supply domains are working without any disturbance. Following text provides description of the extended TMR circuit as well as protection cases.

In Figure 3.8 is presented a standard TMR circuit, supplied by three independent power sources (Power Domain 1, Power Domain 2 and Power Domain 3). For example, in case that high current flow (induced by single event latchup effect) is detected in the digital circuit, supplied by “Power Domain 3”, the related power switch turns off the controlled power domain. Outputs of the “switched off” circuit are

set to low logic level. Therefore, the related inputs of voters in power domains 1 and 2 are driven by '0'. In Figure 3.8, the red line represents a signal, which is switched off (low logic state) because of latchup protection procedure. During the latchup protection phase, other two redundant circuits continue with normal operation. A problem can arise if particle hit the operating part of digital circuit and induce a transient or an upset during latchup protection phase. In this case, the presented circuit cannot operate properly any more.

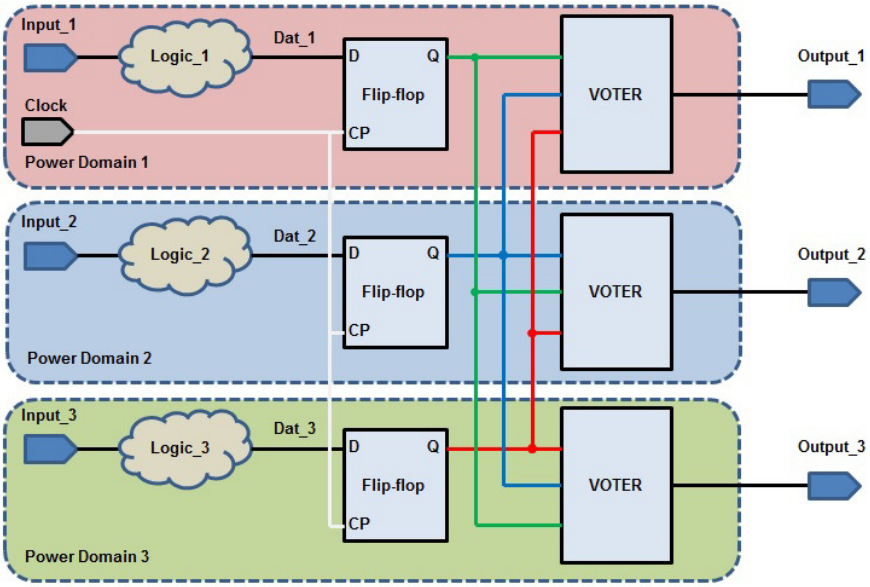


Figure 3.8. Standard TMR circuit with different power domains

Advantage of TMR circuit with controlled power distribution is a power saving mode. In the power saving mode one or two power domains can be switched off in order to decrease the overall power consumption. Redesign of the standard TMR circuit is in this case related to the controlled disabling of the voter input signals, which are defined by switched off (redundant) power domains.

The above mentioned requirements are defining the cases within the TMR circuit with power redundancy operates properly. All cases are listed below.

TMR circuit should operate properly:

- If one of three power domains is switched off
- If two of three power domains is switched off
- If one memory element (flip-flop), due upset effect contains a wrong logic state
- If one voting circuit (voter), due transient effect sets a wrong logic state on its output.

Above mentioned cases are defining the requirements for additional logic which should be implemented in TMR circuit. The additional logic block is named "data keep" and it compares the logic states on the all three voter inputs with states on the flip-flop input and flip-flop output during an active clock edge. The status of the redundant power supplies also represents important inputs for the "data keep" block. During the latchup protection phase, when one redundant module is switched-off, the TMR circuit operates as DMR (double modular redundancy).

Figure 3.9 illustrates a block diagram of “data keep” digital circuit. In the TMR circuit, the “data keep” block is connected to the both redundant flip-flop outputs, power supply relevant signals and to the data input and data output of the “domestic” flip-flop. Term “domestic”, compared to the term “redundant” is related to the components under the same power domain. Therefore, it is possible to define “redundant” components, which are supplied by other power supply domains. “Data keep” block is required for the TMR fault-free operation during the latchup protection phase or during the power-save mode.

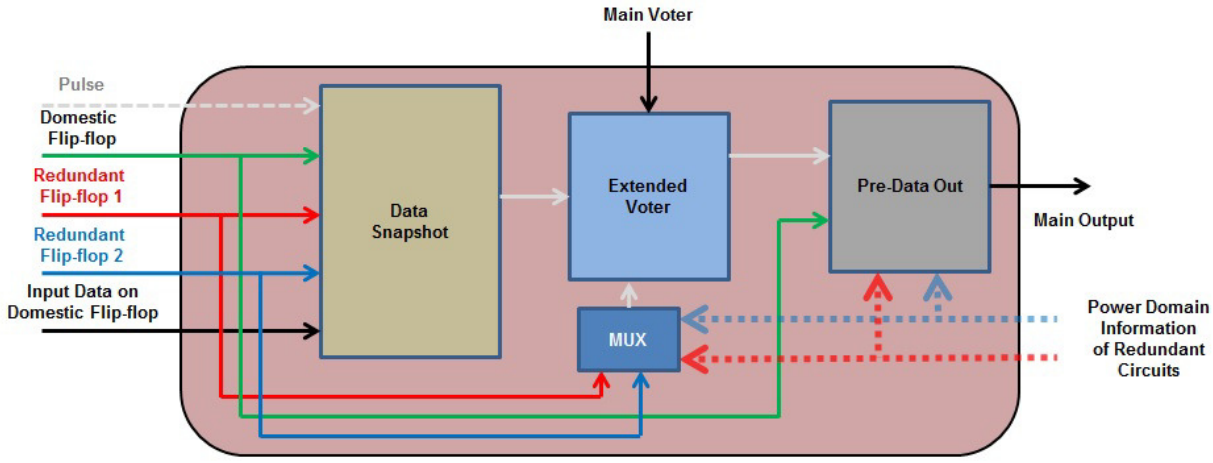


Figure 3.9. “Data keep” block diagram for TMR circuit with latchup protection mode

It is important to notice that in this type of TMR circuit the outputs of sequential elements are dependent on the power domains, which are present as digital inputs (dotted lines). Pulse provides enable signal for latch component to save the data during active clock edge. In order to save the correct value, the signal on the latch input is generated based on the signals of the other redundant circuits as well as information provided by redundant power supplies. Therefore, “data snapshot” block is a circuit composed of a latch and some combinational cells. Extended voter block provides an extra voting which enables the logic value defined by “data snapshot” block. MUX block, presented in Figure 3.9, is used for defining the correct value for the extended voter in case that one of the redundant supplies is switched off. In case that both redundant power supplies are switched off, the “pre-data out” block provides bypass form the “domestic” flip-flop output to the main output. The truth table of TMR “data keep” block is presented in Table 3.2.

Table 3.2. TMR data keep block truth table (when domestic power supply is active)

Input data Domestic Flip-flop (Latch)	Domestic flip-flop Output (FF1)	Redundant flip-flop Output (FF2)	Redundant flip-flop Output (FF3)	Redundant Power Supply (PR1)	Redundant Power Supply (PR2)	Main Output
x	ff1 out	-	-	0	0	ff1 out
x	0	0	0	1	1	0
x	0	0	1	1	1	0
x	0	1	0	1	1	0
x	0	1	1	1	1	1
x	1	0	0	1	1	0
x	1	0	1	1	1	1
x	1	1	0	1	1	1
x	1	1	1	1	1	1
0	0	0	-	1(0)	0(1)	0
0	0	1	-	1(0)	0(1)	0
0	1	0	-	1(0)	0(1)	0
0	1	1	-	1(0)	0(1)	1
1	0	0	-	1(0)	0(1)	0
1	0	1	-	1(0)	0(1)	1
1	1	0	-	1(0)	0(1)	1
1	1	1	-	1(0)	0(1)	1

Table 3.2 represents the truth table of “data keep” block, with clearly divided different power modes:

- Only domestic power supply is active,
- Both redundant and domestic power supplies are active,
- Domestic and one redundant power supply are active.

In Figure 3.10 is shown a block diagram with implemented “data keep” block in TMR circuit. In the following text is provided mathematical analysis of TMR circuit with latchup protection.

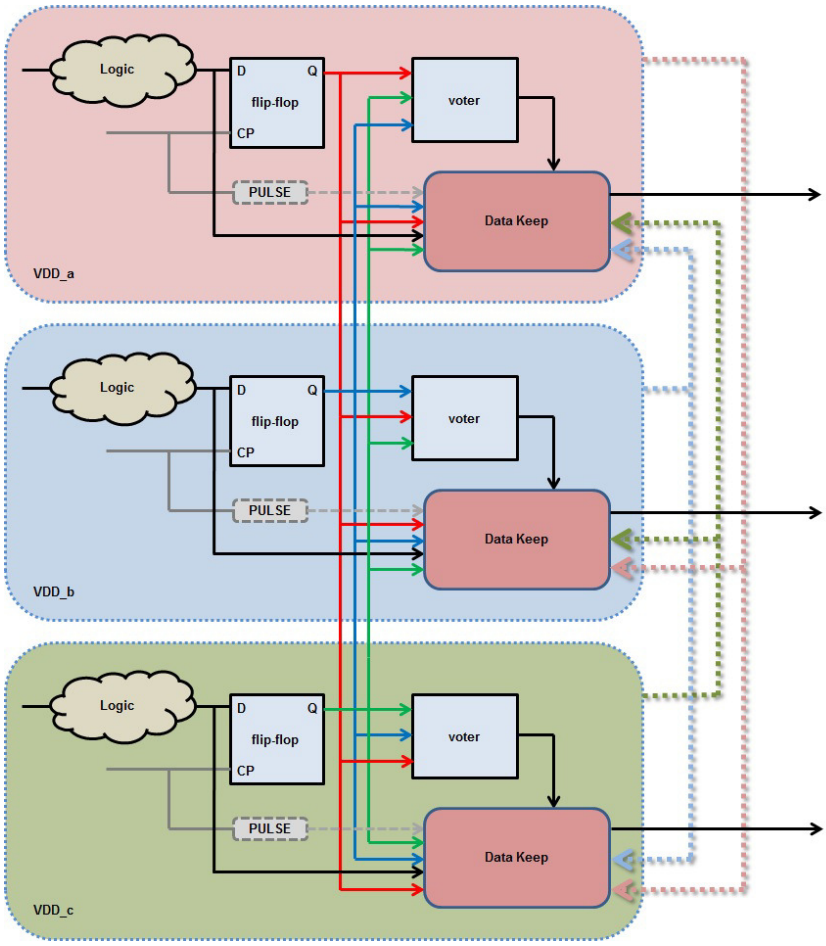


Figure 3.10. TMR circuit with latchup protection mode

The probability that "data keep" block is working without any fault (P_{DK0}) is described by Equation 3.1.

$$P_{DK0} = P_{DS}P_{EV}P_{PDO} \qquad (3.1)$$

The circuit still operates correctly if one memory element is faulty. The related Equation 3.2 describes this case.

$$P_{DK1M} = (1 - P_M)P_M^2 P_{DS}P_V P_{EV}P_{PDO} \qquad (3.2)$$

If main voter is faulty and all other components are functioning without problems the "data keep" circuit still operates. This case is covered by Equation 3.3.

$$P_{DK1V} = (1 - P_V)P_M^3 P_{DS}P_{EV}P_{PDO} \qquad (3.3)$$

The extended voter should be designed to handle the possible effects based on the particle penetration into silicon structure. In case that a transient pulse is detected on the extended voter output, dependent on the particle energy, it can cause a problem. The "pre-data output" is used to select the mode of redundancy dependent on the redundant power domains. It is also used to provide some kind of filtering the transient effects, received from "extended voter", before the logic state is provided on the outputs of the sequential element. Next Equation 3.4 describes analytically an interesting case when one memory element and main voter are faulty in the same time.

$$P_{DK1M1V} = (1 - P_M)(1 - P_V)P_M P_{DS} P_{EV} P_{PDO} \quad (3.4)$$

The failure-free probability of the "data keep" block can be calculated using the Equation 3.5. In this equation all fault-free probabilities are expressed in relation to the fault-free probabilities of flip-flop and voter.

$$\begin{aligned} P_{DK} &= P_{DK0} + 3P_{DK1M} + P_{DK1V} + 3P_{DK1M1V} \\ &= P_M^4 P_V^2 + 3(1 - P_M)P_M^3 P_V^2 + (1 - P_V)P_M^4 P_V + 3(1 - P_M)(1 - P_V)P_M^2 P_V \end{aligned} \quad (3.5)$$

After defining the fault-free probability of the "data keep" block it is possible to calculate the failure-free probability of TMR circuit with implemented latchup protection. It is important once more to note that for the latchup protection is developed a special power switch, described in the previous sections of this chapter.

Based on the Figure 3.11.a. it is possible to define the cases when the TMR circuit with advanced options is working without failures. As we have all required fault-free probabilities, based on the Equation 2.27, provided in Chapter 2.4, the failure-free probability of TMR circuit with "data keep" block can be calculated as:

$$P_{TMRF} = P_M^3 P_V^3 P_{DK}^3 + 3P_M^2 P_V^2 P_{DK}^2 (1 - P_M P_V P_{DK}) = 3P_M^2 P_V^2 P_{DK}^2 - 2P_M^3 P_V^3 P_{DK}^3 \quad (3.6)$$

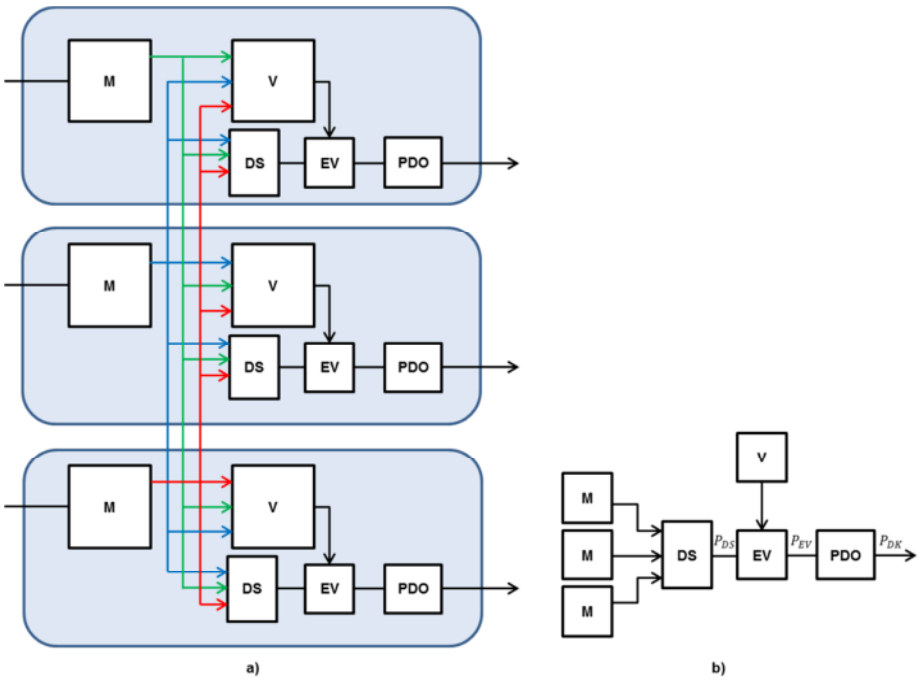


Figure 3.11. a) TMR structure with latchup protection mode;
b) related "data keep" block diagram for mathematical analysis

3.2.2. Self-voting DMR Circuit with Latchup Protection

The DMR circuit, discussed in this chapter, is developed during the research phase related to the automated ASIC design technique with reduced redundancy. It is based on the self-voting approach [TEI08]. The circuit with self-voting approach, without modifications on the feed-back line has low fault-free probability. The reason is high circuit sensitivity on the transient effects near active clock edge. The setup-hold margins are in this situation violated and flip-flops are unable to save a correct state. This situation is better explained following Figure 3.12. In case that a transient effect occurs in the combinational logic during an active clock edge, flip-flops save the different logic states. This is also potential upset initialized by combinational logic. Dependent on the previous logic state, the fault can be propagated through the digital system, causing an error with many possible consequences. The example, presented in Figure 3.12 shows the most important disadvantage of the DMR based redundant circuit. Therefore, the requirement is to mitigate the described effect.

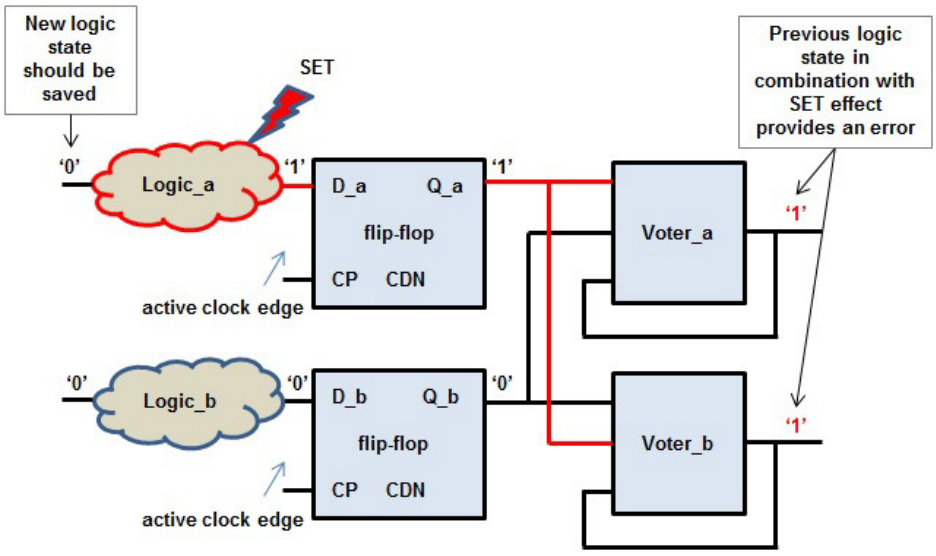


Figure 3.12. An example how a DMR circuit with feed-back enables fault propagation

Lesson learned from the previous example has shown that one more information about saved logic state is very important for correctness of the DMR circuit outputs. The most critical moment is during an active clock edge because of saving wrong logic state or potential metastability. The flip-flop related metastability is an effect which arises as a consequence of changing the data signal within the setup-hold margin. The setup-hold margin represents a period, within the input data should be stable. Therefore, the data should be stable for a setup time before an active clock edge. On the other side, after clock edge the input flip-flop data should be stable for a hold time. In case that the input data is not stable within above defined margin, than it is unknown which logic state will be saved in the flip-flop. This effect is usually present after a transient effect occurs in the combinational logic (marked as "Logic" in Figure 3.12), which drives the flip-flop input.

Related to the previous work in this area [SCH09], the recommendation for accurate voting near active clock edge is to use the logic state on the flip-flop input as third input of the voter. After hold time margin passes, the voter output can be again used for voting until next active clock edge arrives.

There is an important difference how the “data keep” block is connected with other blocks (flip-flop and voter) in the TMR and DMR circuit with supported latchup protection modes. In order to provide more reliability, the DMR circuit with multiplexed feed-back line [SCH09] is modified. Modifications are related to the better handling of upset and transient effects as well as the most important – handling the potential latchup effect. Careful analysis of the DMR circuit with multiplexed feed-back line [SCH09] provides an important fact – if one part of the DMR circuit is turned-off, because of the latchup protection mechanism, the DMR circuit will not operate correctly. Actually, all other cells will be driven by low logic state. Higher reliability of DMR circuit requires re-design of the presented DMR circuit with multiplexed feed-back line, especially in case when it needs to handle the transient effects with longer duration than the flip-flop hold timing margin. The idea of using the input data during an active clock edge is also included in the DMR circuit with latchup protection mode. Therefore, an advanced analysis is required during the latchup protection phase and during the state-recovery procedure after the latchup protection phase or after upset/transient effects.

In Figure 3.12 is presented a DMR circuit with latchup protection support. Further text provides detailed circuit description with important characteristics.

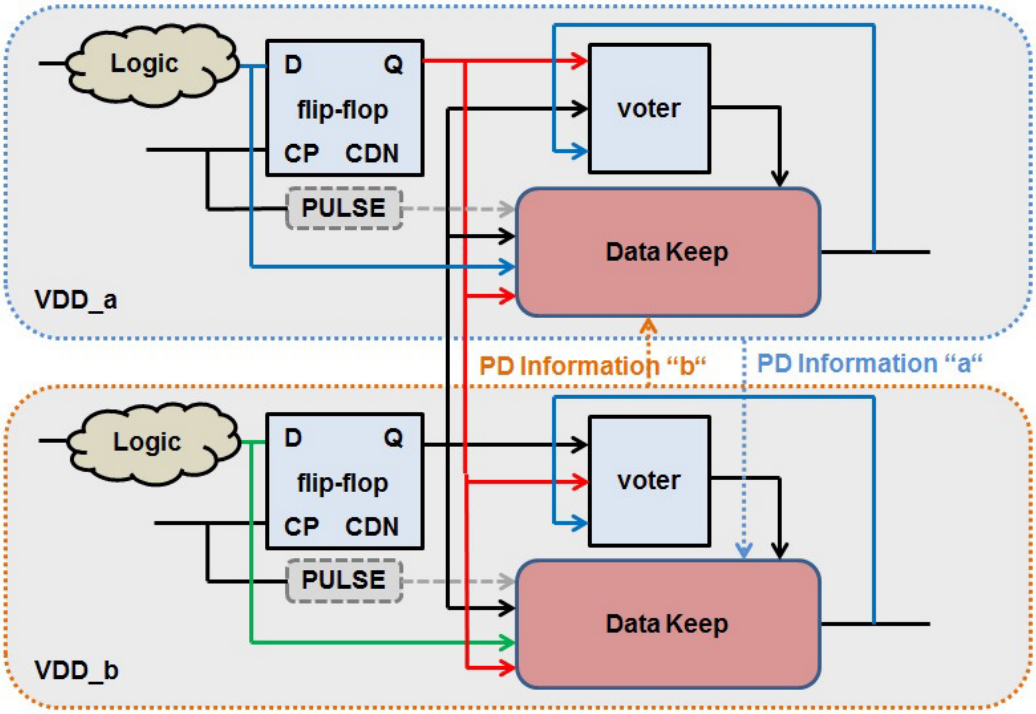


Figure 3.13. Implemented “data keep” block in DMR circuit with latchup protection

Similar as it was mentioned before, the “data keep” block is the most important block in the modified DMR circuit. It observes the logic states of all important blocks in the DMR system as well as power supply condition of the redundant system.

The power supply information, regarding Figure 3.12 is represented as signal “PD information” (PD – power domain). The logic state on the third voter input is dependent on few logic states in the circuit.

In standard DMR circuits the feed-back line is implemented as poor line. Based on the fault-free requirements the C-element will not be discussed in this analysis.

“Data keep” block observes the logic states on the both flip-flop outputs, output of the “domestic” voter and the data input signal for the “domestic” flip-flop as well as the power domain state of the redundant circuit. In order to describe how the modified DMR circuit operates, it is important to go through few examples.

The most critical example is when a transient pulse occurs around an active clock edge. In this case there are two possible consequences: the metastability or saving a wrong logic state (upset) in the flip-flop. Both of them induce erroneous circuit behave. Therefore, the “data keep” block compares logic states on the flip-flop outputs, domestic and redundant, with input data signal of the domestic flip-flop. In case that transient pulse occurs within the setup-hold margin, the flip-flop saves a wrong logic state. A pulse duration defined by “pulse” block, provides enough time to save the stable logic state on the flip-flop data input. Dependent on the saved and stable logic state, the logic states on both flip-flops and the logic state on the domestic voter, the “data keep” block provides accurate logic state as third input to the domestic voter. In the decision process is also included current status of the power supply of redundant circuit.

Here is also important to mention two important advantages of the presented approach. First is related to the configurable protection level. In case that system doesn’t need high protection against radiation effects (terrestrial operation, for example), by using the power protection circuit presented in previous sections, it is possible to reduce the power consumption by selecting a single mode operation. The single mode operation disables the protection against all three radiation effects – transients, upsets and latchup. In this mode, the “data keep” block ignores the signals which are coming from the redundant circuit.

Second advantage compared to the DMR circuit without latchup protection is related to the latchup protection mode. In this case, one part of circuit is switched off regarding high current flow due latchup effect. After protection period, it is important to provide correct “wake-up” procedure of the circuit part which was switched-off. During the “wake-up” procedure it is important that all sequential elements in the digital circuit have the correct logic states on their outputs. The correctness of the data defined by sequential elements is dependable on the “data keep” block. Under term sequential element, refers the complete DMR circuit when dual mode is active. In case that single mode is active, the term sequential element refers to one flip-flop with following “data keep” block. As the single mode doesn’t need redundant signals, the voter and related signals are not included.

“Data keep” block consists of multiplexer, memory element (latch) and voter. The block diagram of “data keep” block is illustrated in Figure 3.13. The truth table of DMR “data keep” block is presented in Table 3.3. In Table 3.3 are presented two different power modes:

- Only domestic power supply is active
- Both power supplies are active

Table 3.3. DMR data keep block truth table

Input data Domestic Flip-flop (Latch)	Domestic flip-flop Output (FF1)	Redundant flip-flop Output (FF2)	Redundant Power Supply (PR1)	Main Output
x	FF1 OUT	-	0	FF1 OUT
0	0	0	1	0
0	0	1	1	0
0	1	0	1	0
0	1	1	1	1
1	0	0	1	0
1	0	1	1	1
1	1	0	1	1
1	1	1	1	1

In case that both power supplies (domestic and redundant) are switched off, all nodes of redundant circuit will be set to low logic state.

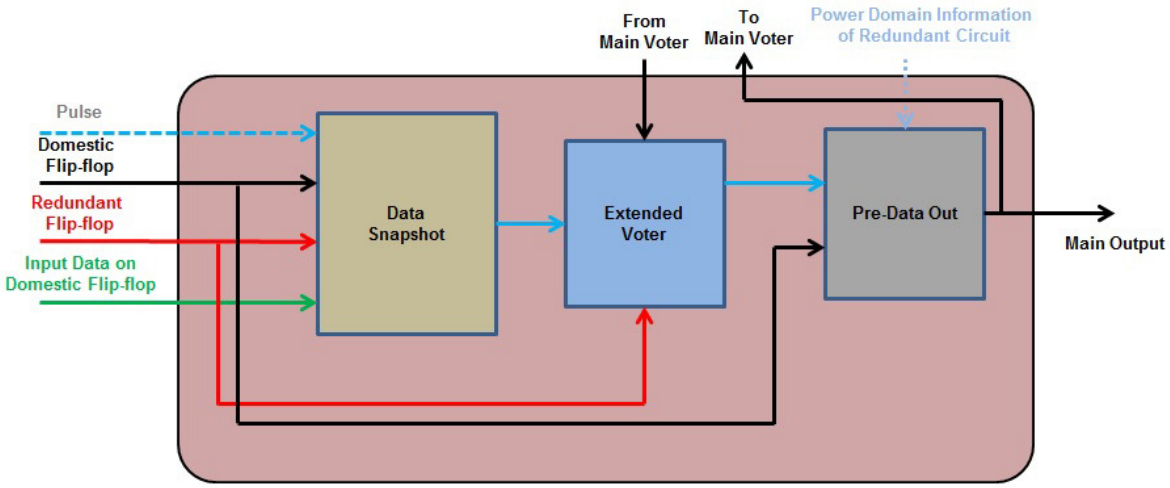


Figure 3.14. Diagram of the "data keep" digital block

"Data snapshot" block is used to provide a correct logic state during complete clock cycle. The saved logic state is dependent on the logic states of the domestic digital circuit as well as redundant flip-flop. Extended voter is used to provide the voting procedure based on the four digital states. The extended voter is also used to provide more accuracy and therefore, higher fault-free probability. In this representation is used a standard TMR voter because of low glitch probability and availability as a standard cell.

The "pre-data out" block is used to select the data, related to the type of circuit protection. In case that circuit is in single mode, then the data will be taken directly from the domestic flip-flop without involving any dependability with the redundant circuit. In case that the system operates in the protection mode

with activated redundancy, then “pre-data out” selects the data which is dependent on the domestic and redundant circuit too.

It is important to note that in this type of circuit is implemented a “voting loop” approach. It is developed during research phase and main component for this data-flow option is “extended voter”. In case that fault occurs in the “pre-data out” block, the main voter recognizes a problem and through “extended voter” the error is immediately corrected. On the other hand, if an error occurs in the main voter, the “extended voter” should correct it. The open question is still the same – does the modified DMR circuit still represents the reduced redundancy compared to the TMR? More details on this topic are provided in Chapter 4.

Before further analysis it is important to provide an extended analytical model for the modified DMR circuit. As there is not anymore only one delay element in the feed-back line, the failure-free probability is changed. It is important to note that previous defined failure-free probabilities in Chapter 2.4, don’t involve the circuit failure-free probability related to the single event latchup (SEL). This is an example how analytical model is dependent of the designer preconception. For the fault-tolerant digital systems it is very important to include all possible effects which can bring a circuit in the faulty state. As the extended DMR circuit includes the latchup protection mode, the analytical model should also involve a corresponding element. On the other hand, the analytical models presented in this work are based only on the fault-free probabilities related to upsets and transients. When a part of the circuit is switched-off due latchup effect, goal is to provide normal operation for the rest of circuit. Therefore, during latchup effect the circuit operates as standard circuit without applied redundancy and failure-free probability is in this case equal to the fault-free probability of the memory elements and other gates.

Similar to the Equation 2.33 (Chapter 2.4), here will be described the failure-free probability of the extended DMR circuit with latchup protection support. Figure 3.14 describes an extended DMR circuit in the analytical form.

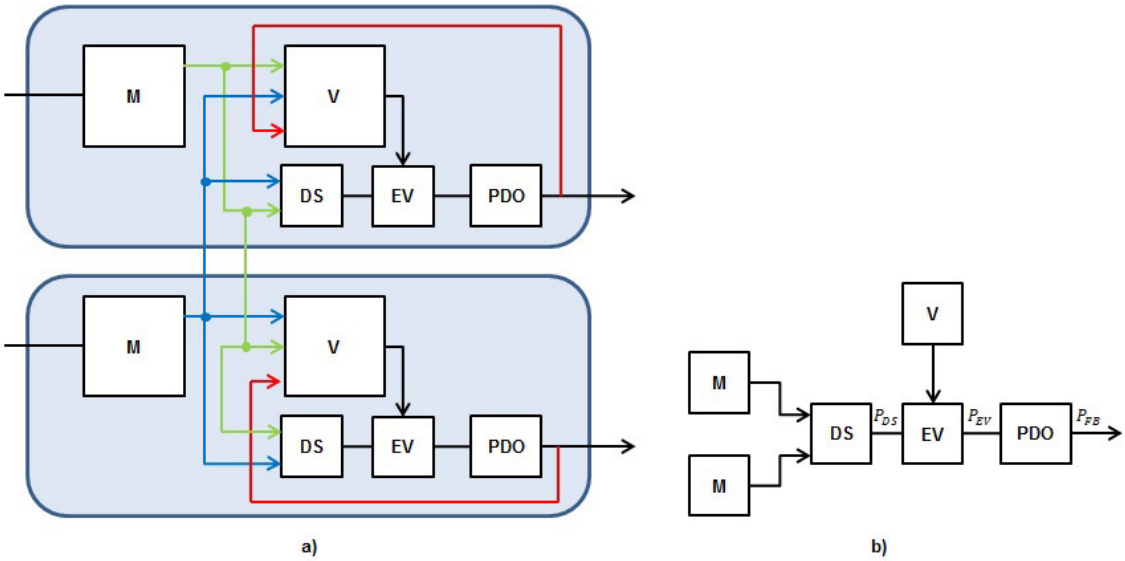


Figure 3.14. a) DMR circuit structure, b) related feed-back block

Based on the DMR structure, shown in Figure 3.14.a, it is important first to define the fault-free probabilities of feed-back line. With red line, in Figure 3.14.a is presented the feed-back connection. In order to provide an easier way to calculate the failure-free probability of the circuit, the feed-back blocks (DS – data save, EV – extended voter and PDO – pre-data out) will be discussed as components with independent inputs (Figure 3.14.b.). Before defining the failure-free probability for complete DMR circuit, first will be discussed the analytical model of the feed-back circuit.

Probability that complete feed-back circuit operates correctly is expressed using Equation 3.7.

$$P_{FB0} = P_M^2 P_{DS} P_V P_{EV} P_{PDO} \qquad (3.7)$$

In case that one flip-flop is faulty, the feed-back circuit will still operate correctly and the failure-free probability is then expressed using the Equation 3.8.

$$P_{FB1M} = (1 - P_M) P_M P_{DS} P_V P_{EV} P_{PDO} \qquad (3.8)$$

From the Equation 3.4 it is possible to notice the dependence of the feed-back circuit failure-free probability and the flip-flop fault-free probability. When voter is faulty, the feed-back circuit still operates correctly only when all other components are without faults. The Equation 3.9 describes this situation.

$$P_{FB1V} = (1 - P_V) P_M^2 P_{DS} P_{EV} P_{PDO} \qquad (3.9)$$

The feed-back circuit still operates correctly if one flip-flop and voter are faulty in the same time. This is because of the “data keep” block which enables a correct logic state during faulty period. This case is defined in Equation 3.10.

$$P_{FB1M1V} = (1 - P_M)(1 - P_V) P_M P_{DS} P_{EV} P_{PDO} \qquad (3.10)$$

Therefore, for the feed-back circuit, the related failure-free probability model is defined by Equation 3.11. It is important to note that the fault-free probability of the “data keep” block is assumed to be the same as fault-free probability of the flip-flop ($P_{DS} = P_M$). The “extended voter” fault-free probability is same as the voter fault-free probability and therefore $P_{EV} = P_V$. The fault-free probability of the “pre-data out” block is in this calculation assumed as ideal ($P_{PDO} = 1$).

$$\begin{aligned} P_{FB} &= P_{FB0} + 2P_{FB1M} + P_{FB1V} + 2P_{FB1M1V} \\ &= P_M^2 P_M P_V P_V + 2(1 - P_M) P_M P_M P_V P_V + (1 - P_V) P_M^2 P_M P_V + 2(1 - P_M)(1 - P_V) P_M P_M P_V \\ &= P_M^3 P_V^2 + 2(1 - P_M) P_M^2 P_V^2 + (1 - P_V) P_M^3 P_V + 2(1 - P_M)(1 - P_V) P_M^2 P_V \end{aligned} \qquad (3.11)$$

In Equation 3.11 there are some members multiplied by two. This is done because there are two possible cases covered by analytical model. The equal situation is described in the [LYO62] and in Chapter 2.4, for the TMR analytical model.

When the feed-back circuit is covered with own analytical model, the analysis goes further in order to describe the failure-free probability of DMR circuit with latchup protection support. The analysis starts from covering all the ceases when the DMR circuit operates correctly. Equation 3.12 expresses first case, when all components of DMR circuit with latchup protection are operating without any faults.

$$P_0 = P_M^2 P_V^2 P_{FB}^2 \qquad (3.12)$$

In case that one memory element is faulty, the DMR circuit still operates correctly as it is expressed in Equation 3.13.

$$P_{1M} = (1 - P_M) P_M P_V^2 P_{FB}^2 \qquad (3.13)$$

In case that voter or fee-back is faulty, the Equations 3.14 and 3.15 are describing these situations, respectively.

$$P_{1V} = (1 - P_V) P_V P_M^2 P_{FB}^2 \qquad (3.14)$$

$$P_{1FB} = (1 - P_{FB}) P_{FB} P_V^2 P_M^2 \qquad (3.15)$$

There is one more situation when the DMR circuit with latchup support operates correctly. In the moment when both main voters are faulty, the feed-back circuits take the main voting tasks. This case is defined by Equation 3.16.

$$P_{2V} = (1 - P_V)^2 P_{FB}^2 P_M^2 \qquad (3.16)$$

Failure-free probability of DMR circuit with latchup support is defined by Equation 3.17. It is very important to notice that the failure-free probability of DMR circuit with supported latchup protection mode describes an independent case during latchup protection phase ($P_{DMR|_{latch-up}}$). The probability that circuit will operate correctly during latchup is equal to the failure-free probability of the circuit without any protection because the voter is excluded in this case ($P_{DMR|_{latch-up}} = P_M$).

$$P_{DMR} = P_0 + 2P_{1M} + 2P_{1V} + 2P_{1FB} + P_{2V}$$
$$= P_M^2 P_V^2 P_{FB}^2 + 2(1 - P_M) P_M P_V^2 P_{FB}^2 + 2(1 - P_V) P_V P_M^2 P_{FB}^2 + 2(1 - P_{FB}) P_{FB} P_V^2 P_M^2 + (1 - P_V)^2 P_{FB}^2 P_M^2 \qquad (3.17)$$

3.3. Power Network Controller

Power network controller (PNC) is a digital subsystem which controls all latchup protection circuits (SPS cells) in the fault-tolerant digital system. It is designed to communicate with all SPS cells independently. It consists of programmable counter and control circuits. Programmable counter defines duration of the latchup protection phase and control circuits are used to provide communication interface with SPS cells. The power network controller complexity is directly related to the redundancy type – DMR or TMR.

Block diagram of power network controller is shown in Figure 3.14. A status/control unit is composed of “control circuits” (CS). “Control circuits” are directly connected to the SPS cells. At the moment when a latchup is detected, the “TSTART” signal activates related “control circuit”, which saves the current counted value. When programmable counter goes again through the saved counted value, the “control circuit” activates the “TSTOP” signal. After “TSTOP” signal activation, the related SPS cell re-activates the controlled power supply where the latchup effect was detected. It is important to notice that this type of power network controller handles more latchup effects simultaneously.

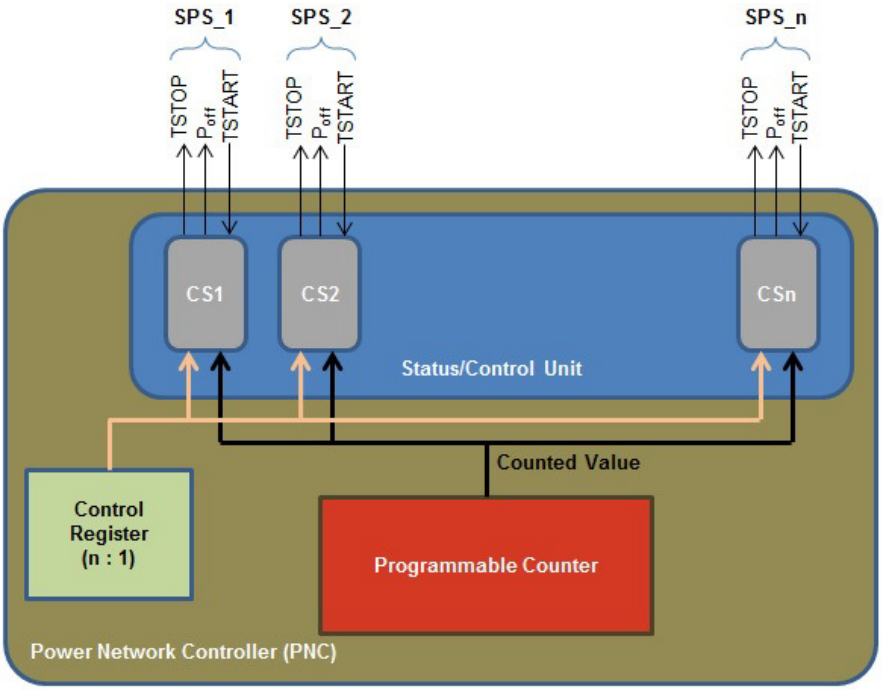


Figure 3.15. Block diagram of power network controller (PNC)

"Control register" (CR) is used for power mode selection. Power mode selection is provided by SPS cells activation/deactivation using the P_{off} signal. Dependent on the application, there are three different power modes:

- Redundant mode (RM) – SPS enables power supply for redundant digital circuits.
- Single mode (SM) – Redundancy is deactivated (Single mode).
- Deactivated mode (DM) –All controlled standard cells (circuits) are switched off.

Dependent on the application, the power network controller can be programmed either externally or internally. External PNC programming is provided using appropriate serial or parallel interface. Internal programming is usually provided using APB AMBA interface [AMB13]. It is important to note that the power network controller has independent clock input. The VHDL code example of power network controller can be found in Appendix D. Therefore, the SPS power network consists of power network controller (PNC) and SPS cells.

4. Fault-Tolerant Circuit Simulation and Analysis

The most important question during development of new design methodology is based on the methods for circuit verification. This point here can induce confusion about single event effects and ability of simulation tools to test them.

Logic simulation is used to predict the behavior of digital circuits and hardware description languages. Simulation can be performed at varying degrees of physical abstraction, such as at the transistor level, gate level, register-transfer level (RTL), electronic system-level (ESL), or behavioral level. Other frequently used term for logic simulation is functional simulation because it provides the answer on the question does the digital circuit under test function properly. Within the functional simulation at the gate level it is possible to observe the digital circuit behave during transient and upset effects. In order to provide the information about circuit behave during latchup effect, it is required to use the functional simulation at the transistor level. Simulations performed at the transistor level are also known as analog simulations.

During the development phase of all circuits, parallel with functional simulations at gate level are performed also analog simulations. This is very important because of the latchup effects which can be simulated on the gate level just for specially designed gates which are not present in the standard cell libraries.

4.1. Fault Injection

Fault-injection environment block diagram, presented in Figure 4.1, shows all required components for the target system test. As the presented work in thesis uses the simulation fault-injection approach, the target system under test is a digital circuit described in the VHDL or in the Verilog hardware description language. The system presented in the VHDL requires a standard digital system description, fault-injectors and description of faults relevant for the test. In case of the Verilog netlist it is important to note that in this work Verilog language is used only for describing the synthetized or gate-level netlist. Each synthetized netlist is composed of the standard cells instances and interconnections. Standard cells are related to the technology library for which a digital system is synthetized.

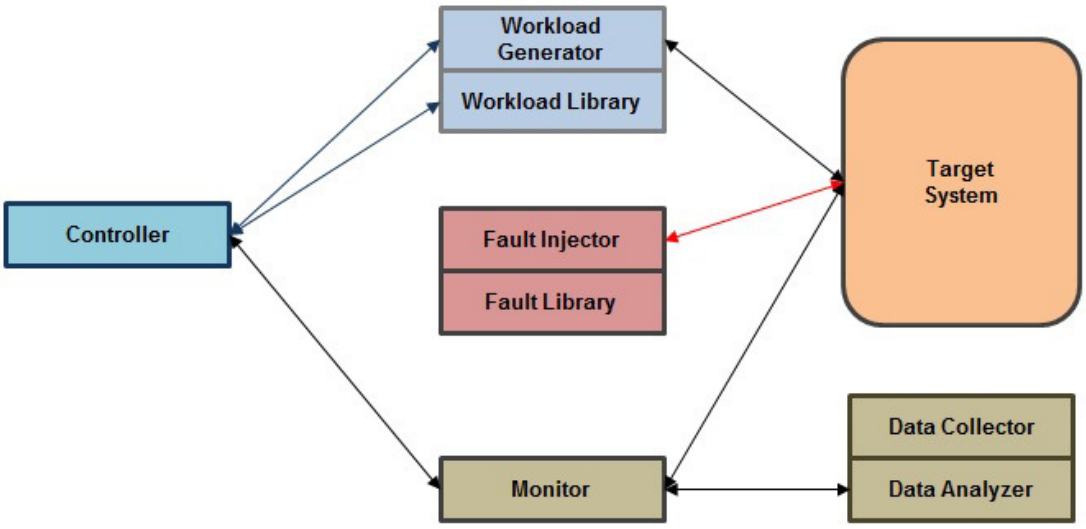


Figure 4.1. Fault injection environment components

Fault-injector enables hardware to take the faults as standard signals, synchronized and processed by the workload generator. In the fault library are the fault stimulus types, the fault timing and the locations where fault should occur.

The workload generator generates input stimulus signals for the system under test (target system). The workload library contains the stored test patterns for the target system. The test patterns are used to provide classic functional tests of the digital system.

The main component in the fault-injection test environment is a controller which controls the test procedure. Controller represents a test-bench with all required definitions and functions.

In order to analyze the test data, data collector and data analyzer are connected with monitor. Monitor is also integrated in test bench and it is used to observe the data in the target system during fault-injection process. The generic fault injection environment [BEN03] can be used for hardware, software, and simulation fault-injection. The work described in thesis is based on a simulation fault injection model.

Fault-injection model needs to be integrated into a simulation model of the designed digital system. It means that the errors or failures, of the simulated system, occur according to the predetermined distribution. The model supports axiomatic, empirical, and physical system abstraction levels. The main advantages of this approach are full control of fault models and injection mechanisms, low cost computer automation, and maximal observability and controllability. The most important disadvantages are high complexity, unverified fault coverage, and long development time.

The simulation fault-injection model, developed for the test purposes used in thesis, covers two types of the single event effects: single event upset and single event transient. In order to design accurate fault-injection models, required for simulation, it is important to analyze the physical effects which are causing the mentioned transient effects (SET) and upset effects (SEU). SET effect is always detected as a short current pulse. For example, an inverter with low logic level (‘0’) on the input terminal and high logic level (‘1’) on the output terminal. In case that n-MOS transistor is hit by radiation particle, it

will conduct the current for the short time, what will be detected as short low logic level ('0') on the output.

Related to the fault-injection environment, presented in Figure 4.1, in next sections will be discussed fault-library for the single event transient effects, fault-library for single event upsets and fault injector.

4.1.1. Transient Fault Library

For the transient effects fault library the most important information are:

- How long a SET effect takes
- The moment when such effect occurs.

The duration of transient effect is dependent on many different parameters as technology, materials used for chip production, source of radiation, etc. The duration of transient effect can be calculated or estimated using the simulation approach. The moment when a SET occurs is defined as in the nature - randomly. Therefore, the SET fault library consists of SET duration library (Transient Duration Library – TDL) and library where is defined the moment when an SET occurs (Transient Starter Library – TSL). It is important to note that the simulation based fault-injection system is designed for the systems which consist of sequential and combinational logic and all fault-injection related operations are related to the main system clock. If the circuit is completely designed using the combinational logic, a virtual clock should be defined.

Calculated SET Duration

Based on the mathematical calculations we can predict the duration of the transient pulse. The calculation is dependent on the particle energy, its physical characteristics and physical characteristics of the material under irradiation. There is extensive research that has previously been conducted in relation to modeling the pulse width of an SET in combinational logic. Messenger presents in his work an approximate analytical solution for the pulse width generated by lightly ionizing particles [MES82], in the double-exponential Equation 4.1, shown below:

$$I_{SET}(t) = I_0(e^{-\frac{t}{\tau_1}} - e^{-\frac{t}{\tau_2}}) \quad (4.1)$$

In this equation, $I_{SET}(t)$ is the approximate pulse generated by the SET effect, I_0 is approximately the maximum current induced during particle penetration, τ_1 represents the charge collection time-constant of the junction. It is the rise time related to the plasma track dynamics. The τ_2 is the down time related to charge drift and diffusion in the transistor or simple the time constant related to the dissipation of the collected charge. The current I_0 is described by Equation 4.2, where Q represents the net charge associated to the transient current through the transistor node. The τ_1 and τ_2 are the process-related factors.

$$I_0 = \frac{Q}{(\tau_1 - \tau_2)} \quad (4.2)$$

The constant τ_1 is mathematically described by Equation 4.3:

$$\tau_1 = \frac{k\varepsilon_0\varepsilon_r}{q\mu N_D} \quad (4.3)$$

The τ_2 constant is not easy to calculate as we did for the τ_1. Related to the previous work, this constant can be calculated for the average estimation through the relation with the constant τ_1. The average ratio $\frac{\tau_1}{\tau_2}$ is 3.5 [LIM12].

The Equation 4.1 is very useful for spice simulation of transient effects but it is not recommended for faults modeling. The reasons are based on the calculation complexity and long time required for analysis. Therefore, it is recommended to use measured or even estimated values of single event transient effect duration. The estimation procedure, based on radiation simulations is presented in the following section.

Estimated SET Duration Based on Simulation Approach

In order to provide a range of durations which single event transient effect can take, it is much easier to use the specially defined simulators. For the scientific purposes it is recommended to use the SRIM (The Stopping and Range of Ions in Matter) simulator [SRI13]. In the simulator is possible to set parameters related to the technology (thickness of different layers, materials…) and parameters related to the type of irradiation source. For the presented work it was important to use the ion mix related to the ESA specification [ESA02] for the simulating the real radiation sources in the space environment. Related to the work [MES82] and using the results of the SRIM simulation done for the IHP technology, in Table 4.1 are presented as well as calculated current intensity.

Table 4.1. Estimated time and current pulses, IHP

Element	Energy [MeV]	Time [ps]	Current pulse [µA]
He	6	217	417
N	139	2097	1266
Ne	186	1397	2509
Si	278	1227	4225
Ar	372	1067	6451
Fe	523	877	10770
Kr	768	847	16249
Xe	1217	807	26655

Based on the previous described method related to the SET pulse duration, it is possible to notice that for the proposed ion cocktail the SET effect can take between 200 [ps] and 2 [ns]. The model for simulation, based on SET fault-injection, involves as much as possible values between 200 [ps] and 2 [ns] in order to provide the scenario near to the reality. Therefore, the defined SET model takes randomized values of SET pulse timing and involves them in the simulation.

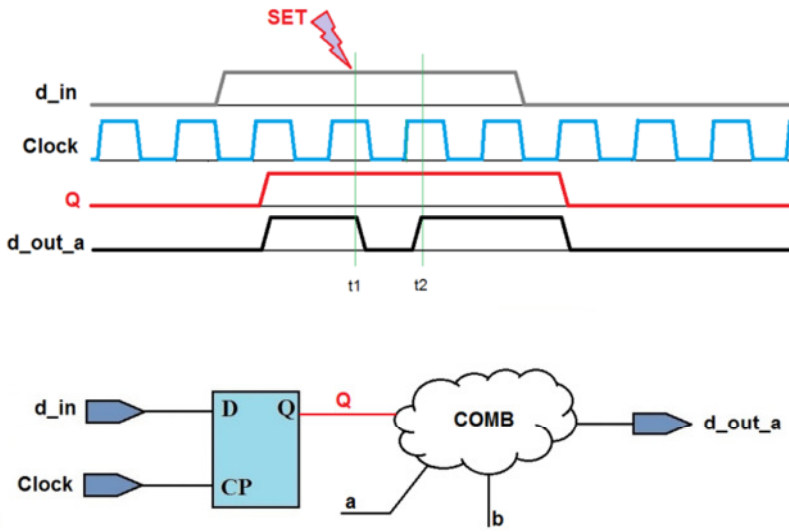


Figure 4.2. SET in the combinational logic

Figure 4.2 presents the signal waveform of the circuit which consists of sequential element (in this case a flip-flop) and combinational block. The figure presents how SET effect behaves in the combinational block. It occurs at the moment t_1 and it is active for the period of time $t_2 - t_1$. The combinational logic has more inputs "**a**" and "**b**" which are not important for this analysis. It is important to note that output "**d_out_a**" is set to wrong logic state during the active clock edge. It is important for the next flip-flop data input, which is connected to the combinational logic output "**d_out_a**". In this case, the next flip flop will memorize the wrong logic state. In this example it is shown how a transient effect can cause system to fail.

The transient effect duration library (TDL) and transient effect starter library (TSL) are generated using the C code. The library example and the source code for its generating are presented in Appendix A.

4.1.2. Upset Fault Library

Single event upsets (SEU) may occur at the moment when deposited charges, by ions and protons, are collected at sensitive nodes of storage elements such as flip-flops, latches, SRAM cells, etc. A single event upset may also be the result of a transient effect being latched on a clock edge after propagating in combinational logic, as it was discussed in the transient effect example, illustrated in Figure 3.4 [ESA10]. The upset effect is available until next clock cycle and it can be propagated through the sequential system. It can cause that digital system fails. The redundant systems are providing the possibility to detect and to correct upset effects.

The most important parameter for a single event upset model is the moment when the effect occurs. It is strong related to the clock cycle. The positive clock edge is used for defining the moment when SEU starts. The Upset Starter Library (USL) consists of the randomly defined timings, within the clock cycle period. That means – an upset can occur immediately after active clock edge (shortest time to an upset) or near to the next active clock edge (longest time to an upset). This situation is presented in Figure 4.3. The period $t_2 - t_1$ (or $t_4 - t_3$) can take any randomly chosen value between 0 [ns] and clock period. It is important to note that dependent on the combinational block (COMB) function, the main

circuit output (d_out_a) can also be in the wrong logic state. In the presented example (Figure 4.3), a wrong logic state will be transferred to the combinational block with potential signal change, which can cause the system to fail.

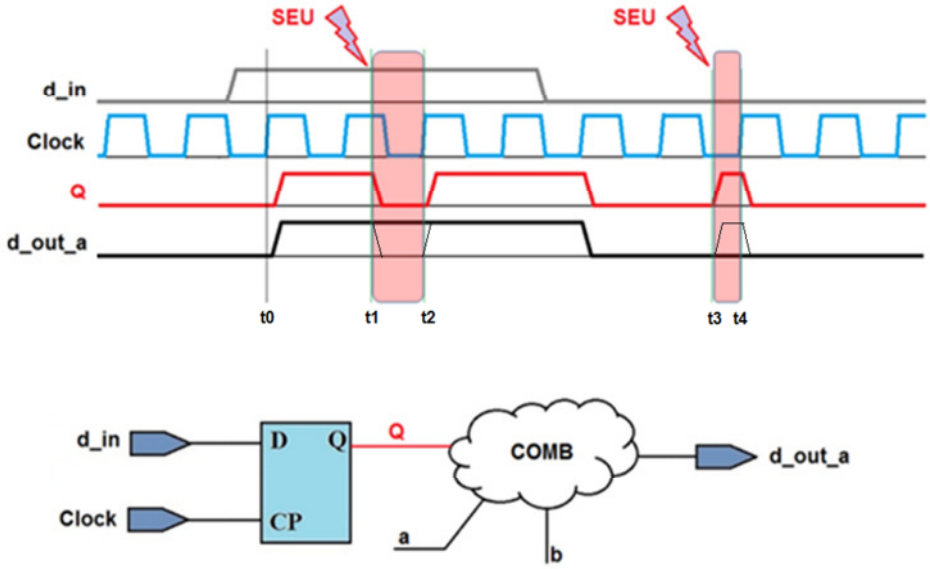


Figure 4.3. SEU effect in flip-flop

The upset starter library (USL) is a textual data type and it is generated using the random number generator function. The USL is generated by C programming language and it is described in Appendix A.

4.1.3. Fault Injector

In order to prepare a digital circuit for fault-injection, it is required to add a XOR gate on each gate output. Sequential circuits should have XOR gate for enabling the SEU faults. On the other hand, a XOR gate on the combinational gates outputs needs to be added in order to inject the SET faults. The fault-injection can be provided in VHDL model or in Verilog netlist.

If we compare the fault-injection on the gate netlist level and on the RTL VHDL level, it is easy to notice that the RTL VHDL fault-injection model requires complex parsers. This is because it needs to recognize sequential and combinational logic in the code. The easiest way to implement faults is to use a synthetized (gate) Verilog netlist, which requires simple parser programs in order to recognize sequential and combinational components and to add the required fault-injectors.

In order to reduce the required simulation resources and to provide automation of fault-injection test, there is one more addition for fault-injector – circuit for test selection. For the fault-injection strategy presented in the thesis work it is important to note that faults are not present for all combinational cells. The most important transient effects for analysis are those which occur in the combinational gates next to sequential cells (flip-flop). For our purpose it is not important to inject faults and to characterize the complete combinational circuits. In reality it is possible to note that transient pulse expire during

transfer through several combinational gates. Therefore, the target combinational gates are those connected to flip-flop data inputs.

On the other hand, all outputs of sequential elements are able to inject the faults. Circuit for the test selection consists of multiplexer and instead of two fault inputs (SEU and SET) it has just one input – the error input. Error input enables the fault on the selected position which is defined just for one cell in one moment for the complete circuit. Position where the fault should be injected is defined in the fault position library (FPL) and it has also random values. In case that digital circuit doesn't have equal number of sequential and combinational cells, then the related SEU or SET lines should stay unconnected (open). The circuit for test selection is presented in Figure 4.4.

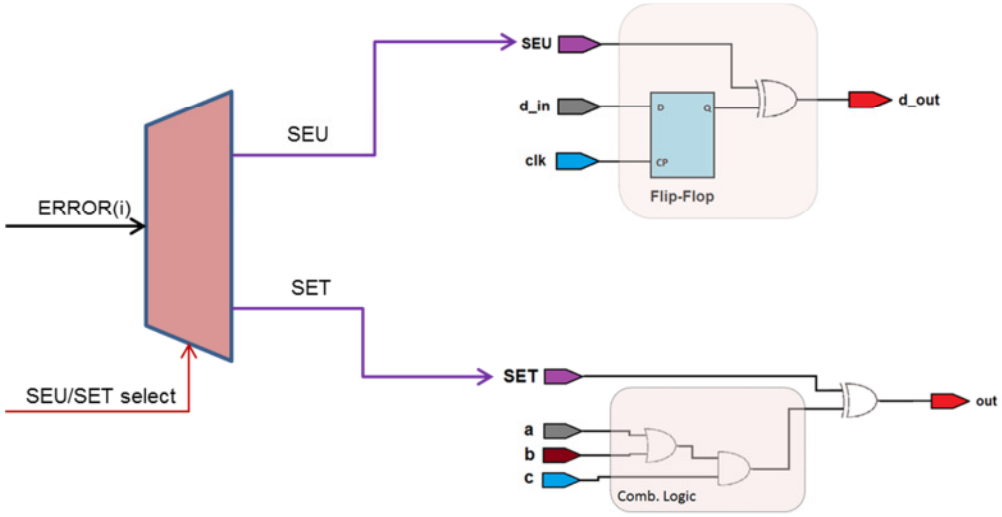


Figure 4.4. Circuit for test selection

Example of TMR fault-injection model with triple voting circuit is shown in Figure 4.5. It consists of three flip-flops and three voters. All flip-flops are connected to all three voters and therefore are providing a complete triple redundant circuit also known as full TMR circuit. Comparing to the full TMR circuit, as it was mentioned before, in order to save area and required power consumption it is possible to use reduced redundancy – DMR. Example of self-voting DMR fault-injection model is illustrated in Figure 4.6.

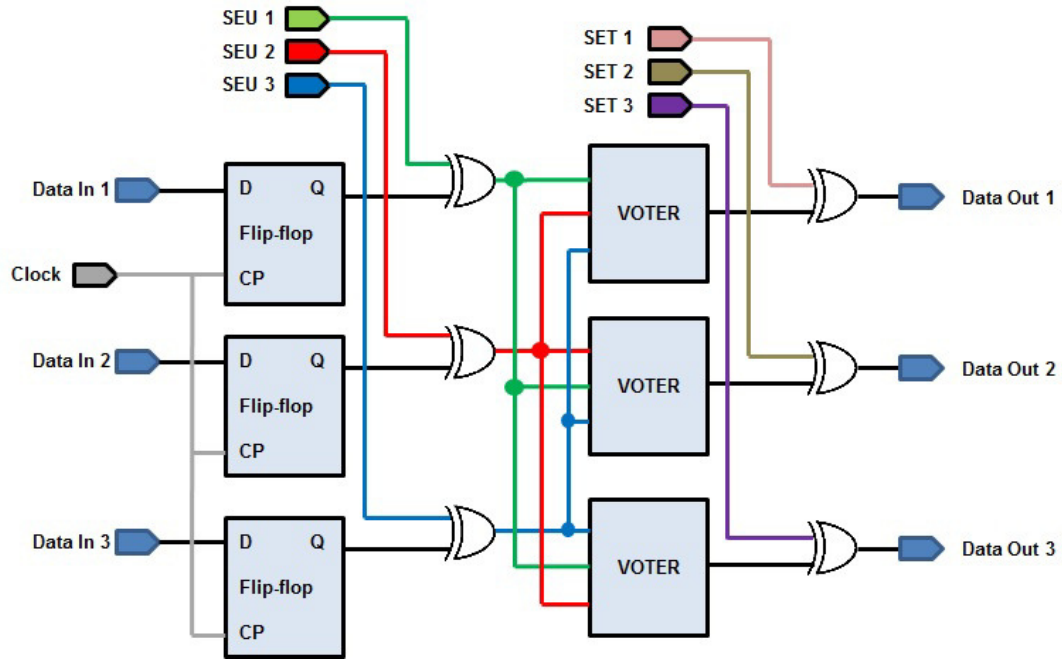


Figure 4.5. Fault-injection model for TMR circuit

As it was mentioned before, the SEU fault injection mechanism is based on flipping the saved state in the flip-flop during clock cycle by using XOR logic. In the next clock cycle, the flip-flop should continue with normal operation. On the other hand, the SET effect is a short transient pulse, detected on a combinational gate output. In case of combinational gate, the SET effect will invert the gate output value for a certain time. A SET fault can be injected using the SET input of the second XOR gate. Both timing parameters (period and start time) are randomly generated.

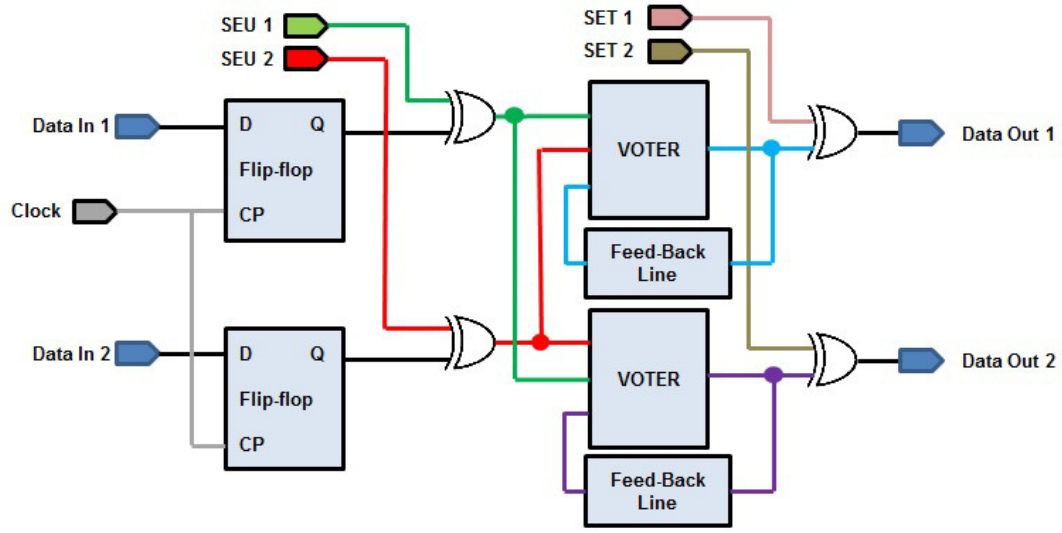


Figure 4.6. Fault-injection model for self-voting DMR circuit

The described fault models for SEU and SET faults can be easy implemented in the circuit which we want to test and characterize.

4.1.4. Fault Injection Mechanism

After provided description of fault-injection libraries and fault-injector with test selector circuit, it is important to describe the mechanism how a fault is injected in the digital circuit – starting from the library until sequential or combinational cell output.

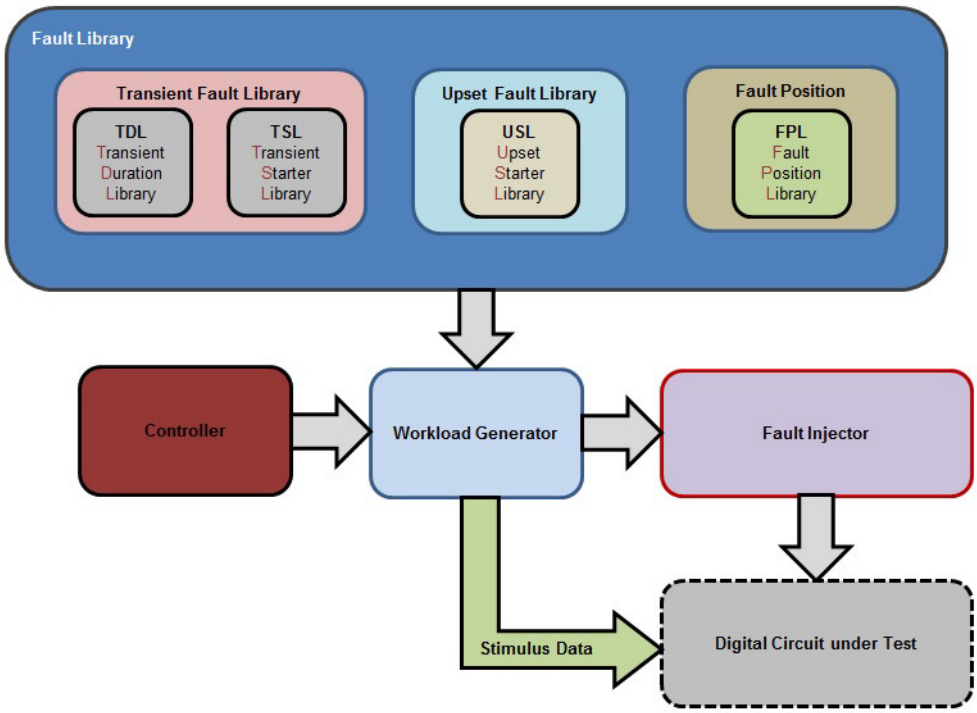


Figure 4.7. Fault-injection flow

In Figure 4.7 is presented modified fault-injection diagram, which describes how the faults are integrated with standard simulation data. The “controller” in Figure 4.7 is a part of the VHDL test-bench and it defines the stimulus data for simulation. Stimulus data imply the clock signals, the reset signals, the stimulus patterns for inputs and other important constants and algorithms for simulation. On the other hand, the “workload generator” represents a part of the test-bench, where the data from “fault library” is converted to understandable values for simulation (from integer to time). Fault libraries are generated in the textual form with numbers as library values. Workload generator integrates the conversion functions (described in Appendix A) and enables automated integration of the stimulus signals and fault-injection. Conversion functions are important for translation from the textual data into time related values.

After the controller initializes the simulation, the workload generator reads the first value from the fault position library (FPL). The range of numbers defined in the FPL take values between one and the number of all sequential elements in the digital circuit under test. For example, if there are 5000 sequential elements in the circuit, the FPL values are 398, 4005, 749 etc. This value defines which element in the digital circuit under test will be selected for the fault-injection. The appropriate ERROR[i] input of the test selector is than set on the high logic level. Dependent on the type of fault-injection (SET or SEU), the fault characteristics are taken from the transient fault library (TFL) or from

the upset fault library (UFL). The selected fault-injector is active for the period of time defined in the related fault library.

4.2. Test Circuits

In order to prove the failure-free probabilities of TMR and DMR circuits with latchup protection, two types of digital circuits are designed and tested – shift register and counter. In the following sections are described the test circuits with implemented models for radiation effects. Shift register and counter are tested with implemented models for upset and transient effects. SPS cell is tested with model for latchup effect. On the end is provided a circuit description for testing a redundant based system on latchup effects.

4.2.1. SET/SEU test circuits

Based on the fault-injection simulation approach, the shift register is implemented using the flip-flops with fault-injectors. In Figures 4.8 and 4.9 are shown implemented 4-bit shift register and 4-bit counter, respectively. The points where faults are injected are also presented in figures as “FI” blocks. FI block implements the circuit described in Chapter 4.1.3. VHDL descriptions of the shift register and counter are presented in Appendix B.

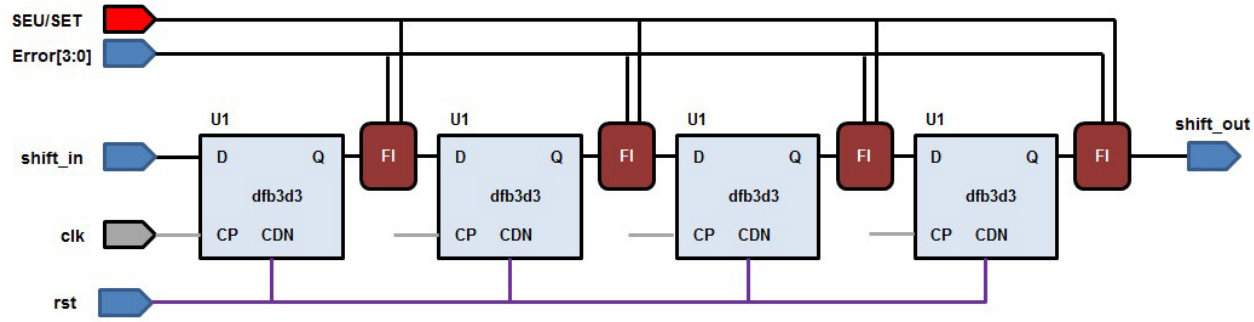


Figure 4.8. Shift register test circuit without mitigation techniques

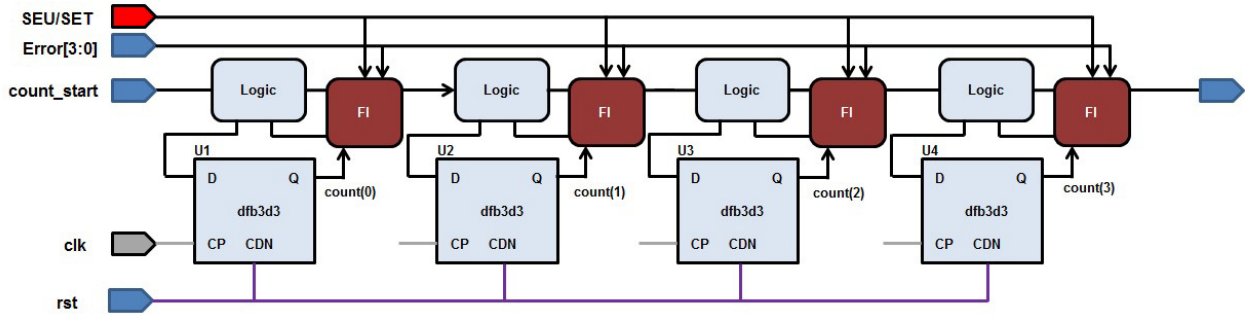


Figure 4.9. Counter test circuit without mitigation techniques

For test purposes, the test architectures are implemented using several redundant circuit approaches:

- Single flip-flop - standard approach
- TMR circuit

- TMR circuit with latchup protection mode
- Self-voting DMR circuit
- Self-voting DMR circuit with latchup protection mode

Shift register is driven by different test patterns in order to provide the circuit sensitivity on the pattern during fault-injection. There are four test patterns used for the tests:

- **All '0'** – test pattern is used to get the circuit sensitivity on faults that force the gate/flip-flop outputs to high logic level;
- **All '1'** – test pattern is used to get the circuit sensitivity on faults that force the gate/flip-flop outputs to low logic level;;
- **Alternate '0-1-0'** – test pattern is used for testing the digital circuit in the case when the data logic states change in each clock cycle. This pattern can also be interpreted as pattern with the most frequent changes of the logic states;
- **Combined test pattern** – is specially developed test pattern, where groups of different numbers of high logic states and low logic states arrive successively. Combined test pattern should provide a real situation in digital circuits when logic states are not changed on every clock cycle but also they are not static. The test pattern is defined by its weight. Patterns for the first three weights are presented in Table 4.2. The number of bits (a_n) for pattern of weight *n* can be calculated as an arithmetical series, represented in Equation 4.3.

$$a_n = 2a_{n-1} - a_{n-2} + 4 \; (n > 3) \qquad (4.3)$$

Table 4.2. Combined Test Pattern Generating

n	Pattern	a_n
1	010101	6
2	0101100101	10
3	01011001110001100101	20

The combined test pattern, used for the fault-injection simulation, is generated for the weight of n=100. After calculation provided by Equation 4.3, the number of bits is 20000. The sequence is repeated few times during simulation in order to provide continuous pattern for the shift register. The C source code developed for automated combined test generating is described in Appendix B.

The simulation based fault-injection is performed using more than million clock cycles (1047300; one clock cycle duration is 10ns) with 104730 injected faults. The injected faults have random positions, durations and occurrence moments. The outputs of shift registers and counters under the fault-injection are compared with the outputs of referent shift registers and counters. Referent shift registers and counters are operating as ideal circuits without fault-injection. In each clock cycle, the result of comparison is written in the "data collector" and after finished simulation, the "data analyzer" provides information about detected errors during fault-injection simulation. The "data collector" and "data analyzer" are described in Chapter 4.1.

4.2.2. SEL test circuits

During the development process of SPS cell, in order to provide the accurate functional verification, it was important to define an appropriate simulation environment. The similar approach is used for the measurements, which are described in next chapter. The simulation environment of SPS cell consists of the three main parts:

a) Latchup generator;
b) Control block;
c) Digital block supplied by the SPS cell.

A SPS cell simulation setup is presented in Figure 4.10. It can also be used as block diagram for the SPS cell measurement environment representation.

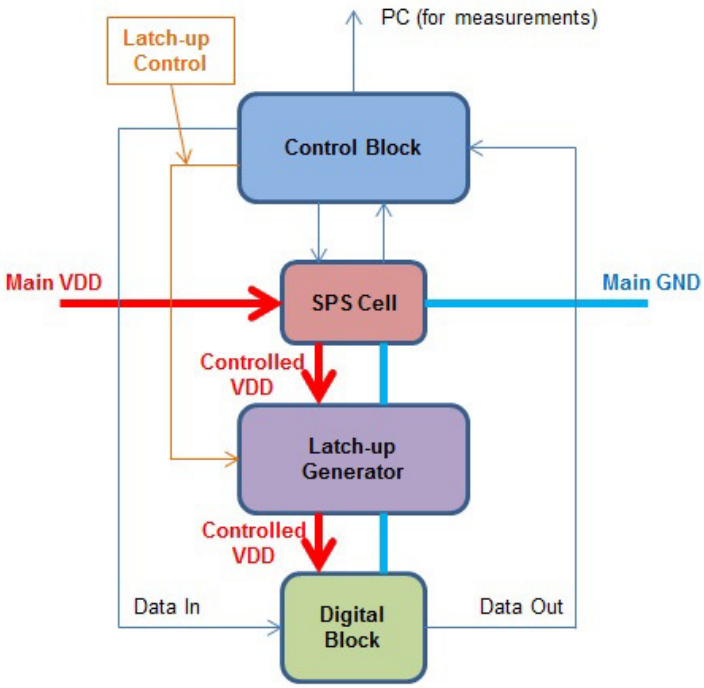


Figure 4.10. SPS cell simulation (measurement) setup

The latchup generator is based on the physical process which describes how a latchup is induced in a CMOS pair. As it is known [HAS05] [RAG02], the latchup effect is based on a parasitic thyristor component, formed in the CMOS pair. Details related to the physical background on the latchup effect are provided in Chapter 2.3. In order to save the time required for designing a technologically dependent thyristor, a simple switch controlled by voltage (VCSW) is used in the simulation process. This approach, used for latchup generator, also provides a simple hardware realization which is required for measurements. The schematic of the latchup generator is presented in Figure 4.11.

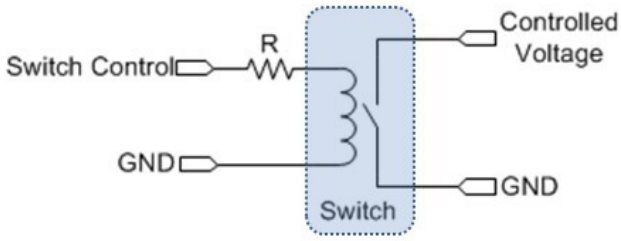


Figure 4.11. Latchup generator schematics used for simulation phase

A control block is used to provide the input control signals for the SPS cell under test. Following the previously provided description of SPS cell (Chapter 3), it is easy to notice the relation between SPS control signals and signals which are describing the current status of SPS cell. Control block is also used to provide initial control signal for the latchup generator, as well as stimulus signals for the digital circuit. For the simulation phase, the control block is a part of the test bench. It is important to note that all simulations are done in the analog environment using Cadence tools and Spectre simulator [CAD13].

Digital block, whose power supply is controlled by SPS cell, is a simple DMR based digital system, which consists of the standard flip-flops (FF) and other gates. The digital block tests are used to provide information what is exactly happening with the data in the moment when the latchup effect occurs, during and after it.

Verification process of the SPS cell, using the test environment presented in Figure 4.10 starts by defining the required operational conditions for the SPS cell. Required operational conditions are defined by control signals, provided from the power network controller (PNC) side. Next step in the verification process is the digital block validation. The control block provides the data input, clocks and other control signals important for the digital block. The control block then verifies the correctness of the digital block using the “data out”, generated by the digital block. When the digital block is verified, the control block activates the latchup generator, which provides short-circuit between “controlled VDD” and “main GND”. The “controlled VDD” is controlled main power supply (“main VDD”). It is controlled by SPS cell. GND is the main ground used in test circuit. All mentioned signals are shown in the SPS simulation environment (Figure 4.10). After “latchup generator” generates short-circuit the “Control Block” detects the “TSTART” signal and generates the required control signals (“TSTOP” and “Poff”).

In order to prove the functional correctness of the presented design methodology, during the development phase of SPS cell and redundant circuits which support latchup protection, several different designs were simulated. Shift register is composed of the sequential elements and therefore useful for observation of the changed functionality during latchup protection phase. On the other hand, the counter is composed of sequential elements and combinational logic. Therefore, the counter can be used as very good example of usually implemented digital circuits. Especially because the counter circuit has feed-back lines, which are required for correct counting and connected to the inputs of sequential elements. These feed-back lines are very useful for early detection of the redundant circuit

"weak points". In the following text are presented two examples of the digital circuits, designed using the DMR redundant circuits with latchup protection.

The schematic of the 4-bit shift register with implemented SPS cell and double modular redundancy is presented in Figure 4.12.

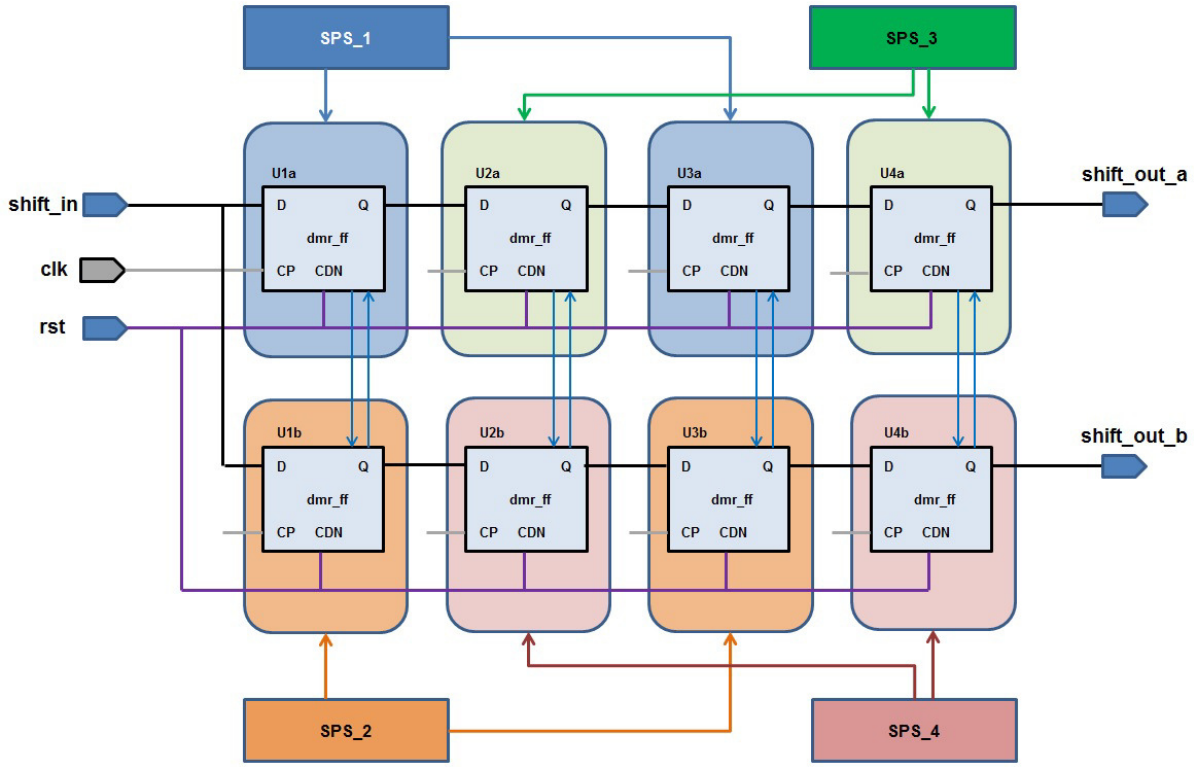


Figure 4.12. 4-bit shift register with implemented latchup protection and DMR

Next example shows a synchronous counter, which integrates SPS cell and DMR redundant circuit with latchup protection. In Figure 4.13 is presented a "counter cell", which is composed from standard flip-flop and combinational logic. Figure 4.14 presents a block diagram of 4-bit counter, designed using DMR approach with latchup protection.

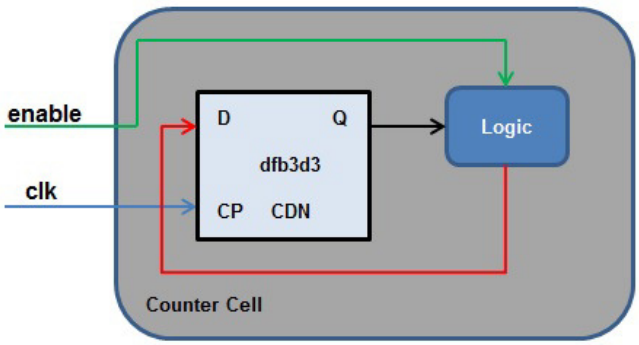


Figure 4.13. Block diagram of the counter cell

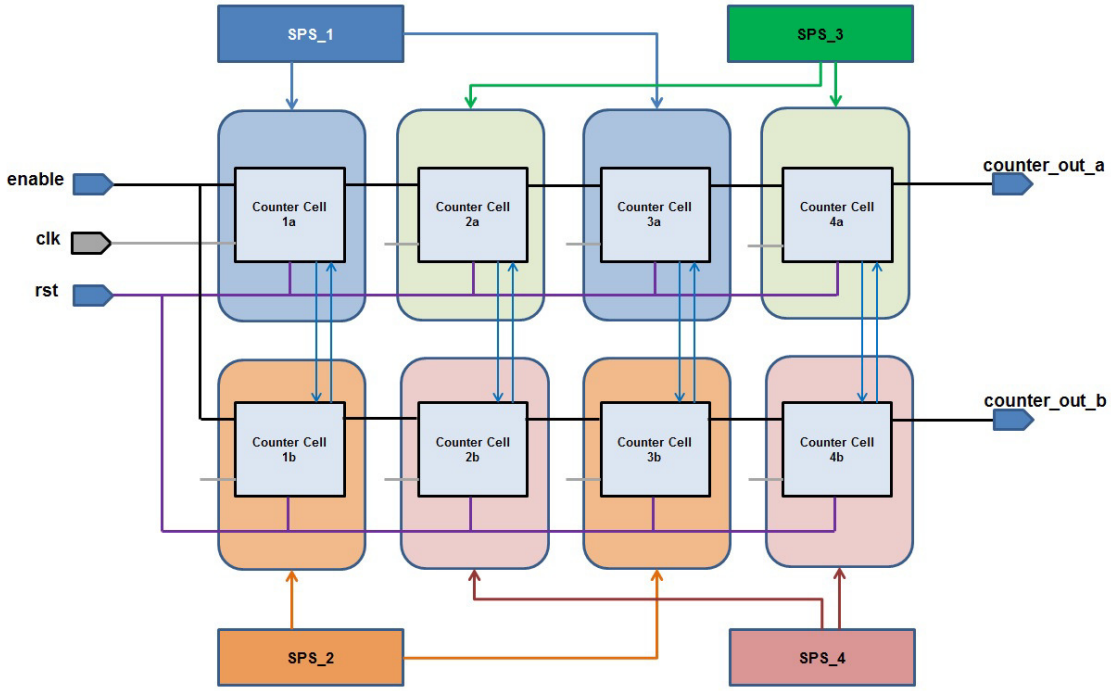


Figure 4.14. Synchronous counter block diagram – DMR based design

4.3. Simulation Results

As it was mentioned before, there are two types of tests. Results of the single event upset and single event transient tests are provided separately from single event latchup tests. This is done in order to get the circuit sensitivity for each type of radiation effect. Dependent on the application (environment) where the digital system will be used, it is possible to define different simulation parameters (frequency of error occurrence) and to decide which redundant system can be used. This section provides simulation results with implemented fault-injection models for each type of the mentioned single event effects.

It is useful to note that during shift register tests (SEU and SET) the highest sensitivity was noticed for the “alternate 0-1-0” test pattern. This test pattern is very useful for testing the shift registers and serial-to-parallel convertors. The mentioned test pattern enables a digital circuit testing, when the data logic states change in each clock cycle. Results related to the counter circuit are independent form any test pattern because the data during test is generated from the counter itself.

In Chapter 2 and 3 are defined analytic models for redundant circuits which provide failure-free probability cross-checking. On one side are failure-free probabilities based on the fault-injection simulation and on the other one the analytically defined (calculated) failure-free probabilities. The analytical model provides statistical information about circuit’s failure-free probability. The circuit in this case represents a sequential element. Therefore, the sequential element can be interpreted as: single flip-flop, TMR circuit, DMR circuit etc. Shift register and counter are digital designs, composed from sequential elements and some other combinational logic. The failure-free probability of sequential

elements and other gates, used for designing the digital circuits, have influences on the failure-free probability of complete digital circuit.

Table 4.3 represents the calculated failure-free probabilities for the DMR and TMR circuits with and without latchup protection mode. In Figure 3.29 are compared the curves which show the failure-free probability for the different TMR and DMR circuit versions, in relation to the fault-free probability of memory element (flip-flop). The voter fault-free probability is fixed ($P_V = 0.99$). In Table 4.3 are defined parameters as Δ_{TMR} and Δ_{DMR}. The Δ_{TMR} and Δ_{DMR} represent the difference between calculated failure-free probabilities of the TMR and DMR circuits with and without latchup protection support, respectively. Equations 4.4 and 4.5 are used for Δ_{TMR} and Δ_{DMR} calculation. The differences are expressed in percentage and they represent the change of the redundant circuit sensitivity on upset and transient effects. The mentioned differences are also visible after fault-injection analysis.

$$\Delta_{TMR} = \frac{P_{TMR} - P_{TRM\ LUT}}{P_{TMR\ LUT}} \times 100\ [\%] \qquad (4.4)$$

$$\Delta_{DMR} = \frac{P_{DMR} - P_{DRM\ LUT}}{P_{DMR\ LUT}} \times 100\ [\%] \qquad (4.5)$$

Table 4.3. Calculated failure-free probabilities for redundant circuits

Flip-flop	Voter	P_{TMR}	$P_{TMR\ LUT}$	$P_{DMR\ FB}$	$P_{DMR\ FB\ LUT}$	Δ_{TMR}[%]	Δ_{DMR}[%]
P_M=0.95	P_V=0.99	0.9898	0.9631	0.8864	0.9327	-2.69	+5.22
	P_V=0.95	0.9733	0.9095	0.8339	0.8868	-6.38	+6.34
	P_V=0.9	0.9430	0.8219	0.7629	0.8231	-12.11	+7.89
P_M=0.99	P_V=0.99	0.9988	0.9954	0.9962	0.9976	-0.34	+0.14
	P_V=0.95	0.9898	0.9631	0.9766	0.9817	-2.67	+0.52
	P_V=0.9	0.9669	0.8901	0.9325	0.9450	-7.68	+1.34

By careful analysis of the data presented in Table 4.3 it is possible to notice an interesting effect. The failure-free probability of the TMR circuit with latchup effect is lower than standard TMR circuit. Main reason why the fault-free probability decreases is additional logic. The decrease of failure-free probability is higher in range of lower fault-free probabilities ($P_M < 0.95$). On the other hand, the failure-free probability of DMR circuit with latchup protection increases. Reason for this effect is based on the circuit topology. In the DMR circuit with latchup protection, the additional logic ("data keep" block) is implemented in the voter feed-back line. Correction circuit in the feed-back line provides correction of transient faults immediately. In the DMR circuit with latchup protection is also implemented the "voting loop", where one voter provides input for second one, but in the same time a second one provides input for the first one. This is not a case in the TMR circuit with latchup protection. In the TMR circuit with latchup protection, the serial voting approach is implemented. The first voter provides a voted value which is transferred to the output of the TMR circuit. Therefore, if some transient effect occurs in the extended circuit, it will be sent to other elements (gates, cells etc.) as a valid signal.

Figure 4.15 shows diagrams of TMR and DMR circuits with and without latchup protection. It is possible to notice the TMR failure-free probability reduction (Figure 4.15.a) and on the other hand, the DMR failure-free probability improvement (Figure 4.15.b).

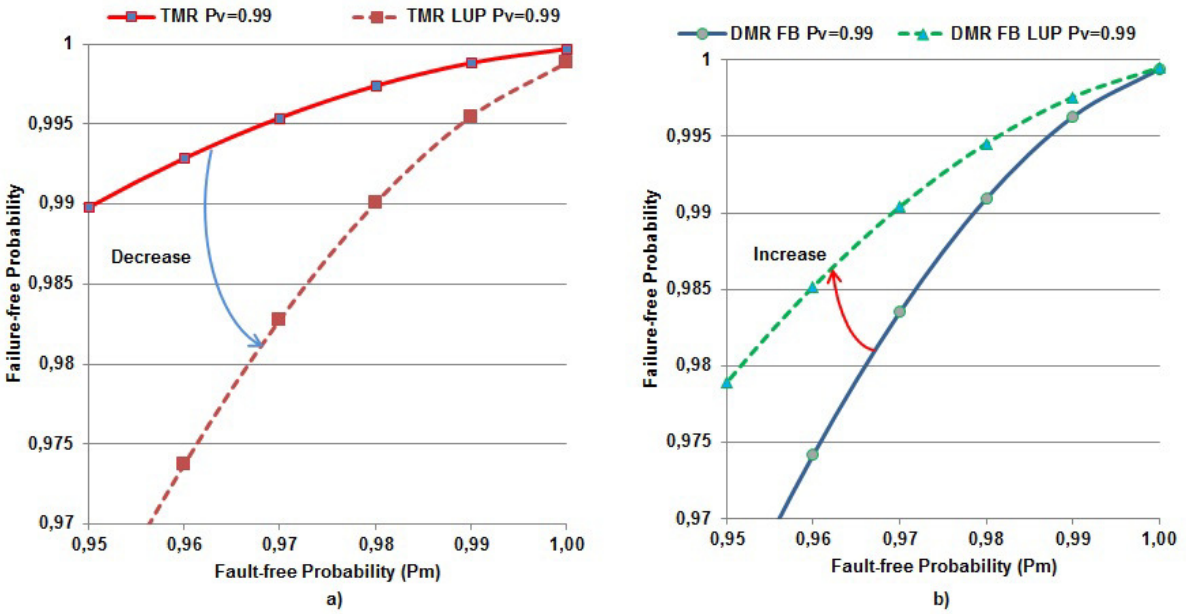


Figure 4.15. a) Failure-free probabilities for TMR circuit with and without latchup protection; b) Failure-free probabilities for DMR circuit with and without latchup protection

Fault-injection simulations are performed on shift register and counter circuits in order to validate the different redundant circuit approaches. As output, the fault-injection based simulation provides a number of errors, which were detected during the simulation process. Dependent on the number of errors, it is possible to calculate the average failure-free probability of tested circuit (shift register or counter), as well as the failure-free probability of the redundant circuit. Failure-free probability of redundant circuit is calculated based on the work provided in [RAC12]. The failure-free probability of the serially connected modules is calculated as multiplied fault-free probabilities of the modules.

Average simulated failure-free probabilities for the tested circuits are calculated using Equation 3.24. The simulated failure-free probability can also be expressed in percentage (Equation 3.25).

$$P_S = 1 - \frac{detected\ errors(SET) + detected\ errors(SEU)}{2 * number\ of\ cycles} \qquad (3.24)$$

$$P_S^{\%} = P_S * 100 \qquad (3.25)$$

In the following Table 4.4 are presented results after fault-injection simulation. The number of errors is defined for the digital test circuit (shift register and counter). The number of cycles used for fault-injection simulation is 1047300.

Table 4.4. Number of detected errors after fault-injection simulation

Sequential Element/Circuit	Shift Register		Counter	
	SEU	SET	SEU	SET
Flip-flop	523522	523336	594381	523761
TMR	0	0	0	0
TMR LUP	758	0	449631	0
DMR FB	1115	12437	520143	528096
DMR FB LUP	983	8306	408964	18561

Failure-free probability of redundant circuit (P_R) can be calculated using Equation 3.26.

$$P_R = \sqrt[n]{P_S} \tag{3.26}$$

In Equation 3.26, the *n* represents the number of sequential elements in the test circuit. For example, in case of 4-bit shift register (same for counter), the number *n* is 4.

Comparison between simulated and calculated failure-free probabilities should confirm the correctness of presented results. In Table 4.5 are presented the calculated and simulated failure-free probabilities of redundant circuits. From Table 4.5 is possible to notice difference which is not higher than 5%.

Table 4.5. Simulated and calculated fault-free probabilities of redundant circuits

Sequential Element/Circuit	Simulated Failure-free Probability	Calculated Failure-free Probability
Flip-flop	0.4999	n/a
TMR	1.0000	0.9898
TMR LUP	0.9412	0.9631
DMR FB	0.9295	0.8864
DMR FB LUP	0.9759	0.9327

After SEU and SET fault-injection simulation, next step is related to the validation that the circuit operates properly under SEL (latchup effect).

Simulation is performed on the DMR based 4-bit shift register and 4-bit counter. In Figure 4.15 are shown timing diagrams of the shift register during latchup protection phase. It is important to note that test covers the case when both SPS cells operate in protection mode. It is important to note that after protection phase, the logic states of the DMR based digital circuit are continuously present and shifting process continues without problems. Block diagram of test circuit (shift register), used for latchup test is presented in Figure 4.12.

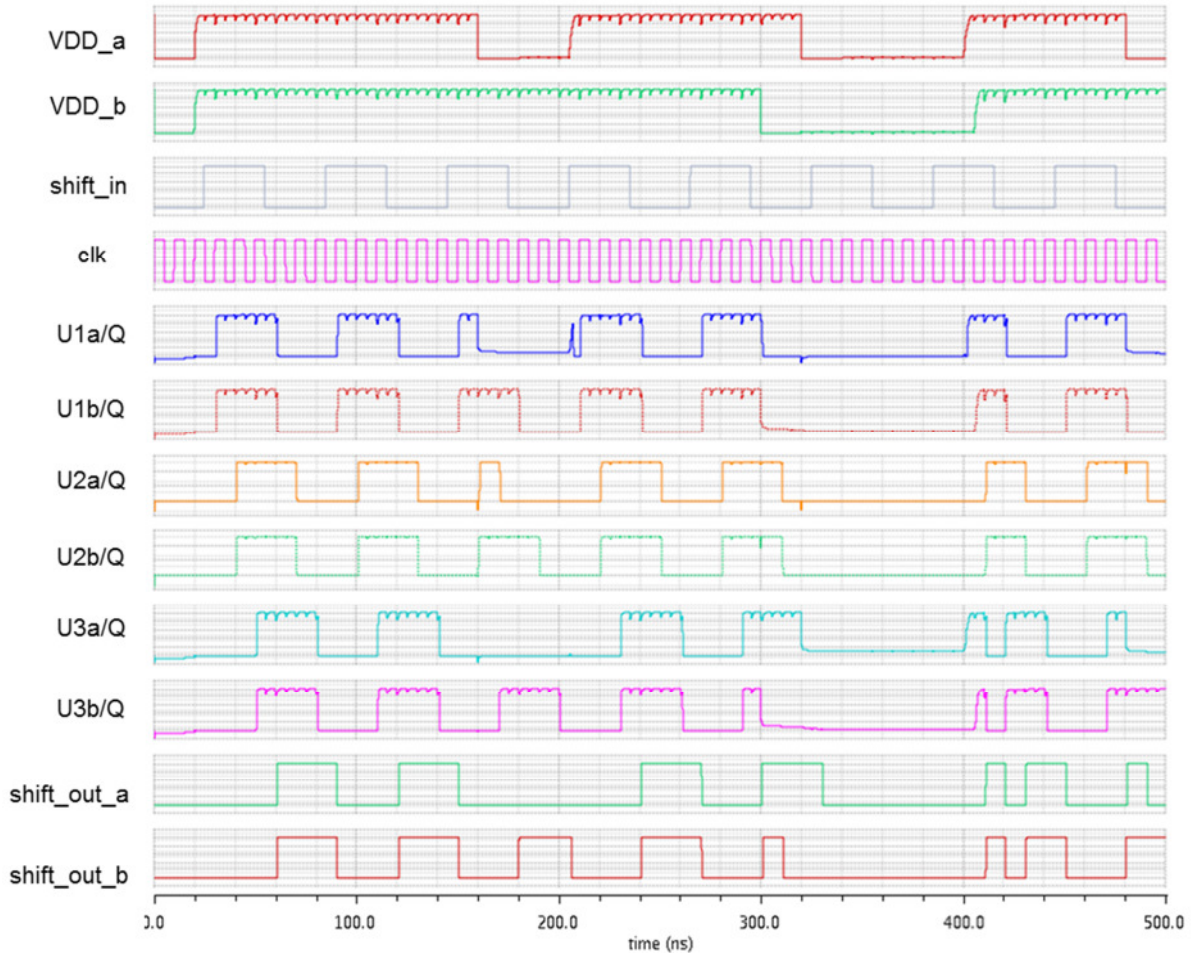


Figure 4.15. Simulation results of latchup protection for DMR based shift register

An important fact, which can be noticed from Figure 4.15, is undisturbed operation of redundant flip-flop (U1b/q) during the latchup protection phase of the other one (U1a/q). This is provided by DMR flip-flop with latchup protection support, which was in details described before (Chapter 3.2.2). The VDD_a is controlled by SPS cell 1 and power supply VDD_b is controlled by SPS cell 2. Other SPS cells are always active.

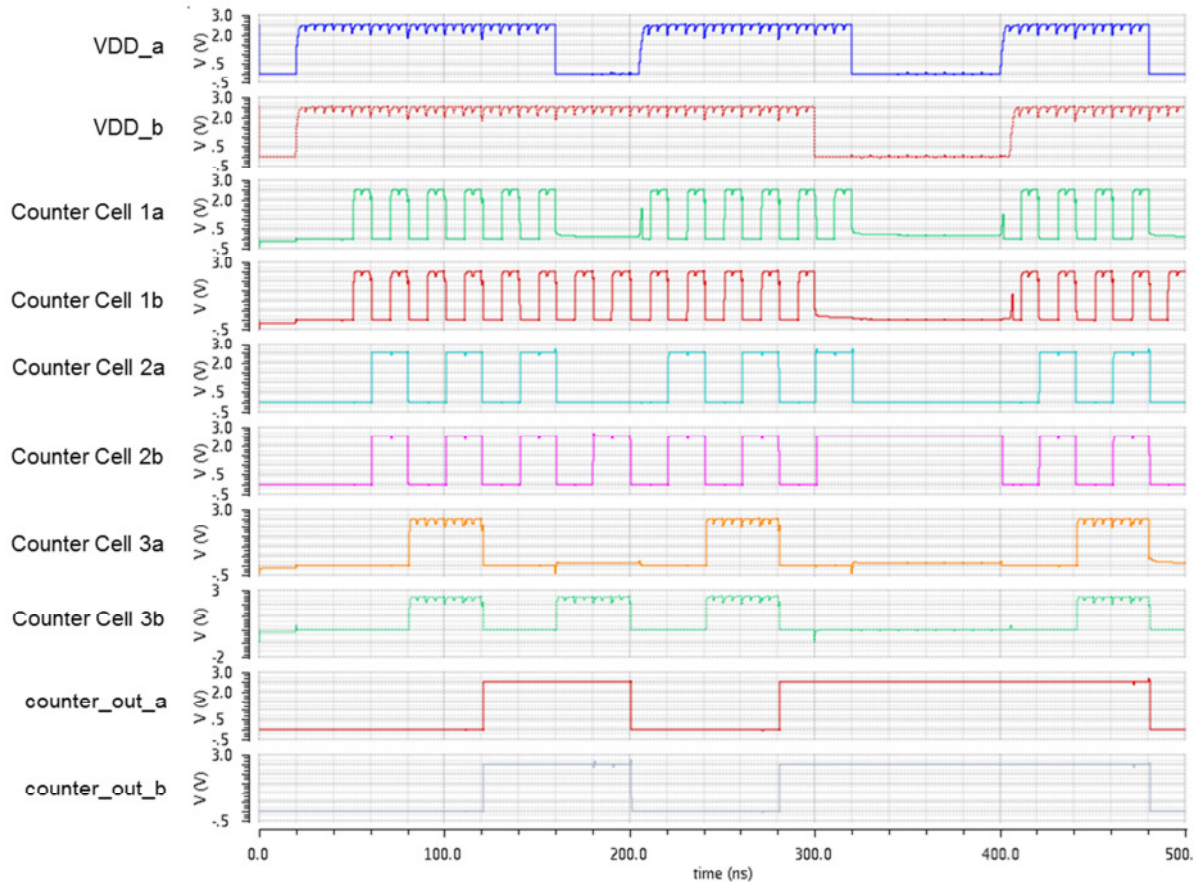


Figure 4.16. Simulation results of latchup protection for DMR based counter circuit

Similar as the shift register timing diagram is discussed, here is also possible to notice the process of the circuit state recovery after latchup protection phase. Block diagram of counter test circuit used for latchup tests is illustrated in Figure 4.14.

Very important result, which should be noticed, is related to the power supplies behavior. From simulation results (for shift-register and counter) it is easy to notice that the power supply has small voltage drops during the clock edges. This effect and nature of its occurrence is already described in Chapter 3. Using a stronger driver (10µm driver transistor in the SPS cell), provides better results regarding the voltage drop. Dependent on application, the designer should decide which SPS cell driver and which type of redundancy should be used. The following text is related to design automation flow for the fault-tolerant digital circuits and examples will be based on the DMR circuit with latchup protection. Same analysis can be provided for the TMR circuit with latchup protection.

5. Modified Design Flow and Circuits Implementation

Fault-tolerant ASICs can be implemented using standard design automation tools [SYN13], [MNT13], [CAD13] and introducing a few additional steps in design flow. An extra step compared to the standard design flow is necessary to generate a new netlist including redundant cells, voters and required protection for memory blocks. The other two extra steps (definition of the power domains and placement of the SPS standard cells) have to be made in the layout phase.

Duplication (for DMR) or triplication (for TMR) of the original net-list is performed using a parsing script which can be included in the synthesis script as an optional subprogram. The parser is generated by Flex-Bison parser generators [GNU13], [LEV09]. Memories must include the protection bits (parity bits, Hamming code, etc.) and be attached to the EDAC logic in order to tolerate SEUs. Duplication of memories and EDACs is necessary for the SEL protection [SCH09].

A simple modification of the technology file suffices in case of the redefinition of the power domains. After the power planning has been finished, the placement of SEL protection switches (SPS) can be performed. The SPS cells are placed under the crossover points of the power stripes and cell rows (where the filler cells are usually placed). A SPS cell (or a group of SPS cells) protects only one power domain and corresponding redundant logic. SPS cells may not be placed under all power crossover points. The maximal current needed for driving the connected logic defines the required (minimal) SPS output current.

5.1. Modified Design Flow

Based on the discussions and conclusions provided in the previous chapters, in this section are provided extended design step descriptions for fault-tolerant ASIC design. Before giving a view on the extended design flow it is important to look back on the requirements related to the redundancy and power protection. In Figure 5.2 is presented the extended design flow diagram. In order to implement a required hardware redundancy based on double-modular redundancy, an extra step after synthesis phase is added – “Netlist Modification”. The netlist modification process is performed using specially developed parser. For the implementation of the specially designed power protection cells (SPS) is used an extra step during placement phase (physical design). The redundancy provides higher protection level against single event upsets and single event transients, as it was mentioned in Chapter 3.1. The protection against single event latchup effect is provided using specially designed power switches, described in Chapter 3.2.

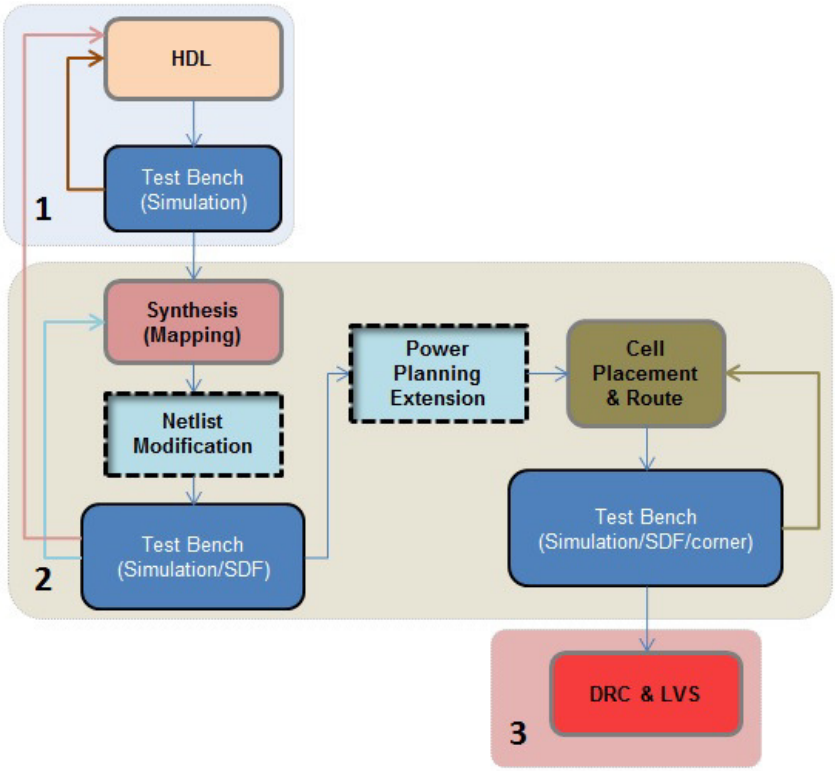


Figure 5.1. Additional steps diagram in Design Flow

The hardware description phase can also integrate a redundancy. It is important to notice that in this discussion, the hardware description phase is mainly related to the VHDL language. Similar to the VHDL it is possible to use Verilog language. Verilog is here used for already mapped design – for example, the netlist after synthesis phase and netlist after physical design (layout) phase.

Redundancy is in VHDL defined by multiplication of the processes which are defining storage elements (flip-flops or latches). For the design from scratch it should not be a problem. In case that it is required to provide the redundancy for finished designs as processor cores for example (Leon, IPMs, MIPS, etc.) the process of redundancy implementation in VHDL development phase can be a very complex. Second critical point is usage of encrypted HDL codes. The redundancy can also be added on the higher levels as modules but this is not the topic of the thesis and it will not be further discussed. Therefore, comparing to the standard design flow the synthesis phase is not changed. As it was mentioned before, the output after synthesis phase are: netlist (described in Verilog), SDF timing data, SDC constraint data and reports. On this point comes an additional step, which is related to the netlist modification.

A special parser is designed for the netlist modification. The parser is designed for the Flex-Bison text processing tools. Bison is a parser generator, which is often used with Flex. Flex is an automatic lexical analyzer, which is used to tokenize input data, needed by the Bison parser [LEV09].

The synthetized netlist is written in Verilog hardware description language and compared to the VHDL it contains just the names of standard cells as well as their interconnections. Therefore, here is important to notice an important difference between VHDL and Verilog hardware description

languages. Although both of them have the same goal – to describe the hardware, VHDL is better for hardware description on a "higher level". The "higher level" implies complex mathematical forms and similar complex logic functions. On the other hand, the Verilog is most commonly used in the design and verification of digital circuits at the register-transfer level of abstraction. It is also used in the verification of analog circuits and mixed-signal circuits.

Following sections are describing how the netlist parser operates and method of integrating the latchup protection cells (SPS cell) into standard designed digital circuit.

5.1.1. Netlist Parser

Besides the netlist modification, parser is also used for automated integration of fault injectors. In the following text are provided the descriptions of netlist parser outputs.

Netlist Modification

After synthesis phase, a Verilog netlist is generated. It contains the standard cells which are defined in the device technology. In the further text is presented an example based on the IHP 250 nm technology. The available storage elements (flip-flops) for the mentioned technology are presented in Table 5.1. **Note:** Thesis work doesn't include redundancy for the level-sensitive storage elements – latch standard cells.

The parser includes options for fault-injection and DMR LUP (DMR with latchup protection support) netlist generating.

Table 5.1. IHP 250 nm Flip-flops

Flip-flop name	Comment
dfb3d1, dfb3d2 and dfb3d3	b – Flip-flop group with asynchronous SET and RESET inputs
dfc3d1, dfc3d2 and dfc3d3	c – Flip-flop group with asynchronous RESET input
dfn3d1, dfn3d2 and dfn3d3	n – Group without asynchronous SET and RESET inputs
dfp3d1, dfp3d2 and dfp3d3	p – Flip-flop group with asynchronous SET input

The developed netlist modification parser operates in two different modes:

- DMR mode with latchup protection support
- Fault-injection mode

The DMR netlist can be generated only with latchup protection support. As it was mentioned before (Chapter 3.2.2.), the DMR circuit which doesn't include a latchup protection has very low failure-free probability. It is important to re-optimize the parsed netlist because of the potential timing problems after parsing.

In Appendix C is provided an example how parser transforms a netlist, which is synthetized using standard approach, into fault-tolerant netlist.

Memory protection

In case that netlist contains the memories, what is usually the case, the parser implements extra hardware for error correction and detection. In the presented design methodology is used EDAC (Error Detection and Correction), based on the Hsiao coding scheme [HIS70].

Hsiao codes are modified version of Hamming codes. Hsiao code is very popular because of error detection and correction, and in the same time Hsiao coding system is used for power-saving implementation. Hsiao codes are single bit error correction and double bit error detection codes (SEC-DED). Using Table 5.1 it is possible to describe a Hsiao coding matrix. The Hsiao matrix generating is described using an example for coding an 8-bit (1B) word [GOE08] [HAO08].

First of all, if Hamming code is used, the control bits are on positions 2^m, where ***m*** is integer. Control bits are parity bits of some sub words, which are explained in Table 5.2.

Table 5.2. Hamming coding scheme

Bit number	1	2	3	4	5	6	7	8	9	10	11	12
First parity bit	P1		X		X		X		X		X	
Second parity bit		P2	X			X	X			X	X	
Third parity bit				P3	X	X	X					X

On bit position 1, in coded word, first parity bit $\mathbf{P_1}$ (marked red) is written. P_1 represents a parity bit of first sub-word. This sub word consists of bits where red **X**s are marked. Same is done for second and third parity check bit (green and blue respectively).

Hsiao code is also based on this principle, but after mathematical analysis for one bit error correction and two bit error detection, it is possible to notice that for every coded bit, from word which we want to code, we need to have an odd number of parity bits. Minimal odd number is three. As we can see, in Table 5.2, only one bit on position 7 has triple covering. Relation 5.1 describes the dependency between number of control bits (***c***) and maximal number of data bits (***d***).

$$d = 2^{c-1} - c \qquad (5.1)$$

From relation 5.1 for 8 data bits example, we need to have minimum 5 control bits. If we use Hsiao coding scheme for 8-bit word example, the result is presented in Table 5.3.

Table 5.3. Hsiao matrix rows from Hamming coding scheme for 8-bit word

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25
P1	X		X		X		X		X		X		X		X		X		X		X		X		X
P2		X	X			X	X			X	X			X	X			X	X			X	X		
P3				X	X	X	X					X	X	X	X					X	X	X	X		
P4								X	X	X	X	X	X	X	X									X	X
P5																X	X	X	X	X	X	X	X	X	X

Fault-injection – in the FI mode, fault injector circuits are added on the storage element inputs and outputs. Fault-injection circuit on input represents a hardware emulated transient effect, which occurs in the combinational logic before flip-flop. On the other hand, the fault-injection circuit on the flip-flop output represents an emulated upset effect. This option enables the verification of designed fault-tolerant digital system.

5.1.2. SPS cell automated placement and routing

A very important design automation step is related to the placement of SPS cells. Parallel to the placement of SPS cells should be provided the integration process of the power network controller (PNC) within the standard design steps. Approach for automated SPS cell placement, which is used in thesis is based on the Cadence Low-Power Implementation flow [CPF13].

After successful design and verification of SPS cell it is necessary to provide the formats which are understandable for other automation design tools. Timing information of SPS cell is provided in .LIB (Liberty) file format and it is partially generated by Cadence Encounter Library Characterizer. The designer needs to add manually I-V characteristics of sensor-driver transistor. The layout abstract view (.LEF) is generated by Abstract Generator, provided by Cadence Virtuoso design tool. Design related files (.LIB and .LEF) for DMR approach with latchup protection support are presented in details in Appendix E.

It is important to note that the presented approach follows the Cadence Low-Power Implementation flow. Therefore, it is required to use a Common Power Format (.CPF) during fault-tolerant ASIC power planning phase. The CPF file is used to describe the power domains, related control pins of power switches and names of the controlled power domains [CPF13].

Power network controller (PNC) should be added manually after netlist parsing. Therefore, the top design consists of the DMR based core and PNC digital circuit. Recommendation is to design PNC related to the number of SPS cells. It is enough to add the PNC as module on the top without any manually provided interconnections. All important interconnections are defined in the CPF file and tool performs automatically routing procedure during the routing procedure of standard cells.

During fault-tolerant ASIC design, based on the proposed methodology, it is recommended to use the following steps:

1) During the floor planning phase, the designer should take care about the core dimensions in order to provide the latchup protection for all rows. SPS cell height is equal to four standard cell rows and therefore during the floor planning it is important to calculate the core height based on Equation 5.2.

$$h_{core} = n_{required} \times 4 \times h_{cell} \qquad (5.2)$$

Member $n_{required}$ is a number of rows required for the optimal core utilization. Usually, the optimal core utilization is in range between 70% and 80%. Core utilization represents the ratio between occupied and free area in the core. It is important to leave some free area for

clock buffers and other optimization related standard cells as well as routing resources. In IHP 250 nm technology the standard cells height h_{cell} is 8.4 µm.

Example: For an ASIC chip it is assumed that 15 rows are enough for the digital core implementation. As the mentioned design requires the latchup protection, there are two possible dimensions for core height:

a) 16 rows (16/4=4 SPS cells); h_{core} = 134.4 µm

b) 12 rows (12/4=3 SPS cells); h_{core} = 100.8 µm

Dependent on the core utilization, one of two calculated core heights should be used for further implementation.

2) After core dimensioning, it is mandatory to create and load .CPF file, which provides information to the tool about redundant power supplies;

3) Defining the area for power network controller (PNC). Recommendation is to place the core in the middle of design, so PNC can be placed in a ring form around core. This is done for better routing results between SPS cells and PNC;

4) Creation of Power stripes and power rings around core and PNC ring;

5) SPS power cell placement. In case that all commands in CPF file are correctly defined, the SPS cells are placed directly under stripes. SPS cells are placed under power stripes (where are usually placed filler cells) and provide interconnection between power supply lines of “row section” and main power supply provided by power stripes.;

6) Special routing – during special routing phase it is important to control how the SPS cells are connected to the stripes and how row supplies are distributed;

7) Placement of standard cells – during the placement phase it is very easy to control how standard cells of redundant power domains are placed on the correct positions. Placement algorithm follows power supply names of standard cells and power supply names of rows (where standard cells should be placed).

8) Clock tree synthesis;

9) Final routing;

10) Geometry and connectivity verification.

Verification of digital circuit which is designed using the presented methodology is same as standard one. The only problem is simultaneously latchup effects simulation during functional verification. Using the fault injection option it is possible to provide the digital circuit verification parallel with fault-injection. The only problem is the test bench complexity because of the many interconnections and functions required for the fault-injection in analog environment.

It is important to note that SPS cell need to be designed for the implementation technology as special power cell. Examples used in thesis are based on the IHP 250 nm process and they are presented and explained in order to understand the principle how fault-tolerant design can be developed using the standard design tools and approaches.

5.2. Implemented Test Circuits

In order to provide tests on silicon, two types of SPS circuits (cells) are implemented in IHP 250 nm technology [IHP13]. Besides the SPS cells, a small DMR based digital system is also developed in order to test how the latchup protection functions together with redundant circuits. Implementation and simulations are provided using Cadence Virtuoso 6.1.5 design tool [CAD13].

5.2.1. SPS network

In this section is in details provided description of SPS network. The SPS network comprises the power network controller (PNC) and all the SPS cells.

A power network of an ASIC is composed of power lines (wires) and it is used for power supply distribution to all standard cells and other circuits (memories, analog blocks, etc.). Before detailed description of SPS network it is useful to define main terms which will be used in the following text.

- **Power supply** – Power supply represents a device used as source of voltage and current, required for normal operation of an electronic system. An ASIC has pads for ground voltage connection, pads for standard cells supply and pads for supply the electronics in I/O pads. Under term “power supply”, in the further text is mentioned pair of power supply for standard cells (VDD) and ground (GND).
- **Power row** – Power row represents a prepared row for the standard cells placement. Compared to the row (area reserved for standard cells), power row has VDD and GND wires, where the standard cells will connect own VDD and GND terminals. Power rows are supplied by power stripes. Power rows are usually designed in the lowest metallization layers – Metal1.
- **Power stripes** – Power stripes are used to provide equal power supply to the power rows and to provide connection between power rows and power rings.
- **Power ring** – Power ring is designed around an ASIC core in order to provide enough power for the proper core operation and to provide connection between power pads and power supply network of an ASIC core (power stripes and power rows).
- **Row section** – Row section represents a region reserved for the standard cells between two power stripes. One row section is usually supplied from left and right power stripes pairs, where both of them, in the same time, define a row section. See Figure 5.6.
- **Standard cells power supply terminals** – Each standard cell has integrated power supply lines and terminals. It was mentioned before that power rows are designed in the lowest metallization layer. Depend on the technology used for ASIC implementation, standard cells

need to have same metallization layer for power supply terminals as power rows. Therefore, after placement of standard cells, the power rows are automatically connected to the standard cell power supply lines (VDD and GND).

Example of an ASIC layout view with implemented power supply terminals is presented in Figure 5.6. Power rings are not shown in the mentioned figure because they are not relevant for the modified power network approach description.

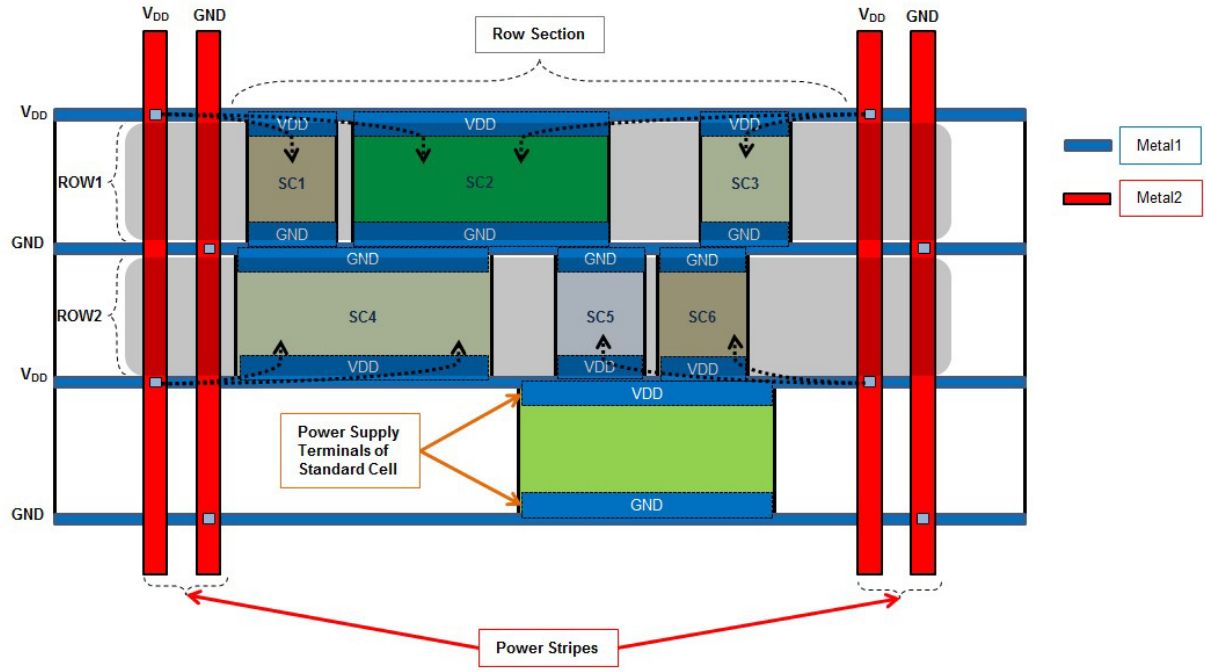


Figure 5.2. Standard power network of an ASIC chip

In Figure 5.6 are also shown the current directions (black dotted arrows). It is possible to notice that the currents, provided by the power stripes are almost equally distributed between standard cells. Standard cells are supplied from left and right power stripe lines. In order to provide basis for the modified power network, which integrates the SPS protection cells, it is required to provide some information related to the “row section” power consumption of standard power network.

The start point for the row power estimation is the power consumption of one “row section”. The “row section” is group of standard cells, placed in one row between two power stripes. The distance between two power stripes is dependent on the technology used for the ASIC development. An example, provided in the following text is based on digital design, optimized for the frequency of 100MHz. For implementation is used standard IHP 250 nm process [IHP13].

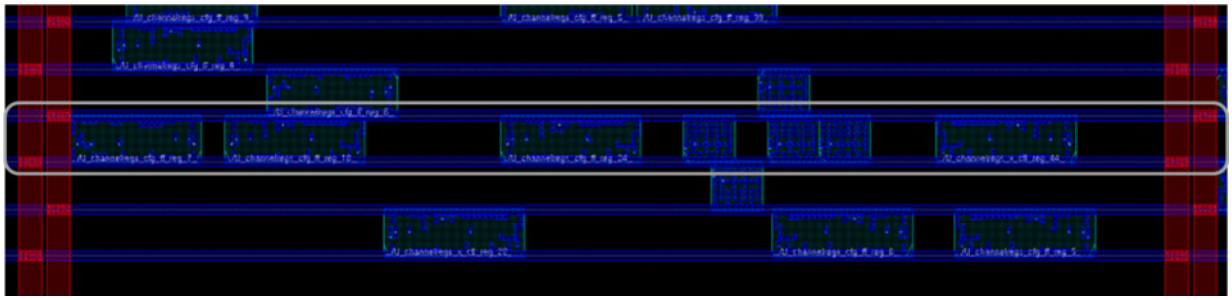


Figure 5.3. "Row section" example

The row section example is taken from a complex design and it is illustrated in Figure 5.7. In the marked row are placed four flip-flops and three different combinational gates – inverters, NANDs and NORs. Average current which the presented "row section" uses is 11.56µA. During the logic state switching in flip-flops and changes of the logic states in the combinational gates, maximal peak current is between 370µA and 710µA.

Related to the analysis provided in Chapter 3, the above described example confirms that SPS cell with driver/sensor transistor of 5µm channel width can drive optimally the mentioned "row section". It is important to notice that in the mentioned example the distance between two stripe pairs is 200µm, what represents the worst case for the IHP 250 nm digital implementation.

In order to provide protection against the latchup effect, it is required to integrate SPS cell in the power network. There are two important steps which are required for the SPS implementation:

1) SPS cell placement should be provided under the power stripes in order to enable that the "row section" power supply is not physically connected to the power stripes. "Row section" should be supplied by SPS cell.

2) Design of the power network controller (PNC), which is used to control all SPS cells in the fault-tolerant system.

In Figure 5.8 is presented a DMR fault-tolerant layout with latchup protection. The represented layout view is based on the layout view shown in Figure 5.6. Power supply distribution is provided separately for redundant components. In Figure 5.8 is shown that row supply is "broken" in neighbor SPS cell. SPS cell 1a and SPS cell 2a are electrically independent but because of the design rules it is important that wires are not floating. Therefore, row power supply provided by SPS cell 1a is broken "internally" in SPS cell 2a. The same approach is used for the redundant power supply, provided by SPS cell 1b and SPS cell2b. For the future releases it is planned to provide that each row is supplied from both sides. This approach requires more routing resources and more area for the implementation of the SPS cell control. On the other hand, it brings uniform row power distribution. This implies that more standard cells could be placed in one row section.

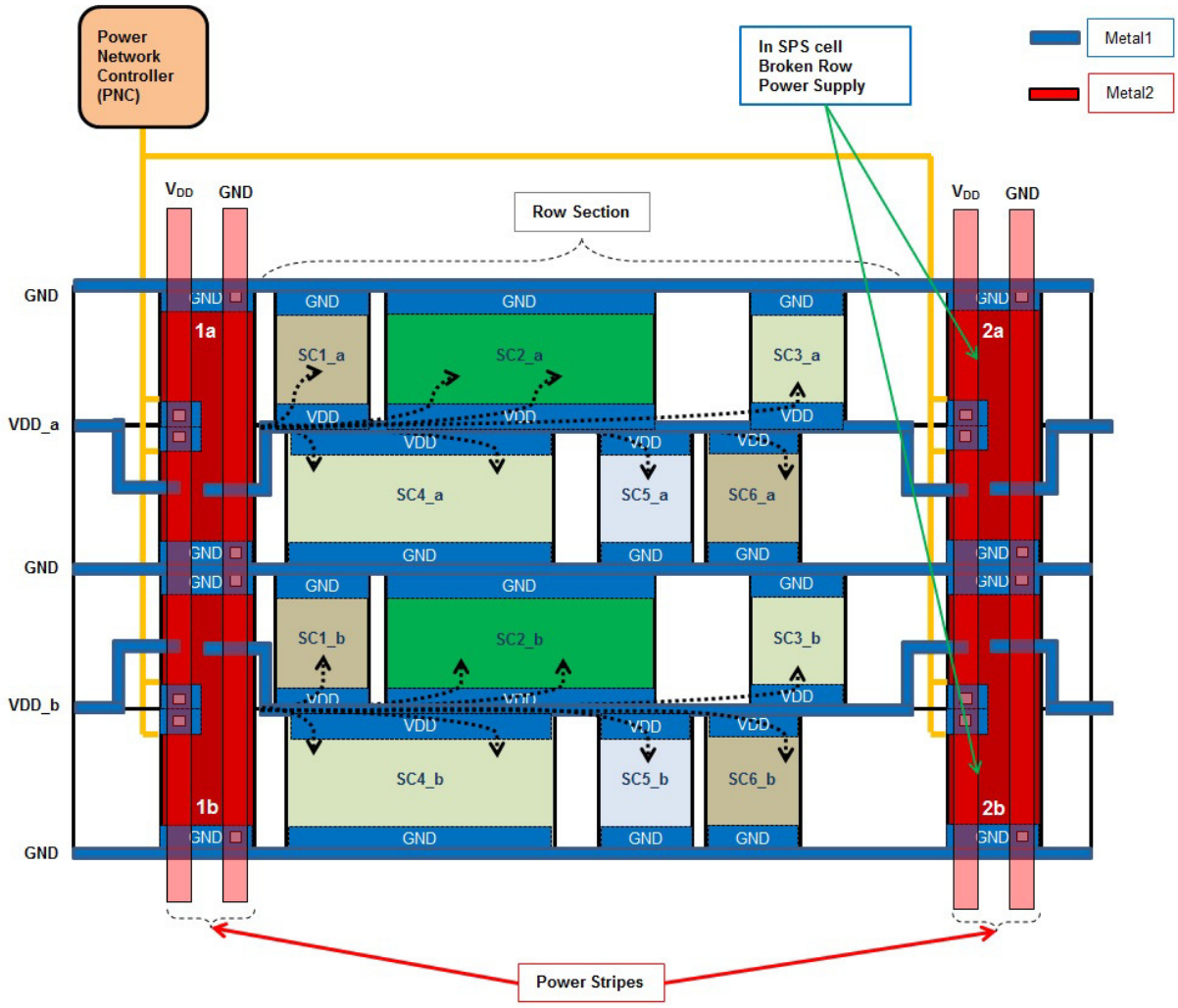


Figure 5.4. Layout view of DMR based circuit with integrated SPS cells

Comparing the standard power network with modified power network, it is not hard to notice the most important difference – one SPS cell provides power supplies for all redundancy levels and it is localized on four "row sections". As it was mentioned before, the approach used for the implementation is based on the DMR. Therefore, the redundancy level is two and one SPS cell covers two redundant power domains. This enables protection of "row section" pair against latchup effect. For example, if latchup effect occurs in the standard cell "SC5_a", the SPS cell 1a detects a higher current intensity than usual and switches off the complete "row section" pair. In the same time, the SPS cell1a informs the PNC that latchup has been detected. In the example presented in Figure 5.8 during the latchup protection phase, logic states of the standard cells in the disconnected "row sections" are lost. This induce that all other standard cells, which are connected to the cells under protection, have on their inputs low logic states. A digital system which includes just latchup protection technique cannot provide correct functionality during latchup protection phase. Therefore, it is necessary to use the specially designed redundant circuits which support the latchup protection (Chapter 3).

Power network controller (PNC) is the most important component, related to the power network. It consists of main programmable counter and status/control unit. Programmable counter is used for

defining a period of latchup protection. In order to provide usage of the presented design methodology with different technologies, the latchup protection period is possible to set from outside. Status/control unit is used for communication between the PNC and all SPS cells in digital circuit. It communicates with all SPS cells in the digital circuit. More details are provided in Chapter 3.3.

5.2.2. SPS cell

Figure 5.2 presents the layout views of SPS cell (circuit) type 1 and type 2. These circuits are used only for functional tests and characterization of SPS structures. This is done to get the related information about maximal current flows, quickness of sensing and switching-off the controlled power supply as well as the other important parameters used during characterization process. Figure 5.2.a and Figure 5.2.b show the layout views of SPS cell type 1 and SPS cell type 2, respectively.

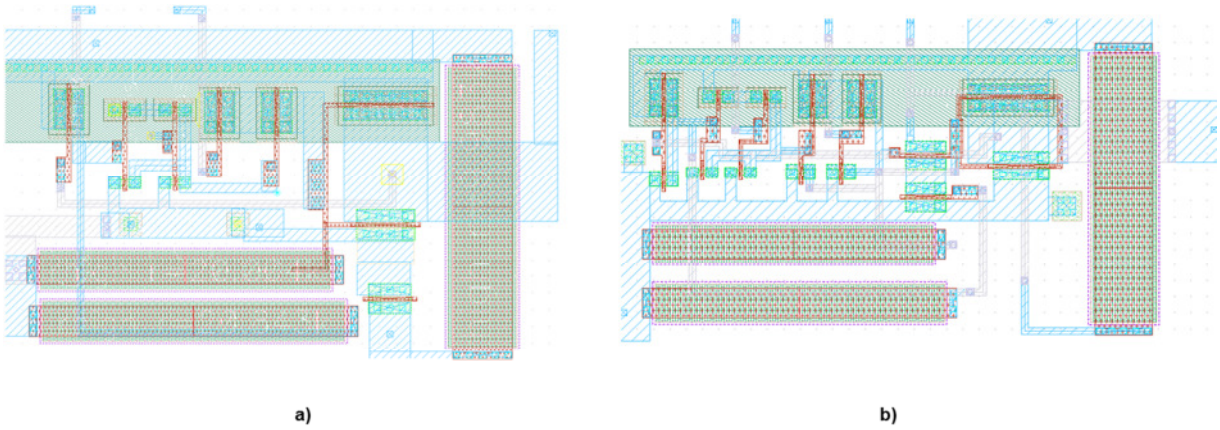


Figure 5.5. SPS cell layout views: a) SPS cell type 1 and b) SPS cell type 2

Based on the SPS circuits descriptions, provided in Chapter 3.1, a careful reader can notice that both SPS circuits include resistors in range of ten kilo-ohms. In order to provide required resistance, the implemented resistors occupy huge silicon area. This is a big disadvantage. Mentioned resistors are used as pull-up or pull-down resistors. Therefore, for usage in automated design flow redesign is required.

5.2.3. DMR Redundant Circuits with Latchup Protection

Following figures are presenting shift register and counter implemented using the presented fault-tolerant design technique. Layout views are related to the block diagrams presented in Figures 5.6 and 5.7. The implementation is provided for DMR circuits because of developed and tested circuits as well as easier presentation. The same analysis can be provided for TMR circuits with latchup protection.

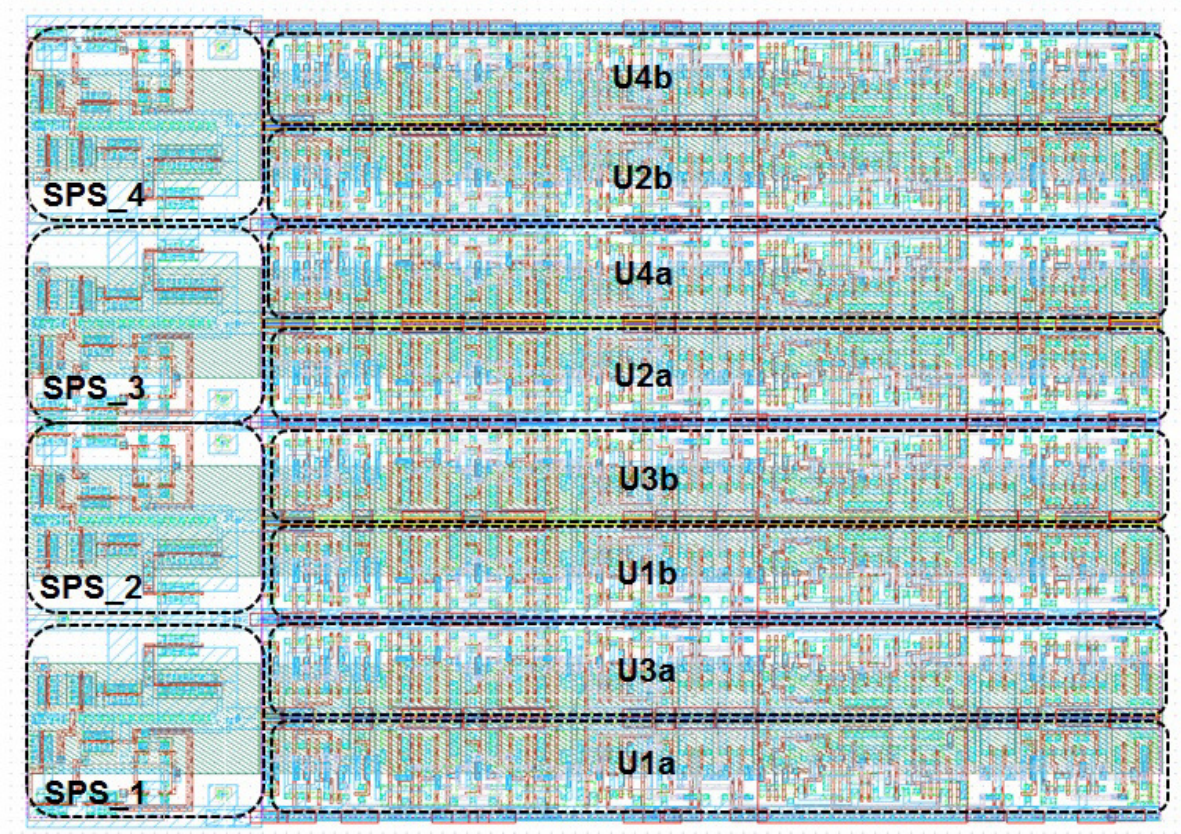


Figure 5.6. 4-bit shift register layout view - DMR approach with integrated SPS cells

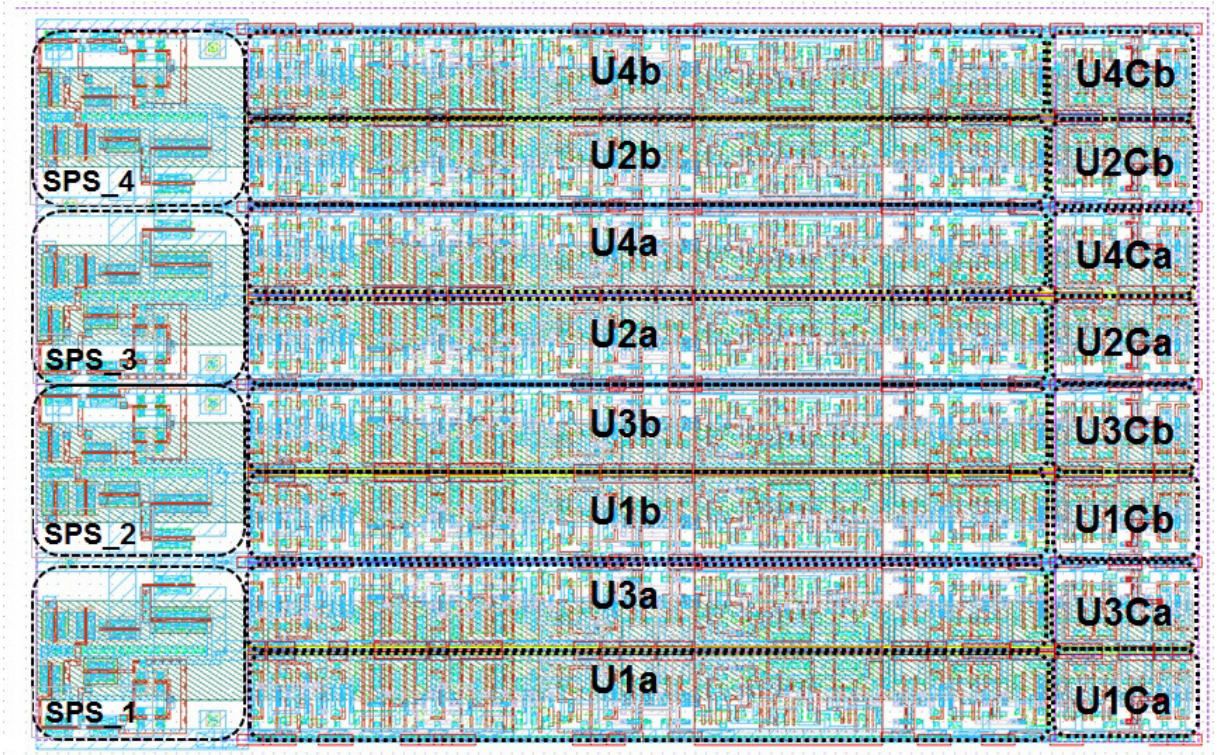


Figure 5.7. Counter layout view – DMR approach with implemented SPS cells

5.3. Results and Analysis

This section provides implementation results of different test circuits. The implementation results are related to:

- Required power consumption,
- Occupied silicon area
- Timing degradation.

Timing degradation implies a lower maximal operating frequency and therefore it should be reduced as much as possible.

According to different measurement methods the test circuits used for analysis are separated in three groups:

1) Redundant circuits.
2) SPS cells.
3) Power network controller.

End of section provides the measurement results and discussion of above mentioned test circuits.

5.3.1. Power Network Controller

Power network controller analyze is provided for SPS cell implementation in TMR and DMR redundant digital circuits. Implementation characteristics of power network controller have shown that the hardware control of one DMR SPS switch occupies area of 655.2 µm² with power consumption of 86 µW. For the TMR SPS switch, required PNC area per switch is 982.8 µm² with power consumption of 111.8 µW. Related to the number of required SPS cells it is possible to get estimated area and power consumption for PNC controller.

5.3.2. Redundant circuits analysis

Several circuits are implemented in order to get a view on the impact which redundancy has on the implementation characteristics (area, power and timing). The most important result of this discussion is related to the redundant circuits with latchup protection. In Table 5.4 are presented the implementation characteristics for the redundant circuits with and without latchup protection as well as the implementation characteristics of the non-redundant circuit (standard one). Redundant circuits which include the latchup protection contain in the name “LUP” suffix. All circuits are implemented using a standard IHP 250nm CMOS process [IHP13]. From Table 5.4 it is possible to notice that the higher reliability level bring the degradation which is related to the power consumption, occupied silicon area and timing. On the other hand, the failure-free probability is significantly increased. It is important to note that DMR circuit without latchup protection, used in this analysis is based on the work provided by [SCH09]. More details on this DMR topology are provided in Section 2.4.

Table 5.4. Redundant circuits implementation characteristics

Redundant Circuit	Area [µm^2]	Power [mW]	Timing Degradation [ps]
Flip-flop	155.23	6.881e-03	0
TMR	684.43	43.919e-03	340.34
DMR	691.48	17.450e-03	420.81
TMR LUP FF *	1563.23	155.719e-03	465.12
DMR LUP FF *	1303.94	128.213e-03	879.54

*) In area and power estimations are included implementation characteristics for power network controller (PNC) related to the latchup protection control

If we compare the TMR circuit with single flip-flop implementation, it is possible to notice that occupied silicon area is 4.4 times higher than occupied silicon area of single flip-flop. On the other hand, the DMR circuit requires almost the same area as TMR circuit. This interesting result has background in the standard cells used for redundant circuit implementation. TMR circuit is implemented using three flip-flops and three voters. Based on the work provided in [SCH09], DMR circuit is implemented using two flip-flops, two voters, two multiplexers and some buffers. Related to the power consumption, the DMR circuit has shown better results than the TMR circuit. Reason for this is lower number of flip-flops, which are implemented in DMR circuit. Flip-flops are components which are switching in every clock cycle and this implies higher power consumption. Difference in timing degradation between TMR and DMR circuits is similar.

Analysis of implementation characteristics showed that if a design doesn't need the latchup protection, it is possible to notice that TMR circuit has more advantages than DMR circuit with multiplexer in the feed-back line. Related to the fault-free probability (Section 4.3), comparison between TMR and DMR circuits, has shown that TMR circuit provides better results than DMR circuit.

If latchup protection is integrated in above mentioned redundant circuits, the implementation characteristics are changed. As it was described in Section 3.2, the redundant circuits which support the latchup protection require extra hardware. This hardware extension affects the implementation characteristics and therefore the area, power and timing characteristics degrade. Second degradation source is hardware used for the SPS cell control – power network controller (PNC). Compared to the standard TMR approach, the TMR circuit with latchup protection requires 2.28 times more area, 3.54 times higher power consumption with small timing degradation. On the other hand, the TMR circuit with latchup protection provides very high protection level against transient effects, upset effects and latchup effects.

DMR circuit with latchup protection also required extra hardware. Same as for the TMR with latchup support, extra hardware in DMR circuit is required for latchup protection and control of SPS cells. Compared to the DMR circuit, proposed by [SCH09], the DMR circuit with latchup support requires 1.88 times more area, 7.34 times increase of power consumption and two times worse timing. Timing degradation is higher because of the "voting-loop" circuit (Section 3.2.2). In any case, the timing degradation is still lower than 1ns.

On the end it is interesting to compare two redundant circuits, where both of them integrate the latchup protection. Based on the implementation characteristics, provided in Table 5.4, it is possible to notice that DMR circuit with latchup support has lower power and area overhead than the TMR circuit with latchup protection. Only significant degradation is related to the timing parameters of circuit. Based on the results provided in Section 4.3, the fault-free probability of TMR circuit with latchup protection decreases compared to the standard one. On the other hand, the DMR circuit with latchup protection has better fault-free probability than DMR circuit proposed by [SCH09].

Based on the discussion, provided in Section 4.3, the fault-free probability of TMR LUP circuit is in any case higher than the fault-free probability of DMR LUP circuit. Therefore, dependent on the design requirements (power, area, fault protection), can be decided which redundant approach should be used. DMR LUP approach represents a trade-off between protection level and very important hardware overhead. On the other hand, the TMR LUP approach provides very high protection against single event effects, with high power and area overhead as big disadvantage.

5.3.3. SPS cells analysis

The analysis of SPS cells is based on power consumption and timing characteristics. In previous chapters were presented two different types of protection circuits (cells) (SPS circuit 1 and SPS circuit 2) and their main disadvantages. Current section describes which type of SPS circuit should be redesigned according to the requirements for the automated placement and routing. As it was mentioned before (Section 5.2.2), the redesigned version of SPS switch excludes resistors in order to decrease the occupied silicon area.

Power consumption analysis is based on simulations within different functional modes:

- NM – Normal mode (without latchup effects),
- PM – protection mode (with latchup occurrence),
- PO – power-off phase (SPS cell deactivated).

It is important to notice that the current intensities of SPS circuit type 1 are higher than current intensities of SPS circuit type 2. This effect is related to the PMOS transistor characteristics. Compared to NMOS transistor the leakage current of PMOS transistor is much higher. It is easy to notice that in SPS circuit type 1, the transistor P_6 is connected through resistor R_0 directly to the ground (Figure 5.9 – red dotted rectangle). The same is for PMOS transistors P_3 and P_4, which are connected to the ground over resistor R_1. In the SPS circuit type 2 all PMOS transistors have related NMOS transistors, which are reducing the average current and therefore power consumption. Table 5.5 represents the implementation characteristics of both SPS circuit types. All results, presented in Table 5.5 are based on the simulation.

Table 5.5. Comparison of the SPS circuits implementation characteristics

Circuit / Parameter	SPS1	SPS2
Occupied Area [µm^2]	667.832	756.807
NM Current (Normal Mode) [pA]	38.531	34.358
PM Current (Protection Mode) [µA]	464.981	454.662
PO Current (Power-Off) [pA]	37.131	35.931

Based on the functional analysis of two presented SPS circuit types it is possible to notice a difference how the signals are generated. In SPS circuit type 2 the “TSTART” signal is generated slower than it in SPS circuit type 1. Main reason why “TSTART” signal is generated slower in SPS circuit type 2 is higher complexity of circuit. Therefore, for faster communication with power network controller is recommended to use SPS circuit type 1.

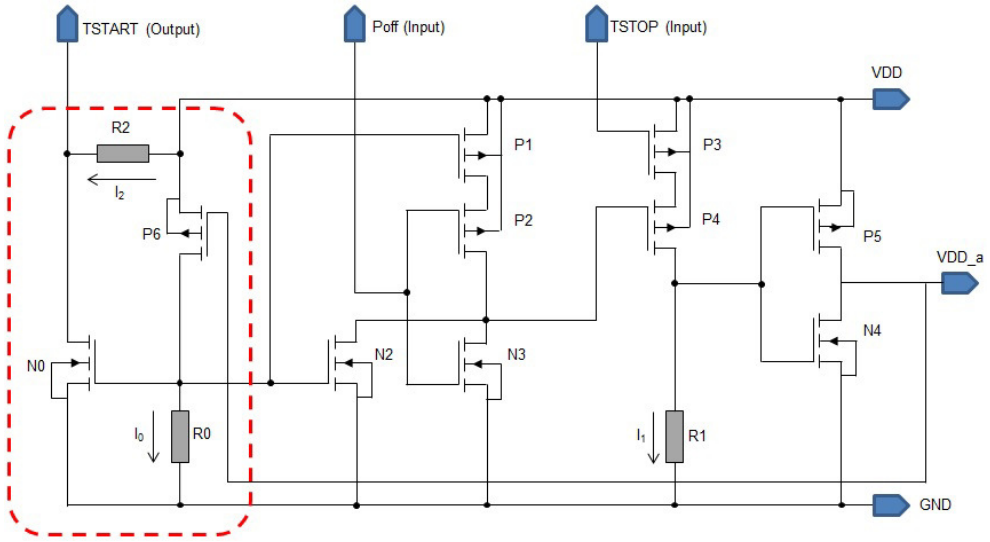


Figure 5.8. Source of higher power consumption (red rectangle) – SPS switch type 1

On the other hand, the SPS circuit type 2 provides continuous protection between SPS cell and power network controller. Therefore, in case that latchup effect is active after protection phase is finished, the SPS circuit type 2 provides continuous information to power network controller that in the controlled digital section a latchup effect is present. This option is advantage and provides simple power network controllers. In Table 5.6 are summarized the advantages and disadvantages of SPS circuit type 1 and SPS circuit type 2.

Table 5.6. Comparison table of the SPS circuits

Circuit Parameter	SPS type 1	SPS type 2
Fast Communication With Power Network Controller	+	-
Support for Permanent Latchup Detection	-	+
Circuit Complexity	+	-
Power Consumption	-	+
Occupied Area	+	-

Regarding the parameter comparison, for further implementation and design process automation is used SPS circuit type 1. Main reasons for this decision are circuit complexity and occupied silicon area.

In order to provide a power cell which can be integrated in a digital design using the standard design tools, three important steps are performed during redesign:

- Resistor replacement with corresponding NMOS or PMOS transistors
- Redesign of “TSTART” interface
- Layout design to support four-row approach

The schematic of redesigned SPS circuit type 1 is illustrated in Figure.

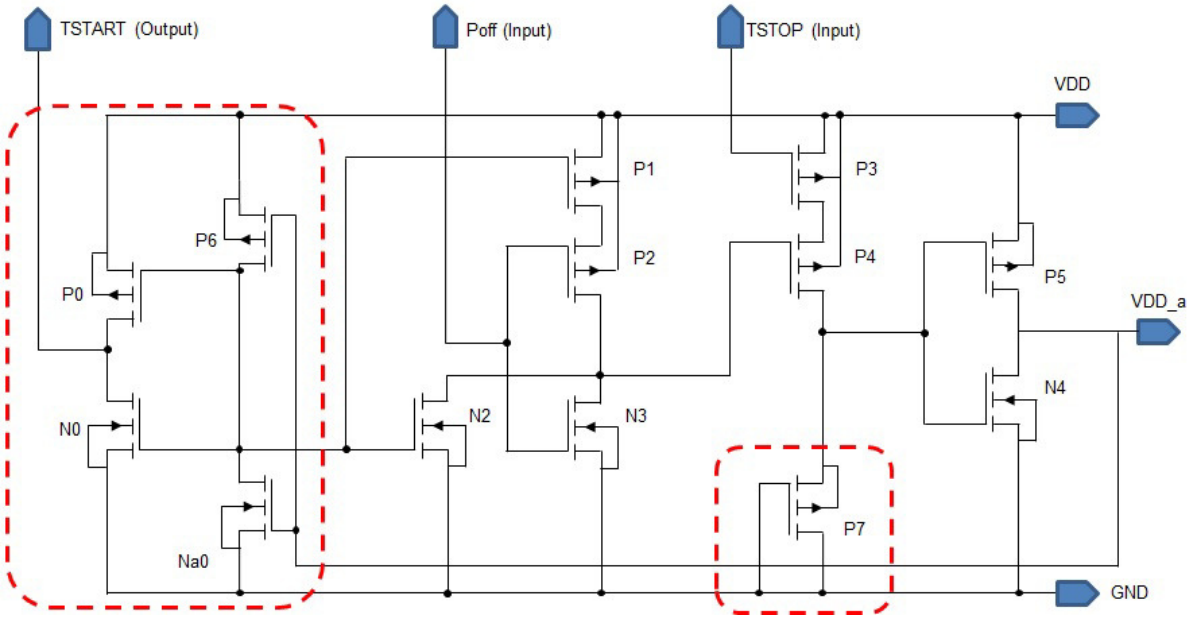


Figure 5.9. Redesigned SPS circuit type 1

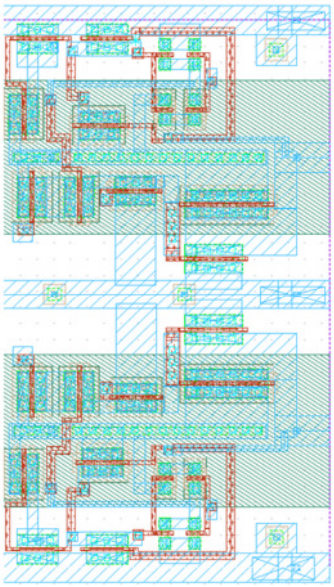


Figure 5.10. SPS circuit prepared for automated fault-tolerant design (four-row approach)

The layout view of redesigned SPS cell used for automated fault-tolerant design and based on the DMR approach is presented in Figure 5.10. Modified SPS circuit version is based on SPS circuit type 1. More details on implementation characteristics of redesigned SPS cell and scripts for automated placement and integration in a digital system are provided in Appendix E.

5.3.4. Measurements

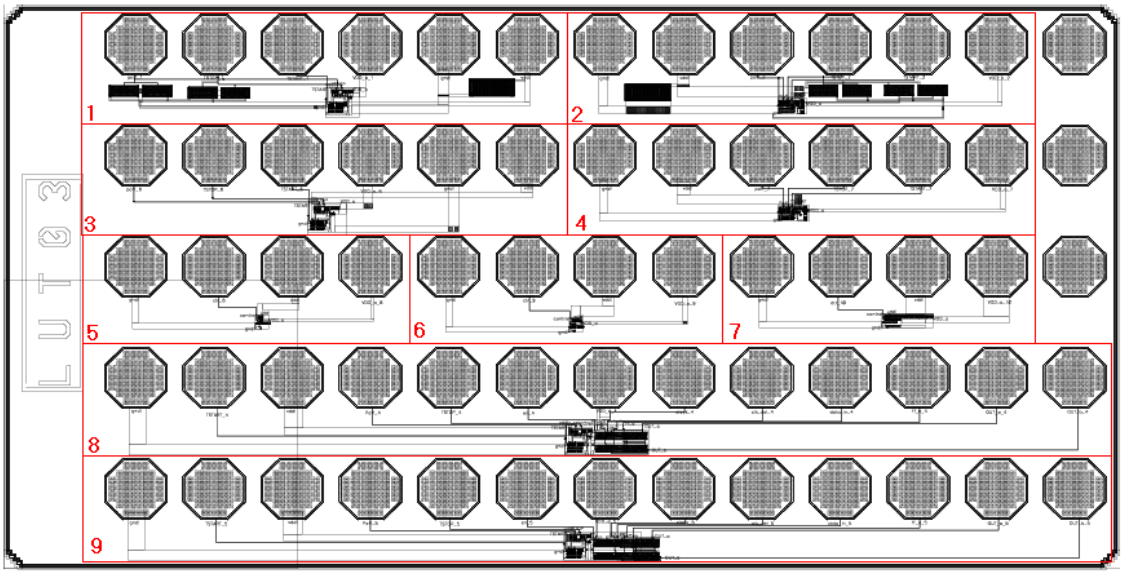
Three groups of test circuits are produced in standard IHP 250 nm process [IHP13], in order to prove the functionality of SPS cells. First group of test circuits is used for main functional tests. They are also used for timing measurements in moment when the latchup effect occurs. Second group of test circuits is used for measuring the burn-off time of the sensor/driver transistor. Burn-off time is measured in the latchup (short-circuit) mode. Transistors are categorized in three different groups according to the width-length ratio: the lowest (20.83), medium (41.67) and highest (208.33).

Third group of test circuits is used for the functional analysis of SPS cell which is integrated in a small DMR digital system. This group of test circuits provides the most important results because it integrates all developed circuits together (latchup protection circuit and sequential elements with latchup protection). Chip name LUT-03 indicates latchup test chip, 3th version. Control system for automated measurements is based on the microcontroller system and signal generators.

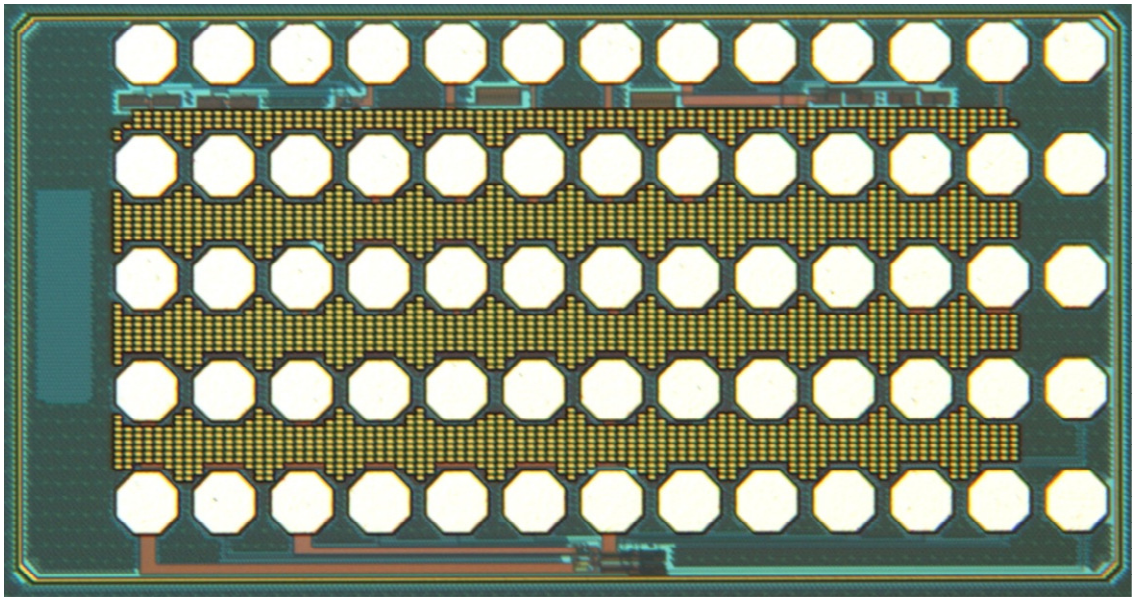
The radiation tests of SPS cells were performed at the Radiation Effects Facility at the Cyclotron Institute located on the campus of Texas A&M University – TAMU – in College Station, Texas (USA) [TAM13]. Tests were based on SEU/SET tests with particle energies up to 50 MeVcm^2/mg and SEL tests in different temperatures and ion energies up to 74.8 MeVcm^2/mg. Ion cocktail used during radiation measurement was composed of different radiation sources: ^{4}Ne, ^{40}Ar, ^{63}Cu, ^{84}Kr, ^{141}Pr, ^{181}Ta, and ^{197}Au. Effective linear energy transfer (LET) was in range 2.6 MeVcm^2/mg up to 82.8 MeVcm^2/mg.

The SPS cell operated correctly during the irradiation tests.

LUT-03 represents a package of nine different circuits prepared for tests which are described in the following text. Circuits are separated in the different groups, based on functionality and MOS transistor parameters. Pads of LUT-03 test circuits are provided in line across the chip so that probe with 13 (12) pins can connect more than one circuit with minimal usage of area. Figure 5.11 a) presents a layout view of LUT-03 test chip and Figure 5.12 b) illustrates chip-photo of LUT-03 test chip.



a)



b)

Figure 5.11. a) Layout view of LUT-03 test chip; b) Chip-photo of LUT-03 test chip

In Table 5.7 are presented circuit details and corresponding circuit numbers, based on Figure 5.11 a). The sensor/driver transistor and the discharger transistor are in details explained in Chapter 4.3. In the following text are provided descriptions of the test circuits, implemented in the LUT-03 chip.

Table 5.7. Circuits and related parameters of the sensor/driver transistor and discharger

Circuit	PMOS sensor/driver transistor width	NMOS discharger width
1 – Switch type I	5 µm	3.44 µm
2 – Switch type II	5 µm	3.44 µm
3 – Switch type I	10 µm	5 µm
4 – Switch type II	10 µm	5 µm
5 – Maximal Current Test	5 µm	3.44 µm
6 – Maximal Current Test	2 x 5 µm (10 µm)	5 µm
7 – Maximal Current Test	10 x 5 µm (50 µm)	20 µm
8 – Switch Type 1 and DMR	5 µm	3.44 µm
9 – Switch Type 2 and DMR	5 µm	3.44 µm

Functional description of test circuits 1 and 3 is provided in Section 3.1.1. The description is based on SPS cell type 1. The difference between test circuit 1 and test circuit 3 is just in the W/L ratio of the sensor/driver transistors and discharger transistors. Functional description is same for both of them. On the other hand, the test circuits 2 and 4 are already described in Section 3.1.2. Test circuits 2 and 4 are related to the SPS cell type 2. Difference between test circuit 2 and 4 is based on the W/L ratio of sensor/driver transistor. The discharger transistor dimensions are also different and values are shown in Table 5.7.

Test circuit 5, 6 and 7 are used for maximal current tests. The maximal current tests are based on longer time shorted-circuit. The measurements are related to the destruction of driver/sensor transistor integrated in SPS cell. Schematics of the test circuit 5, 6 and 7 are same but channel dimensions differ. The transistor dimensions are also presented in Table 5.7. The schematic of tests circuit used for burn-off tests is presented in Figure 5.12. Test circuit has four terminals required for test – power pins, control and output. Test board is specially designed for this type of tests and in the moment when the test transistor is destroyed, the test procedure is automatically stopped.

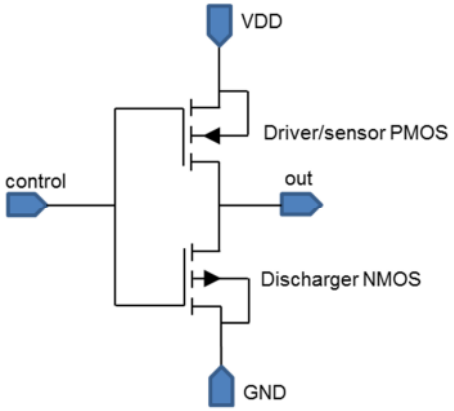


Figure 5.12. Schematic of the test circuits 5, 6 and 7

Test circuits 8 and 9 are used for functional tests, where SPS cell is connected together with simple DMR digital circuit. These tests provide information about power switch cell behave, when it is integrated with standard cells. The most important measurement result of the test circuits 8 and 9 is related to the DMR circuit state changes during the power-off and power-on phases. Block diagram of

implemented test circuits 8 and 9 is shown in Figure 5.13. Difference between test circuit 8 and test circuit 9 is based on the SPS cell type – test circuit 8 implements SPS cell type 1 and test circuit 9 implements SPS cell type 2.

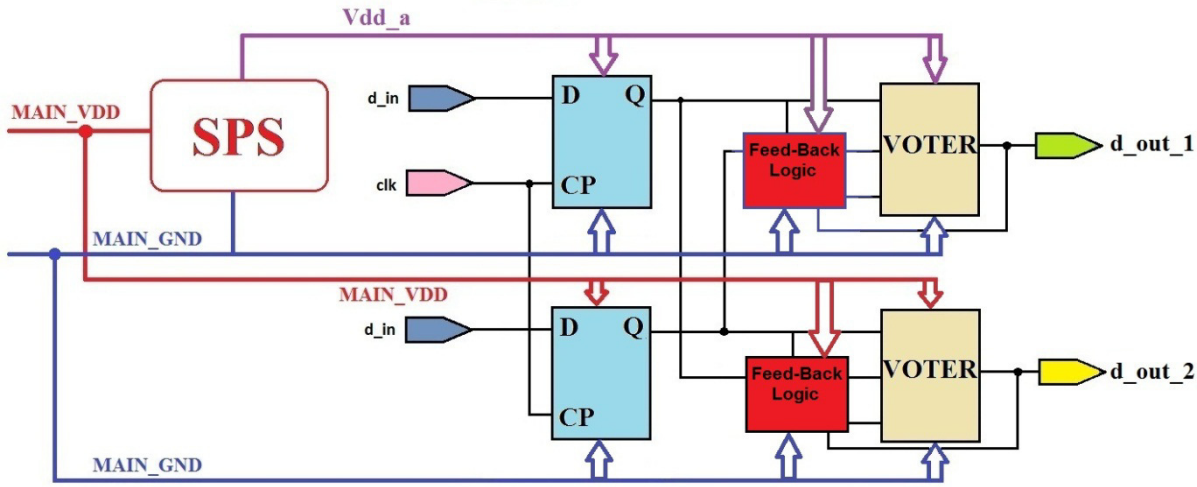


Figure 5.13. Block diagram of test circuits 8 and 9

The power supply "Vdd_a" is controlled power supply, used for latchup tests. Careful reader can notice that the digital circuit presented in Figure 5.13 implements just one SPS circuit. Involving a second SPS circuit, as it should be in final design, requires more complex test environment. Therefore, it is enough to test one part of the DMR circuit with protection cell, where other part is supplied directly from the DC source.

Measurement equipment foresaw for the mentioned test circuits is separated in two groups:

a) Stimulus generator
b) Data acquisition

Stimulus generator is a circuit based on Microcontroller, Thyristor and related connections for stimulus and measured data. Microcontroller (Microchip®) PIC16F177A is used to control the latchup occurrence moment. In the same time, from other terminal the microcontroller sends a signal to the oscilloscope, which synchronizes the data recording. Board is designed to support extensions of the microcontroller usage – personal computer (PC) controlled application for circuit control, extra timers, external memory for data storage, etc. Basic block scheme of the measurement equipment (test system) is presented in Figure 5.14.

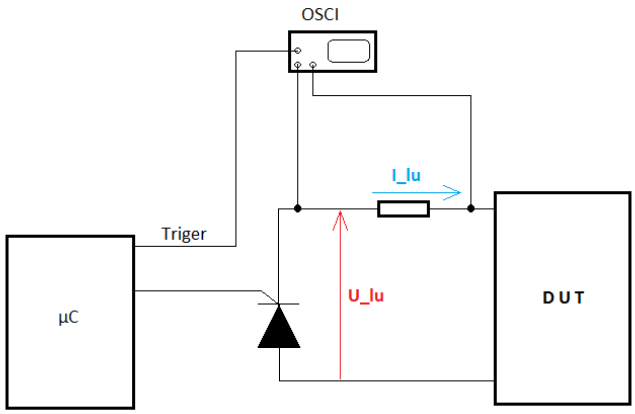


Figure 5.14. Block diagram of the measurement equipment (test system)

Data acquisition part consists of Oscilloscope (OSCI), PC, microcontroller (µC), voltmeters and ammeters. Oscilloscope used for measurements has four channels and an external triggering option. All cables and connections between equipment and test board are coaxial based, due to maximal possible measurement accuracy. PC is connected to the test board through RS232 communication protocol. As µC is used as stimulus generator, there is possibility to use an extended option, based on the data processing and proportional A/D conversion. Therefore, the measurement and data processing can be done in µC and transferred to PC or another mass memory device. In Appendix E is provided C code used for microcontroller automated measurements.

The test board has its own power supply unit and provides some of the stimulus signals for test chip. On the other hand, the test board provides required trigger signal for oscilloscope and other information related to the measurement as temperature, current status of the terminals, etc. The schematic of power supply unit is provided in Figure 5.15. View of test equipment is provided in Figure 5.16. Schematic of complete board is presented in Appendix E.

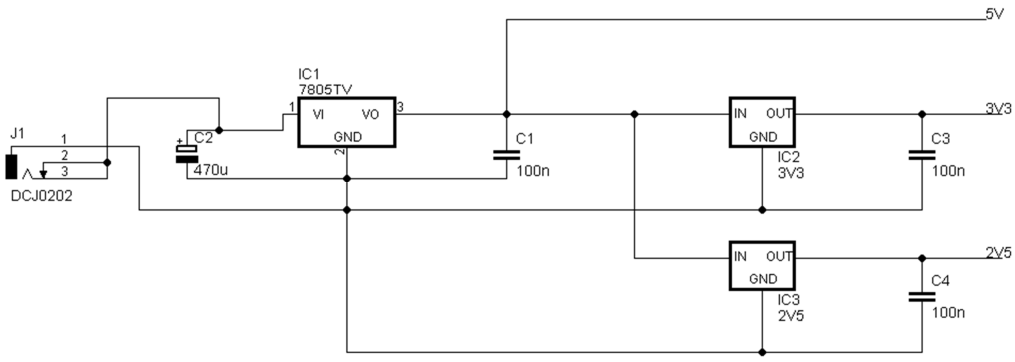


Figure 5.15. Power supply unit of the test board

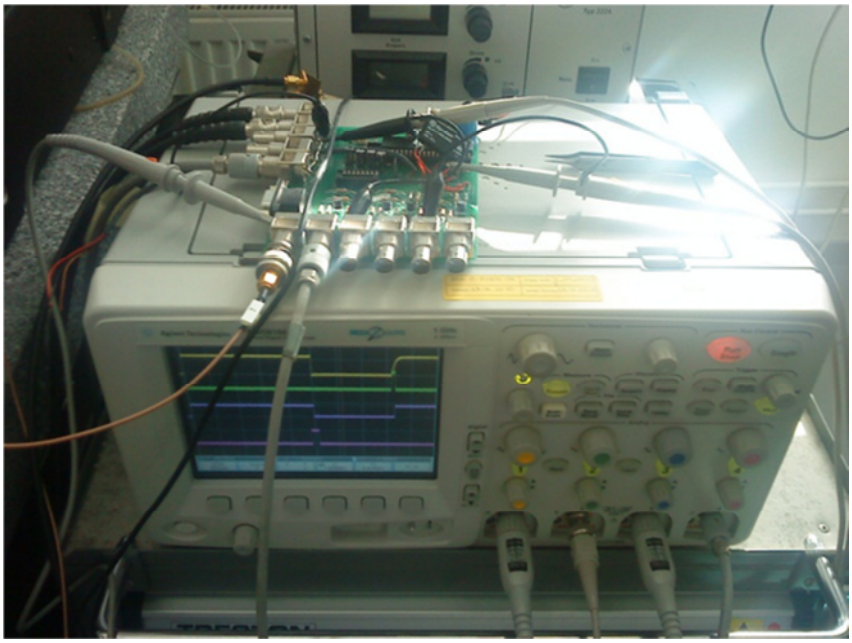


Figure 5.16. Measurement equipment for latchup protection measurements

Timing measurements are done in order to characterize the SPS circuit as a standard power switch cell for usage in the automated design flow. In Figure 5.17 is presented a waveform diagram of the mentioned signals and required timings. The strobe signal, in Figure 5.17, represents the latchup activation signal. Table 5.8 presents the simulated and measured signal timings. The timing measurements are done based on the 50% voltage level during the signal transitions.

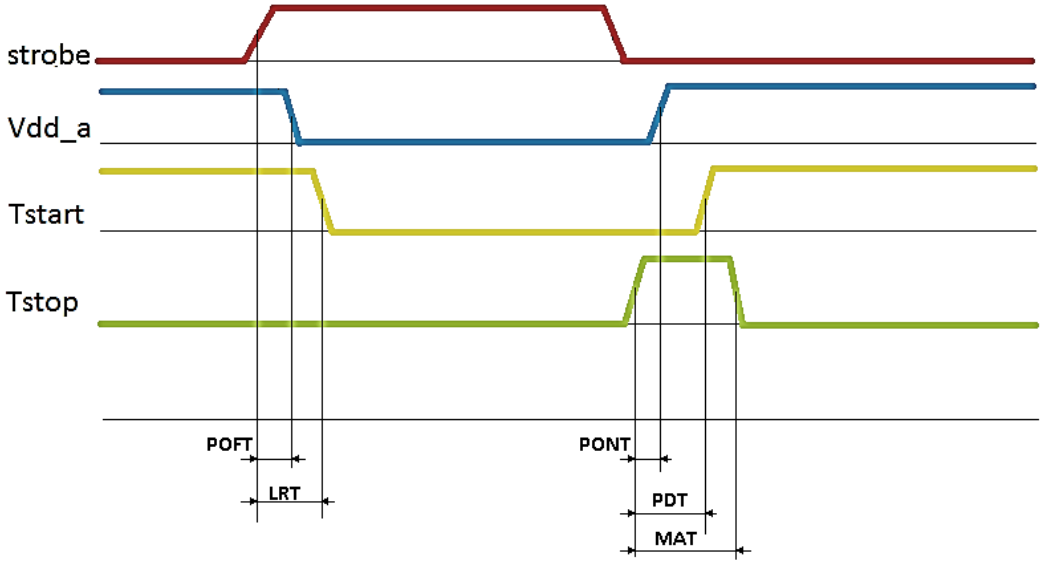


Figure 5.17. Timing diagram of SPS circuit

The time required to set output pin “Vdd_a” in the low logic level is defined as a power off time (POFT in Figure 5.17). Time required for activation of “Tstop” signal, after latchup, is defined as a latchup recognition time (LRT in Figure 5.17).

In order to “wake-up” the SEL power switch circuit (SPS) from the latchup protection mode, it is required to provide an impulse to the “Tstop” terminal. The minimal length of the “Tstop” pulse is defined as a minimal activation time (MAT in Figure 5.17). The “Tstop” pulse brings back the controlled power supply “Vdd_a” into active state. The time required for this process is defined as a power on time (PONT in Figure 5.17). The “Tstart” pin is simultaneously set in high logic state, whereby latchup protection sequence is finished. Time required for this process is defined as a protection deactivating time (PDT in Figure 5.17).

Before the discussion of simulated and measured SPS timings, it is important to note that the measurements are done using standard equipment. This involves different parasitic effects and therefore some re-calculations are required. Measured values are normalized by correction factors for each pad on the test circuit due parasitic capacitances and resistances in cables and connections. Results of simulation and measurements are provided in Table 5.8.

Table 5.8. SPS signal timing (with corrected measured values)

	POFT	LRT	PONT	PDT	MAT
Simulated	55.19 [ps]	76 [ps]	487 [ps]	786.5 [ps]	700 [ps]
Measured	120 [ps]	440 [ps]	1.42 [ns]	1.71 [ns]	2 [ns]

It is important also to note that in case of permanent short circuit on the “Vdd_a” output pin, a SPS cell will automatically be in the protection mode.

Maximal current tests are based on longer time short-circuit and measurements when the output transistor of power switch cell will be destroyed. The burn-off test is done for the driver/sensor PMOS transistor, in order to prove that transistor will survive high current flow in the moment when latchup occurs. Figure 5.18 shows the simulation result of the voltage and current for the driver/sensor PMOS transistor in the moment when latchup occurs. Controlled voltage is “Vdd_a” and current pulse represents the current through the PMOS driver/sensor transistor.

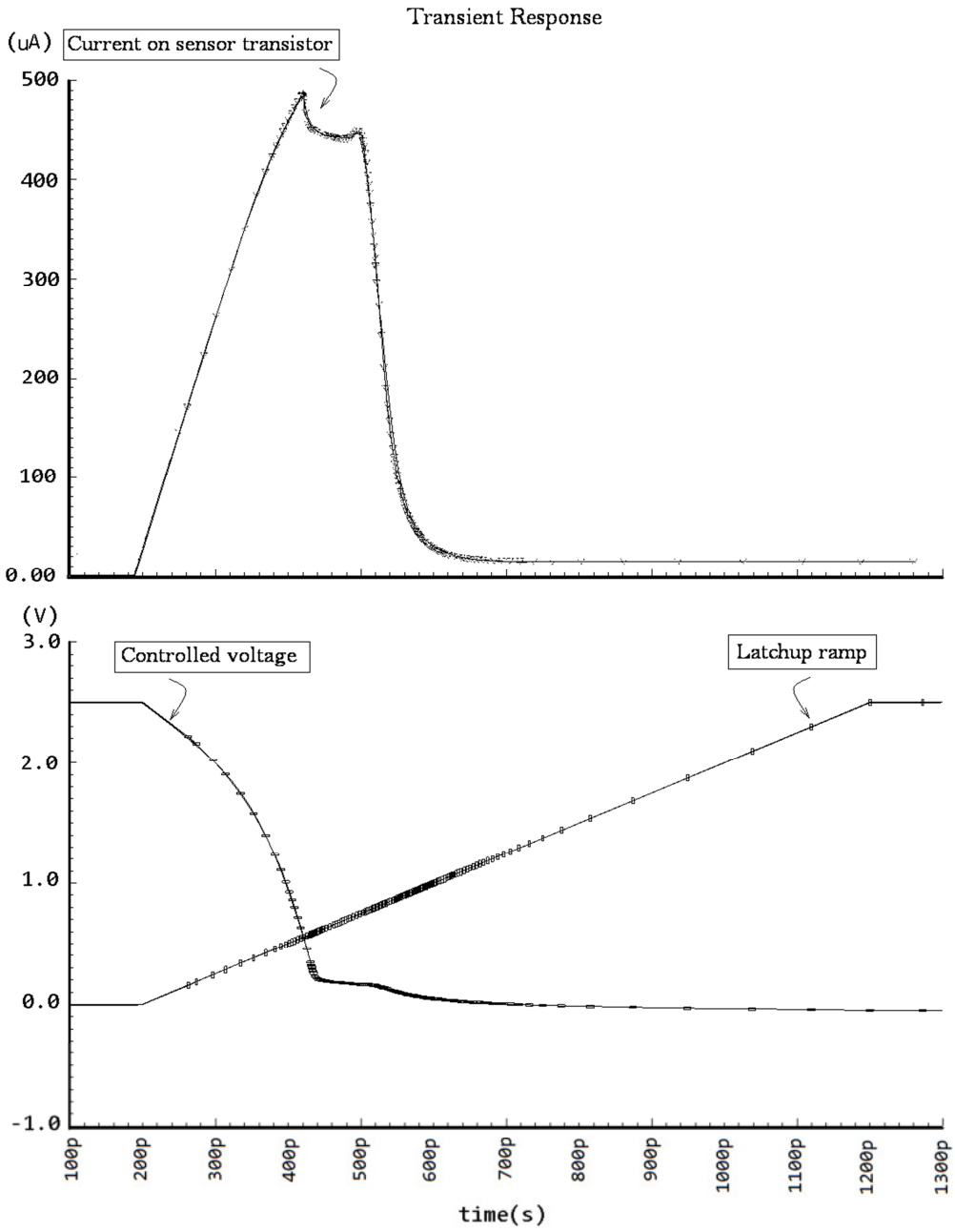


Figure 5.18. Simulation current and voltage diagrams after latchup occurrence

In Table 5.9 are presented burn-off timings of different test transistors, which can be used as driver/sensor transistor.

Table 5.9. Burn off timings (with corrected measured values)

W/L	20.83	41.67	208.33
$t_{burn\text{-}off}$ [ns]	20	30	38

The measured power consumption, of SPS itself, is about 500 µW in normal conditions and it goes up to 1.25 mW when stimulated latchup occurs. A simulated value of the SPS power consumption is 75 pW in normal conditions and 1.16 mW in the latchup mode. This power consumption difference,

between two mentioned modes, is due to the pull-down resistors implemented in circuits and parasitic effects in the measurement equipment.

Measured SPS cell signals are depicted in Figure 5.19. Latchup trigger signal is used for activation of the reed-relay used as short-circuit switch and for synchronization of measurement equipment. “Vdd_a”, “Tstop” and “Tstart” signals are in details described in the related chapters. During measurements the “Poff” signal was always in active logic state and therefore, it is not present in the following analysis.

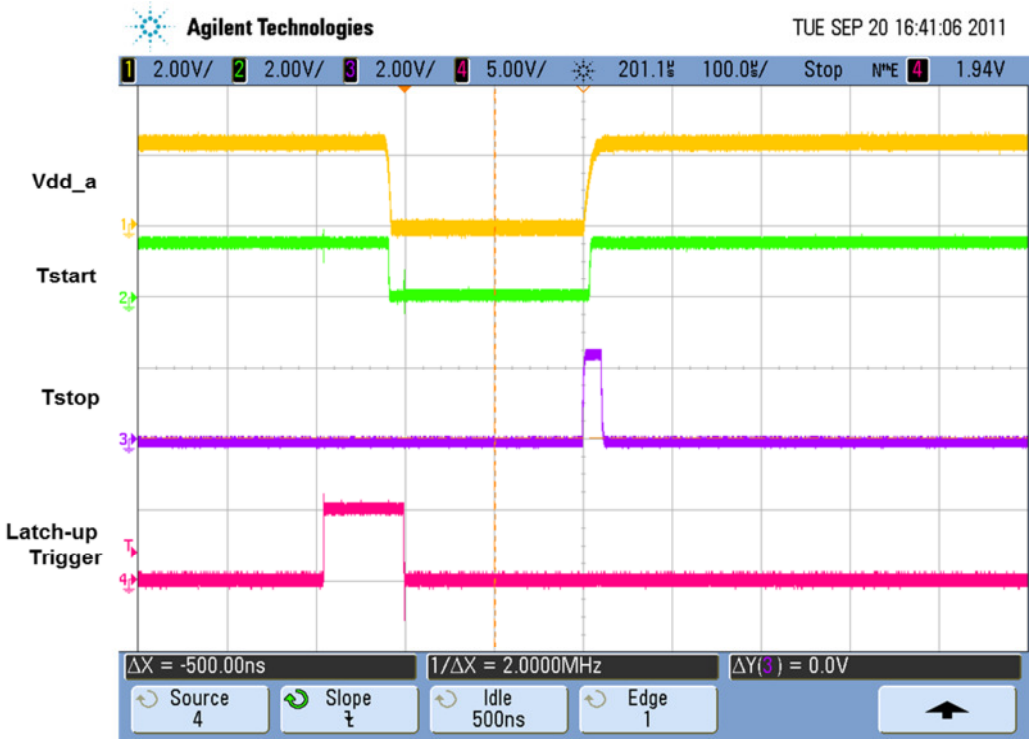


Figure 5.19. SPS signals and controlled voltage (Vdd_a) during latchup protection phase

Measurement results, based on DMR and SPS cell are provided in Figure 5.20. It is easy to notice that during the latchup protection phase, output of digital circuit, controlled by SPS cell is switched off. Other part of the DMR circuit operates without problems. After latchup protection phase is finished, the complete circuit continues to operate normally. The measurement results are copied directly from oscilloscope and shown in Figure 5.20 [AGI08].

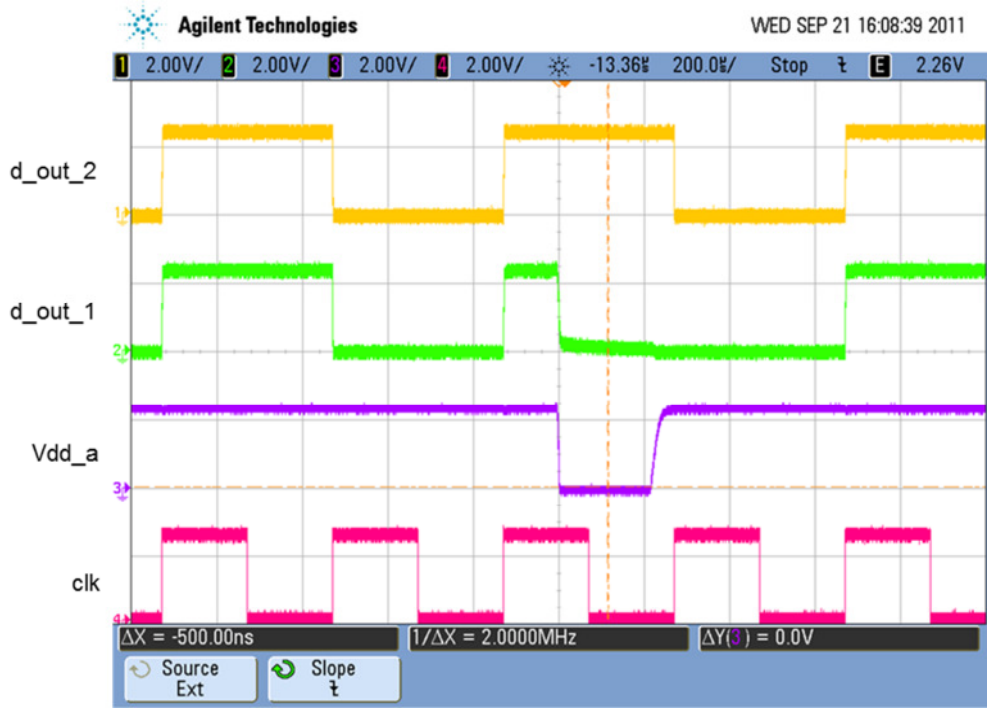


Figure 5.20. Measurements of the SPS cell included in digital circuit

From previous figures, recorded during measurements on silicon it is clear that the usage of a redundant digital circuit (DMR) provides the circuit states recovering after short-circuit switch is relaxed (opened). From Figure 5.20 it is possible to see that short-circuit protection is active as long as it is required.

6. A Case Study: Middleware Switch Processor

6.1. Introduction

During designing a new satellite system, it is usual to face with very complex problems. It is possible to divide them in groups and based on this to find the most optimal solution. The first problem is the long development time of an avionics system. Parallel with it are present the huge costs because of long time required for defining the new interface specifications. The development of very expensive board computers represents the second very important problem. All devices in a satellite communicate with each other through the board computer. Therefore, for every new satellite (or space related) system it is required to define a new device configuration and redesign the board computer. The way for solving these problems was the implementation of a SCAN network, [MON9a] [MON9b] [MON10] [PET11]. The central part of the whole SCAN system is the MW switch processor.

Spacecraft Area Network (SCAN) approach is based on a message distribution protocol defined by an interconnection link. The message distribution system is based on a publisher / subscriber model where each message has its own message topic identifier (TID). The switch controller is routing the received message to all subscribers which are related to the received TID. First ASIC implementation of MW switch processor was provided by IHP and DLR. This version implements two types of serial interfaces – S3P (Simple Synchronous Serial Protocol) and UART (Universal Asynchronous Receiver Transmitter). Publishers (sensors, computing nodes) provide messages, which are public under a given topic. Subscribers (memories, actuators, computing nodes), configured to receive the messages from the defined topic, can receive all published messages (Figure 6.1) [MON10] [SCH12].

This system, based on the topic share between providers and consumers, provides an interconnect service. Services will be published as topics, regardless of whether they are produced by software tasks or by hardware devices. Therefore, the SCAN network can attach and use COTS (Commercial off the Shelf) devices (with their own protocols) although they are not space verified. The network provides required interfaces and protocol converters. The COTS devices receive and send their own messages, and then the protocol converters translate them into our internal “universal language”. The network performs all required transformations in order to make the message transfer transparent. In this respect, the SCAN based system can integrate COTS components and provide a reliable architecture.

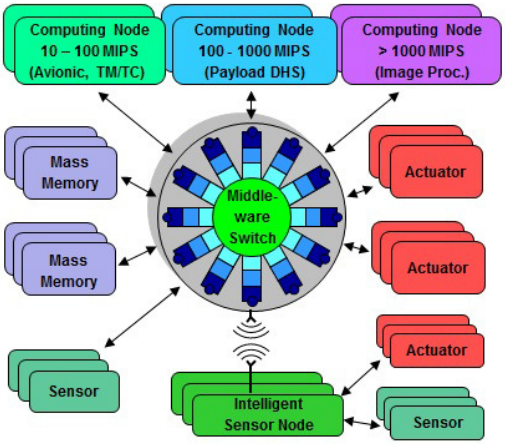


Figure 6.1. MW switch processor integrated in the satellite system

In Figure 6.1 is presented a SCAN network. As we can see, the heart of SCAN network is MW Switch processor, which performs all reliable actions.

The mentioned processor version showed very good testing results but the main disadvantage is fault-tolerance. In the mentioned system only the memory controller was implemented with EDAC (Hsiao coding scheme) in order to provide protection for the data between processor and external memory.

As the MW switch processor is the main component of SCAN system, it needs to fulfill three important architectural requirements.

The first requirement is high reliability - if MW processor is not functioning regarding failure, the complete SCAN system is not usable. In order to provide the reliable usage of a SCAN system in space environment, it is necessary to apply a fault-tolerant design concept. In the following text is described how middleware switch processor can be designed as fault-tolerant system using the presented fault-tolerant design methodology. The ASIC design concept, presented in thesis, provides tradeoff between production costs and high system reliability.

The second important architectural requirement is scalability – fast and uncomplicated changes of active transferring data port number. Regarding to the current data transfer standards in space industry [8], the MW switch processor can achieve the required data throughput only in target topologies (parallel MW switch architecture or combination of the pipelined-parallel architecture).

The third architectural requirement is flexibility – highly configurable ports for the communication link between different device types connected to the SCAN. It is important to note that more MW switch processors connected to the network also provide higher reliability and scalability of complete satellite system.

In the following sections are provided details of MW Switch architecture and short description about produced middleware switch but without applied fault-tolerant techniques. This is the first produced version, which is very important for the functional verification of SCAN approach (DLR). End of chapter provides details related to the development of fault-tolerant Middleware Switch processor. The thesis work has a goal to provide a fault-tolerant processor using the developed design technique. The fault-

tolerant Middleware Switch processor has reduced hardware compared to the non-FT Middleware Switch. Hardware in the fault-tolerant middleware switch version is reduced because of DMA controllers, which are owned by Synopsys without rights on hardware changes.

6.2. Middleware Switch Architecture

The non-fault-tolerant Middleware Switch processor has three main modular groups: communication ports, communication links and a central processing unit (LEON-2 CPU). Communication ports are connected via the switch matrix or DMA channels. The DMA channels are configured to transfer data between ports and main memory. The switch matrix may be used to forward high speed messages directly between ports using a multi cast protocol. The internal CPU may communicate with every port, the switch matrix and the DMA controller to monitor and control their configuration, data traffic and flow control.

The Processor reads the topic ID directly from the message and after the software processing, gives a corresponding instruction to the switch matrix. The primary ASIC implementation consists of 6 high speed Simple Synchronous Serial Ports (S3P), 8 UART ports, Leon 2 core processor and switch matrix (Figure 6.2). The additionally implemented functions are a debug unit, programmable timers, pulse generators and counters as well as built-in-self-test (BIST) features.

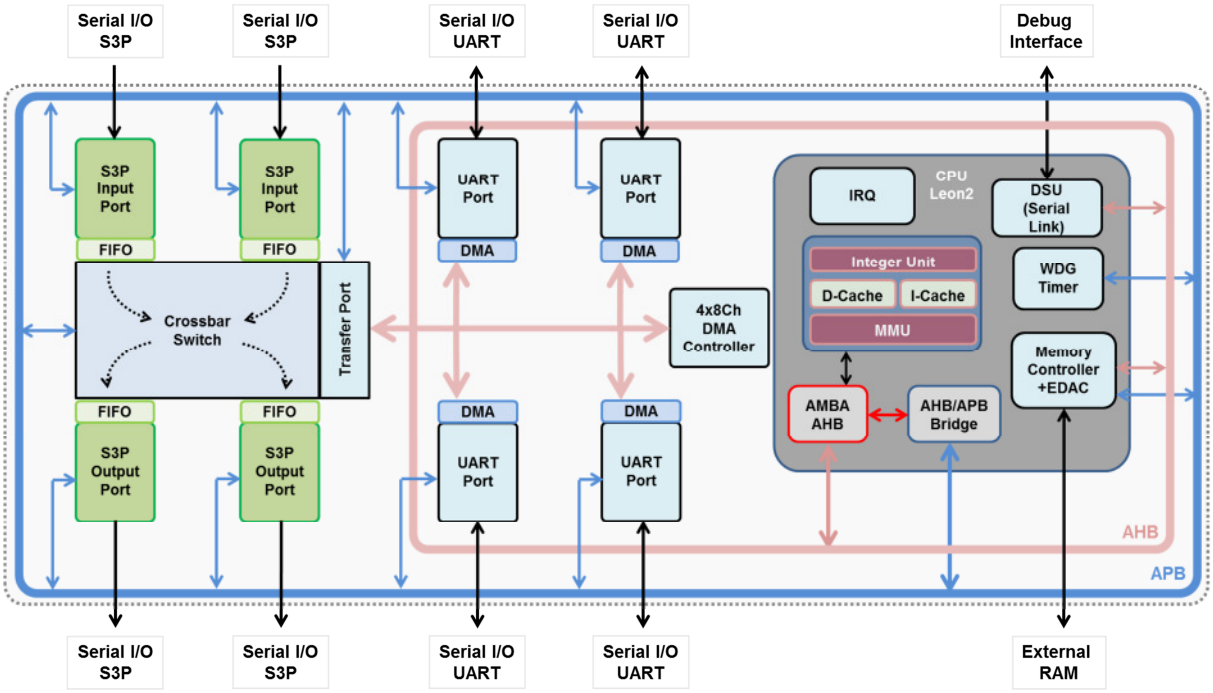


Figure 6.2. Basic concept of the Middleware Switch ASIC

In the architecture development process, a necessity to have two different systems for the port connections was determined. For S3P ports a switch matrix is used, which basically consists only of multiplexers. The received data are buffered in the FIFO buffers, which are implemented together with the ports. For UART ports we are using DMA controllers and an AHB processor bus to transfer data between the ports and external memory. All messages which are stored in the memory may be

manipulated by the CPU before their being transmitted to any destination device.

The Middleware Switch uses a uniform internal protocol to allow simple switchable connections between very heterogeneous devices. Therefore, all ports are using protocol converters to convert between the externally used protocols and the internal format. In order to allow different device types to be connected to the same port pin, the protocol converters must be configurable. The most important configurations are related to message framing, flow control, synchronization and header information (topic ID, message length). If any new protocol format is not yet implemented into the port hardware, the message must be transferred into the memory and the CPU can do the protocol conversion. Therefore the implemented concept is very flexible and even cascading several ASICs is possible.

Middleware Switch ASIC implementation uses the integrated hardware handshaking but also supports the software handshaking. Because of the different communication protocols, even the software handshaking must be recognized and handled by the port logic. This results in the devices being able to send and receive flow control commands via the Middleware Switch to avoid data losses and re-transmissions. Such commands (Stop, Continue) can be transmitted anytime within or between messages. This is very useful and necessary when devices communicate with each other at different speeds and when the internal buffers are running out of the memory space.

The CPU control of a S3P port is accomplished through configuration registers connected to the APB processor bus. The CPU can configure the switch matrix, initiate data transfer through the switch matrix, send flow control commands to the external device and start or stop data transfer from port to the external device. On the other hand, the S3P port can initiate the interrupts in different cases – FIFO is full, detected flow control sequence on input etc. The UART port is controlled by the CPU through APB processor bus, where the CPU can define the mode of the protocol convertor. The DMA controllers are set by the CPU in order to control the data traffic.

Additional implemented functions are: a debug unit, programmable timers, pulse generators, and counters scan chains and built-in-self-test (BIST) features. An EDAC (error detection and correction) block between the memory controller and an external memory device is also implemented into the MW switch. The error detection and correction (EDAC) module is used to protect the data written into the memory. The EDAC algorithm is based on Hsiao coding scheme and provides one bit error correction and two bits error detection. Registers, flip-flops and latches should be protected through TMR (Triple Modular Redundancy) or DMR (Double Modular Redundancy) which we will implement into a future Middleware Switch designs.

The presented ASIC implementation of a Middleware Switch uses Leon 2 IP core from Aeroflex Gaisler© (formerly Gaisler Research) as the processor unit; the DMA core from Synopsys© Design Ware library; and all other components (memories, FIFOs, EDAC, ports and all control logic) from IHP GmbH.

6.2.1. Functional Description

Functional description is based on the developed S3P communication protocol [MON10]. Detailed S3P port description is divided into two subsections – input port section and output port section. The input port section describes components related to data processing on the receiving side: the receiving serial to parallel convertor, the input S3P port status and configuration registers, FIFO write/read controller, the mechanism of the internal port handshaking functioning during data transfer and parallel to serial convertor, which prepares the data for the switch matrix transfer.

On the other hand, the output port section describes the components related to the data preparation for the transmission: serial to parallel convertor for the data received from the switch matrix, output FIFO read/write controller, programmable output clock generator and parallel to serial data convertor.

The S3P protocol or SSSP (Simple Synchronous Serial Protocol) is similar to the internal Middleware Switch data representation. It uses two signals **S3P clock** and **S3P data**. Reception (input S3P port) and transmission (output S3P port) are independent and it is possible to have different data rates (defined by related S3P clock frequencies). The output S3P port sends the most significant bit first. Description of the aforementioned components will be done following the way data are processed through the S3P port. Let's start from the receiver activation to see how S3P ports, integrated into the Middleware Switch, are processing the data.

Before providing the S3P ports functional description, it will be useful to mention the message framing and other commands used for this type of ports. Messages are framed by using begin of message (BOM) and end of message (EOM) commands. Commands consist of two bytes as they are defined in Table 6.1.

Figure 6.1. S3P protocol messages framing commands

Short	Hex.	Binary
BOM	0xFF02	1111 1111 0000 0010
EOM	0xFF03	1111 1111 0000 0011

0xFF is also used to trigger other commands that control the data transfer through S3P ports. All other commands, related to the S3P protocol data flow control, are presented in Table 6.2.

Figure 6.2. S3P protocol data flow commands

Short	Hex.	Binary
STOP	0xFF18	1111 1111 0001 1000
CONTINUE	0xFF01	1111 1111 0000 0001
FILLER	0xFF00	1111 1111 0000 0000
SYNC	0xFFFF	1111 1111 1111 1111

Input S3P Port

Data Receiver (Input)

All components related to the S3P data reception are connected to the main shift register. The block diagram of the S3P data receiver is presented in Figure 6.3.

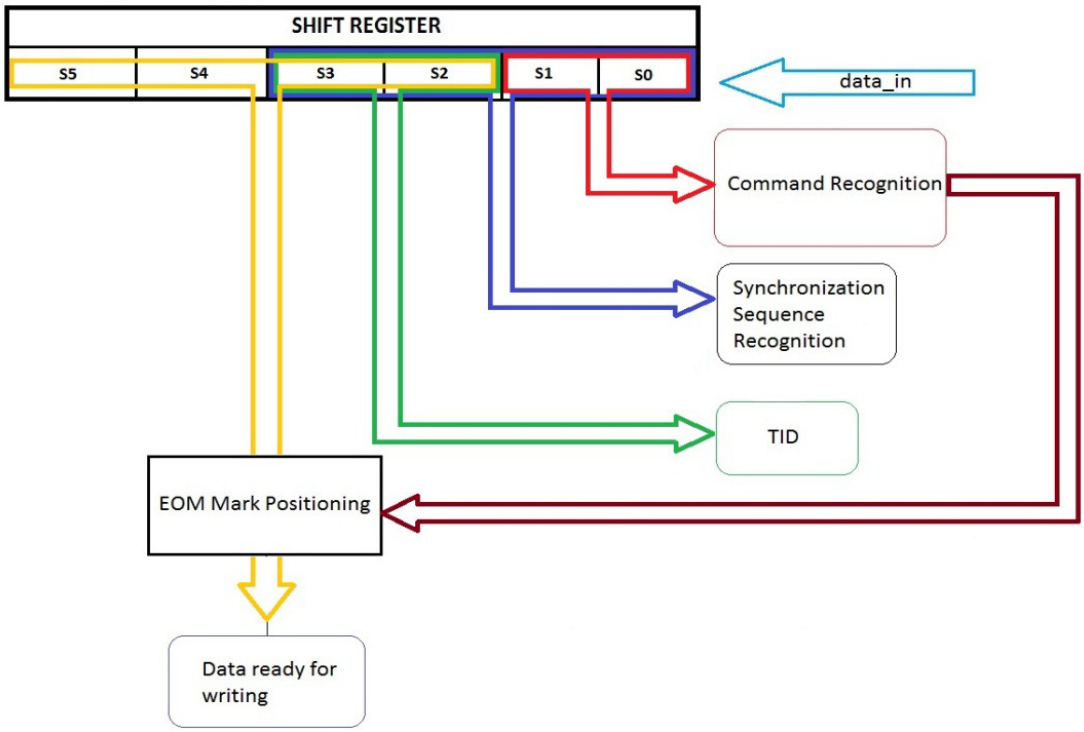


Figure 6.3. S3P data receiver

Data reception process is described through the following items:

- Port synchronization;
- Removing all data flow commands from transferring messages (STOP, CONTINUE, FILL); except message framing commands BOM and EOM. In case where STOP or CONTINUE command is detected, data receiver sets the interrupt and corresponding flag. Those commands are related to the S3P port output section and present data flow control for the units communicating with the MW Switch processor using different clock frequencies and provide a complete messages transfer without losing the data;
- Serial to parallel conversion into 32bit words (4Bytes) using a shift register;
- In case that new SYNC is detected after the initial synchronization (during legal message transfer), the S3P port needs to be re-synchronized. This means that the SYNC should not be included in the message which we want to transfer. An external unit which sends data can send SYNC command only if a new synchronization is required. All previously written data will be deleted;
- EOM mark positioning for the 32-bit representation.

Port Synchronization

Based on the specification of SCAN system, data transfer can start only after the input S3P port has been synchronized. The synchronization process starts by receiving the 16 bits, all '1' (SYNC command - 0xFFFF). After the SYNC command has been received, the S3P port waits for the BOM (0xFF02) and the data transfer process continues. The bit sequence required for synchronization is presented in Figure 6.4.

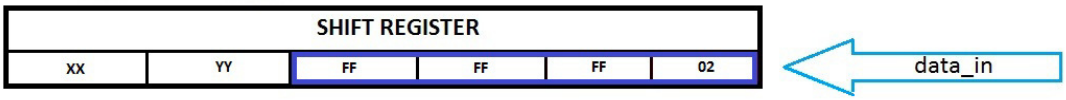


Figure 6.4. S3P synchronization

TID recognition

After the S3P port has been synchronized, the next received data is Topic Identifier (TID). The MW-SW-ASIC is responsible for establishing transfer paths between different devices. All communication in the system is based on a publisher / subscriber protocol. In this protocol, publishers send messages under a given topic and subscribers (0, 1 …, *) receive all messages published under the given topic (multicast behavior). The topic is represented by the topic identifier (TID). To establish a transfer path, the publisher and the subscriber have to share the same TID.

Figure 6.5 shows the shift register in the moment when BOM (0xFF02) and TID (D0 and D1) are received and ready for write process in inputs FIFO. When data receiver receives the TID, it will not be automatically written in the TID register, readable for processor. The TID value will be ready for the processor in the moment when first 32-bit word is read out of the FIFO buffer. The address offset for S3P TID register is 0x04 in hexadecimal form.

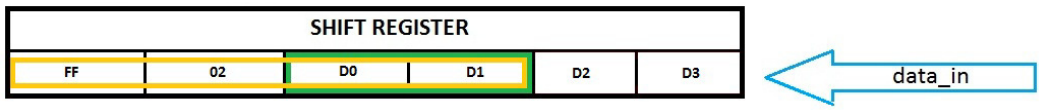


Figure 6.5. BOM and TID ready for write in FIFO

After receiving the BOM and TID, the shifting process goes further. The data ready for writing in the FIFO will be collected in 32 S3P clock cycles only in case that no data flow command is received. Figure 6.6 presents this situation, in relation to Figure 6.5:

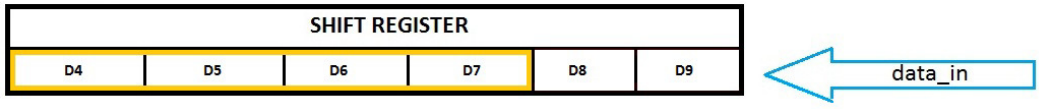


Figure 6.6. Received data ready for writing into the FIFO

Data Flow Command Detection

Related to the SCAN concept, it is required to remove all data flow commands from the transferring messages (STOP, CONTINUE, FILL), except message framing commands BOM and EOM. In case

where STOP or CONTINUE command is detected, the data receiver sets the interrupt and corresponding flag. Those commands are related to the S3P port output section and presenting a data flow control for the units communicating with the MW Switch processor using different clock frequencies, and provide a complete messages transfer without losing the data.

Command detection is done in first two bytes of the shift register. In case that data flow command is recognized, the shift register will shift just those two bytes and continue with the normal data shift process. Example of STOP command recognition is presented in Figure 6.7.

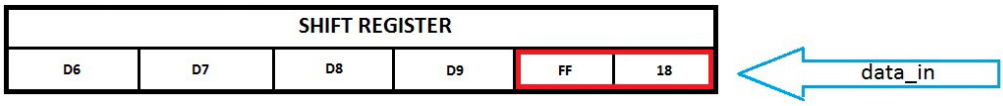


Figure 6.7. Recognized STOP command

When the data flow command has been detected, port logic activates an interrupt and sets a bit of the recognized flow command in status register. Commands and related offset addresses are presented in the table below. The only way to find the activated interrupt (stop or continue) is to detect from which port it is coming and then to search through the status register. After the interrupt routine has been activated, it is important to mask the activated interrupt in order to avoid loop. This type of interrupt will be self-disabled after a new bit has been written in input shift register. The offset address for the status register of the input S3P port is 0x0C. The **stop** flow command flag can be found at the position '0'; the **continue** flow command flag can be found at the position 1

EOM Mark Positioning

When the data have been parallelized and EOM mark detected, it is possible to make four different EOM's positioning the 32-bit word before it is written in the FIFO:

a) EOM (0xFF03) detected after writing a 32-bit data word. In this case EOM (0xFF03) will be written in the new 32-bit word with extra added '1'(0xFFFF)

D_{n-3}	D_{n-2}	D_{n-1}	D_n
FF	03	FF	FF

b) EOM is detected after 8 received bits. In this case EOM mark will be added in word bit position 23:8 and additional 0xFF in order to provide full 32-bit word

D_{n-4}	D_{n-3}	D_{n-2}	D_{n-1}
D_n	FF	03	FF

c) EOM is detected after 16 received data bits. In this case EOM mark will be added in the lower 2By position – 15:0. This automatically forms the full 32-bit word ready for writing in the FIFO

D_{n-5}	D_{n-4}	D_{n-3}	D_{n-2}
D_{n-1}	D_n	FF	03

d) EOM is detected after 24 received data bits. First part of EOM mark (0xFF) will be added in order to form complete 32-bit word and will be written in FIFO. Second part of the EOM mark (0x03) and 24 logic '1s' (0xFFFFFF) will be added in the new 32-bit word and then will be written in the FIFO

D_{n-6}	D_{n-5}	D_{n-4}	D_{n-3}
D_{n-2}	D_{n-1}	D_n	FF
03	FF	FF	FF

Byte Stuffing

To prevent the command byte value 0xFF from occurring in the transparent data and to be misinterpreted, it must be skipped. The command byte value (0xFF) is skipped whenever it appears in the user data stream. Moreover, the command byte value is forbidden in both bytes of the topic ID. To reduce the amount of "forbidden sequences/bytes", the used escape byte is also set to 0xFF. When the sender detects a forbidden byte, it is transformed according to the rule, presented in Figure 6.8. STX represents BOM command and ETX represents EOM command.

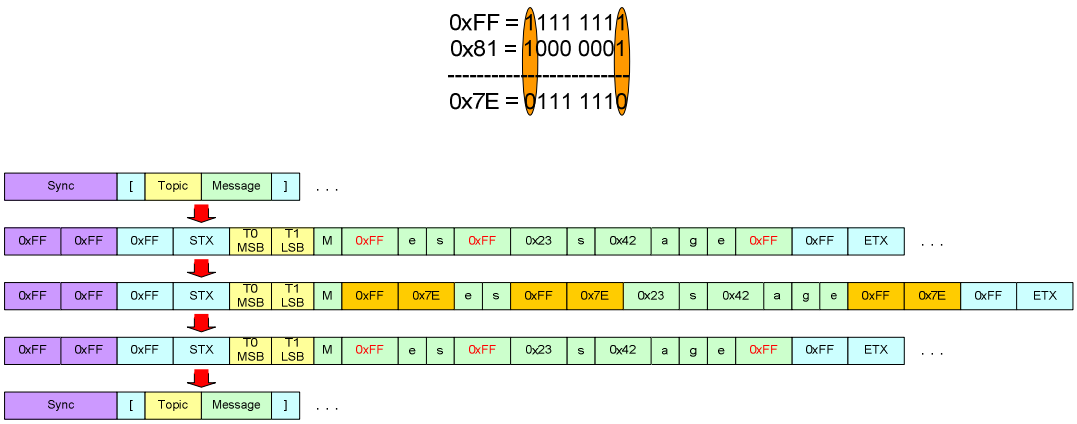


Figure 6.8. Data view from receiving phase to transmitting phase

FIFO Controller

FIFO controller is the main component of input S3P port and its environment is presented in Figure 6.9. In this section is described a working algorithm and functionality of all other components, following the data flow (how the data are processed). Same FIFO controller unit is also used for the output S3P port. Therefore, in the following text the FIFO controller section of the output S3P port will not be described in details.

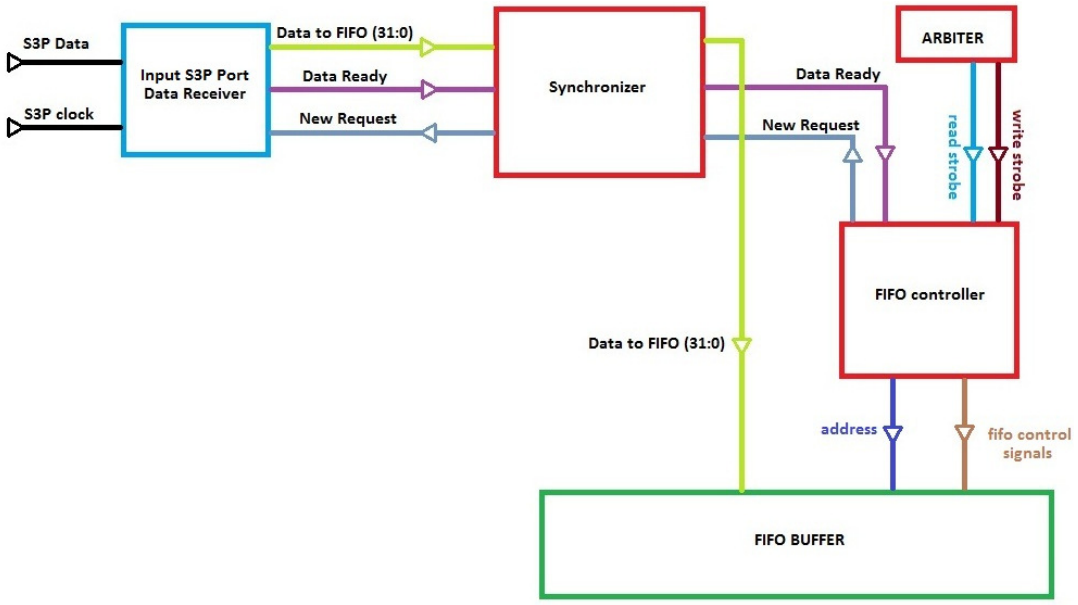


Figure 6.9. FIFO controller environment (input S3P port)

When data receiver block is received a 32-bit word, it communicates with FIFO controller using a handshaking interface. The handshaking interface comprises two signals: **data ready** (from the data receiver side) and **new request** (from the FIFO controller side). It is important to note that the synchronization system (presented as synchronizer in Figure 6.9) synchronizes the signals between internal processor clock domain and S3P external clock domain defined by the transmitting device.

In order to provide access for writing and reading operations, a memory arbiter is used. Memory arbiter provides writing (reading) operation in every second system clock cycle, as presented in Figure 6.10.

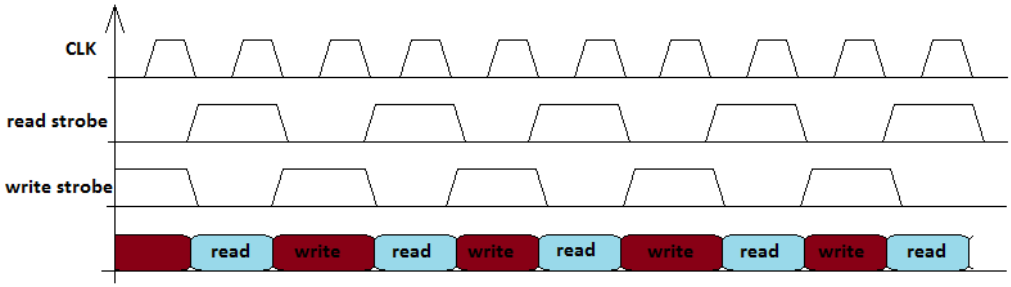


Figure 6.10. Arbiter strobe signals and enabled read/write memory access slots

After the system initialization, FIFO controller sends **new request** signal to data receiver block. When

the first 32-bit word is ready for writing in the FIFO buffer, the logic in data receiver activates (set to high logic level) the **data ready** signal. In the moment when data are ready for writing, first the FIFO controller needs to get a slot from arbiter, so that the writing process can be performed. Second, the reading process needs to be inactive in order to set correct control signals and addresses for the FIFO buffer.

When all these requirements have been fulfilled, the FIFO controller sets related FIFO control signals group (chip select – CS, write strobe – WRSTR, read strobe – RDSTR) and in the next system clock cycle a new value is written in the FIFO buffer. During writing process, the **new request** signal is inactive. After successfully written 32-bit word, the FIFO controller again activates the **new request** signal and system is ready to receive a new 32-bit word. Figure 6.11 presents the writing process.

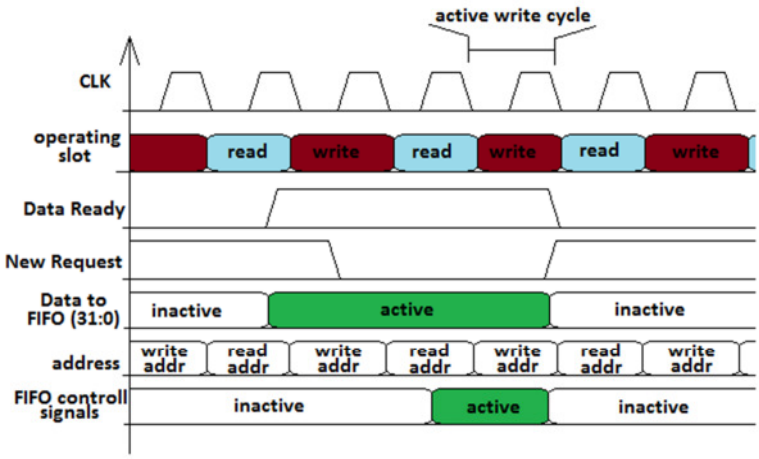


Figure 6.11. Writing process – FIFO buffer

FIFO controller is not used only for the write/read control. It is also used to provide the status of FIFO buffer. Blocks which are able to provide a current status are: read/write control block, read pointer, write pointer, message counter and pointer control block. All mentioned blocks are connected to the FIFO controller and they are presented in Figure 6.12.

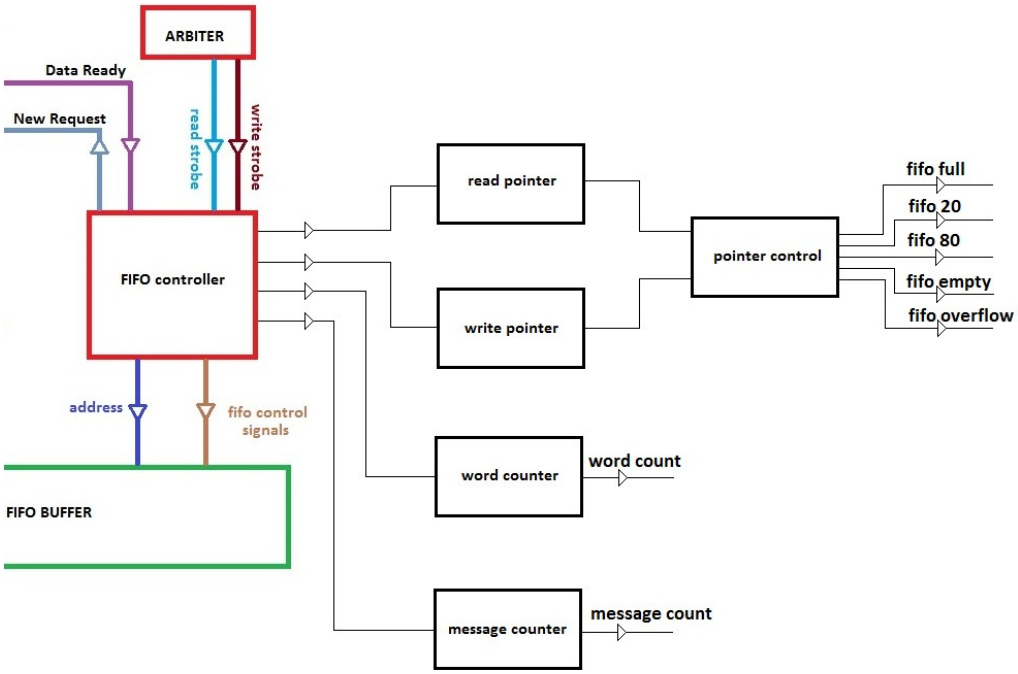


Figure 6.12. FIFO controller with status counters

Read/Write Pointer

Read (write) pointer is used for generating the addresses for reading (writing) process. It increases after each data read (write) phase from (in) the FIFO buffer. Read and write pointer outputs are connected to the pointer control block. Pointer control block monitors the current status of FIFO buffer and sets the related flag. Related flag descriptions and addresses offsets for the input-port FIFO controller are shown in Table 6.3.

Table 6.3. Description of flags used for status information of input port FIFO

Flag Name	Description	Offset Address
FOV_I	FIFO overflow flag is set when the FIFO is 100% full and there are no more free data locations for the data	0x0C bit position:2
FFULL_I	FIFO full flag is set when 95% of data locations are used	0x0C bit position:3
F80_I	This flag is used mainly for the data flow control. The pointer controller sets this flag when more than 80% of data locations are used	0x0C bit position:4
F20_I	This flag is used mainly for the data flow control. The pointer controller sets this flag when less than 20% of data locations are used	0x0C bit position:5
FEMPTY_I	This flag is set when there are no data in the FIFO buffer	0x0C bit position:6

Word Counter

Word counter is used to inform how many bytes are written during one message transfer. The message is framed by BOM and EOM and the maximum allowed number of bytes in one message is 1300. ***NOTE:*** In this version of Middleware Switch Processor the word counter is included but not readable by processor.

Message Counter

The number of messages written in FIFO memory is one of the most important status information. Reading a message from the input FIFO buffer can be performed only when one complete message has been written in the FIFO buffer. Until a pair of BOM and EOM commands has been written, the reading process can't be started. If message counter value is greater than 0x00 hexadecimal, related signals in the reading control part of FIFO controller are providing two interrupts:

1) One or more messages are in FIFO buffer
2) Link request for internal transmission

The link request (LKRQ_I) interrupt, for the internal switch matrix transmission, is set if one or more messages have been written in the FIFO buffer. Software reads out the interrupt and performs the internal data transmission (between different S3P ports) related to the defined TID. The data will be directed to the output S3P port (or directly to the external memory unit), where one or more units are able to receive the data from an input S3P port (broadcasting). The data transfer is performed by using the switch matrix (crossbar switch), which is explained in details in Section 6.2.1.4 (related to the Output S3P port).

Parallel to Serial Convertor (Internal Serial Transmitter)

Before the data have been transmitted through the switch matrix, they should be serialized. This is done in order to save hardware resources. If we compare the occupied silicon space and power consumption of the switch matrix for the serial data transfer to another switch matrix for the parallel 32-bit data transfer, we can note 7 times more power consumption and 10 times more silicon usage of the latter! The implementation results applied to the IHP 250 nm technology are presented in Table 6.4.

Table 6.4. Implementation results of different switch matrix (crossbar switch) versions

	Area [um^2]	Power [mW]
Serial Switch Matrix	2009.904	1.286
Parallel Switch Matrix	33974.640	7.911

In order to read data from the FIFO buffer it is required to have a complete message, saved in FIFO buffer. Serial to parallel convertor first read a BOM and TID from FIFO buffer and then sets an interrupt (LKRQ), whereby it requests a link for the defined topic ID (TID). During the serialization process of received data, there is enough time to get a free frame for reading (in case the writing process is active in the same moment). The reading process takes more time than the writing process because of the time required for data to be stabilized on the FIFO buffer outputs. The applied handshaking system (**read request** and **FIFO data enable** signals) for reading process is similar to the handshaking system used for the writing process. A standard reading procedure is presented in Figure 6.15. Active read cycle starts in the moment when all FIFO control signals are stable and when reading slot (defined by arbiter block) is active. Data are read in next clock cycle but because of

memory characteristics used for the FIFO buffer implementation, the data will be stable and ready for serial conversion after one clock cycle more. This process is internal and description in this section is provided in order to understand the functional principle of the Middleware Switch S3P protocol.

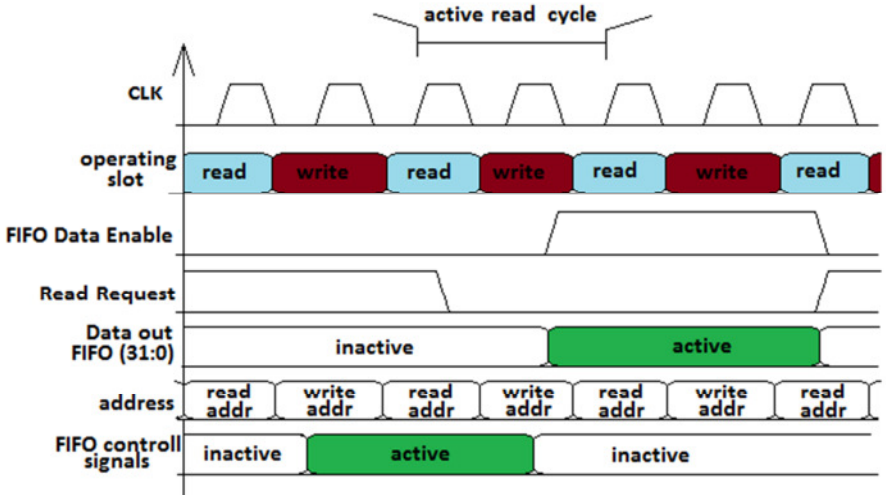


Figure 6.15. Reading process – FIFO buffer

Figure 6.16 shows the block diagram of complete S3P input port. Outputs of "parallel to serial convertor" block are directly connected to the switch matrix. Signal "tx_enable" provides synchronization signal which defines the valid data on the receiver side (output S3P port).

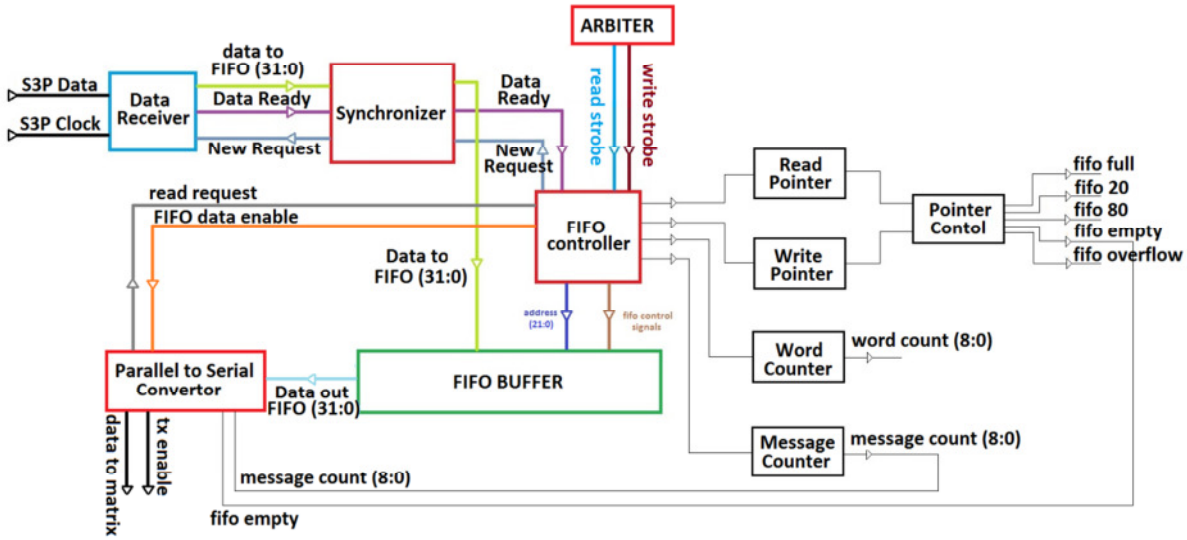


Figure 6.16. Input S3P port – Block Diagram

In next sections is provided description of output S3P port and switch matrix. In order to provide better explanation how the switch matrix and S3P protocol operate, in the following section are provided few examples.

Output S3P Port

Data transferred through the switch matrix consists of two signals: the serialized transferred data and the transfer valid signal. On the input S3P port side, these signals are named respectively: “data to matrix” and “tx_enable”. Same as in the data receiver on the input side, the output S3P port has a receiver device. Instead of the data receiver used for the input S3P port, a simplified version, based on the shift register, is implemented here. FIFO controller, used for the output S3P port, is same as FIFO controller in the input S3P port. The FIFO buffer has the same characteristics as FIFO buffer used for the input S3P port. According to the data processing flow, further text provides details of following digital blocks:

- Switch Matrix
- Data Receiver (Output S3P Port)
- FIFO Controller
- Word Counter
- Message Counter
- Parallel to Serial Convertor.

Switch Matrix and Data Receiver (output S3P port)

Output-port data receiver receives bits transferred through the switch matrix and does the data parallelization. This version of data receiver does not need to recognize any message framing mark (BOM, EOM, FILL, STOP…). The shift process starts when the “tx_enable” (generated by input port) signal activates. All signals are fully synchronized with the system clock. The related signals of the output S3P port data receiver are presented in Figure 6.17.

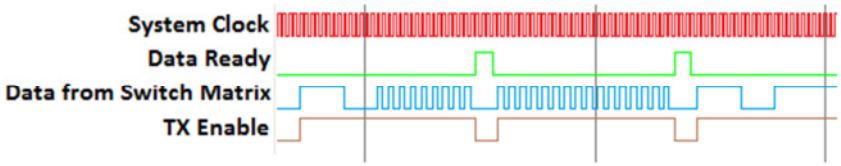


Figure 6.17. Switch matrix transferred data – 0xFF|02|A5|A5; 0xA5|A5|A5|A5; 0xFF|03|FF|FF

Data Ready signal informs the FIFO controller that a 32-bit word is formed in the data receiver and that word can be written in the FIFO buffer. For example shown in Figure 6.17, 3 different 32-bit words are used. This short data sequence was used in the initialization development process of the S3P ports and it is presented in Figure 6.18. As it is easy to notice, this 3x32-bit word sequence forms a complete message.

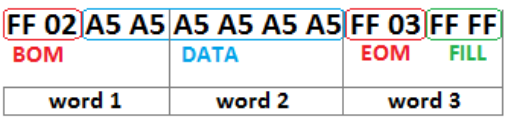


Figure 6.18. Message example in S3P protocol

It is important to notice that the switch matrix (crossbar switch) is integrated into the output S3P port.

Data transfer between five S3P ports is described in Figure 6.19. The control registers, related to the switch matrix, are **LINK_O (2:0)** and **WREN_O**. Figure 6.19 a) presents the switch matrix after the initialization without any active transfer. Figure 6.19 b) presents a data transfer (together with related **LINK_O** and **WREN_O** registers set) from port 2 to port 5, from port 3 to port 1 and from port 1 to the port 4. Data transfer from port 2 to the ports 1, 3, 5 and from port 5 to the ports 2 and 4 is presented in Figure 6.19 c).

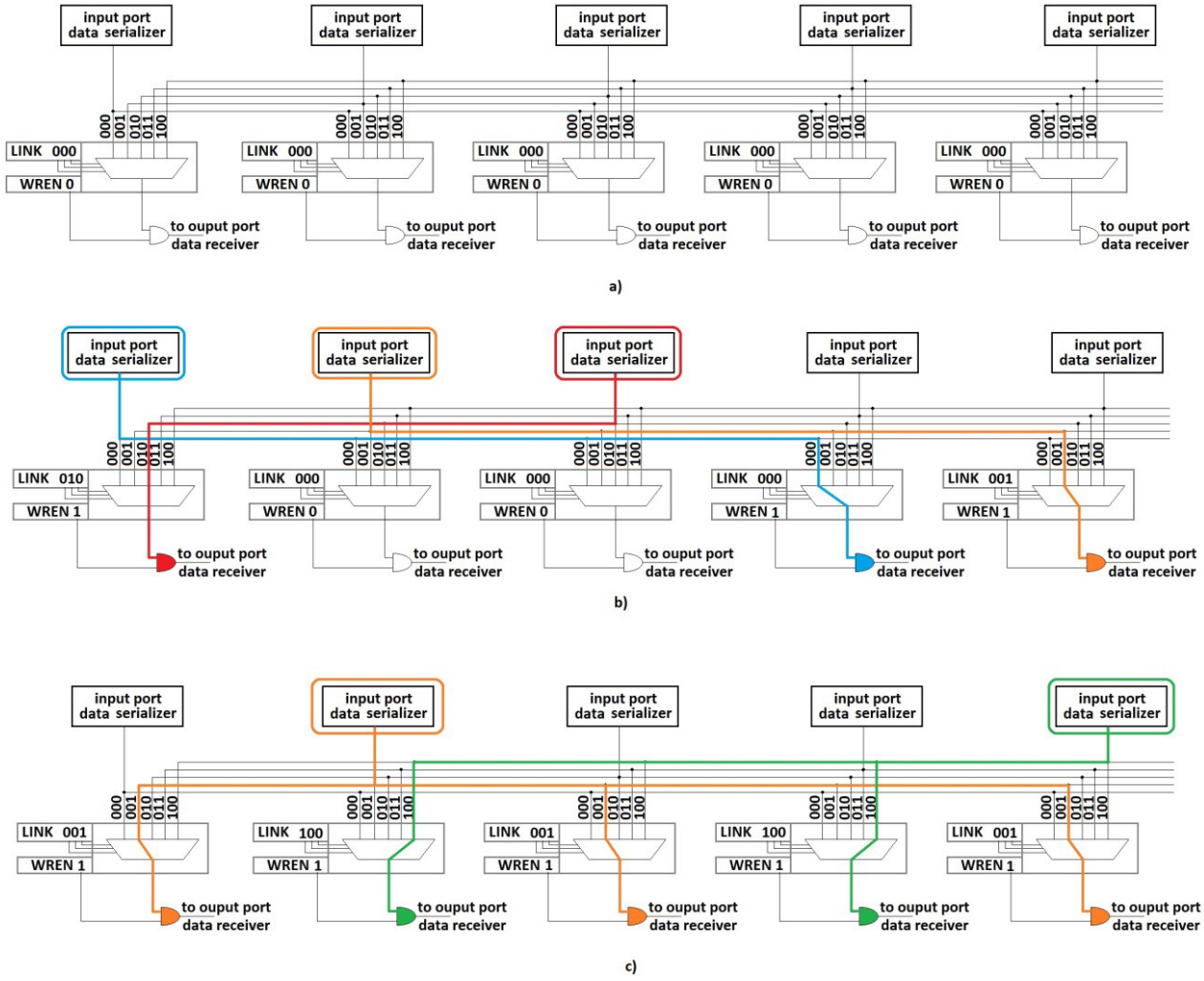


Figure 6.19. Data transfer through the switch matrix: a) LINK and WREN registers after initialization; b) from Input 1 to Output 4, from Input 3 to the Output 1 and from Input 2 to the Output 5; c) from Input 2 to the Outputs 1, 3, 5 and from the Input 5 to the Outputs 2 and 4

Important status/control registers for internal data transfer (with offset addresses) are presented in Table 6.5.

Table 6.5. Internal data transfer status/control register

Signal Name	Description	Offset Address
LKRQ_I	LKRQ interrupt sets when input S3P port requests a message link	0x28 bit position: 8
LKEN_IN	The CPU generated signal after the LKRQ interrupt is set and after the data link is provided for data transfer	0x10 bit position: 0
LINK_O	Address of the input S3P port, from which the related output S3P port should receive the data	0x1C bit position: (2:0)
WREN_O	The data receiver does not need to receive all data transferred through the switch matrix at the related address. This software is given the freedom to provide the selective data reception, especially in case when broadcasting data transfer is used.	0x1C bit position: 3

Received data are collected in form of 32-bit words and they are ready for writing in the FIFO buffer. Data receiver consists of shift register and bits number control. This is done in order to provide reliable data reception. In case that a transient effect happens on the “tx_enable” line and data receiver receives 31 or 33 and more bits for the one word, this word will be not be written in the FIFO buffer – it is corrupted. Output port will wait until the next BOM message frame comes and continue with the reception process.

FIFO Controller – Output S3P Port

FIFO controller used for the output S3P port is same as the FIFO controller used for the input S3P port. The only difference between input and output S3P ports is in the synchronization circuit. In output S3P port a synchronization circuit between data receiver and FIFO controller is not required. The same handshaking interface is used for the communication between the data receiver and the FIFO controller in the output S3P port. Figure 6.20 are presents the data receiver block, FIFO controller and other components related to the data reception part of the output S3P port.

It is possible to notice that in Figure 6.20 the communication system between data receiver and FIFO controller is same as the system described in section related to the input S3P port and it will not be described in details here. The writing and reading procedures for both FIFO controllers are the same.

As it was discussed in the input S3P section, the FIFO controller is not used just for the write/read control. It is also used to make the status of the FIFO buffer readable to the CPU. The status related blocks are: read/write control block, read pointer, write pointer, message counter and pointer control block. All mentioned blocks are connected to the FIFO controller and they are presented in Figure 6.21.

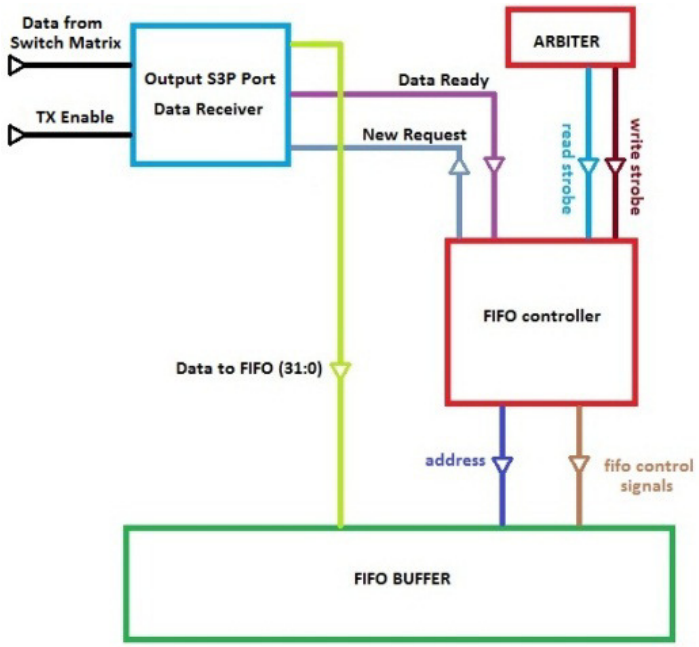


Figure 6.20. FIFO controller environment (output S3P port)

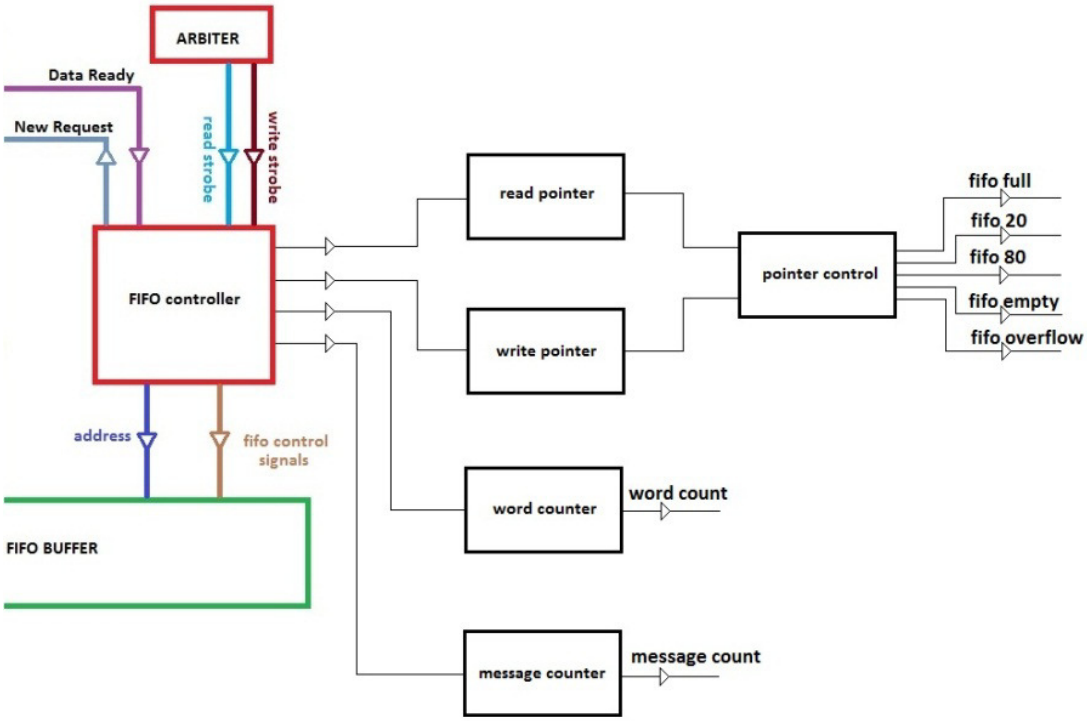


Figure 6.21. FIFO controller with status counters

It can be noted that Figure 6.21 is the same as Figure 6.12 because of the same structure and functionality. The read (write) pointer is used to generate the address for the reading (writing) process. It increases the address value after each data read (write) phase from (in) the FIFO buffer. Read and write pointer outputs are connected to the pointer control. The pointer control monitors the current status of FIFO buffer and sets the related flag. Flag descriptions and addresses offsets are shown in the following Table 6.6.

Table 6.6. Description of flags used for status information of input port FIFO

Flag Name	Description	Offset Address
FOV_O	FIFO overflow flag is set when the FIFO is 100% full and there are no more free data locations for the data	0x14 bit position:9
FFULL_O	FIFO full flag is set when 95% of data locations are used	0x14 bit position:10
F80_O	This flag is used mainly for the data flow control. The pointer controller sets this flag when more than 80% of data locations are used	0x14 bit position:11
F20_O	This flag is used mainly for the data flow control. The pointer controller sets this flag when less than 20% of data locations are used	0x14 bit position:12
FEMPTY_O	This flag is set when there are no data in FIFO buffer	0x14 bit position:13

Word Counter

Word counter is used to inform how many bytes are written during one message transfer. The message is framed by BOM and EOM. Maximum allowed number of bytes in one message is 1300.

Message Counter

The number of messages written in FIFO memory is one of the most important status information. Reading a message from the output FIFO buffer can be performed only when one complete message has been written in the FIFO buffer (same as in input S3P port). Until a pair of BOM and EOM has been written, the reading process can't be started. The message counter for the output S3P port has address offset 0x14 with bits positioned from 0 to 8. When the message counter value is greater than 0x00, the related signals in the reading control part of the FIFO controller provide two interrupts:

1) One or more messages are saved in the output FIFO buffer
2) Port request for external transmission

The port request (**PTRQ_O**) interrupt, for the external port-to-port or port-to-device transmission, is set when one or more messages have been written in the output FIFO buffer. Software reads out the interrupt and performs the data transmission, related to the defined TID by software. The data will be directed to another S3P port or to the device which can communicate within the S3P protocol (or to the external mass memory).

Parallel to Serial Convertor (External Serial Transmitter)

Parallel to serial convertor, integrated into the output S3P port, does the data serialization, provides the related S3P generated clock, initiates the reading process from the output port FIFO buffer and communicates with the CPU through interrupt, status and control registers.

When a complete message has been written in the output FIFO buffer, the message counter increase the value from 0x00 and the external data serial transmitter sets an interrupt to the CPU for the

external serial data transmission – **PTRQ_O** (port request). The external serial data transmitter has an integrated clock generator in order to make the mentioned S3P protocol available to another connected device, which receives the data from the related S3P port.

After the CPU has registered an interrupt from the external serial transmitter, it transmits a transmission enable (TXEN_O) signal and the process of data transmission starts. Because of the S3P data flow control commands, the external serial transmitter can send STOP, CONTINUE, FILL and SYNC commands. The SYNC command is sent automatically when the bit rate register (BRREG_O) has been set. The output clock generator starts sending clock cycles and, when first data have been serialized, the data transmission process starts. In case that between BOM and EOM mark, a FIFO buffer need to wait for the data received from the switch matrix, the port will automatically start sending FILL words in order to keep the synchronization and inform the receiving device that the transferring message is not corrupted – just on hold. The output port needs to send the STOP or CONTINUE mark, only if the CPU sets the corresponding registers TXSTOP_O and TXCONT_O, respectively.

The block diagram of the complete S3P port, with input and output S3P ports is presented in Figure 6.22.

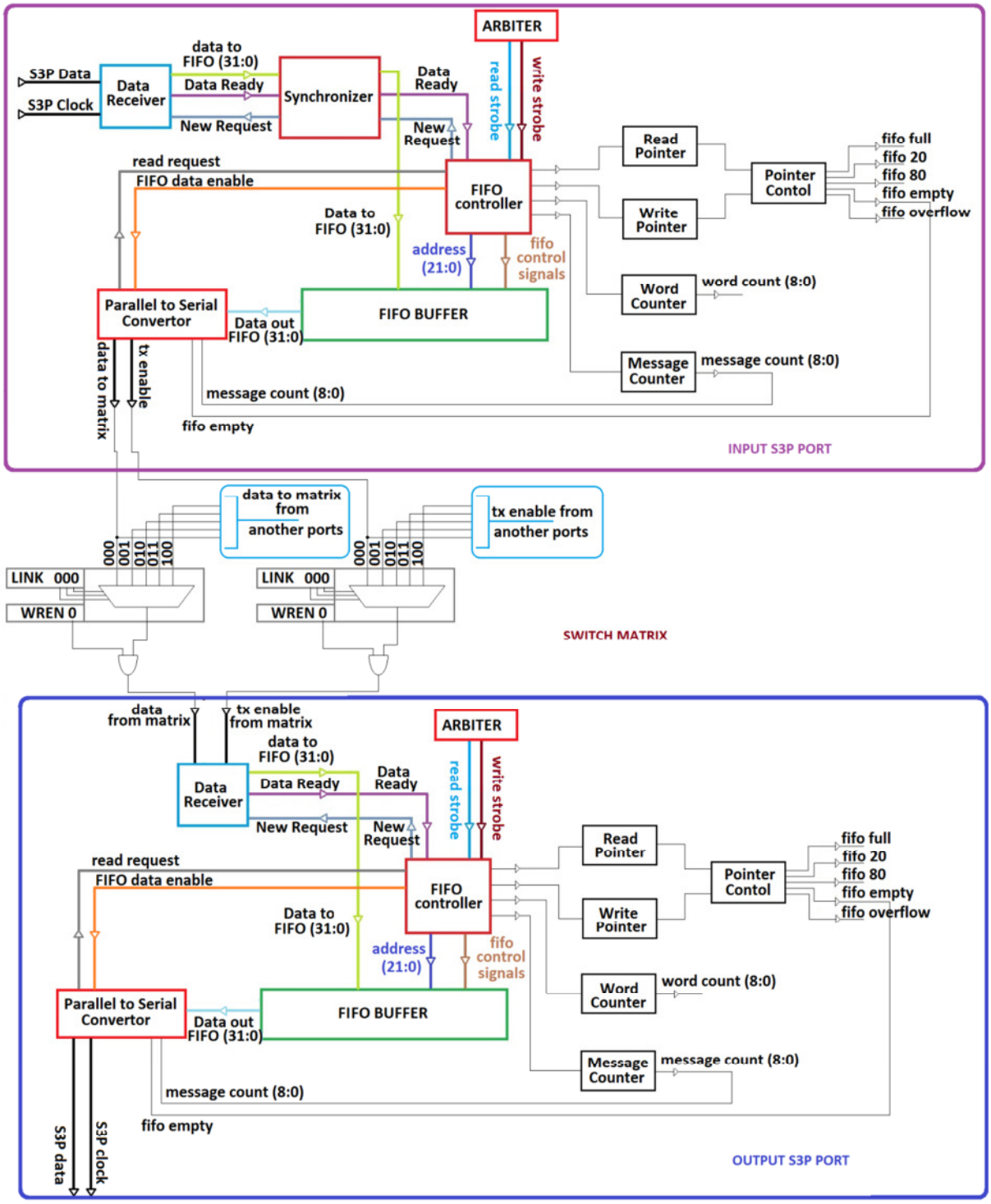


Figure 6.22. Complete S3P port (Input S3P Port, Switch Matrix and Output S3P Port)

Data Flow Control

In case that a S3P input port (or more precisely, its assigned input FIFO) is not able to store the data any more, a STOP command is sent by the output port. External serial transmitter of the same port provides this action. As pointed out, the device on the other side of link needs to be able to understand the flow control commands. The data flow commands can be sent at any time, between messages or inside of messages. As soon as a STOP mark is received in the input S3P port, the external transfer,

on the output port side, will be stopped and either FILLERs (inside of a message) or SYNCs (between messages) will be transmitted.

When the associated input port is able to store more data, the corresponding output port sends a CONTINUE mark. The external sender then continues the interrupted transmission. NOTE: The flow control marks are never written to the FIFO data buffers. In the current version, the flow control within S3P shall be handled by software.

To achieve this, the FIFO controller sets ALMOST FULL (f80_i or f80_o) and ALMOST EMPTY (f20_i or f20_o) signals in the port status register, which, if not masked out, leads to a port interrupt. The ALMOST FULL signal is set at fill level of approx. 416 words, the ALMOST EMPTY signal at fill level of approx. 96 words.

The CPU can then handle the port interrupt by issuing commands to the hardware, e.g. **Stop External Transmission (TXEN_O = '0')**, **Continue External Transmission (TXEN_0 = '1')**, **Send STOP Mark (TXSTOP = '1'; TXCONT = '0')** or **Send CONTINUE Mark (TXSTOP = '0'; TXCONT = '1')**. To achieve flow control, each port controller is able to react to these CPU commands directly, bypassing the output FIFO.

As pointed out, the maximum S3P message length is 1300 bytes. To prevent a deadlock situation if an (illegal) extremely large message is in the receiving FIFO and cannot be moved due to a missing EOM mark, all messages longer than the maximum message length will be deleted. Apart from that, every new message generating an overflow is discarded, too. Figure 6.23 a) presents a situation where the input S3P FIFO is almost full and the external serial transmitter sends a STOP command to the device sending data to the corresponding input S3P port, which is sending the data to the output S3P port through the switch matrix.

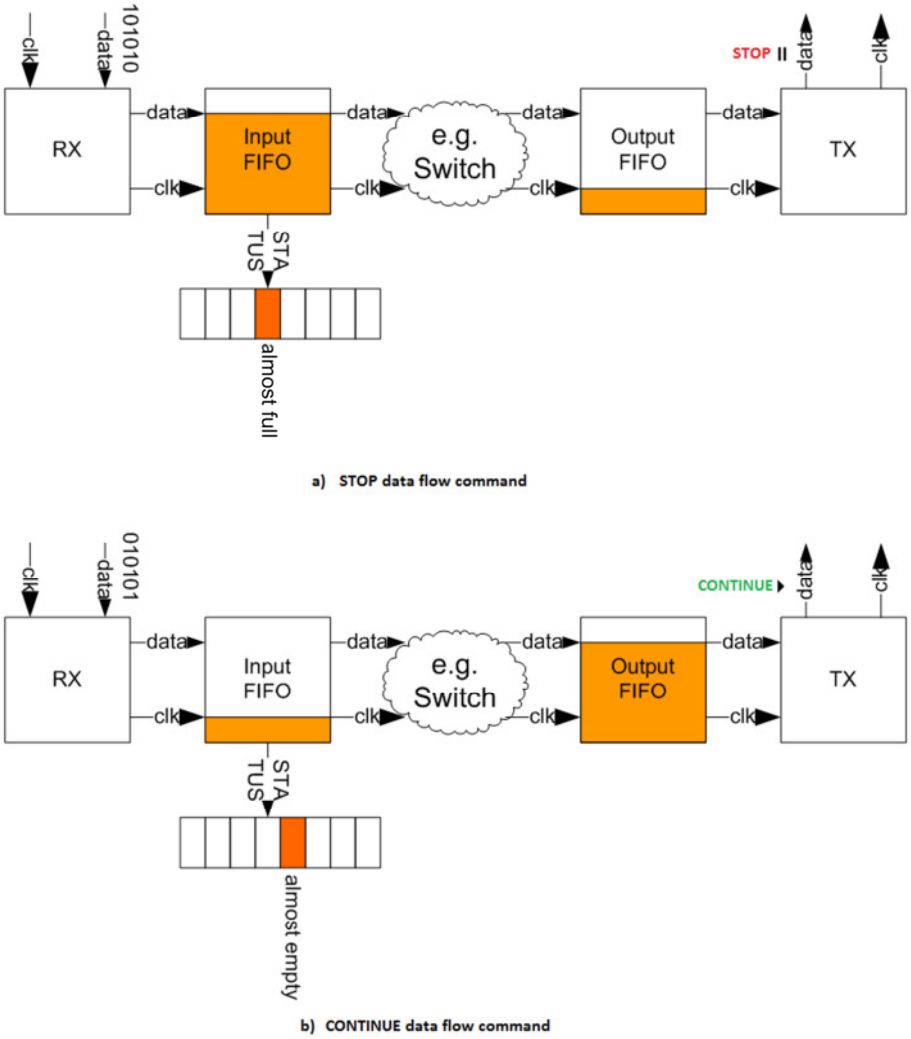


Figure 6.23. Data Flow Control S3P Port

Figure 6.23 b) presents a situation where the input S3P FIFO is almost empty and the output S3P FIFO is almost full. The external serial transmitter sends then a CONTINUE command to the device which sends data to the related input S3P port. This re-enables the data transfer from the transmitter to the related input S3P port and input port FIFO buffer fills with new data.

Internal Flow Control

Flow control has to be applied for internal transmissions as well. E.g. it could happen that an output FIFO is filled faster than it is emptied and can therefore overflow.

The CPU has to avoid the overflow situation by stopping the corresponding internal transmission. To achieve this, the FIFO's ALMOST FULL flag of the output FIFO is evaluated. As the CPU software knows which port is involved in the transfer, it can send a "**Stop Internal Transmission**" (LKEN_I = '0') command to the currently assigned sender (an input FIFO). When the corresponding (output) FIFO is able to store more data, the CPU sends a "**Continue Internal Transmission**" (LKEN_I = '1') command to the currently assigned internal sender (an input FIFO).

6.2.2. Middleware Switch Implementation

The Middleware ASIC is developed in standard SGB25V - 0.25µm IHP BiCMOS technology. The mentioned technology is not radiation hard technology. The measurement results of the MW switch processor in the nominal temperature, voltage and process conditions are provided in Table 6.7.

Table 6.7. Measurement results of non-FT Middleware Switch processor

Implemented Architecture ***Parameter***	Non-FT MW Switch Processor
Maximum Frequency [MHz]	105.4
Power Consumption [W]	1.3
Maximum Operating Temperature [℃]	49.07
Power Supply Core [V]	2.5
Power Supply Pad [V]	3.3
Area [mm²]	64

During measurements the temperature analysis of the processor is provided in order to find potential "hot spots". The results of the thermal analysis are presented in Figure 6.24. We can see that the MW switch processor has almost uniform temperature distribution. Maximum temperature differences are less than ± 2℃.

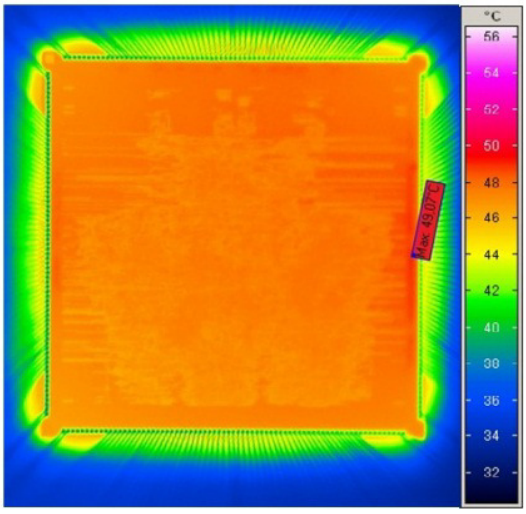


Figure 6.24. Thermal analysis of the MW switch during operation in nominal conditions

The implemented non-FT MW Switch processor architecture, after production, packaging and bonding is presented in Figure 6.25.

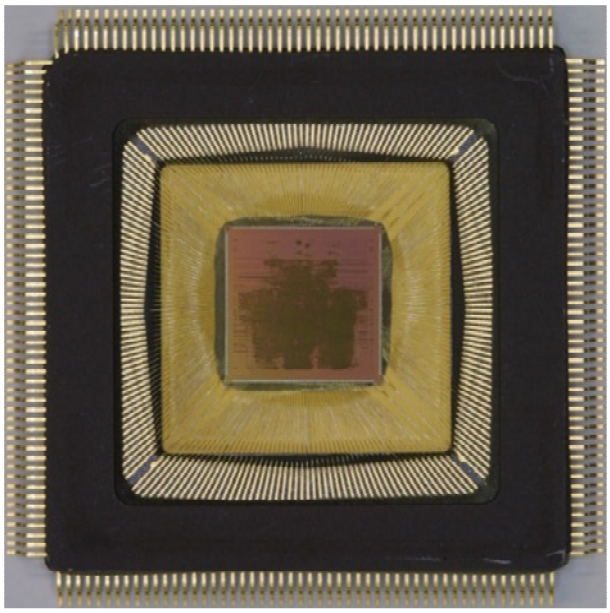


Figure 6.25. Non-FT MW Switch processor (open package)

Following sections describe fault-tolerant MW switch architecture and implementation characteristics. The implementation is based on the developed design methodology using DMR approach with latchup protection.

6.3. Fault-tolerant Middleware Switch Processor

The developed fault-tolerant design methodology is applied on the middleware switch architecture with reduced hardware. Hardware reduction is based on the exclusion of UART communication ports and DMA controllers. Therefore, beside Leon2 core the implemented fault-tolerant architecture involves S3P ports and switch matrix. Block diagram of the fault-tolerant Middleware Switch processor is presented in Figure 6.26.

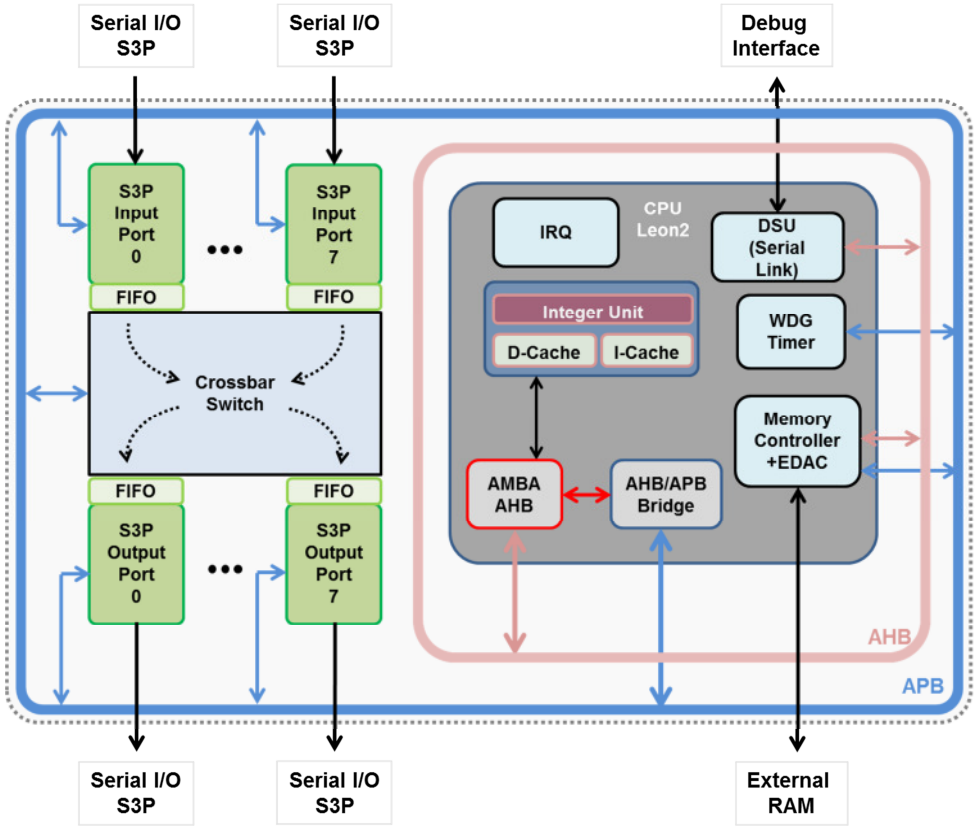


Figure 6.26. Reduced Middleware Switch architecture – design methodology test case

6.4. Fault-tolerant MW Switch Implementation Characteristics

In this section are provided the most important information related to the implementation characteristics of the fault-tolerant middleware switch processor. Beside information about power consumption and area of the fault-tolerant processor version, here are represented implementation characteristics of non-fault-tolerant design too. This is done in order to compare two same architectures, which are implemented using two different design approaches. The comparison result provides information related to effects on the implementation characteristics, which a fault-tolerant design has. It is important to notice that test case is realized using reduced version of middleware switch processor.

In the fault-tolerant MW switch version, complete hardware is doubled and for memory protection against potential SEUs is used EDAC and against latchup effect is used the same SPS approach as for the standard cells. As the memory requires more power, few SPS cells are connected in parallel

mode. Power network controller is in this example implemented in the SGB25RH process without latchup protection. It is possible to provide two parallel power network controllers with integrated latchup protection but this is not the goal of this discussion.

As it was explained in previous Chapter 5, the synthesis is same for both architectures. Power consumption, occupied silicon area and maximal frequency after synthesis are presented in Table 6.8.

Table 6.8. Power, area and max operating frequency of synthetized circuit

Architecture	Power [mW]	Area [mm²]	Max. Frequency [MHz]
MW_Switch	343.690	24.876	94.07

In order to provide better view on the standard cell type used during implementation, in Table 6.9 is presented occupied area in relation to the combinational or sequential cells. It is important to notice that non-combinational cells involving memory blocks and flip-flops. Test case is implemented without latch cells.

Table 6.9. Area regarding cell type using standard design approach

Standard cell type		Required area [mm²]
Combinational Cells		4.116
Non-combinational Cells	Cache Memory – data + instruction	0.813 3.165
	S3P FIFO	11.626
	Sequential Cells	5.012
TOTAL		24.876

After netlist parsing and timing analysis of new DMR-based netlist it is possible to notice increase of power consumption and required silicon area. The implementation results of DMR netlist together with power network controller are presented in Table 6.10.

Table 6.10. Power, area and max operating frequency of circuit after DMR insertion

Architecture	Power [mW]	Area [mm²]	Max. Frequency [MHz]
MW_Switch_DMR	831.2	79.616	83.33

Changes of area after netlist parsing are presented in Table 6.11.

Table 6.11. Area regarding cell type using DMR design approach

Standard cell type		Required area [mm²]
Combinational Cells		8.232
Non-combinational Cells	Cache Memory – data + instruction	1.626 6.330
	S3P FIFO	23.525
	Sequential Cells	39.903
TOTAL		79.616

In the following figure is presented a view of MW switch chip floor-plan, prepared using the developed fault-tolerant design methodology. In order to provide better routing, the PNC is placed around

processor core as it is possible to notice in Figure 6.25.

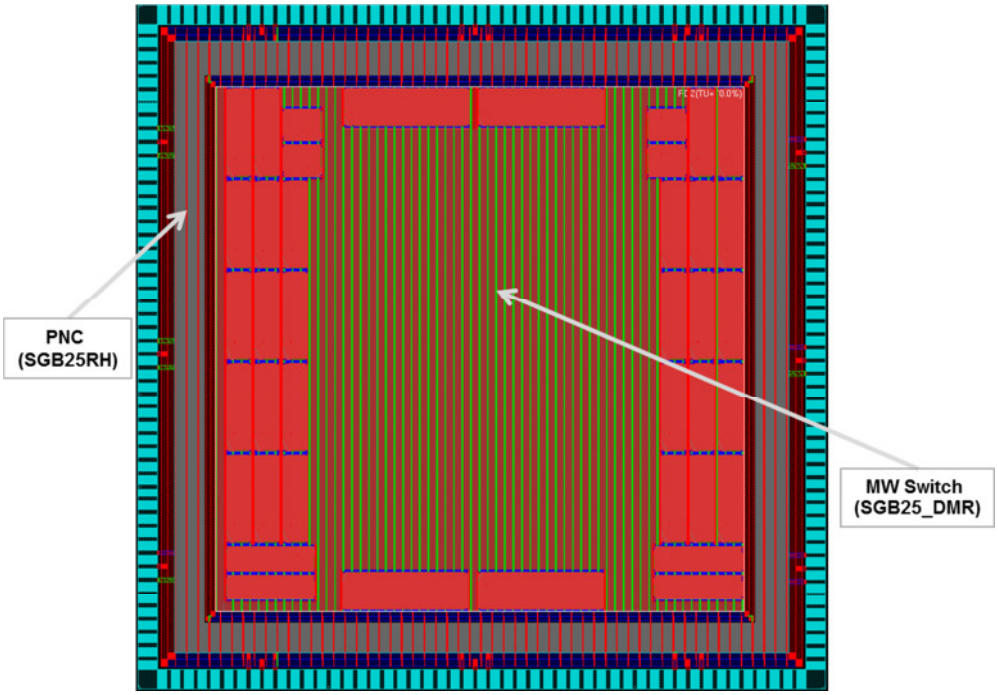


Figure 6.24. Floor-plan view of fault-tolerant middleware switch processor

Placed SPS cell is presented in Figure 6.25. It is important to note that stripes in MW switch processor core are generated using Metal3 layer. This is done because of control signals which are routed in Metal2 layer. Complete DMR based MW switch design is presented in Figure 6.26.

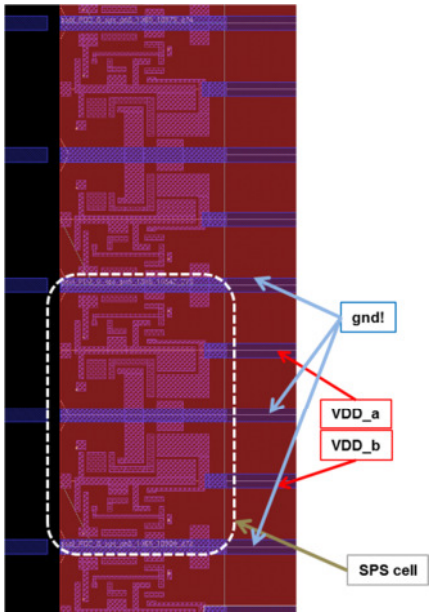


Figure 6.25. SPS cells after placement and power connections with rows

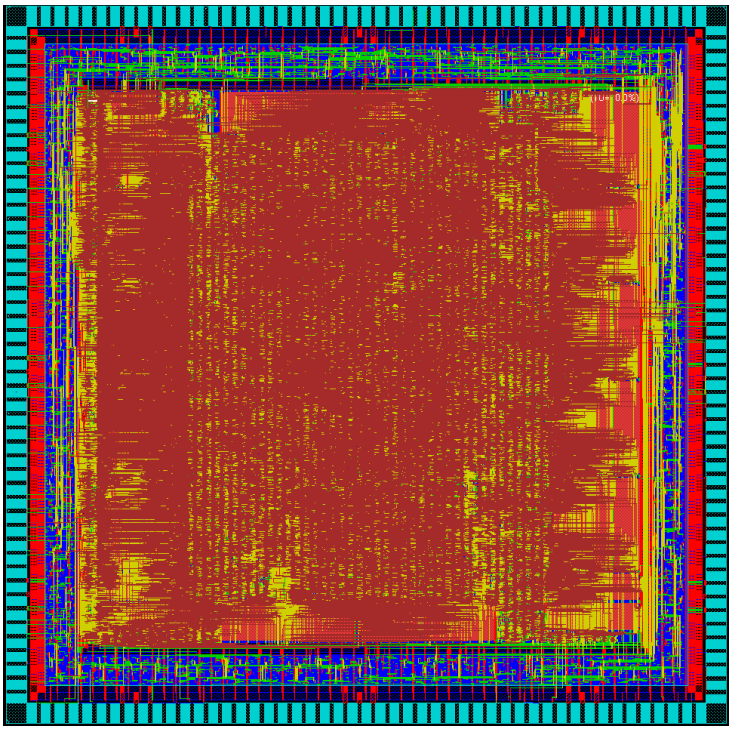


Figure 6.26. Finished ASIC layout designed using represented DMR approach

Implementation characteristics after physical design phase (layout) are presented in Table 6.12.

Table 6.12. Implementation characteristics after layout

Architecture	Power [mW]	DMR Core Area [mm²]	DMR Core Area with PNC [mm²]	Complete Chip Area [mm²]	Max. Frequency [MHz]
MW_Switch_DMR	1063	86.490	110.250	144.25	76.92

In order to discuss the above presented results it is very useful to compare them with usually used full-TMR mitigation technique. Based on the synthesis estimation done for the TMR circuit which supports latchup protection (Table 5.4; Section 5.3.2), the estimated area required for PNC and TMR based circuit is 88.781 mm². This estimation shows that expected area overhead by TMR approach is about 11.5%, compared to the DMR approach. The main reason for the area and power overhead is the power network controller. SPS cells doesn't involve any area overhead because they are implemented under power stripes where are usually placed filler cells. This is an important result because in this example are placed 21335 SPS cells.

It is clear that presented design methodology provides designs with reduced maximal operational frequency. Power consumption and required silicon area are also degraded. On the other hand, the protection against latchup effect, as well as protection against single event upsets and transients require trade-off between required hardware, power consumption, maximal operating frequency and sufficient protection level against radiation effects.

7. Conclusion

Advanced redundant circuits need a latchup protection in order to operate in a reliable manner. Therefore, this thesis introduces and describes newly developed single event latchup (SEL) protection switches and a technique for their integration into an ASIC design. Based on simulation results, it is concluded that the integration of latchup protection support decreases the failure-free probability of triple modular redundant (TMR) circuits and increases the failure-free probability of double modular redundant (DMR) circuits. The opposite effect on the two types of circuits is due to the different circuit topologies. In a TMR circuit with latchup protection, the additional “data keep” logic introduces a feed-back line which multiplies the original fault-free probability. In contrast, a self-voting DMR circuit already includes a voter feed-back line and the “data keep” logic does not introduce a reduction of the circuit failure-free probability. The simulations were performed using the fault-injection approach with predefined fault libraries. The final impact analysis of single event effects with a high fault coverage at circuit and system levels is provided by developing practical, accurate, and simple simulation fault models. The results demonstrate that SEL power switch (SPS) cells integrated with redundant circuits provide a reliable protection with a fast reaction time against the destructive latchup effect.

The SPS cells were characterized by measurements. Using a specially designed hardware, the two switch types (SPS Type 1 and SPS Type 2) and the two driver/sensor transistor widths (5 µm and 10 µm) have been proven as a good hardware infrastructure for the proposed latchup protection technique. It was also shown that redundant circuit logic state recovery is successful after the latchup protection phase is over. Irradiation measurements have proven the correct design and successful implementation of the SPS cell.

The main contribution of the presented work is the introduction and development of a methodology for highly reliable digital ASIC designs based on redundant circuits with latchup protection. The digital ASIC design flow was modified in order to generate triple modular redundant and double modular redundant netlists and to integrate single event latchup power switches (SPS) at the layout level. These SPS cells are designed to be handled as standard power cells and their placement is done using standard Cadence low power design flow. The developed design methodology has been used for protection against radiation induced single event effects, such as single event upset (SEU), single event transient (SET) and single event latchup (SEL). It protects circuits and systems with embedded memories using error detection and correction algorithms. Compared to the other developed mitigation techniques, the presented technique in this thesis has the goal to integrate different levels of protection in order to provide the highest possible protection level with additional costs.

As a practical case, the described approach and the developed redundant and protection techniques were applied to design a middleware switch processor for network-centric systems. This example illustrates the advantages and weaknesses of the used methods and techniques. It reveals the methodology parts that need further improvements as well as preferred directions for future developments.

Based on the implementation figures, it is easy to notice that the proposed design methodology comes at the price of an overhead of area and power. It is important to note that the area and power overheads have two different causes. The first cause is the additional control logic used to support the power protection technique in redundant circuits (the “data keep” block). The “data keep” block for both TMR and DMR redundant circuits implements a memory element, a voter and a multiplexer, which directly increase the occupied silicon area and power consumption. On the other hand, the mentioned hardware extension provides a stable and failure-free operation in a radiation environment. The second cause for the area and power overhead is the implementation of the power network controller (PNC), important for proper operation of all integrated SPS cells. The SPS cell itself does not affect the area overhead due to the mechanism by which it is integrated into an ASIC. This was achieved by an innovative technique to utilize area normally reserved for filler cells, while staying in the standard design flow.

This thesis presents an analysis for finding a compromise between implementation resources and a high protection level, working within standard design tools and standard technologies. Future work should focus on the power network controller (PNC) single event effect analysis related to upset and transient effects and develop a protection mechanism for it. Even then, the PNC is not protected against latchup effects and therefore it must be implemented in a radiation hard technology.

The combination of the work presented in this thesis with the proposed further development of a fault-tolerant PNC will provide a methodology for implementing radiation hard digital designs which are resistant against major radiation effects – SET, SEU, and SEL. Moreover, this methodology will operate completely within the standard ASIC design flow.

Appendix A

Fault-injection library generators

```
#include <stdlib.h>
#include <stdio.h>
#include <time.h>

FILE *vp;

int main( void )
{
int i;
int var;
f1=fopen("<file_name>.txt", "wt");

/* Seed the random-number generator with current time so that
* the numbers will be different every time we run.
*/
srand( (unsigned)time( NULL ) );

/* Display 10 numbers. */
for( i = 0; i < 104729; i++ ) {
var = rand() % <number> + 1;

/* <number> defines range in which random numbers will be generated
* for example: number= 1000 ->


fprintf(vp, "%d\n", var);
}
system("PAUSE");
    return 0;
}
```

Based on the represented C-code it is possible to generate all required libraries. In the following text are provided descriptions of the VHDL test-bench fault-injection functions and blocks.

Conversion functions (VHDL):

```
function to_time( n : integer := 0 ) return time is
begin
      return( n * 1 ps );
end function to_time;

function to_time_ns( x : integer := 0 ) return time is
begin
      return( x * 1 ns );
end function to_time_ns;

function to_time_ns_res( d : integer := 0 ) return time is
begin
      return( d * 100 ps );
end function to_time_ns_res;
```

Part of test-bench used for fault generating:

```
-------------------------------------------
-- ERROR GENERATOR
-------------------------------------------
error_read:process(clk)
       variable row_er : line;
       variable data_er : integer;
       variable count : integer := 0;
begin
       if (clk'event and clk = '1') then
              if count = see_res then
                     new_time <= '1';
                     count := 0;
                     if (read_start = '1') then
                            readline(seul,row_er);
                            read(row_er,data_er);
                     else
                            data_er := 0;
                     end if;
              else
                     count := count + 1;
                     new_time <= '0';
                     data_er := 0;
              end if;
       end if;
       for i in 0 to 14 loop
              if i = data_er and i > 0 then

              -- !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
              if see_sel = '0' then
                     err_inter_1(data_er) <= '1';
              else
                     err_inter_1(data_er) <= '1' after start_seu; -- for SEU
                                                          -- error UNCOMMENT!!!!
              end if;
              -- !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
              else
                     err_inter_1(i) <= '0';
              end if;
       end loop;
end process;

process(err_inter_1)
begin
       for i in 0 to 14 loop
          if (start_set + delta_set) >= clk_per and start_set < clk_per then
              if err_inter_1(i)' event and err_inter_1(i) = '1' then
                     err_inter_2_1(i) <= '1' after start_set;
              elsif err_inter_1(i)' event and err_inter_1(i) = '0' then
              err_inter_2_2(i) <= '1' after (start_set + delta_set - clk_per);
                     else
                            err_inter_2_1(i) <= '0';
                            err_inter_2_2(i) <= '0';
                     end if;
              elsif start_set < clk_per then
                     if err_inter_1(i)' event and err_inter_1(i) = '1' then
                            err_inter_2_1(i) <= '1' after start_set;

                       err_inter_2_2(i) <= '1' after (start_set + delta_set);
                     else
                            err_inter_2_1(i) <= '0';
                            err_inter_2_2(i) <= '0';
```

```
                        end if;
                elsif start_set >= clk_per then
                        if err_inter_1(i)' event and err_inter_1(i) = '0' then
                        err_inter_2_1(i) <= '1' after (start_set - clk_per);

                err_inter_2_2(i) <= '1' after (start_set + delta_set - clk_per);
                        else
                                err_inter_2_1(i) <= '0';
                                err_inter_2_2(i) <= '0';
                        end if;
                end if;
        end loop;
end process;
set_factory:for i in 0 to 14 generate
        err_inter_2(i) <= err_inter_2_1(i) and not(err_inter_2_2(i));
end generate;
--err <= err_inter_1; -- SEU test
--err <= err_inter_2; -- SET test
err <= err_inter_2 when see_sel = '0' else err_inter_1;
-------------------------------------
-------------------------------------

time_to_SEE_read:process(clk)
        variable row_t : line;
        variable data_t : integer := 0;
begin
        if (clk'event and clk = '1') then
                if (read_start = '1' and new_time = '1') then
                        readline(time_to_effect,row_t);
                        read(row_t,data_t);
                        --else
                        --     data_t := 0;
                end if;
        end if;
        if data_t > 0 then
                see_res <= data_t;
        else
                see_res <= 1;
        end if;
end process;

--------------------------
-- set timing random form
--------------------------
set_delta_read:process(clk)
        variable row_sd : line;
        variable data_sd : integer := 0;
begin
        if (clk'event and clk = '1') then
                if (read_start = '1' and new_time = '1') then
                        readline(set_delta_file,row_sd);
                        read(row_sd,data_sd);
                        --else
                        --     data_sd := 0 ps;
                end if;
        end if;
        delta_set_1 <= data_sd;
end process;
delta_set <= to_time(delta_set_1) when read_start = '1' else 0 ps;

seu_start_read:process(clk)
        variable row_sus : line;
        variable data_sus : integer := 0;
begin
        if (clk'event and clk = '1') then
                if (read_start = '1' and new_time = '1') then
                        readline(seu_start_file,row_sus);
```

```
                    read(row_sus,data_sus);
                    --else
                    --     data_sd := 0 ps;
              end if;
       end if;
       start_seu_1 <= data_sus;
end process;
start_seu <= to_time_ns(start_seu_1) when read_start = '1' else 0 ns;

set_start_read:process(clk)
       variable row_sut : line;
       variable data_sut : integer := 0;
begin
       if (clk'event and clk = '1') then
              if (read_start = '1' and new_time = '1') then
                    readline(set_start_file,row_sut);
                    read(row_sut,data_sut);
                    --else
                    --     data_sd := 0 ps;
              end if;
       end if;
       start_set_1 <= data_sut;
end process;
start_set <= to_time_ns_res(start_set_1) when read_start = '1' else 0 ns;
```

Appendix B

VHDL descriptions of shift register and counter used for fault-injection tests

```
-----------------
-- Shift Register
-----------------
library IEEE;
use IEEE.std_logic_1164.all;


entity shift_reg_no_error is
  port(
       clk : in STD_LOGIC;
       data_in : in STD_LOGIC;
       rst : in STD_LOGIC;
       shift_orig : out STD_LOGIC
  );
end shift_reg_no_error;

architecture shift_reg_no_error of shift_reg_no_error is

component flip_flop
  port (
       clk : in STD_LOGIC;
       d : in STD_LOGIC;
       rst : in STD_LOGIC;
       q : out STD_LOGIC
  );
end component;

signal NET371 : STD_LOGIC;
signal NET375 : STD_LOGIC;
signal NET473 : STD_LOGIC;
signal NET530 : STD_LOGIC;
signal NET575 : STD_LOGIC;
signal NET616 : STD_LOGIC;
signal NET629 : STD_LOGIC;

begin

U1 : flip_flop
  port map(
       clk => clk,
       d => NET473,
       q => NET530,
       rst => rst
  );

U2 : flip_flop
  port map(
       clk => clk,
       d => NET530,
       q => NET575,
       rst => rst
  );

U3 : flip_flop
  port map(
```

```
        clk => clk,
        d => NET575,
        q => NET629,
        rst => rst
  );

U4 : flip_flop
  port map(
        clk => clk,
        d => NET629,
        q => NET616,
        rst => rst
  );

ff1 : flip_flop
  port map(
        clk => clk,
        d => data_in,
        q => NET371,
        rst => rst
  );

ff2 : flip_flop
  port map(
        clk => clk,
        d => NET371,
        q => NET375,
        rst => rst
  );

ff3 : flip_flop
  port map(
        clk => clk,
        d => NET375,
        q => NET473,
        rst => rst
  );

ff4 : flip_flop
  port map(
        clk => clk,
        d => NET616,
        q => shift_orig,
        rst => rst
  );

end shift_reg_no_error;

----------------------
-- Counter
----------------------
library IEEE;
use IEEE.std_logic_1164.all;

entity counter_no_error is
  port(
        clk : in STD_LOGIC;
        enable : in STD_LOGIC;
        rst : in STD_LOGIC;
        carry : out STD_LOGIC;
```

```
       count_0 : out STD_LOGIC;
       count_1 : out STD_LOGIC;
       count_2 : out STD_LOGIC;
       count_3 : out STD_LOGIC;
       count_4 : out STD_LOGIC;
       count_5 : out STD_LOGIC;
       count_6 : out STD_LOGIC;
       count_7 : out STD_LOGIC
  );
end counter_no_error;

architecture counter_no_error of counter_no_error is

component flip_flop
  port (
       clk : in STD_LOGIC;
       d : in STD_LOGIC;
       rst : in STD_LOGIC;
       q : out STD_LOGIC
  );
end component;

signal feed_back_0 : STD_LOGIC;
signal feed_back_1 : STD_LOGIC;
signal feed_back_2 : STD_LOGIC;
signal feed_back_3 : STD_LOGIC;
signal feed_back_4 : STD_LOGIC;
signal feed_back_5 : STD_LOGIC;
signal feed_back_6 : STD_LOGIC;
signal feed_back_7 : STD_LOGIC;
signal NET1659 : STD_LOGIC;
signal NET1933 : STD_LOGIC;
signal NET2068 : STD_LOGIC;
signal NET2126 : STD_LOGIC;
signal NET497 : STD_LOGIC;
signal NET624 : STD_LOGIC;
signal out_orig_0 : STD_LOGIC;
signal out_orig_1 : STD_LOGIC;
signal out_orig_2 : STD_LOGIC;
signal out_orig_3 : STD_LOGIC;
signal out_orig_4 : STD_LOGIC;
signal out_orig_5 : STD_LOGIC;
signal out_orig_6 : STD_LOGIC;
signal out_orig_7 : STD_LOGIC;
signal veza_1 : STD_LOGIC;

begin

U1 : flip_flop
  port map(
       clk => clk,
       d => feed_back_1,
       q => out_orig_1,
       rst => rst
  );

veza_1 <= NET1659 and out_orig_3;

count_3 <= out_orig_3;
```

```
U12 : flip_flop
  port map(
       clk => clk,
       d => feed_back_4,
       q => out_orig_4,
       rst => rst
  );

feed_back_4 <= out_orig_4 xor veza_1;

NET1933 <= veza_1 and out_orig_4;

count_4 <= out_orig_4;

U16 : flip_flop
  port map(
       clk => clk,
       d => feed_back_5,
       q => out_orig_5,
       rst => rst
  );

feed_back_5 <= out_orig_5 xor NET1933;

NET2068 <= NET1933 and out_orig_5;

count_5 <= out_orig_5;

feed_back_1 <= out_orig_1 xor NET497;

U20 : flip_flop
  port map(
       clk => clk,
       d => feed_back_6,
       q => out_orig_6,
       rst => rst
  );

feed_back_6 <= out_orig_6 xor NET2068;

NET2126 <= NET2068 and out_orig_6;

count_6 <= out_orig_6;

NET624 <= NET497 and out_orig_1;

count_1 <= out_orig_1;

U5 : flip_flop
  port map(
       clk => clk,
       d => feed_back_3,
       q => out_orig_3,
       rst => rst
  );

feed_back_2 <= out_orig_2 xor NET624;

feed_back_0 <= out_orig_0 xor enable;
```

```
NET497 <= enable and out_orig_0;

feed_back_7 <= out_orig_7 xor NET2126;

carry <= NET2126 and out_orig_7;

count_7 <= out_orig_7;

count_0 <= out_orig_0;

NET1659 <= NET624 and out_orig_2;

count_2 <= out_orig_2;

feed_back_3 <= out_orig_3 xor NET1659;

ff0 : flip_flop
  port map(
       clk => clk,
       d => feed_back_0,
       q => out_orig_0,
       rst => rst
  );

ff2 : flip_flop
  port map(
       clk => clk,
       d => feed_back_2,
       q => out_orig_2,
       rst => rst
  );

ff3 : flip_flop
  port map(
       clk => clk,
       d => feed_back_7,
       q => out_orig_7,
       rst => rst
  );

end counter_no_error;
```

Combined pattern generator used for testing shift register - C source code

```
-----------------------------------------------
-- C source code for combine pattern generator
-----------------------------------------------
#include<stdio.h>
#include <stdlib.h>

int main(int argc, char** argv)
{
if (argc != 2) {
return 200;
}
FILE *fp;
if (fp = fopen("result.txt", "w")) {
int wc = atoi(argv[1]);;
int i = 0;
for (i = 0; i < wc; i++) {
```

```
switch (i) {
case 0:
fprintf(fp, "Wg0: 000000\n");
break;
case 1:
fprintf(fp, "Wg1: 010101\n");
break;
default:
fprintf(fp, "Wg%d: 010", i);
int j = 0;
for (j = 2; j < i+1; j++) {
int k = 0;
for (k = 0; k < j; k++) {
fprintf(fp, "1");
}
for (k = 0; k < j; k++) {
fprintf(fp, "0");
}
//fprintf(fp, " ");
}
for (j = j-1; j > 2; j--) {
int k = 0;
for (k = j; k > 1; k--) {
fprintf(fp, "1");
}
for (k = j; k > 1; k--) {
fprintf(fp, "0");
}
//fprintf(fp, " ");
}
fprintf(fp, "101\n", i, j);
}
}
}
return 0;
}
```

Appendix C

Parser example for automated fault-tolerant design based on the DMR circuit

In the DMR mode all flip-flops are replaced with specially designed DMR flip-flops. As it was mentioned in Chapter 3.2.2., the DMR flip-flop contains the voting system and data-keep logic in order to provide enough high protection level, compared to the standard DMR approach.

Next example shows DMR generating from standard digital circuit. Example is based on the “dfb3d1” flip-flop. Digital circuit after synthesis is presented in following Figure.

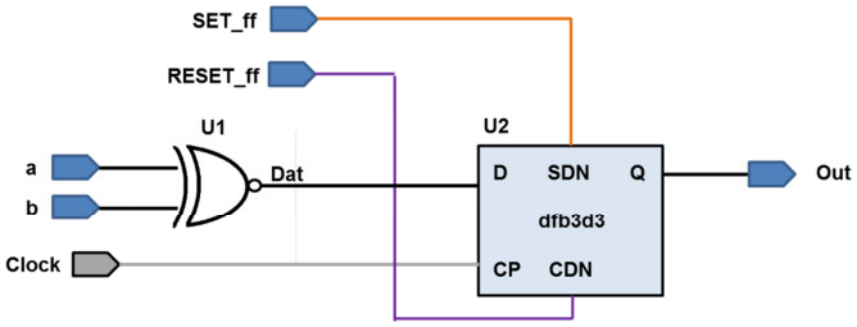


Figure: Digital circuit after synthesis phase

Verilog code of the digital circuit (shown in Figure 5.3) after synthesis is represented in the text below. It is important to notice the Verilog structure in order to follow how the parser is working.

```
// Circuit name and Input/Output Ports
module test_des ( SET_ff, RESET_ff, a, b, Clock, Out );

// Port Declaration
input SET_ff, RESET_ff, a, b, Clock;
output Out;

// Internal Wire Declaration
wire Dat;

// Instances (Standard Cells)
xn21d3 U1 ( .A1(a), .A2(b), .ZN(Dat));
dfb3d3 U2 ( .SDN(SET_ff), .CDN(RESET_ff), .D(Dat), .CP(Clock),
.Q(Out));

endmodule
```

Parser exchanges standard flip-flops with DMR LU flip-flops and provides required interconnections for the data and power supply information between the redundant components (standard cells). The name conversion scheme for the flip-flops defined in the IHP 250 nm standard cell library is shown in the following table:

dfb3d1-2-3	dfb3d1-2-3_dmr
dfc3d1-2-3	dfc3d1-2-3_dmr
dfn3d1-2-3	dfn3d1-2-3_dmr
dfp3d1-2-3	dfp3d1-2-3_dmr

It is easy to notice that for the naming of DMR LU storage cells is used the same approach as for the standard technology with suffix "_dmr" on the end of name.

The Verilog code of "test_des" digital circuit, after parsing in the DMR mode with latchup protection support, is presented in the following text:

```
// Circuit name and Input/Output Ports
module test_des_DMR_LU ( SET_ff, RESET_ff, a, b, Clock, Out );

// Port Declaration
input SET_ff, RESET_ff, a, b, Clock;
output Out;

// Internal Wire Declaration
wire Dat_0;
wire Dat_1;

// Signals for communication between redundant blocks
wire v_out_0;
wire v_out_1;

// Power signals from redundant blocks
wire power_0;
wire power_1;

wire out_inv;

// Instances (Standard Cells)
xn21d3_a U1_0 ( .A1(a), .A2(b), .ZN(Dat_0));
xn21d3_b U1_1 ( .A1(a), .A2(b), .ZN(Dat_1));

dfb3d3_dmr_a U2_0 ( .SDN(SET_ff), .CDN(RESET_ff), .D(Dat_0),
.CP(Clock), .rff_1(v_out_1), .pwr_1(power_1), .Q(v_out_0));
dfb3d3_dmr_b U2_1 ( .SDN(SET_ff), .CDN(RESET_ff), .D(Dat_1),
.CP(Clock), .rff_1(v_out_0), .pwr_1(power_0), .Q(v_out_1));

sp01d1_a fin_voter_Output ( .A(v_out_0), .B(v_out_1), .C(out_inv),
.ZN (out_inv));

in01d3_a U3 (.I(out_inv), .ZN(Out) );

endmodule
```

Block diagram of digital circuit after parsing is shown in Figure 5.4. In Figure 5.4 are not presented power interconnections.

There are two possible ways to provide status information of redundant power domains. As it was described in Chapter 3, the power domain status is very important for correct DMR circuit behave, especially during latchup protection phase. First and easiest way is to use a parser option to generate two extra wires "power_0" and "power_1". In order to avoid conflicts with power supply names it is recommended that signal names are different. Specially designed flip-flops have different suffixes because of the automated placement and routing approach. Both flip-flops are same but power domains are different. In Appendix 5 it is possible to notice that .LEF files have different power supply names.

In this phase is already prepared power separation between redundant circuits. It is known that HDL codes don't involve information about power supply but using the presented trick based on the components naming provides automated physical design with separated power supply lines. Before physical design phase explanation, it is important to provide explanation how memories are protected.

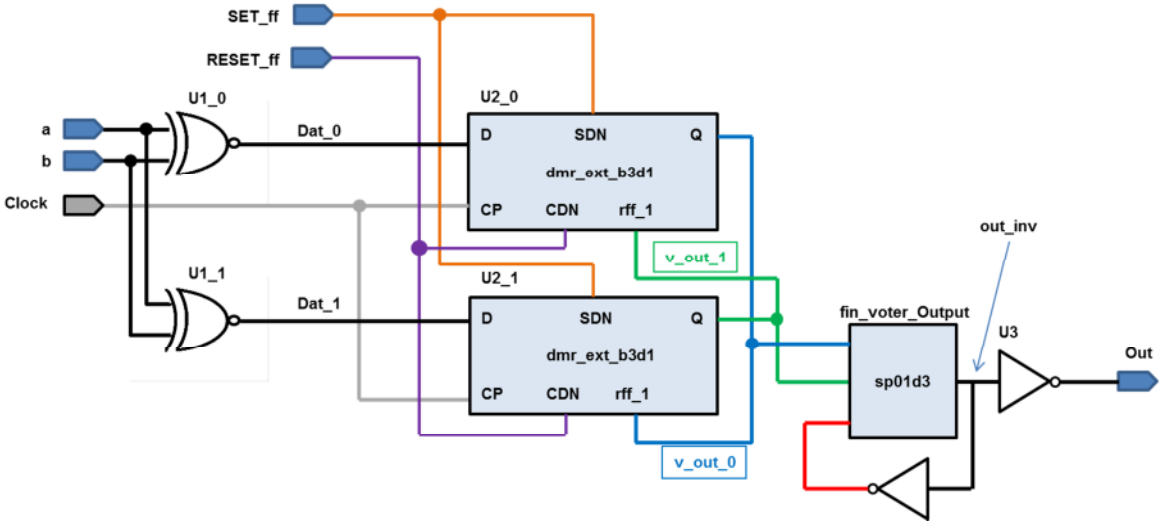
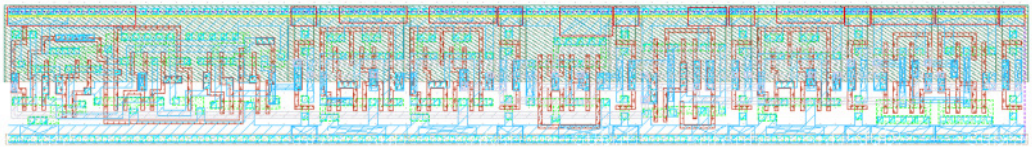


Figure: Block diagram of the DMR digital circuit

DMR Flip-flop implementation details

Layout view of "dfc3d3_dmr" cell:



Area: 627.983 μm²

Power: 27.4 μW

Appendix D

Power network controller (PNC) example for controlling 6 SPS cells (PNC_6)

VHDL code of programmable counter:

```
library IEEE;
use IEEE.STD_LOGIC_1164.all;
use IEEE.std_logic_unsigned. all;


entity pnc_counter is
 port(
        clk : in STD_LOGIC;
        rst : in STD_LOGIC;
        active : in std_logic;
        load : in std_logic;
        DIN : in std_logic_vector(8 downto 0);
        count : out std_logic_vector(8 downto 0)
        );
end pnc_counter;

architecture rtl of pnc_counter is
signal COUNT_INT: STD_LOGIC_VECTOR(8 downto 0);
begin
 process (clk, rst)
 begin
        if rst = '1' then
               COUNT_INT <= (others => '0');
        elsif clk'event and clk='1' then
               if load = '1' then
                      COUNT_INT <= DIN;
               else
                      if active = '1' then
                             COUNT_INT <= COUNT_INT - '1';
                      end if;
               end if;
        end if;
 end process;
 COUNT <= COUNT_INT;

end rtl;
```

Control Circuit (CS) VHDL code:

```
library IEEE;
use IEEE.STD_LOGIC_1164.all;

entity control_circuit is
 port(
        clk : in STD_LOGIC;
        rst : in STD_LOGIC;
        count_val : in STD_LOGIC_VECTOR(8 downto 0); -- current counter state
        TSTART : in STD_LOGIC;     -- latchup detected
        switch_off : in std_logic; -- switch-off current SPS set by CPU
        TSTOP : out STD_LOGIC; -- Counted periode
        Poff : out STD_LOGIC -- Poff interface to SPS
        );
end control_circuit;

architecture rtl of control_circuit is
 signal curr_cnt : std_logic_vector(8 downto 0);
begin
 p0 : process(clk, rst) is
```

```
begin
        if rst = '0' then
                tstop <= '0';
                Poff <= '0';
                curr_cnt <= (others => '0');
        elsif clk'event and clk = '1' then
                if tstart = '1' then
                        curr_cnt <= count_val;
                        tstop <= '0';
                        poff <= '0';
                elsif curr_cnt = count_val then
                        curr_cnt <= count_val;
                        tstop <= '1';
                        poff <= '0';
                elsif switch_off = '1' then
                        poff <= '1';
                else
                        tstop <= '0';
                        Poff <= '0';
                        curr_cnt <= (others => '0');
                end if;
        end if;
 end process;
end rtl;
```

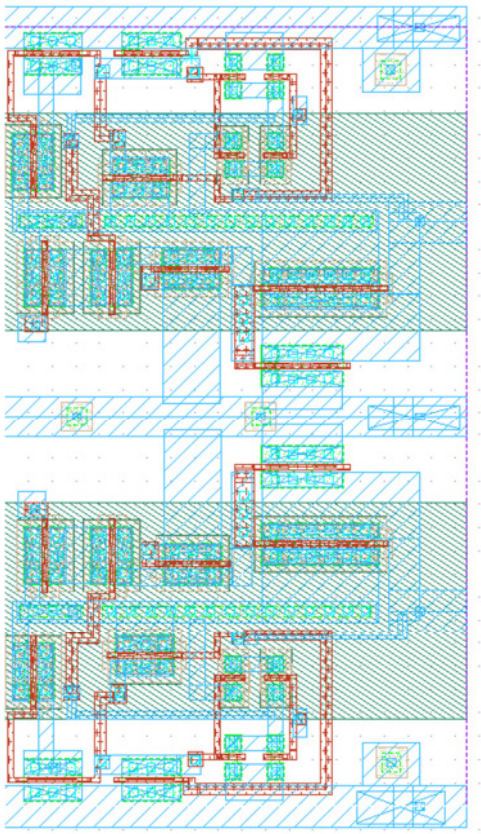
Estimated silicon area after layout for PNC_6 architecture: 3125.8 µm²
Estimated power consumption for PNC_6 architecture: 0.1195 mW

Appendix E

SPS_pn5 (four-row cell)

SPS_pn5 consists of two SPS cells – for VDD_a and VDD_b redundant power supplies. Driver transistor width is 5 µm.

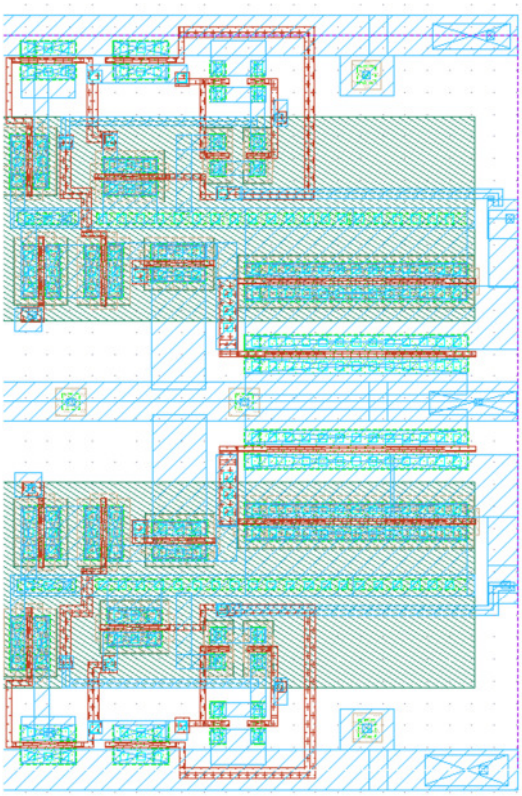
Layout view:



SPS_pn10 (four-row cell)

SPS_pn10 consists of two SPS cells – for VDD_a and VDD_b redundant power supplies. Driver transistor width is 10 µm.

Layout view:



Cadence CPF file for SPS row insertion:

```
set_cpf_version 1.1

##############################
# Technology Part
##############################

define_library_set -name set25_wc -libraries
{/home/petrovic/design/LayoutDes/test_circuit/output/ihp25_slow_a.lib \
 /home/petrovic/design/LayoutDes/test_circuit/output/ihp25_slow_b.lib \
 /home/petrovic/design/LayoutDes/test_circuit/output/dmr_ff_extension_a_slow.lib \
 /home/petrovic/design/LayoutDes/test_circuit/output/dmr_ff_extension_b_slow.lib \
 /home/petrovic/design/SPS_characterisation/lib/SPS_switch.lib}

define_library_set -name set25_bc -libraries
{/home/petrovic/design/LayoutDes/test_circuit/output/ihp25_fast_a.lib \
 /home/petrovic/design/LayoutDes/test_circuit/output/ihp25_fast_b.lib \
 /home/petrovic/design/LayoutDes/test_circuit/output/dmr_ff_extension_a_fast.lib \
 /home/petrovic/design/LayoutDes/test_circuit/output/dmr_ff_extension_b_fast.lib \
 /home/petrovic/design/SPS_characterisation/lib/SPS_switch.lib}


set_design leon_top

create_power_domain -name PD1 -default

create_power_domain -name PD2 -instances {processor} -shutoff_condition {Poff_1 |
Poff_2} -base_domains PD1

create_nominal_condition -name off -voltage 0
create_nominal_condition -name on -voltage 2.5

create_power_mode -name PM1 -domain_conditions {PD1@on PD2@on} -default
create_power_mode -name PM2 -domain_conditions {PD1@on PD2@off}



update_nominal_condition -name on -library_set set25_wc

##############################
# Synthesis Related
##############################

update_power_mode -name PM1 -sdc_files
{/home/petrovic/design/MW_switch_VP/report/leon.sdc}
update_power_mode -name PM2 -sdc_files
{/home/petrovic/design/MW_switch_VP/report/leon.sdc}


##############################
# Layout Related
##############################

create_power_nets -nets vdd! -voltage 2.5
create_power_nets -nets {VDD_a VDD_b} -internal
create_ground_nets -nets gnd! -voltage 0

create_global_connection -domain PD2 -net VDD_a -pins VDD_a
create_global_connection -domain PD2 -net VDD_b -pins VDD_b

update_power_domain -name PD1 -primary_power_net vdd! -primary_ground_net gnd!
update_power_domain -name PD2 -primary_power_net VDD_a -primary_ground_net gnd! -
equivalent_power_nets VDD_b

end_design
```

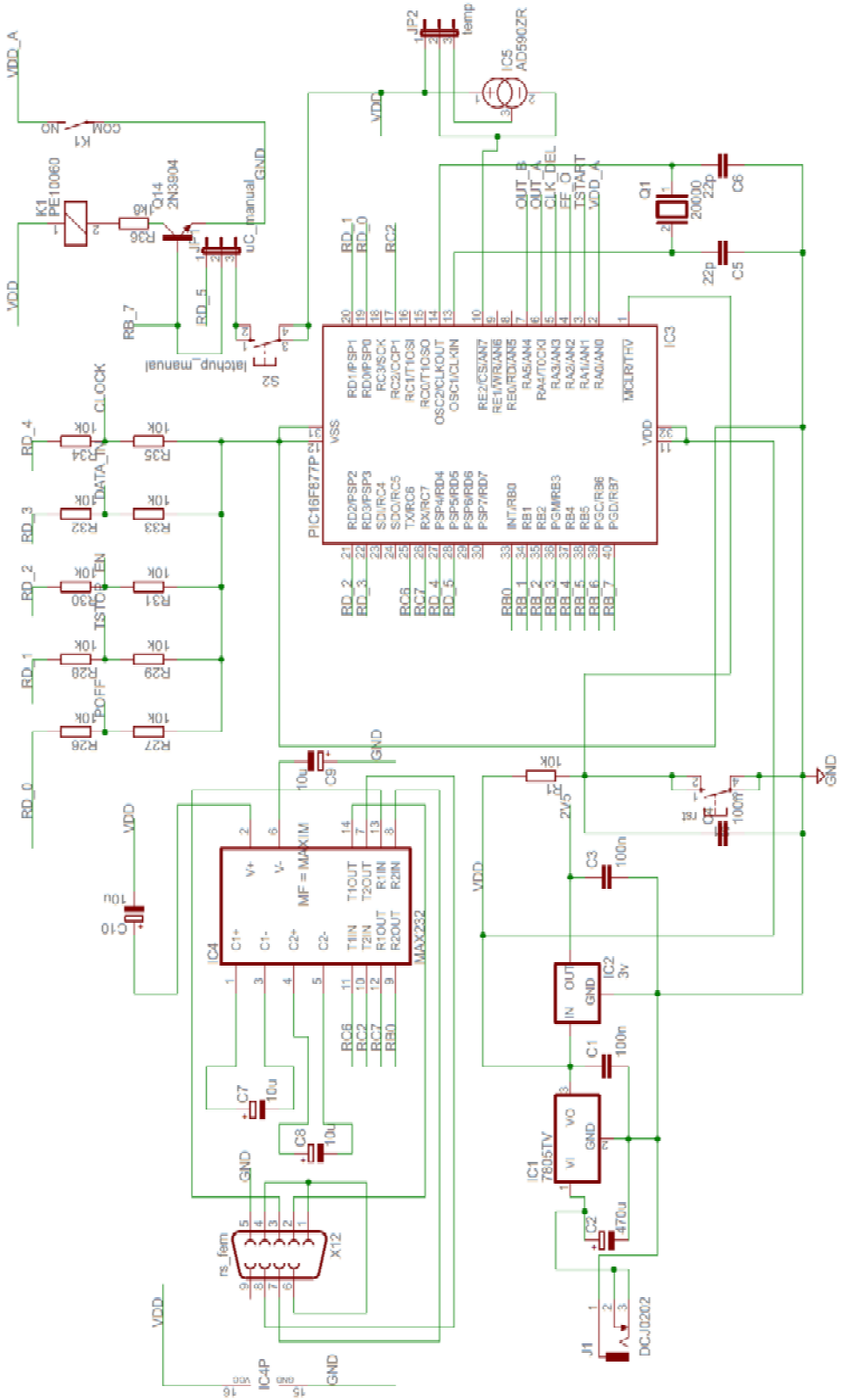
PIC Microcontroller C code (16F877A) for automated Latchup Test

```
/*
 * Project name:

     SEL_control

     Firmware for PIC 16F877A with main purpose to drive and to control Latcup
     control circuit produced in IHP GmbH, with standard 250nm process.

     Control circuit triggers measurement devices in the moment of latchup and
     provides required synchronization for correct measurements.

 * Copyright:

     (c) IHP GmbH
     Dipl. Ing. Vladimir Petrovic
     Im Technologiepark 25
     15236 Frankfurt Oder
     Germany

 * Revision History:
     13072011:
       - Initial release;
 * Description:
     System for measurement and acquisition data from LUT-03, latchup protection
     for ASIC components.
 * Test configuration:
     MCU:             P16F877A
     Dev.Board:       EasyPIC4
     Oscillator:      HS, 08.0000 MHz
     Ext. Modules:
     SW:              mikroC v8.2.0.0
 * Notes:
     -----------------------------------------------------------------------
     NOTE: THIS WORK IS DONE FOR PhD THESIS DEFENCE FOR SPACE MICROELECTRONICS
     COMPONENTS AND IT IS CONFIDENTIAL!
     -----------------------------------------------------------------------
*/

//Declarations-----------------------------------------------------------------
// declaration of data for sending on RS232

//unsigned i;//brojac

/*void interrupt() {
            if (INTCON.TMR0IF) {
            //cnt++;                    // Increment value of cnt on every interrupt
            TMR0   = 0xEA;
            INTCON = 0xA0;           // Set T0IE, clear T0IF, enable port B
interrupts
             if (portd.f4 == 1) {
                   PORTD.f4 = 0;      // Toggle PORTB LEDs
                   }
                   else {
                   PORTD.f4 = 1;
                   }
            }
            }
*/


void main(){
//ADCON1 = 9; //ADC setup -> ports A0-A5 are analog, A6-A7 are digital and VDD is
Vref
TRISA = 0xFF; //port A direction defined as inputs
```

```
TRISB = 0xFF; //port B is defined as input from SPC board (after voltage
conversion)
USART_Init(19200); //UART is initialized on port C
PORTD = 0;
TRISD = 0x00;// port d is defined as output port
trisc = 0b10000001; //C7 and C0 are inputs - others are outputs on portC
PORTD.f4 = 0; // triger signal for osci and acquiring equipment
PORTD.f0 = 0; // low active switch controll signal -> when 0 controlled supply is
0[V]
//PORTB.f1 = 1; // low active signal from circuit for latchup relax start TSTART
PORTD.f1 = 0;// high active signal to stop protection TSTOP

//cnt = 0;

//option_reg = 0x80; // prescaller is assigned to TMR0
//TMR0 = 0xEA; // initial value for TMR0

//INTCON = 0xA0; //enabled interrupt on portb(7:4) changing values and TMR0
interrupt enabled
first_step:
           while(1){// if signal for shortcut is detected
              delay_ms(5000);
                   //if (portb.f1 == 0) {// is shortcut recognized - should be 0
                   //   portd.f1 = 0;//tstop deactivated for sure
                       goto latchup_time;// go to shortcut routine
                   //   }
                   //else{
                   //    goto first_step;// until circuits respond
                   //}
                   }
           //else {
           //goto first_step;
           //}



standard_condition:
       while(1) {
                if (portb.f7 == 1){// if signal for shortcut is detected
                      portd.f4 = 1;// strob active, recording start
                      portd.f1 = 0;// TSTOP inactive
activate_latchup:
                      if (portb.f1 == 0) {
                            portd.f4 = 1;// strob active, recording start
                            portd.f1 = 0;
                            portd.f5 = 0;
                           goto latchup_time;
                           }
                      else
                          goto activate_latchup;
                      }
                else
                    {
                    portd.f1 = 0;
                    delay_ms(5000);
                    portd.f5 = 1;// automated latchup signal
                    portd.f4 = 1;// strob active, recording start
                    goto standard_condition;
                    }
                }

latchup_time:
              //if (portb.f1 == 0){
              //latchup long survive test
                 portd.f4 = 1;// strob active, recording start
                 portd.f5 = 1;
```

```
   portd.f1 = 0;// TSTOP inactive

   delay_ms(10000);
   portd.f5 = 0;// automated latchup deactivated
   delay_us(2);
   portd.f4 = 0;// strob deactivated
   delay_us(500);   //time "d"
   portd.f1 = 1;// tstop signal sent
   delay_us(15); // time "g"
   goto first_step;
//   }
//else
//    goto latchup_time;
}
```

SPS Test Board Schematic



References

[AGI08] Agilent Technologies, MSO6104A Mixed Signal Oscilloscope: 1 GHz, 4 scope and 16 digital channels, Available: http://www.home.agilent.com

[AMB13] Advanced Microcontroller Bus Architecture, AMBA, Available: http://www.arm.com/products/system-ip/amba/amba-open-specifications.php

[BAZ00] M. P. Baze, S. P. Buchner, D. McMorrow, "A Digital CMOS Design Technique for SEU Hardening", IEEE Transaction on Nuclear Science, Vol. 47, No. 6, pp. 2603-2608, December 2000

[BEN03] A. Benso and P. Prinetto, "Fault injection techniques and tools for embedded systems reliability evaluation", Kluwer Academic Publishers, Dordrecht 2003

[BOL07] C. Bolchini, A. Miele, and M. D. Santambrogio, "TMR and Partial Dynamic Reconfiguration to Mitigate SEU Faults in FPGAs", Proc. 22nd IEEE International Symposium on Defect and Fault-Tolerance in VLSI Systems, Rome (Italy) 2007, pp.87-95

[BRE05] C.J. Brennan, K. Chatty, J. Sloan, P. Dunn, M. Muhammad, R. Gauthier, "Design automation to suppress cable discharge event (CDE) induced latchup in 90nm CMOS ASICs", Electrical Overstress/Electrostatic Discharge Symposium, 2005. EOS/ESD '05.

[CAD13] Cadence Custom IC Design – Circuit Design, Available: http://www.cadence.com/products/cic/Pages/default.aspx

[CAR99] C. Carmichael, E. Fuller, M. Caffrey, "SEU Mitigation Techniques for Virtex FPGAs in Space Application", MAPLD Conference 1999, Available: http://www.xilinx.com/appnotes/VtxSEU.pdf

[CHA01] K. Chakraborty, S. Kulkarni, M. Bhattacharya, P. Mazumder, Fellow and Anurag Gupta, "A Physical Design Tool for Built-In Self-Repairable RAMs", Very Large Scale Integration (VLSI) Systems, IEEE Transactions on (Volume: 9, Issue: 2, Pages: 352-364), IBM – IEEE 2001

[CHR05] D. Christiansen, C. K. Alexander, "Standard Handbook of Electrical Engineering (5th ed.)", McGraw-Hill, ISBN 0-07-138421-9, 2005

[CHU06] P. P. Chu, "RTL Hardware Design Using VHDL – Codding for Efficiency, Portability, and Scalability", Cleveland State University, John Wiley & Sons, Inc., ISBN-13: 978-0-471-72092-8, 2006

[CPF13] Cadence Low Power Implementation, Available: http://www.cadence.com/products/fv/pages/lp_flow.aspx

[DRE02] N. Dressnandt, M. Newcomer, O. Rohne, S. Passmore, "Radiation hardness: design approach and measurements of the ASDBLR ASIC for the ATLAS TRT", Nuclear Science Symposium Conference Record, (pp. 151-155), 2002 IEEE

[ESA02] Single Event Effects Test Method and Guidelines, ESCC Basic Specification No. 25100, 2002, Available: https://escies.org/webdocument/showArticle?id=229

[ESA10] ECSS-E-HB-10-12A, Space engineering, Calculation of radiation and its effects and margin policy handbook, ECSS Secretariat, Noordwijk, The Netherlands, 2010

[ESA12] Microelectronics Technologies for Space, ESA Microelectronics, Europe Space Agency, 15.10.2012 Available: http://www.esa.int/TEC/Microelectronics/SEMSTCV681F_0.html

[EST78] D.B. Estreich, A. Ochoa, and R.W. Dutton, “An Analysis of Latchup Prevention in CMOS IC’s Using Epitaxial Buried Layer Process”, International Electron Device Meeting, 1978, pp. 76-84

[GAL11] R. Gaillard, M. Nicolaidis (ed.), Soft Errors in Modern Electronic Systems, Frontiers in Electronic Testing 41, DOI 10.1007/978-1-4419-6993-4_2, 2011

[GNU13] GnuWin packages, http://gnuwin32.sourceforge.net/packages.html

[GOE08] M. Richter, K. Oberlaender, M. Goessel, “New Linear SEC-DED Codes with Reduced Triple Bit Error Miscorrection Probability”, 14th IEEE International On-Line Testing Symposium, IOLTS08, Page(s): 37 - 42, Digital Object Identifier: 10.1109/IOLTS.2008.27, 2008

[HAM98] S. N. Hamilton, A. Orailoglu, “Efficient self-recovering ASIC design”, Design & Test of Computers, (Volume: 15, Issue: 4, Pages: 25-35), IEEE 1998

[HAO08] L. Hao, L. Yu, “A Study on the Hardware Implementation of EDAC”, Third International Conference on Convergence and Hybrid Information Technology, ICCIT08, Volume: 2, Page(s): 222 – 225, Digital Object Identifier: 10.1109/ICCIT.2008.14, 2008

[HAS05] A. Hastings, “The Art of Analog Layout - Second Edition”, Chapter 4 – Failure Mechanisms, Page: 171, ISBN-10: 0131464108, 2005

[HOS02] A. Holmes-Siedle and L. Adams, “Handbook of Radiation effects, second edition”, Oxford University Press, ISBN 978-0-19-850733-8, 2002

[HSI70] Hsiao, M.Y., “A Class of Optimal Minimum Odd-weight-column SEC-DED Codes”, IBM Journal of Research and Development, Volume: 14, Issue: 4, Page(s): 395 – 401, Digital Object Identifier: 10.1147/rd.144.0395, 1970

[HSU97] Mc. Hsueh, T. K. Tsai, R. K. Iyer, “Fault Injection Techniques and Tools”, IEEE Computer, Vol. 30, No. 4, April 1997, pp. 75-82.

[IEE13] “IEEE Standards”, http://www.eda.org/sdf/

[IHP13] IHP GmbH, Leibniz Institute Frankfurt Oder, Available: www.ihp-microelectronics.com

[INI11] K. Iniewski (Ed.), “Radiation Effects in Semiconductors”, CRC Press, Boca Raton, 2011.

[JAV12] A. Javanainen, “Particle radiation in microelectronics”, Academic Dissertation for the Degree of Doctor of Philosophy, Department of Physics, University Jyväskylä, Finland, 2012

[JIA07] D. Jia, “A Framework on Mitigating Single Event Upset using Delay-Insensitive Asynchronous Circuits”, Region 5 Technical Conference, April 20-21, Fayetteville, AR, 1-4244-1280-3/07/$25.00 ©2007 IEEE

[JON10] D. H. Jones, R. McWilliam and A. Purvis, “Design of a self-assembling, repairing and reconfiguring Arithmetic Logic Unit”, University of Durham England, 2010, www.intechopen.com

[JUG07] Y.-K. Jung, “Fault-recovery Non-FPGA-based Adaptable Computing System Design”, Second NASA/ESA Conference on Adaptive Hardware and Systems (AHS 2007) 0-7695-2866-X/07 (pp. 709-716), 2007 IEEE

[KHA93] A. A. Khalid, Master thesis, “Reliability Analysis of Triple Modular Redundancy System with Spare”, Department of Computer Engineering, Rochester, New York, December 1993

[KOL33] A. N. Kolmogorov, “Grundbegriffe der Wahrscheinlichkeitsrechnung“, Julius Springer, Berlin 1933

[KOR79] K. Israel, Y. H. Stephen, “Reliability Analysis of N-Modular Redundancy Systems with Intermittent and Permanent Faults”, IEEE Transactions on Computers, vol. c-28, No. 7, July 1979

[LAC08] R. C. Lacoe, “Improving integrated circuit performance through the application of hardness-by-design methodology”, IEEE Transaction on Nuclear Science, vol.55, pp.1903-1925, 2008.

[LAL03] P. K. Lala, “A single error correcting and double error detecting coding scheme for computer memory systems”, Proc. 18th IEEE International Symposium on Defect and Fault Tolerance in VLSI Systems, Cambridge (USA) 2003, (pp.235-241)

[LAY97] P.J. Layton, D. R. Czajkowski, J. C. Marshall, H. F. D. Antony, R. W. Boss, “Single event latchup protection of integrated circuits”, Radiation and Its Effects on Components and Systems, 1997, RADECS 97, 327-331, ISBN: 0-7803-4071-X

[LEV09] J. R. Levin, “Flex & Bison – Unix Text Processing Tools”, Published by O’Reilly Media, Inc., ISBN: 978-0-596-15597-1, 2009

[LIM12] D. B. Limbrick, W. H. Robinson, “Characterizing single event transient pulse widths in an open-source cell library using SPICE, Proceeding: IEEE Workshop on Silicon Errors in Logic – System Effects (SELSE 2012), At Urbana, IL

[LIT96] V. Litovski, “Osnovi Elektronike” (Translation: The Fundamental Electronics), 2006, Nis, Serbia

[LYO62] R. E. Lyons, W. Vanderkulk, “The Use of Triple-Modular Redundancy to Improve Computer Reliability”, IBM Journal April 1962

[LYE09] L. Ye, G. Xiaohan, X. Weiwei, H. Zhiliang, and D. Killat, “An Experimental Extracted Model for Latchup Analysis in CMOS Process”, Proc. 8th IEEE International Conference on ASIC, Hunan (China) 2009, pp.1035-1038

[MAK03] G. K. Maki, P. S. Yeh, “Radiation tolerant ultra-low power CMOS microelectronics: Technology development status”, Proc. NASA Earth Science Technology Conference, Hyattsville (USA) 2003, (pp.1-4)

[MAX13] Maxwell Technologies, “Latchup Protection Technology™ (LPT) Overview”, Available: http://www.maxwell.com/products/microelectronics/latchup-protection

[MCH97] H. Mei-Chen, T. K. Tsai, and R. K. Iyer, “Fault injection techniques andtools”, IEEE Computer, Vol.30, No.4, pp.75-82, 1997.

[MES82] G. Messenger, “Collection of charge on junction nodes from ion tracks,” Nuclear Science, IEEE Transactions on, vol. 29, no. 6, pp. 2024–2031, 1982.

[MNT13] Mentor Graphics design tools, Available: www.mentor.com/products

[MON9a] S. Montenegro, E. Haririan, “A Fault-Tolerant Middleware Switch for Space Applications”, Third IEEE International Conference on Space Mission Challenges for Information Technology, 2009. SMC-IT09, Page(s): 333 – 340, IEEE, 2009

[MON9b] S. Montenegro, “Network Centric Core Avionics”, Advances in Satellite and Space Communications, SPACOM 2009, Page(s): 197 - 201

[MON10] S. Montenegro, V. Petrovic, and G. Schoof, “Network Centric Systems for Space Application,” Advances in Satellite and Space Communications, SPACOM 2010,Page(s): 146 - 150

[NEU56] J. Von Neumann, “Probabilistic Logics”, Automata Studies, Princeton University Press, 1956

[NIC06] M. Nicolaidis, “A Low-Cost Single-Event Latchup Mitigation Scheme”, Proceedings of the 12th IEEE International On-Line Testing Symposium (IOLTS'06), 0-7695-2620-9/06 $20.00, © 2006 IEEE

[NIC11] M. Nicolaidis, “Soft Errors in Modern Electronic Systems”, volume 41, Springer 2011

[PET10] V. Petrovic, G. Schoof „Design Flow Approach for Reliable ASIC Designs“, the 7th International New Exploratory Technologies Conference (NEXT 2010), Turku, October 19 - 21, 2010, Finland

[PET11] V. Petrovic, M. Ilic, G. Schoof, S. Montenegro, “Implementation of middleware switch ASIC processor”, 19th Telecommunications Forum, Page(s): 574 – 577, Digital Object Identifier: 10.1109/TELFOR.2011.6143613, 2011

[PET12a] V. Petrovic, M. Ilic, G. Schoof, and S. Montenegro, “Implementation of Middleware Switch ASIC Processor”, IEEE Telfor Journal, Vol. 4, No. 2, 2012

[PET12b] V. Petrovic, M. Ilic, G. Schoof, Z. Stamenkovic “SEU and SET Fault Injection Models for Fault Tolerant Circuits”, Proc. of the 13th Biennial Baltic Electronics Conference (BEC2012), (2012)

[PET12c] V. Petrovic, M. Ilic, G. Schoof, Z. Stamenkovic, “Design Methodology for Fault Tolerant ASICs”, Proc. of the 15th IEEE Symposium on Design and Diagnostics of Electronic Circuits and Systems Symposium (DDECS 2012), 8 (2012)

[RAB03] J. M. Rabaey, A. Chandrakasan, B. Nikolic, “Digital Integrated Circuits (2nd Edition)” ISBN 978-0-13-090996-1, 2003

[RAC12] Reliability Analysis Center (RAC), Selected Topics in Assurance Related Technologies, “Understanding Series and Parallel Systems Reliability”, Volume 11, Number 5, Available: http://rac.alionscience.com

[RAG02] H. Ragaie, S. Kayed, “Impact of CMOS device scaling in ASICs on radiation immunity”, 2002. (EWAED) - The First Egyptian Workshop on Advancements of Electronic Devices, 2002

[RAS11] M. H. Rashid, “Power Electronics (3rd ed.)”, Pearson, ISBN 978-81-317-0246-8, 2011

[ROL03] N. Rollins, M. Wirthlin, P. Graham, M. Caffrey, “Evaluating TMR Techniques in the Presence of Single Event Upsets”, Military and Aerospace Programmable Logic Devices International Conference, Washington DC, USA, 2003

[SAM04] P. K. Samudrala, J. Ramos, and S. Katkoori, “Selective Triple Modular Redundancy (STMR) Based Single-Event Upset (SEU) Tolerant Synthesis for FPGAs”, IEEE Transactions on Nuclear Science, Vol.51, No.5, pp.2957-2969, 2004.

[SAN03] A. Sangiovanni-Vincentelli, “The tides of EDA”, Design and Test of Computers, Volume: 20, Issue: 6, Pages: 59–75, IEEE, 2003.

[SCE06] L. Scheffer, L. Lavango, G. Martin, “Electronic Design Automation for Integrated Circuits Handbook”, Published in 2006 by CRC Press Taylor & Francis Group, ISBN: 978-0-8493-7924-6

[SCH09] G. Schoof, M. Methfessel, and R. Kraemer, “Fault-tolerant ASIC design for high system dependability”, Proc. 13th International Forum on Advanced Microsystems for Automotive Applications, Berlin, Germany, 2009, pp. 369-382.

[SCH12] G. Schoof, V. Petrovic, S. Montenegro, “Systemarchitektur für Raumfahrtanwendungen”, Proc. 24. GI/GMM/ITG-Workshop Testmethoden und Zuverlässigkeit von Schaltungen und Systemen, 29 (2012)

[SHA09] A. Sharma and B. Singh, “Simulation of Fault Injection of Microprocessor System using VLSI Architecture System”, TENCON 2009, 978-1-4244-4547-9/09/$26.00 ©2009 IEEE

[SHE08] W. Sheng, L. Xiao, Z. Mao, “An Automated Fault Injection Technique Based on VHDL Syntax Analysis and Stratified Sampling”, 4th IEEE International Symposium on Electronic Design, Test & Applications, (pp.587-591), DOI 10.1109/DELTA.2008.36, ©2008 IEEE

[SHI08] K. Shinohara and M. Watanabe, “A double or triple module redundancy model exploiting dynamic reconfigurations”, NASA/ESA Conference on Adaptive Hardware and Systems, 978-0-7695-3166-3/08, 2008 IEEE DOI 10.1109/AHS.2008.67, (pp.114-121)

[SIM12] A. Simevski, E. Hadzieva, R. Kraemer, M. Krstic, “Scalable Design of a Programmable NMR Voter with Inputs State Descriptor and Self-Checking Capability”, Proc. NASA/ESA Conference on Adaptive Hardware and Systems (AHS 2012), 182 (2012)

[SRI13] J. F. Ziegler, SRIM – The Stopping and Range of Ions in Matter Simulator, Available: http://www.srim.org/

[STE06] L. Sterpone, M. Violante, “Hardening FPGA-based systems against SEUs: A new design methodology”, Journal of Computers, Vol. 1, NO. 1, April 2006, Academy Publisher

[SYN13] Synopsys Accelerating Innovations, Available: www.synopsys.com/tools

[SZC07] R. Szczygiel, “ASIC for HEP (High Energy Physics) – detector systems of several millions of measurement channels”, Measurement Science and Technology, 2007, IOP Publishing

[TAM13] Radiation Effects Facility, the Cyclotron Institute, MS #3366 / College Station, TX 77843, Available: http://cyclotron.tamu.edu/ref/index.php

[TAR11] J. Tarrillo, R. Chipana, E. Chiele, F. L. Kastensmidt, “Designing and Analyzing a SpaceWire Router IP for Soft Error Detections” Designing and analyzing a SpaceWire router IP for soft errors detection Test Workshop (LATW), (pp.1-6), 2011 12th Latin American, 2011

[TEI08] J. Teifel, “Self-Voting Dual-Modular-Redundancy Circuits for Single-Event-Transient Mitigation”, IEEE Transactions on Nuclear Science, VOL. 55, NO. 6, (pp. 3435-3439), December 2008

[TKS82] T. K. Shridharbahai, “Probability and Statistics With Reliability, Queuing, And Computer Science Applications”, Prentice Hall, India (1982)

[TRO83] R.R. Troutman, “Epitaxial Layer Enhancement of n-Well Guard Rings for CMOS Circuits”, IEEE Electron Devices, 1983, (pp. 438-440).

[TRO86] R.R. Troutman, “Latchup in CMOS Technology: The Problem and its Cure”, Kluwer Publications, Boston, MA, 1986

[VIR5Q] Space-grade rad-hard Virtex-5QV FPGA http://www.xilinx.com/products/ silicondevices/fpga/virtex-5qv/index.htm

[VOL08] S. H. Voldman, “Latchup”, Wiley; 1 edition, ISBN-10: 0470016426, 2008

[WES11] N. H. E. Weste and D. M. Harris, “CMOS VLSI design: A circuits and systems perspective”, Fourth Edition, Addison-Wesley, Boston, 2011.

[WIR03] M. Wirthlin, E. Johnson, N. Rollins, M. Caffrey, and P. Graham, “The Reliability of FPGA Circuit Designs in the Presence of Radiation Induced Configuration Upsets”, Proc. 11th Annual IEEE Symposium on Field-Programmable Custom Computing Machines, Napa CA (USA) 2003, pp.133-142

[YAN12] I. Yang, S. H. Jung, K. H. Cho, “Self-Repairing Digital System with Unified Recovery Process Inspired by Endocrine Cellular Communication”, Very Large Scale Integration (VLSI) Systems, IEEE Transactions on, ISSN: 1063-8210 ©2012 IEEE

List of acronyms

AHB	Advanced high performance bus
AMBA	Advanced microcontroller bus architecture
APB	Advanced peripheral bus
ASIC	Application specific integrated circuit
BIST	Build-in-self-test
BJT	Bipolar Junction transistor
CMOS	Complementary metal-oxide semiconductor
CPF	Common power format
CR	Control register
CS	Control circuit
CSD	Current sensor/driver
DLR	Deutsches Zentrum für Luft- und Raumfahrt
DMA	Direct memory access
DMR	Dual modular redundancy
DRC	Design rule check
DTMR	Double/triple modular redundancy

EDA	Electronic design automation
EDAC	Error detection and correction
ESA	Europe space agency
ESL	Electronic system level
FF	Flip-flop
FI	Fault-injection
FIFO	First-in first-out
FPL	Fault position library
FT	Fault-tolerant
GDSII	Graphic database system 2
GND	Ground
HDL	Hardware description language
HEP	High-energy physics
HSECT	HIT Soft Error Characterization Toolkit
IC	Integrated circuit
LPT	Latchup protection technology
LUT	Latchup test
LVS	Layout vs. Schematic
MTFB	mean time between failures

MW	middleware switch
NASA	National Aeronautics and Space Administration
NMOS	N type metal-oxide semiconductor
NMR	N-modular redundancy
PCB	Printed circuit board
PMOS	P type metal-oxide semiconductor
PNC	Power network controller
RH	Radiation hardened
RISC	Reduced instruction set computer
RT	Register transfer
RTL	Register transfer level
S3P	Simple synchronous serial protocol
SCAN	Spacecraft area network
SCR	Silicon-controlled rectifier
SDF	Standard delay format
SEE	Single event effect
SEL	Single event latchup
SET	Single event transient
SEU	Single event upset

SOC	System on chip
SPS	SEL power switch
SR	Shift register
SRIM	Stopping and Range of Ions in Matter
TDL	Transient duration library
TID	Topic identifier
TMR	Triple modular redundancy
TSL	Transient starter library
UART	Universal Asynchronous Receiver Transmitter
USL	Upset starter library
VCSW	Voltage-controlled switch
VHDL	Very high speed hardware description language

Bibliography

Vladimir Petrovic, born December 13, 1982 in Pozarevac, Serbia, Yugoslavia received his Diploma Engineering degree in 2006 from the Faculty of Electronic Engineering in Nis, University Of Nis, Serbia, Yugoslavia. Following his graduation, he continued working at the Integration Microsystems Austria as junior researcher in the area of medicine electronics applications. From 2008 he works as a scientist in the System Department of IHP, Frankfurt (Oder), Germany. Research areas are aerospace microelectronics, radiation effects on microelectronics, development of techniques for radiation effects mitigation on system and circuit level.