ISSN 0002-306X. Proc. of the RA NAS and NPUA Ser. of tech. sc. 2020. V. LXXIII, N2

UDC 621.382.13 MICROELECTRONICS

V.A. JANPOLADOV, A.A. PETROSYAN, S.S. ABAZYAN, H.V. MARGARYAN

RANDOM FAULTS INJECTION AND SIMULATION IN AUTO-CORRECTION CIRCUITS

The random faults impact on auto-correction circuit, such as the Hamming code scheme, is demonstrated. For their injection a Verilog model was created which focuses on the output buses of correctly working Hamming code circuit. The gate level simulation shows that even verified output data from the auto-correction scheme can be corrupted under the random faults influence such as radiation effects. The total failure ratio after random fault injection is 61% compared to simulation without faults injection. As an example of solving the observed issues, the Triple Modular Redundancy (TMR) methodology implementation with Hamming code circuit is presented. The proposed simulation and implementation methods can be used in modern automotive system integrated circuits (ICs) where data transmission is of critical importance and is included in the standard digital design flow.

Keywords: fault injection, fault generation, radiation, simulation, digital design, Hamming code, Triple Modular Redundancy, verification.

Introduction. The modern pace of technology progress and technology node scaling at a tremendous speed bring new challenges to the design of ICs. In addition, the old problems begin to gain much more weight than before. Entering the market of automotive system (AS) devices further tightens the requirements for the design of ICs, as well as for their protection. There are many types of faults that can affect the normal operation of ICs and data transmission through them, such as Single Event Effects (SEE) caused by radiation, stack-at and bridging faults, which appear after the manufacturing process or under thermal influence. Because of this, verification of the developed ICs has become one of the most important and resource-demanding part, since human life is beginning to depend on their correct operation. For self-verification and assurance in the scheme proper behavior in AS systems auto-correction circuits are required.

Auto-correction circuits are used for verifying input data and correcting it. But under the influence of radiation their functionality can be corrupted, in other words, the already verified or corrected data can be changed in the outputs of such circuits after all the verification stages which will lead to failures in the whole system. The Hamming code block was chosen as an example of auto-correction circuits which can detect and auto-correct single-bit

errors. On the other hand, for multibit errors, the reset signal is generated to report their existence in the input data.

The work [1] presents fault detection and correction by Hamming code. In work [2] Hamming code and the TMR technique are analyzed separately in ICs. It is proved that the TMR methodology implemented on registers is more effective than the single Hamming code circuit usage in multibit systems because of the area increase. But the input data can not be verified and auto-corrected in a simple TMR register implementation, that is why such circuits as Hamming code are required on the inputs of ICs. The Single Event Transient (SET) influence on voters designed for TMR is described in work [3] when this paper studies the SET influence on Hamming code block. Automation of the TMR methodology implementation for complex digital circuits is described in work [4].

As an example of the mentioned issue solution TMR implementation for the enhanced data transmission protection is shown. This technique with the Hamming code can be used in systems where data protection has high priority, such as AS.

Random faults injection on the Hamming code circuit. To simulate a random impact on the circuit, the SET injection model is used. For that, an appropriate Verilog model is created (Fig.1). It randomly chooses a bit from the output data of the auto-correction circuit, then injects a SET pulse at a random simulation time and with a random pulse width. The placement of the random SET injection is shown below.

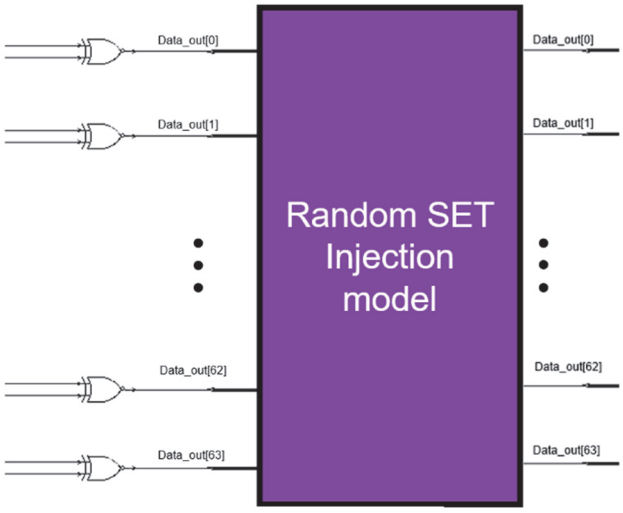

Fig. 1. The placement of the created fault injection model

The SET pulse width, injection time and probability calculation are based on the proposed method from [2]. SET can be injected with a frequency twice as high as the design working frequency (Fig.2).

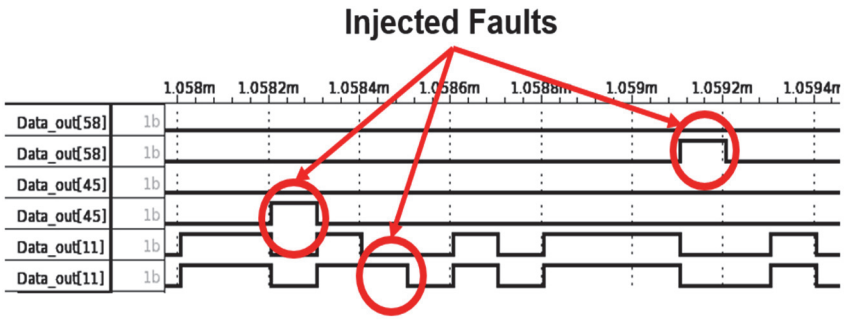

Fig. 2. An example of waveform comparison of correct and faulted data for Data_out[11], Data_out[45] and Data_out[58] output signals

Single Hamming code implementation. For a single Hamming code implementation, a design supporting this structure is developed by using Verilog Hardware Description Language (HDL). The mentioned design with a single Hamming Code circuit consists of several modules (Fig.3):

• Input registers module – 64-bit registers for driving the input data;

• Hamming code module – as an example of an auto-correction circuit which operates with 64-bit data;

• Data capture registers – 64-bit registers for reading data from Hamming code.

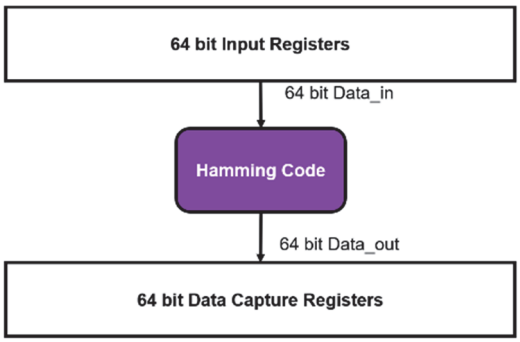

Fig. 3. Hierarchy without TMR

The created Register Transfer Level (RTL) description is verified by functional simulation for assurance in the expected behavior. The input registers drive received data to the Hamming code block, which verifies 64-bit data and then sends it to the output data capture registers.

After that logic synthesis is performed to generate the gate level netlist. 14 nanometer FinFET technology node is used as a target library for synthesis and analysis, which is developed by Synopsys Armenia Educational Department (SAED) [5]. The operation frequency is 500 *MHz*. The results are demonstrated in Table 1.

Table 1

Timing, area, power reports without TMR after logic synthesis

Worst Slack [*ns*]	Total Power [*uW*]	Total Area [μm^2]
1,49	56,5189	359,9659

Synthesis was done without collapsing the RTL hierarchy and with a high mapping effort.

TMR implementation with Hamming code. The idea is to implement the TMR methodology with the Hamming code circuit for enhanced data transmission protection in ICs by using Digital Design Flow (Fig.4).

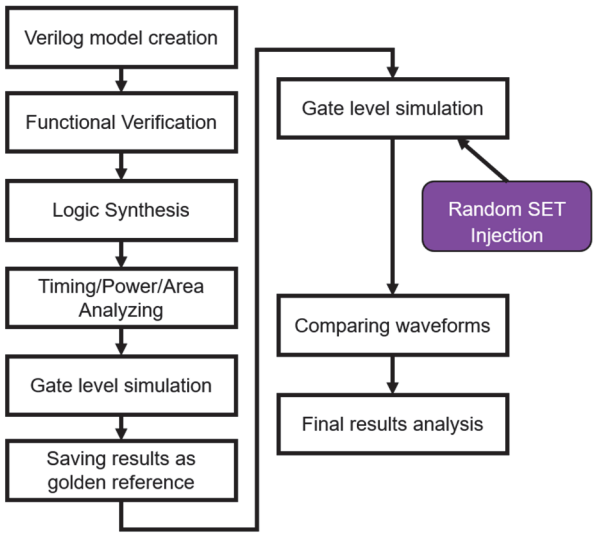

Fig. 4. Implementation flow

The developed RTL description for a single Hamming code implementation was modified to support the TMR methodology (Fig.5).

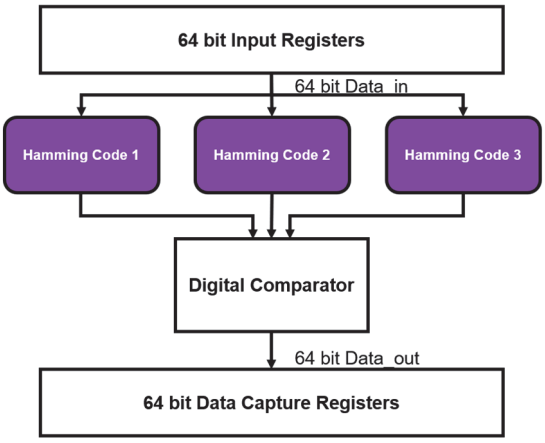

Fig. 5. The TMR methodology with a Hamming code

The quantity of the Hamming block modules become three. These blocks are operating parallelly, sending verified data to the created comparator. A digital comparator is added in the design hierarchy to work as a voter that chooses the correct data by the majority principle algorithm (Fig.6).

```
If(data_1 == data_2 == data_3) {
      out = data_1
} else if (data_1 == data_2) {
            out == data_1
      } else if (data_2 == data_3) {
                  out == data_2
            } else if (data_3 == data_1) {
                        out == data_3
                  } else {
                              // If all data differs
                              // report system reset
                        }
```

Fig. 6. Comparator algorithm

After the logic synthesis the reports regarding performance, power consumption and design area are collected and saved for comparison with synthesis reports of the single Hamming code implementation. Simulation is done based on the generated gate level netlist with and without random fault injection to check the stability of this model. Also gate level netlist generated in the previous single Hamming code implementation is simulated with the same fault injection mechanism for comparison with the TMR-based design.

Experiment results. Single Hamming code is implemented and simulated with random fault injection for comparison with the proposed application of TMR methodology on the Hamming code circuit by tripling its quantity and

adding separate comparator block for the output data analysis. The results for logic synthesis and gate level simulation are provided below.

The target technology and frequency remained the same in comparison with the single Hamming code circuit implementation results. The results for timing, area and power factors after synthesis are shown in Table 2.

Table 2

Timing, area, power reports with TMR after logic synthesis

Worst Slack [*ns*]	Total Power [*uW*]	Total Area [μm^2]
1,02	137,7133	2058,6153

Gate level netlists for both designs are simulated ~250000 times with and without SET injection and then compared. The failure ratio in final output data waveform in comparison with the single Hamming code circuit is 61% (Fig.7).

Fig. 7. Waveform comparison without TMR

In case of the TMR implementation and fault generation on one of the three Hamming code blocks, 100% valid data will pass through the comparator and the next stages of the IC (Fig.8).

Fig. 8. Waveform comparison with TMR

Sections A and B demonstrate two implementations' synthesis and simulation result comparison to understand the difference ratio, advantages and disadvantages of the proposed method.

A. Synthesis Comparison

As described earlier, two synthesis iterations are performed for the analysis. The first is with a single Hamming code module and the second one with the TMR methodology implementation. Logic synthesis is done by using the Synopsys' Design Compiler [6] tool.

The worst slack comparison is shown in Table 3. Timing degradation is expected because of additional logic insertion on the critical path.

Table 3

Critical Path Slack Comparison

Report	Without TMR	With TMR
Worst Slack [*ns*]	1,49	1,02

Area comparison is illustrated in Table 4. The total area increased approximately 3 times, which is expected since the number of the Hamming code blocks increased by two.

Table 4

Area Comparison

Report	Without TMR	With TMR
Net Interconnect area [μm^2]	359,9659	1176,2985
Combinational area [μm^2]	177,2898	744,3216
Sequantial area [μm^2]	119,3471	137,9951
Total Area [μm^2]	656,602	2058,6153

To analyze the area increase in case of big designs, the ORCA processor reports were used from work [7], which is designed on the same SAED 14 nanometer FinFET technology node. The total area of proposed circuit with the TMR implementation is ~0,773% of the ORCA's total area.

The comparison of static, dynamic and total power consumption is shown in Table 5. The same reference processor was used for analysis.

Table 5

Power Dissipation Comparison

Report	Without TMR	With TMR
Cell Leakage Power [*nW*]	132,8312	418,8779
Cell Switching Power [*uW*]	15,6808	82,3838
Cell Internal Power [*uW*]	40,7052	54,9106
Total Power [*uW*]	56,5189	137,7133

The total power consumption increases approximately 2.4 times. In case of a single Hamming code circuit total power is equal to 56,5189 *uW*, which is only ~0,03% of the ORCA's total power value, then the TMR implementation is equal to only ~0,07% of the ORCA's total power consumption with low power techniques implementation in it. Power increase is also expected, as a number of standard cells increase due to the TMR methodology implementation.

B. Simulation Results

To prove the effectiveness of the proposed TMR implementation with the Hamming code circuit in data transmission protection both designs' gate level netlists are simulated with the random SET faults injection. For simulation and waveform analysis Synopsys' VCS [8] and Custom WaveView [9] tools are used.

After the previously described simulations the final output data is separated for each design and compared. All failures from the first iteration are waved with TMR methodology implementation. The results are demonstrated in Table 6. Error waving is 100% in case of the TMR implementation

Table 6

Waveform Comparison

Signal	Iteration	Runs	Errors
Data_out	Without TMR	250000	~154000
Data_out	With TMR	250000	0

These results clearly demonstrate the effectiveness of the proposed method in data transmission quality with the expected losses in power consumption and area.

Conclusion. The paper presents the random faults injection and simulation method to show the faults' consequences on the correctly working auto-correction circuit such as the Hamming Code. The failure ratio observed by using the proposed method in the final output data with the Hamming code circuit is 61%. The presented method can be included in the commonly used standard digital design flow. As an example of solution of the considered issues caused by faults injection, TMR methodology implementation is used. The total failure waving is almost 100% compared to the single Hamming code implementation in case if one Hamming code circuit data is corrupted.

REFERENCES

1. **Anil Kumar Singh.** Error Detection and Correction by Hamming Code // International Conference on Global Trends in Signal Processing, Information Computing and Communication (ICGTSPICC). - 2016. - P. 35-37.
2. Analyzing area and performance penalty of protecting different digital modules with Hamming code and triple modular redundancy / **R. Hentschke, F. Marques, F. Lima, et al** // Proceedings. 15th Symposium on Integrated Circuits and Systems Design. - 2002. - P. 95-100.
3. **Danilov Igor A., Gorbunov Maxim S., Antonov Andrey A.** SET tolerance of 65 nm CMOS majority voters: A comparative study // 14th European Conference on Radiation and Its Effects on Components and Systems (RADECS). - 2013. - P. 1597-1602.

4. **Benites L.A. C. and Kastensmidt F. L.** Automated design flow for applying triple modular redundancy (TMR) in complex digital circuits // IEEE 19th Latin-American Test Symposium (LATS). - 2018. - P. 1-4.
5. **Melikyan V., Martirosyan M., Piliposyan G.** 14nm Educational Design Kit // Capabilities Deployment and Future Small Systems Simulation Symposium. - 2018. - P. 37-41.
6. Design Compiler® User Guide. - Synopsys Inc., 2019. - P. 90-276.
7. Multi-Voltage and Multi-Threshold Low Power Design Techniques for ORCA Processor Based on 14 nm Technolog / **V. Melikyan, M. Martirosyan, D. Babayan, et al** // 2018 IEEE 38th International Conference on Electronics and Nanotechnology (ELNANO). - 2018. - P. 116-120.
8. VCS® User Guide. - Synopsys Inc. 2019. - P. 551-562.
9. Custom WaveView™ User Guide, - Synopsys Inc., 2019. - P. 113-124.

Russian-Armenian University, National Polytechnical University of Armenia, Yerenav State University. The material is received on 20.01.2020.

Վ.Ա. ՋԱՆՓՈԼԱԴՈՎ, Ա.Ա. ՊԵՏՐՈՍՅԱՆ, Ս.Ս. ԱԲԱՋՅԱՆ, Հ.Վ. ՄԱՐԳԱՐՅԱՆ

ԱՆԿԱՆՈՆ ՍԽԱԼՆԵՐԻ ՆԵՐՄՈՒԾՈՒՄԸ ԵՎ ՄՈԴԵԼԱՎՈՐՈՒՄԸ ԻՆՔՆՈՐՈՇՄԱՆ ՇՂԹԱՆԵՐՈՒՄ

Ցուցադրվում է անկանոն սխալների ազդեցությունը ինքնորոշման սխեմայի վրա, ինչպիսին է Հեմմինգի կոդի սխեման: Անկանոն սխալների ներմուծման համար նախագծվել է Verilog լեզվով մոդել, որը կենտրոնանում է ճշգրիտ աշխատող Հեմմինգի կոդի ելուստների վրա: Փականների մակարդակի մոդելավորումը ցույց է տալիս, որ նույնիսկ այն տվյալները, որոնք ստուգում են անցել ինքնորոշման սխեմայի միջոցով, կարող են աղավաղվել անկանոն սխալների ազդեցությամբ, ինչպիսիք են ճառագայթման էֆեկտները: Վերջնական ձախողումների տոկոսը սխալների ներմուծումից հետո կազմում է 61%, ի համեմատ առանց սխալների մոդելավորման: Որպես նկատված ձախողումների կասեցման միջոց ներկայացված է եռակի մոդուլային պահեստավորման մեթոդի կիրառումը՝ Հեմմինգի կոդի սխեմայի օգտագործմամբ: Առաջարկված մոդելավորման և կիրառման մեթոդները կարող են օգտագործվել ժամանակակից ավտոմատացված ինտեգրալ սխեմաներում, որտեղ տեղեկույթի ճշգրիտ փոխանցումն ունի վճռորոշ նշանակություն, ինչպես նաև կարող են օգտագործվել թվային նախագծման ընթացակարգում:

Առանցքային բառեր. սխալի ներմուծում, սխալի գեներացում, ճառագայթում, մոդելավորում, թվային նախագծում, Հեմմինգի կոդ, եռակի մոդուլային պահեստավորում, ստուգում:

В.А. ДЖАНПОЛАДОВ, А.А. ПЕТРОСЯН, С.С. АБАЗЯН, А.В. МАРГАРЯН

ВНЕДРЕНИЕ И СИМУЛЯЦИЯ СЛУЧАЙНЫХ ОШИБОК В АВТОКОРРЕКТИРУЮЩИХ СХЕМАХ

Показано влияние случайных ошибок на схему автокоррекции, такую как схема кода Хэмминга. Для их внедрения разработана модель на языке Verilog, которая фокусируется на выходных шинах корректно работающей схемы кода Хэмминга. Симуляция на уровне вентилей показывает, что даже данные, прошедшие проверку в автокорректирующих схемах, могут быть искажены под влиянием случайных ошибок, таких как радиационные эффекты. Конечный процент сбоев после внедрения случайных ошибок составляет 61% в сравнении с симуляцией без внедрения ошибок. В качестве примера предотвращения наблюдаемых сбоев представлена реализация методологии тройного модульного резервирования со схемой кода Хэмминга. Предлагаемые методы симуляции и реализации могут быть использованы в современных интегральных схемах для автоматизированных систем, где передача данных имеет критически важное значение, а также могут быть внедрены в стандартный маршрут проектирования.

Ключевые слова: внедрение ошибки, генерация ошибки, радиация, симуляция, цифровое проектирование, код Хэмминга, тройное модульное резервирование, верификация.