# Proposal of a functional safety methodology applied to fault tolerance in FPGA applications

Bruna F. Flesch, Bianca Brand, Rodrigo Marques de Figueiredo, Lúcio Rene Prade and Márcio Rosa da Silva
Polytechnic School
University of Bells River Valley
950 Unisinos ave - São Leopoldo / RS - Brazil
Email: brunaf.flesch@gmail.com, bianca_sb@hotmail.com, {marquesf, luciorp, marcior}@unisinos.br

***Abstract*—This paper proposes a functional safety methodology applied to fault tolerance in FPGA applications by evaluating a case study in digital signal processing (DSP) field. This study is based on the safety standard IEC 61508 that states a safe system must be capable of detecting faults and it should have a safety mode. So, functional evaluations of a configurable DSP module were needed to define this methodological proposal. Besides, identification of its critical paths and proper definition of safety architectures were done. The case study was developed in configurable hardware. Furthermore, due to the influences of external factors, such as temperature, electromagnetic fields, among others, together with the advance of technology that reduces the components' size, entails in a much bigger occurrence of faults in a circuit. Hence, preliminary results from the use of this methodology indicated a reduction of up to 95 % in Single Event Transients (SETs) occurrence as well as a hardware area increase from 60 % to 163 %.**

## I. Introduction

According to [1], Digital Signal Processing (DSP) was first used in the 1960s and is an important technology to both science and engineering. Nowadays, safe DSP applications are largely used in several fields. Some examples are listed below: in space applications to improve quality of pictures obtained; in the medical field to diagnosis by image, ECG analysis, medical image storage; in commerce for special effects in movies and video calls; in military applications for radars, sonars and secure communication; in scientific field for simulation, modeling, data acquisition; in the industrial field to control and test processes. All these areas present terrestrial or space applications that may require fault tolerance, functional safety or even both.

It might be mentioned that dependability of a system will strongly depend on the application for which it was developed. Due to the advance of technology and device shrinkage, circuit components are getting smaller, which results in a higher vulnerability towards particles that reach a system [2]. The impact of a charged particle in combinational elements may result in a Single Event Transient (SET). These SETs may be propagated and stored in memory elements, producing Single Event Upsets (SEU) [3]. This fact coupled with influence of external factors increase chances of faults in a system. External factors are examplified as temperature, pressure and electromagnetic field.

Assuring the safety and reliability of the circuit in most basic elements and signals, making the process less expensive and more effective and improving use of resources will be obtained from this proposal. So, the purpose of this study is to present such methodology, and furthermore, to also develop configurable and flexible codes.

This paper is organized as follows: in the second section safety standards are addressed by presenting relevant definitions and fault tolerance techniques. The third section focuses on some related works. The methodological proposal is studied in the fourth section. The case study in which such methodology was applied is explained in the fifth section, followed by preliminary results and discussion.

## II. Safety Standard and fault tolerance

Safety Instrumented Systems (SIS) are safety actuation systems of an industrial unit, made of sensors, logical and final elements. Due to technology advance, such systems became more complex, leading to the establishment of safety standards that assist both their development and maintenance. These standards should automatically lead a system to a safety state or mitigate consequences of hazards.

The standard IEC 61508 was developed with the aim of being a guide to help diverse industries to develop additional standards specifically for their applications [4]. Such standard states that a safety integrity level (SIL) is responsible for defining the risk reduction that a SIS may provide. Moreover, a SIL is calculated in the beginning of a project.

From the standard IEC 61508, the standard IEC 61511 was created, intending establishment of a SIL goal, based on hazard and risk analysis. This is made in order to develop safety products with adequate safety integrity level. So, one or more safety instrumented functions (SIF) is applied so that an expected safety level might be reached. SIF is an action made by a SIS to bring a process or equipment to a safety state. A SIF has a specific SIL, which is needed to achieve functional safety.

SIL represents the amount of risk reduction that a SIS can provide, which is defined as the probability of failure on a demand interval. It can be defined in a scale of 1 to 3, according to [5] or 1 to 4, according to [6].

From the aforementioned, it might be said that IEC61508 standard recommendations were applied to establish a top level sequence of steps that integrate this methodological proposal.

978-1-5090-1331-9/16/$31.00 ©2016 IEEE

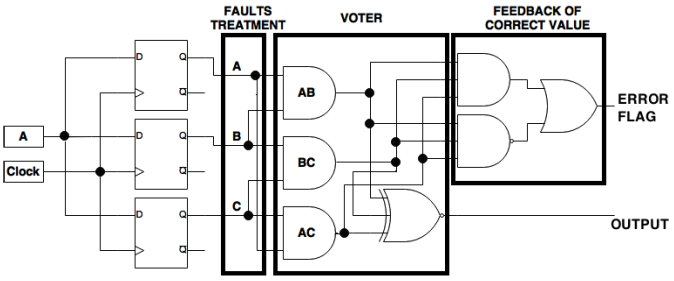


Fig. 1. TMR implemented.

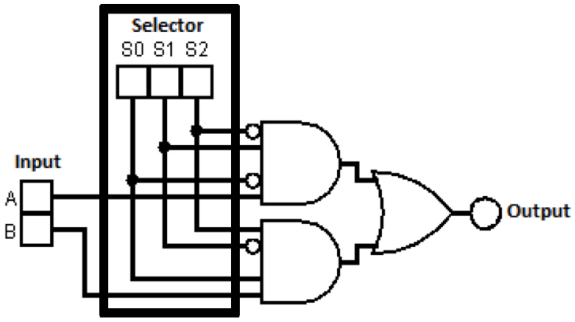


Fig. 2. Multiplexer with MooN technique.

Furthermore, two fault tolerance techniques are applied in this study. They are the N Modular Redundancy (NMR) and MooN technique. The first one is a variation of the standard Triple Modular Redundancy (TMR or 3MR), which is used in sequential components. Regarding the mitigation technique, MooN was applied in combinational components. Both techniques may treat faults occurred in internal signals of the circuit as well as those faults occurred in inputs and memory elements [2].

### A. NMR Technique

The NMR technique was applied to sequential components in the case study by using redundancy to deal with faults. Once it was done, the system was driven to a safety mode. It has an error flag and a voter, which decides the output signal through majority. For example, the TMR replicates three times the original circuit, as shown in Fig. 1. As can be seen in Fig. 1, the TMR contains a feedback of the correct value. Its aim is to assure that this methodology not only correct errors but mitigate them too. So, whenever all TMR inputs are equal, the error flag will be in high logical level. This fact indicates there was no error. Besides, for the TMR, it must have at least two signals with the same logical value so that such value may be transmitted to the output signal [7].

Fig. 1 highlights the voter, the error flag circuit and signals that may be affected by SETs. These signals result from the combination of input A being replicated three times through D-type flip-flops. The resulting signals are then transmitted to the voter and to the error flag circuit. It might be stated that whenever a SET occurs in the input of the NMR block, its output will be compromised. In a similar way, the output will be in error whenever the majority of voter inputs be changed by SETs.

### B. MooN Technique

This technique is specified in [6] and may be applied to combinational components, such as multiplexers and demultiplexers, by using redundancy to detect faults. In this case study, all multiplexers presented in the DSP module were modified by triplicating their selectors. These selectors are responsible for defining which input will be connected to the output. Consequently, the output will be changed by only two out of eight different combinations of the selector.

For example, by analyzing the 1oo3 technique, it replicates the control signal three times, as shown in Fig. 2. Furthermore, the truth table of it is presented in Table I. It may be seen the output will maintain its value for 24 out of 32 possible input combinations. Consequently, the chance of the multiplexer commute incorrectly the output is of one out of eight, that is, 12.5 %. In contrast, any wrong change in selector will lead to an erroneous value the output of a multiplexer without use of fault tolerance techniques.

TABLE I
TRUTH TABLE FOR THE MULTIPLEXER WITH 1OO3 TECHNIQUE

| A | B | S0 | S1 | S2 | Output |
|---|---|---|---|---|---|
| X | X | 0 | 0 | 0 | Does not change |
| X | X | 0 | 0 | 1 | Does not change |
| A | X | 0 | 1 | 0 | A |
| X | X | 0 | 1 | 1 | Does not change |
| X | X | 1 | 0 | 0 | Does not change |
| X | B | 1 | 0 | 1 | B |
| X | X | 1 | 1 | 0 | Does not change |
| X | X | 1 | 1 | 1 | Does not change |

## III. RELATED WORK

In spite of methodologies combining both fault tolerance and functional safety concepts to prevent occurrence of SETs, there is none that clearly does it nowadays. So, ensuring identification and properly treatment of faults that occur in sequential, logical and memory elements will also be possible with use of the proposed technology. We propose usage of the functional safety recommendation presented in [6]. By doing this, a set of reliability evaluation steps and treatment of SETs in a basic level will be addressed.

Increasing reliability of FPGA designs is addressed in [8] by proposing different fault tolerant strategies. In [9], spare memory use and replacement of a memory cell by its neighbor is suggested. Duplication with comparison as well as concurrent error detection are examined in [10]. Temporal redundancy, among others, is compared to TMR in [11] as alternative solutions to reach fault tolerance in FPGA applications. Furthermore, frames configuration is suggested in [12].

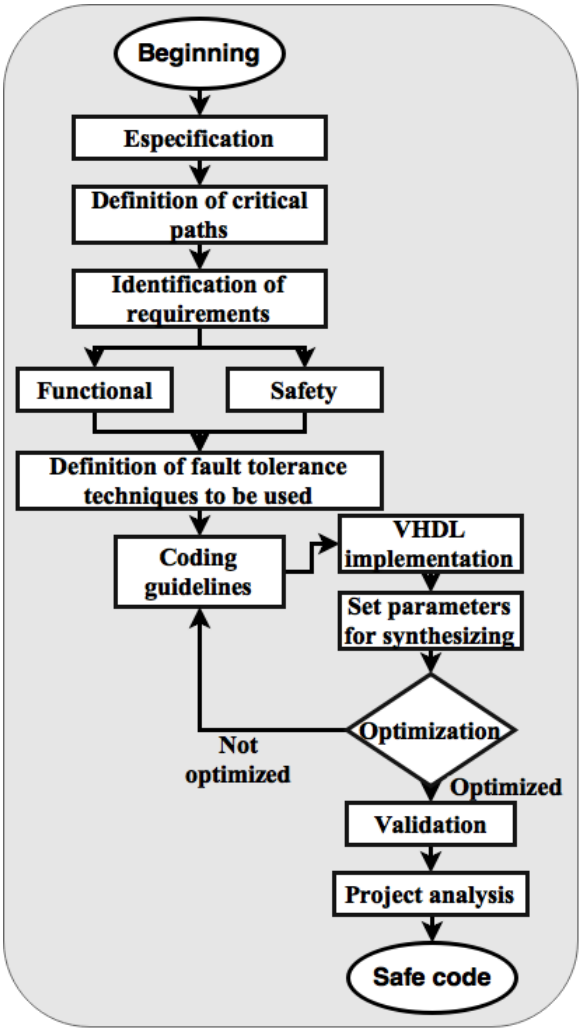


Fig. 3. Block diagram of proposed methodology

## IV. Proposed Methodology

The proposed methodology aims to specify a suggested safety management process that could be applied in accordance with IEC61508 [6] to SRAM-based FPGA applications. Furthermore, the technique MooN should be assigned to combinational elements and the NMR technique should be applied to sequential elements. Besides, codes should be as effective as the techniques being currently used. So, less expensive, more flexible, portable and optimized circuits would be obtained as well as there would have a better use of available resources.

Fig. 3 presents a block diagram of the proposed methodology. The first task to be accomplished is to set all project specifications, which includes definition of basic elements, diagrams, figures, tables and all relevant data that specifies its functionality. Once it is done, possible points of failure should be identified by looking at both combinational and sequential elements of the project [8]. So, with regard to the safety assessment, one or more critical paths of the circuit may be defined to further actions.

The third task in this approach is to identify functional and safety requirements in order to not only guarantee that all faults will be identified but also to ensure that the project will do what it was meant to. After that, fault tolerance techniques and mitigation ones should be evaluated to define which of them would best fit in the project.

In *coding guidelines* stage the manufacturer's suggestions should be consulted to develop a cleaner code. The target used in this case study was a Spartan 3E, from Xilinx. So, in order to avoid conflict of control signals, [7] suggests that flip-flops should use set or reset, never both; it is also recommended use of pipeline so that a performance increase may be addressed. By considering that flip-flops usually consume approximately 1 % of area in a project, it represents a valid gain. Further suggestions are made to minimize the number of control signals, as mentioned previously, to avoid conflict of control signals. Finally, it proposes an efficient use of lookup tables (LUTs), which may vary according to the family of the FPGA used.

The following step is implementation of the hardware description and proceed to set carefully synthesizing parameters. VHDL was used in this approach. At this point, it might be relevant to consider that the parameter *Keep hierarchy* in Xilinx's ISE parameters because it specifies whether a design unit should or not be merged to other parts of the design.

The style used for the hardware description is analyzed in *Optimization* stage, aiming to optimize it by using a best practice coding. To achieve this, a code should present the following requirements, as mentioned in [13]:

1) Code should abide by the VHDL language rules, meaning it should follow the legal constructs;
2) Code should have a common look, enhancing familiarity to the code in order to provide a better understanding;
3) Code should be easily readable and maintainable;
4) Code should yield expected results, that is, it has to behave as expected in its specification;
5) Obsolete or outdated hardware description should be avoided.

The next step is to validate the project, by confirming whether it satisfies all functional and safety requirements. Finally, a project analysis should be done so that resources usage may be quantified and optimized. At this stage, techniques applied should also be verified. By doing this, it might be possible to confirm that results are consistent to the expected ones, as well as other aspects, such as consumed area, skew, clock, among others.

All these steps may also be visualized in the life cycle for FPGA projects with functional safety, which is presented in Fig. 4.

Equally important is to consider the difference of implementing this methodology for FPGAs or for Application Specific Integrated Circuits (ASICs). It relies on the fact that it might be easier to modify a FPGA-based design than an ASIC one, by considering that several integrated circuits (ICs) would be needed to build components with 1ooN and NMR techniques.

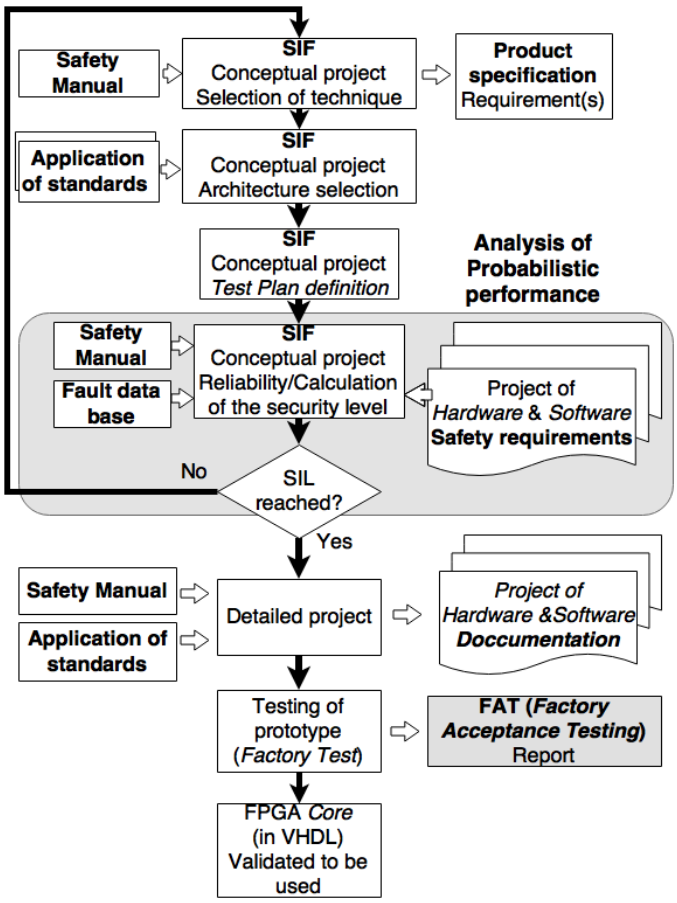


Fig. 4. Life cycle for FPGA projects with functional safety

Applying this methodological proposal instead of using error correction codes (ECCs) enables the possibility of both correct faults and inform the system about their occurrence. In contrast, these two tasks may not be possible by using ECCs. It will depends on the ECC-type that is being used. For example, parity is capable of just detecting faults whereas hamming and Reed Solomon can detect and correct them. It is likely that user modifications would be needed to use an ECC and to notify the system about a fault existence.

## V. Case Study

The DSP module studied in this case was described in VHDL and presents the following major characteristics:

1) May be configurable by the user;
2) Operates with fixed point of inputs with up to 18 bits, due to Spartan 3E primitives used;
3) Implements addition, subtraction, multiplication and division;
4) May operate in rising or falling edge of clock.

The entire design process of the DSP module studied in this case followed all methodological steps proposed in the previous section. Also, it might be mentioned that none multiplexer was used in addition and subtraction. Because of that, only TMR technique was applied on them. Similarly,

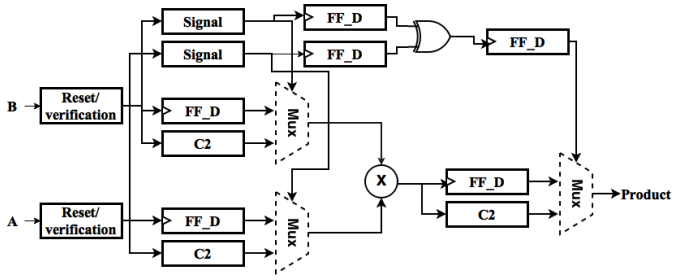


Fig. 5. Configurable multiplier without fault tolerance techniques

both multiplier and division blocks contain TMRs in their sequential elements. The 1oo3 technique was applied in combinational elements of both multiplication and division blocks, as can be seen in Fig. 5.

For the multiplier, A and B input widths are verified in the first clock pulse inside the block *Reset/Verification* in Fig. 5, whose output will be an 18 bit signal which is a copy its input. In sequence, the two signals are simultaneously connected to both D-type flip-flops and the *C2* blocks. These blocks perform two's complement of their input signals or extract information whether the data is positive or negative. Outputs of D-type flip-flops and *C2* blocks are then connected to multiplexers, whose selectors contain information obtained previously from inputs.

Outputs of the first two multiplexers are then connected to an embedded multiplier in Spartan-3E FPGA named *MULT18X18S*. Its output, along with inputs A and B is connected to a D-type flip-flop and a *C2* block. By doing this, a multiplexer may pick up the positive or the negative product of multiplication, depending on signals A and B.

It might be said that all continuous connections and blocks in Fig. 6 are in the critical path of the multiplier and were treated with the technique TMR, excepting memory elements. Moreover, the technique MooN was applied in all multiplexers. Such elements are framed by a dashed line in Fig. 5 and Fig. 6. They are also part of the critical path, once any SET that occur will lead to an error in the output of the multiplier. By considering these modifications, the configurable multiplier with fault tolerance techniques applied is presented in Fig. 6.

In Fig. 6 it might be seen that several TMRs were used to treat SETs in the circuit. Both verification of input width and two's complement of it are implemented in blocks *TMR_A* and *TMR_B*. TMRs were not inserted in memory elements in order to not increase overly circuit area. These elements are the *FF_D* blocks in Fig. 5 and Fig. 6.

Regarding identification of product signal to define whether it is positive or negative, it is developed in *TMR_Signal* block. Finally, a two's complement of the product is done in *TMR_product* which is then connected to a multiplexer along with the positive product of multiplication.

Similar evaluations were developed to all four operations implemented in this DSP module and that is the reason why not all of them were presented in this paper.

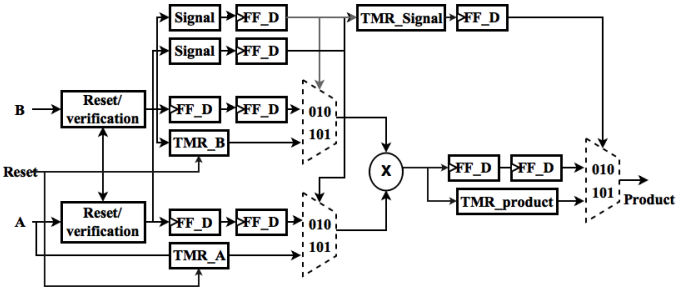


Fig. 6. Configurable multiplier with fault tolerance techniques

## VI. Results and Discussion

Simulations were made through use of three softwares: ISE and ISIM from Xilinx and a fault injector simulator named *SH_lapse*. This simulator was designed to insert SETs in a VHDL project [14]. So, each arithmetic operation was tested separately by inserting faults in both circuits: the circuit without any fault tolerance technique and in the fault tolerant one. Then, the number of outputs with wrong values, time response and area were compared between them.

Furthermore, each signal that belonged to the critical path was identified. The number of inserted SETs varied from 0 to 100 per simulation. That is, in the first simulation, one SET was inserted. In the second one, 10 SETs were included and in the third one, 100 SETs were inserted in each design. Besides, two different types of simulations were executed: random and sequential. They differ because in a sequential simulation faults are inserted in signals by obeying a specific order. In contrast, faults are randomly injected in signals in random simulations, as the name suggests. So, for each arithmetic function, 6 different simulations were executed.

It was also defined that all SETs would last 10 % longer than the clock period, so that all of them could be registered. Stimuli duration for input signals is bigger than for SET duration in order to reach a feasible simulation. Regarding width of inputs A and B, they were chosen to be different so that we can prove our DSP operates with inputs with different number of bits. Other simulation parameters are presented as follows:

1) Stimuli duration for input signals: 100 ns;
2) SET duration: 11 ns;
3) Total time employed for the fault simulation: 10,000 ns;
4) Number of inserted SETs per simulation: 1, 10 and 100;
5) Width of A input: 15 bits;
6) Width of B input: 13 bits;
7) Clock operates in rising edge;
8) Clock frequency: 100 MHz;
9) Target: Spartan3E-100 CP132;
10) All simulations were post synthesis;

Fig. 7 summarizes error reduction per arithmetic function that occurred in sequential simulations. As can be seen, there is a slightly difference between addition and subtraction results, due to several similarities they share, such as architecture and number of signals that integrate their critical paths.

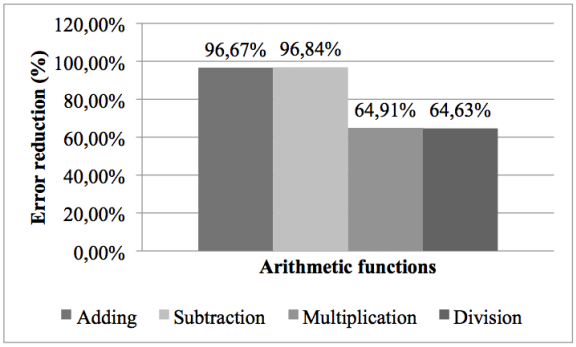


Fig. 7. Error reduction per arithmetic function (sequential simulation)

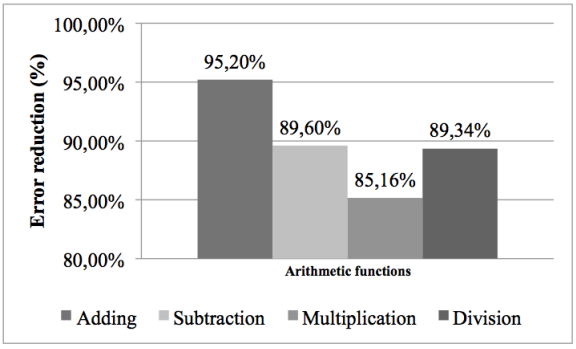


Fig. 8. Error reduction per arithmetic function (random simulation)

The average rate of error reduction identified in sequential simulations of the multiplier was smaller than for addition and subtraction. It occurred because both multiplier and divider contain larger numbers of critical signals than addition and subtraction. In addition, it might be said that most part of differences between multiplier and divider is due to the number of memory elements they have. Therefore, these are some of the reasons that may explain why they presented similar results.

Referring to error reduction rates for random simulations, they are presented in Fig. 8. For addition and subtraction functions, error reduction rates were similar to the ones obtained from sequential simulations. However, these rates vary significantly for multiplication and division, which may be related to the name of critical signals and how they are selected to a fault insertion in SH_lapse simulator.

With regard to clock pulses increase related to the use of aforementioned fault tolerance techniques, it might be said that it varied from 33.3 % to 40 % and is presented in Fig. 10. It should also be mentioned that area increase, which is presented in Fig. 9, varied from 60.77 % to 163.49 %. The lowest increase of area is related to the divider, as expected, by considering it is the arithmetic function that uses the biggest number of slices among them.

Table II shows the number of errors reduced in all operations.

From the aforementioned results, it might be stated that the methodology proposed in this paper might be used in critical

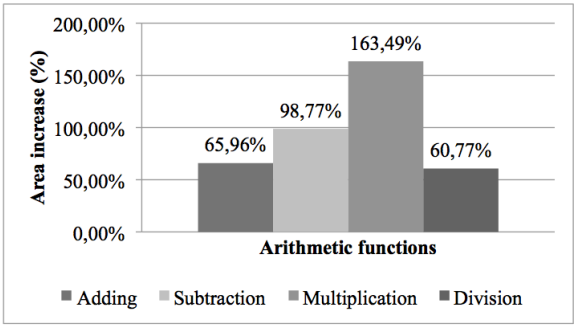


Fig. 9. Area increase

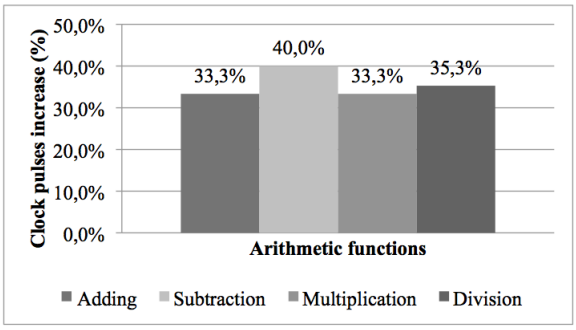


Fig. 10. Clock pulses increase

TABLE II
NUMBER OF ERRORS REDUCED FOR EACH ARITHMETIC OPERATION

| Simulation | No. SETs applied | Error reduction per operation | | | |
|---|---|---|---|---|---|
| | | + | - | x | : |
| Sequential | 1 | 1 | 1 | 1 | 1 |
| | 10 | 11 | 11 | 6 | 8 |
| | 100 | 99 | 105 | 61 | 78 |
| Random | 1 | 1 | 1 | 1 | 1 |
| | 10 | 10 | 8 | 6 | 8 |
| | 100 | 143 | 119 | 76 | 80 |

applications that require such safety. Furthermore, it leads to the fact that hardware use, available energy and possibility of increase the circuit time response are limitations of this proposal.

Fig. 7 and Fig. 8 highlight a significant reduction in the number of errors that occurred in sequential and random simulation. So, identification of the exact moments in which these faults occurred and error reduction are both advantages and an improvements of the functional safety of a design, obtained from this proposal. Code optimization and improving its readability might also be benefits of this methodology.

## VII. CONCLUSION

This study proposed to aggregate a functional safety methodology to fault tolerance FPGA applications. MooN and NMR were suggested as fault tolerance techniques that could be applied to sequential and combinational elements, respectively, in a FPGA project. Moreover, the aforementioned methodology was applied to develop safe codes in VHDL in order to implement a parameterized DSP module that executes addition, subtraction, multiplication and division.

Two DSP modules were compared: the one without fault tolerance techniques and designed with no concern about safety and its modified version, which was planned with use of the proposed methodology and fault tolerance techniques. So, from the presented results, it might be stated that this methodology may be a relevant tool to be used in the design of safe applications for FPGAs, once it enables implementation of several SIFs inside a system as well as establishes a specific sequence of tasks that should be done to create an organized and optimized safe core in VHDL.

As a future work it is proposed study of error correction codes and their comparison with MooN and NMR techniques that were addressed in the present study. In sequence, multiple bit upsets (MBUs) may be defined as a future work too.

## REFERENCES

[1] S. W. Smith, *Digital Signal Processing*. 200 Wheeler Road, MA 01803, Burlington, The United States of America: Newnes, 2003.
[2] F. Kastensmidt, "Designing single event upset mitigation techniques for large sram-based fpga components," Ph.D. dissertation, Federal University of Rio Grande do Sul. Computer Institute, PO box 15064, zip code 91501-970, Porto Alegre - RS - Brazil, 2003.
[3] R. C. Baumann, "Radiation-induced soft errors in advanced semiconductor technologies," in *IEEE Transactions on Device and Materials Reliability*, vol. 5, no. 3, 2005, pp. 305–316.
[4] A. P. A. V. Melo, "Safety integrity level (sil) integrated with human and organizational factors," diploma thesis, School of Chemistry of the Federal University of Rio de Janeiro, March 2012.
[5] ISA, *Safety instrumented functions (SIF) - Safety integrity level (SIL) evaluation techniques Part 3: determining the SIL of a SIF via fault tree analysis (ANSI/ISA TR 84.02)*, The Instrumentation, Systems, and Automation Society (ISA) Std., 2002.
[6] IEC, *IEC 61508 - Functional safety of electrical/electronic/programmable electronic safety-related systems*, International Electrotechnical Commission Std., 2010.
[7] P. Adell, G. Allen, G. Swift, and S. McClure, "Assessing and mitigating radiation effects in xilinx sram fpgas," in *Radiation and Its Effects on Components and Systems (RADECS), 2008 European Conference on*, Sept 2008, pp. 418–424.
[8] M. Berg, "Fault tolerance implementation within sram based fpga designs based upon the increased level of single event upset susceptibility," in *IEEE International On-Line Testing Symposium (IOLTS'06)*, no. 12, 2006.
[9] F. Hanchek, "Methodologies for tolerating cell and interconnect faults in fpgas," in *IEEE Transactions on Computers*, vol. 47, no. 1, 1998, pp. 15–33.
[10] F. G. Kastensmidt, G. Neuberger, R. F. Hentschke, L. Carro, and R. Reis, "Designing fault-tolerant techniques for sram-based fpgas," in *IEEE Design & Test of Computers*, 2004, pp. 552–562.
[11] B. H. P. Keith S. Morgan, Daniel L. McMurtrey and M. J. Wirthlin, "A comparison of tmr with alternative fault-tolerant design techniques for fpgas," in *IEEE Transactions on Nuclear Science,*, vol. 54, no. 6, Dec. 2007, pp. 2065, 2072.
[12] E. C. Farid Lahrach, Abderrahim Doumar, "Fault tolerance of sram-based fpga via configuration frames," no. 14, 2011, pp. 139–142.
[13] B. Cohen, *VHDL Coding styles and Methodologies*, 2nd ed. Dordrecht: Kluwer Academic Publishers, 2002.
[14] B. R. R. Reis, "Sh_lapse simulator," 2014.