Fault Tolerant FSM on FPGA Using SEC-DED Code Algorithm

Sooraj S[1], Manasy M[2], Ramesh Bhakthavatchalu[3]
Department of Electronics and Communication Engineering
Amrita University, Amritapuri, India
soorajsreedhar90@gmail.com[1], pmanasy6@gmail.com[2], rameshb@am.amrita.edu[3]

***Abstract*—This paper focuses on the design of an automatic error detection and correction of FSMs for soft errors. Single Event Upsets (SEUs) due to radiation impose an increasing problem to the reliable operation of FPGA used in communication, mile-aero, automotive and industrial designs. By developing safe case sequential logic and fault tolerant state machines with error mitigation circuitry logic, the FPGA design can be protected from SEUs. When a soft error occurs, these method ensures safe design operation by returning the design to a known safe or reset state of operation. This logic provides reliable system operation. The objective of this work is to implement automatic error detection and correction of FSMs for single event upsets is presented. We also analyze how bit errors within same clock cycle can be removed by the addition of error checking bits using SEC-DED code algorithm. The SEC-DED is commonly used as error detection and correction since it is more efficient t han p arity c hecker. T he S EC-DED c ode c an b e used as the error detection and correction circuitry behind the FSMs to improve the reliable operation. The proposed architecture is simulated in VCS. This structure is synthesized in Xilinx Virtex 7 for implementing on FPGA. The result of the proposed design is analyzed and compared to determine its performance in terms of reliability.**

***Keywords*- Finite State Machines(FSM), Single Event Upsets(SEUs), Hamming algorithm, SEC-DED code, Verilog, FPGA, Synplify premier.**

I. INTRODUCTION

In this technologically advanced world, FPGA's find a number of critical application in operating regions involving adverse environmental conditions (military and space). For the proper operation, such circuits must be fault tolerant, especially FSMs in which fault detection is a hazardous task. At present, the available FPGA synthesis tools are capable of error recovery to a specified r ecovery s tate w ithout providing error correction. In this paper, we are focusing on an efficient and reliable FPGA design which is capable of protecting against single event upsets. Radiation-induced SEUs pose a great threat to the proper working of devices. FPGA designs can be protected from such errors using safe sequential logic as well as fault tolerant machines with custom error mitigation. When a soft error occurs, these techniques ensure safe design operation by returning the design to a known safe state or reset state of operation. Single event upset is generating on-chip memory element fault. If such faults are not corrected in proper time, the FPGA disaster may occur. Adding the Error Detection and Correction(EDAC) circuit in FPGA to avoid soft error faults is a good option. The SEC-DED (Single Error Correction and Double Error Detection) hamming code has different applications [1]-[2]-[3]. In this design, hamming code method is used to protect FSM from unknown state.

The radiation emits alpha particles as they decay and distract the electrons in the semiconductor. These disturbances can cause drastic changes in the voltage levels in digital logic. The voltage disturbance will be mostly transient. The transient error gets stored and propagate through the state machines, which results in hardware failure. These type of error is known as single event upsets(SEUs) [4] . A flipped bit in the state machine's state register can put the FSM into an invalid state. Safe FSM or safe case FSM implementation using error detection circuit try to move from unknown state to reset state. These methods need to start the FSM from the initial state.The SEC-DED code can be used to detect double bit error and correct single bit errors of the hamming distance of four, ensuring that the content of a state register reaching a correct state would be detected and that correct operation of FSM would continue. Many research works have been done to protect FSM from SEUs [5]-[6].

In this paper, FSM is designed which can protected from single bit error and double bit error has been designed and implemented. The proposed design provides flexibility in terms of correction of errors protect the FSM from the unwanted state. We also concentrate on improving the reliability of FPGA and the system becomes working without errors. It is simulated and synthesized using Verilog on the synplify premier for implementation in Xilinx Virtex 7 FPGA [7].

The structure of the paper proceeds as follows: Section II describes the previous work. Section III discusses Finite State Machines (FSM). Section IV discusses the SEC-DED hamming code algorithm. Section V discusses how fault tolerant FSM is implemented using the hamming algorithm. Section VI shows simulation and synthesis results. Section VII concludes the paper and discusses future scope in this field.

II. PREVIOUS WORK

Referring the previous work, it takes many fault tolerant approaches have been proposed to protect all the parts of FSM [8]. Most commonly used method is TMR(Tripple Modular Redundancy Architecture). This method triplicates the design and the voter logic algorithm is used to select the valid next state. This approach is already implemented in Xilinx FPGAs [9].

State machines can perform single bit error correction by using hamming distance 3. When one error is detected, the present state is routed to the feedback loop. Error correction take place for single bit only [10] .

Based upon the classical approaches ,a double modular redundancy (DMR) solution have been proposed [11]. This method perform duplication of memory with parity bit included in each word. This method require two memory locations and large FSM implementation is difficult. FSM can be protected from SEUs by using embedded memories in FPGA [5] . This method uses a number of block RAM memories. The safety and reliability of FPGAs are obtained by using delay based physically unclonable functions [12]. The design optimization to improve the various parameters like area, memory and speed in FPGA is shown [13].

In this paper proposed architecture correct errors and minimal extra circuit used for error detection and correction methods.

III. FSM MODEL

The best method in which we can model the behavior of a system is by using FSM [14]. States usually describe the value of a circuit at different points and state machine transverses through different states in a special manner. A basic state machine will have some logic which is usually PAL based, output registers and state registers. State machines are used in a number of system control applications. The behavior of FSM can be graphically described by state transition diagram as in Figure 1. In this transition diagram, the bubble represents the states, and arrows represent the transitions. The arrow labels indicate the input value corresponding to the transition. The figure represents the 4 state FSM transition diagram for 1010 pattern. The diagram represents all the possibility of that particular FSM working in normal condition.

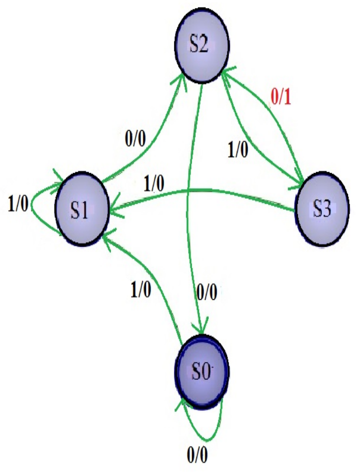


Fig. 1. State Transition Diagram.

Basically, FSMs are of two types: Mealy [15] and Moore state machines. For Moore machines, output generation is described by assigning outputs with states. Similarly, for a Mealy machine, we are assigning outputs to transitions [14]. Using FSMs we can analyze the state of a system at a particular point of time under our consideration and also we can use that state to formulate the behavior of the system. The systems will be modeled by considering the number of states that the system can occupy along with the behavior of the system in these states. The transition between states and connectivity between states also define the efficient modeling of a system. Conceptually we can define FSM modeling by the following key factors.

- Description of system by a finite number of states
- Behavior of system in each state depends on input/event
- Transition between states can be triggered by either inputs/events
- Systems will have an initial input
- In short FSM is a random sequential next state logic machine.

The state transitions are complicated and do not follow a simple pattern. Figure 2 shows basic block diagram of FSM.

Here in Figure 2, FSM is synchronous indicating that transitions will be controlled by the edge triggering of clock signals only. The Moore outputs are functions of states whereas Mealy machines are functions of states and inputs. For synchronous FSM, at rising edge, the FSM enters a new state. During the clock period, it performs functions such as the assertion of output signals and results will be simultaneously examined and exit path determination will be done at the next rising clock edge. State machines can also be described using flow

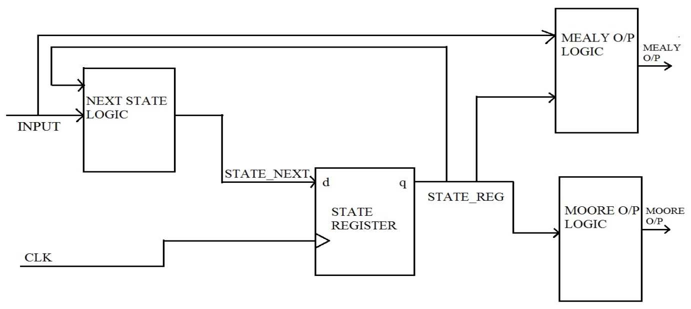


Fig. 2. A Basic Block Diagram of FSM with Mealy and Moore Logic.

charts as well as state transition table representation. Once we finish describing a state machine, we can implement it on a device. Software packages will be used for implementation of a state machine in the device. For that purpose, state machine description is converted into transition and output functions [15]. During synthesis, the physical components will provide delay threats. In total, the performance of FSM is determined by maximal clock rate. Consider an FSM with four states S0, S1, S2 and S3 encoded using one–hot coding. The corresponding states are defined in Table I.

Each bit in the state is representing state registers. If a single event upset occurs in any one of this FSM state registers, FSM may go to an invalid state. In such conditions, it is very difficult to determine the correct state of FSM. Hamming

TABLE I. STATES IN FSM MODULE USING ONE-HOT CODE

State Name	State
(State 0)S0	0001
(State 1)S1	0010
(State 2)S2	0100
(State 3)S3	1000

algorithm can be used to recover the FSM from the invalid state.

IV. SEC-DED CODE

The Single error correction and double error detection is more reliable error correction and detection method[4].The advanced version of the hamming code is called as SEC-DED code.

A. *Hamming Code*

Hamming code is a binary error correcting block code which contains n bit data and (k-n) parity bits[16]. This group of k message bits per block is called as code word as in Figure 3. Consider an n-dimensional message space which contains n bit of data. This particular space contains 2^n possible combinations and space is fully filled up. The code word is representing a k dimensional code word space and contain 2^k possible combinations .i.e., $2^k \gg 2^n$.

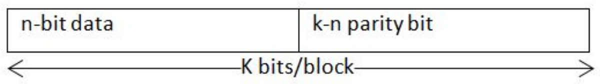


Fig. 3. Figure of Code Word.

B. *Hamming Distance*

There is a one to one mapping in each of the n-message bits to one point in high-dimensional space. The two points in that particular space have been very near and have only one-bit difference. i.e., hamming distance,d_H=1. Suppose one error occur in that particular different bit in any point, both the points are same and error cannot be detected. In order to avoid this scenario, we want to encode properly with a minimum hamming distance between any code word per point as high as possible [17].

Hamming distance is the number of bit positions at which two code words are different. Good encoding strategy is to choose the point in k-dimensional code word space in such a way that minimum hamming distance is maximum as possible. Hamming distance 2, d_H=2 same as parity checker and it can only detect the single bit error. To guarantee the detection of up to f errors in all scenario, the Minimum Hamming distance in a block code must be d_{min}= f + 1. To guarantee correction of up to s errors in all scenario, the minimum Hamming distance in a block code must be d_{min} = 2s + 1. Hamming distance, d_H=3 can correct single bit error or detect double bit error and hamming distance 4(SEC-DED) can be used to correct single bit error and detect a double bit error.

C. *Hamming Rule and Parity Bits*

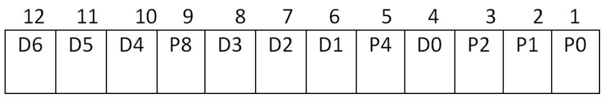
Hamming algorithm should use extra parity bits and it can be use to find the errors. Number of parity bit can find by using hamming rule[18].ie,

$$2^m \geq m + n + 1.........(HammingRule)$$

This rule is used to find the number of parity bits which depends upon the number of message bits. Hamming code can be converted to a SEC-DED code by adding one check bit, which is a parity bit on all the bits in SEC code word.

D. *SEC-DED Code Algorithm*

The most commonly used error correcting codes are SEC-DED codes[19]. The logic is very similar to that of the single error correcting code but we now need a hamming distance of four. One way to do this is by increment the check bit by one and using this bit as a parity for the check bits. Actually, one can think of the extra check bit as the XOR of the rest of the code word. In Figure 4 , 7- bit code require five parity bits that can be added at the end of the data unit. The parity bits are placed in positions 0,1,2,4 and 8. For clarity in the examples below, these parity bits are referred to as P0, P1, P2, P4 and P8[20].The value of the P can be determined by substituting the value of the original length of the data to be transmitted.



12	11	10	9	8	7	6	5	4	3	2	1
D6	D5	D4	P8	D3	D2	D1	P4	D0	P2	P1	P0

Fig. 4. Position of Parity Bits in SEC-DED Code.

In the SEC-DED code ,each m bit is the parity bit for one combination of data bits as shown in Table II.

TABLE II. PARITY BITS POSITIONS

Parity Bits	Positions
P8	9,10,11,12
P4	5,6,7,8
P2	3,4,7,8,11,12
P1	2,4,6,8,10,12
P0	2,3,4,5,6,7,8,9,10,11,12

Each data bit may be included more than one calculation. The new code word can detect double bit errors with single bit error correction that are not correctable[21]. For error detection and correction, all the parity groups are checked except P0.The corresponding decimal value of parity bits are called as the syndrome. This syndrome value and overall parity bit P0 shows the possible errors[22]. The possible error types and respective solutions are summarized in Table III.

TABLE III. ERROR DETECTION AND CORRECTION

Syndrome	Overall Parity Bit	Error Type
0	0	No Error
0	1	Overall Parity Bit Error
Not equal to zero	0	Double bit error
Not equal to zero	1	Single Bit error

Calculated syndrome and overall parity bits are zero and can represent the error-free state of FSM. The Figure 5 shows the syndrome and overall parity calculation. The overall parity

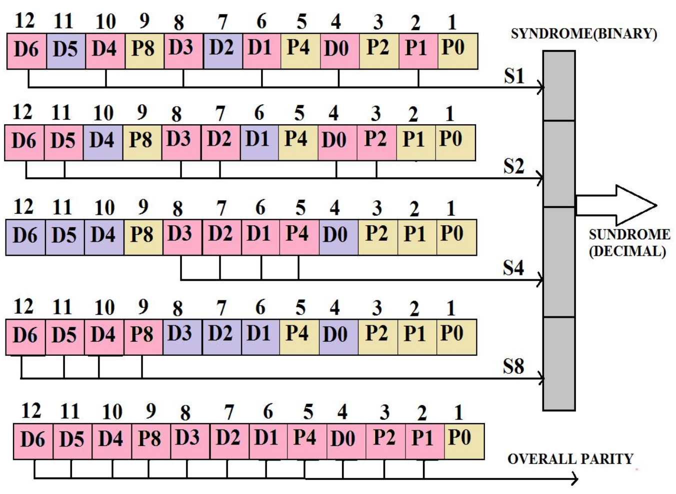


Fig. 5. Error Detection and Correction using SEC-DED Code.

can represent odd value and syndrome value is zero means error present in the LSB bit. If an error present in the parity bit, error correction is not taking place. In double bit error detection, if one of the two errors is in the overall parity bit, then the other error in the SEC portion of the block. Syndrome non-zero value is representing error present in the SEC portion. If the errors is present in the data section of the code word, then syndrome will also be non-zero, although error correction is needed in the data section.

V. PROPOSED FAULT TOLERANT FSM ARCHITECTURE

This paper presents design of a fault tolerant FSM using SEC-DED code. Normally, FSM implementation involves using error detection circuitry to force a state machine in to reset state or in to a user defined state. In this design increase the reliability of the system and errors within same clock cycle can be removed by the addition of parity bits using EDAC algorithm.

The proposed architecture is given in Figure 7. The proposed architecture have four sections.

- Parity Bit Generator(PBG)
- States and Parity bit Registers(SPR)
- Syndrome Generator(SG)
- Error correction Circuit (ECC)

Depending upon the number of state bits, a number of parity bits can calculate by using hamming rule and by using hamming encoding algorithm the PBG module give correct parity and the overall parity bit. The correct state and parity bits are stored in SPR module which contains flip- flops to store the corresponding bits. The present state and parity are generated by this module and present state output should give for FSM combinational circuit for finding the correct pattern. The Present state of the FSM can use for finding the syndrome value. The SG module has been identified unwanted upset hit in the present state or parity. The syndrome value and overall parity will be decided error present in the FSM. Depending upon the SG output one flag bit should set. The syndrome is representing an error in the parity bit, the flag bit should change to a low value. The SG module output representing double bit error, double bit error detected flag should set to high.

The Syndrome value and overall parity bit should decide the single bit error correction and double bit error detection. The ECC module can use for comparing the PBG output and SG output. Depending upon the flag bit received from the SG module, corresponding syndrome value representing bit position in SEC portion is flipped. The ECC module can compare the correct state and parity. The correct output is given for SPR module always.

This architecture should take place each clock cycle. It tries to maintain FSM in the original state. The SEC-DED code implemented FSM is used for correcting single bit error and double bit error detection. If any error occurs in the parity bits, comparator module will not try for error correction

VI. SIMULATION AND SYNTHESIS RESULTS

The proposed design of fault tolerant FSM is developed in Verilog HDL, simulated in VCS and synplify premier is the software tool used for synthesis in FPGA.

A. Simulation Results

The Figure 8 shows the result after simulation of 4 state fault tolerant FSM detecting the 1010 pattern.

B. Synthesis Results

The result were analyzed after synthesis Xilinx Virtex 7 XC7VX485T part FFG1157 package -1 speed grade FPGA for 4 state fault tolerant FSM to detect pattern 1010 . On analysis, it has been observed that fault tolerant FSM needs extra some flip-flops and combinational circuit for error free operations.

The Figure 6 shows the synthesis result in synplify premier by using Xilinx Virtex 7 (V7) FPGA. In this FSM needs only extra some flip-flops and combinational circuits.

TABLE IV.
PARAMETER ANALYSIS FOR FAULT TOLERANT 4 STATE FSM USING SEC-DED CODE FOR DIFFERENT PATTERN IN SYNPLIFY PREMIER

Performance parameters	1010	0001	1111
FPGA	V7	V7	V7
Number of FFs	08	08	08
Number of 2 input LUTs	01	0	01
Number of 3 input LUTs	04	0	04
Number of 4 input LUTs	03	06	04
Number of 5 input LUTs	11	12	11
Number of 6 input LUTs	06	07	05
Total number of LUTs	25	25	25
Number of bonded IOBs	33	33	33
Numbre of GCLK	1	1	1
Max frquency of operation(MHz)	388.7	339.6	419.58

The Table IV shows parameter analysis of fault tolerant FSM. Here we are given 3 different patterns and corresponding implementation analysis. From that table, we can observe that minimum number of sequential elements used and depending upon the pattern frequency of operation is changing. Synplify premier is the industry's most advanced FPGA design and

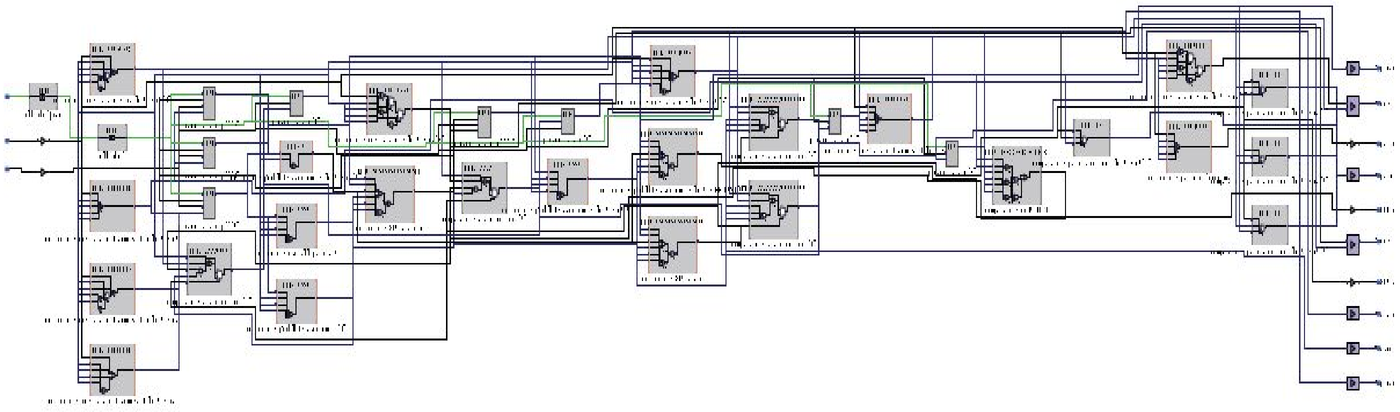


Fig. 6. Fault Tolerant FSM using SEC-DED Code Algorithm in Xilinx Virtex 7.

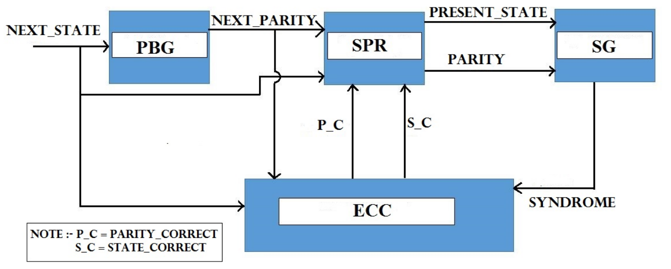


Fig. 7. Block Diagram of Fault Tolerant FSM using Hamming 4 Algorithm.

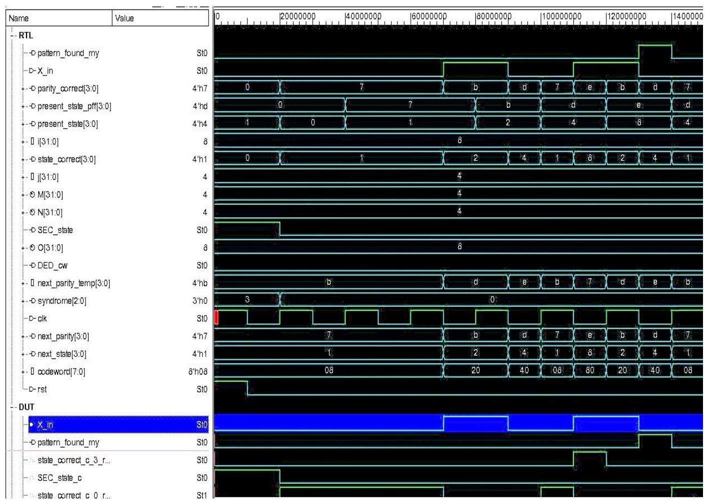


Fig. 8. Waveform of Fault Tolerant FSM using SEC-DED Code Algorithm.

debugs tool. The tool provides good performance, fast runtime, and area optimization for cost and power reduction. The tool includes features that automate the creation of high reliable designs. The tool is used to implement the fault tolerant FSM in Xilinx Virtex 7 FPGA.The four state FSM require extra 4 flip-flops to store the parity bit and some combinational circuits only. i.e, the FSM output should obtain with less time and error correction is take place in each clock cycle.

VII. Conclusion and Future Scope

This paper presents a fault tolerant FSM using SEC-DED code. The error detection and correction FSM is introduced by using SEC-DED code in each states. The design structure proposed in the paper can be configured for any type of state machines. This type of FSM can correct single bit error and detect double bit error. The analysis of this work shows that fault tolerant FSM using hamming 4 protect any type of FPGAs from soft errors and error checking occur in each clock cycle. This type of FSM requires extra parity flip-flops and the combinational circuits. This architecture is more reliable and area of the FSM is increased due to extra flip-flops and combinational circuit. The FSM is always working in the safe condition and we can operate any system without any damage and it is more reliable.

As future scope, the multi-bit error correction structure can be used and it will increase the reliability of the system.

References

[1] Mohr, Karl C., Giby Samson, and Lawrence T. Clark, “A Radiation Hardened by Design Register File With Lightweight Error Detection and Correction,IEEE Trans. On Nucl. Sci.” vol. 54, pp. 1335–1342, August 2007.

[2] M. Tang, G. Zhang, and H. Zhang, “Auto error correction hardware implementation of AES algorithm,Acta Ele. Sin,” vol. 33, pp. 2013–2016, November 2005.

[3] R. Xu, Z. Gu and J. Luo, “Design of radiation hardened error detection and correction circuit,Microelectronics,” vol. 40, pp. 547–550, Aug 2010,.

[4] K. H. Yearby,M. Balikhin, S. N. Walker, “Single-event upsets in the cluster and double star digital wave processor instruments,pace Weather,” vol. 12, Jan. 2014 .

[5] Laura Frigerio and Fabio Salice , “RAM-based fault tolerant state machines for FPGAs,22nd IEEE International Symposium on Defect and Fault-Tolerance in VLSI Systems,” pp. 312–320, Sept. 2007.

[6] M. Cassel and F. Lima,, “Evaluating one-hot encoding finite state machines for SEU reliability in SRAM-based FPGAs,12th IEEE International,On-Line Testing Symposium,” Apr. 2006.

[7] Wilson José , Ana Rita Silva and Horácio Neto, “Analysis of matrix multiplication on high density Virtex-7 FPGA, 23rd IEEE on Field Programmable Logic and Applications (FPL),” 2013.

[8] S. Niranjan and J. F. Frenzel , “A comparison of fault-tolerant state machine architectures for space-borne electronics,IEEE Transactions on Reliability,” vol. 45, pp. 109–113, 1996.

[9] R. Senhadji-Navarro, I. Garcia-Vargas, G. Jimenez-Moreno and A. Civit-Ballcels , “ROM-based FSM implementation using input multiplexing in FPGA devices,Electronics Letters,” vol. 40, pp. 1249–1251, 2004.

[10] R. Leveugle, R. Rochet, G. Saucier, L. Martinez and C. Pitots , “A synthesis tool for fault-tolerant finite state machines,FTCS-23 The Twenty-Third International Symposium on Fault-Tolerant Computing,” pp. 502–511, 1993.

[11] A. Tiwari and K. A. Tomko , “A synthesis tool for fault-tolerant finite state machines,IEEE Transactions on Reliability,” vol. 54.

[12] Ramesh Bhakthavatchalu and N H N Sai Kiran , “Implementing delay based physically unclonable functions on FPGA,Advanced Communication Control and Computing Technologies (ICACCCT),” 25-27 May 2016.

[13] Ramesh Bhakthavatchalu, Ammu Kripalal and Suchitra Nair , “Modified FPGA based design and implementation of reconfigurable FFT architecture,Automation, Computing, Communication, Control and Compressed Sensing (iMac4s),” 22-23 March 2013.

[14] A. Tiwari and K.A. Tomko, “Enhanced reliability of finite-state machines in FPGA through efficient fault detection and correction,IEEE transaction on reliability,” vol. 54, sep 2005.

[15] Kamil Mielcarek , Alexander Barkalov and Larisa Titarenko, “Designing LUT-based mealy FSM with transformation of collections of output functions,5th IEEE on Modern Circuits and Systems Technologies (MOCAST),” 2016.

[16] R. W. Hamming , “Error detecting and error correcting codes,The Bell System Technical Journal,” vol. 29, april 1950.

[17] Eng-Jon Ong and Miroslaw Bober , “Improved Hamming Distance Search Using Variable Length Hashing,IEEE on Computer Vision and Pattern Recognition (CVPR),” 2016.

[18] Ronnie O. Serfa Juan , Min Woo Jeong and Hyeong Woo Cha , “FPGA implementation of hamming code for increasing the frame rate of CAN communication, IEEE Asia Pacific Conference on Circuits and Systems (APCCAS),” 2016.

[19] Yuanyuan Cui , Mian Lou and Jianqing Xiao , “Research and implementation of SEC-DED Hamming code algorithm,IEEE Region 10 Conference on TENCON ,” 2013.

[20] Karl C. Mohr , Giby Samson and Lawrence T. Clark, “A Radiation Hardened by Design Register File With Lightweight Error Detection and Correction,IEEE Transactions on Nuclear Science,” vol. 54, 2007.

[21] M. Y. Hsiao , “A Class of Optimal Minimum Odd-weight-column SEC-DED Codes,IBM Journal of Research and Development ,” vol. 14, july 1970.

[22] C.G. Oh , H.Y. Youn and V.K. Raj , “An efficient algorithm-based fault tolerance design using extended rearranged Hamming checksum,IEEE International Workshop on Defect and Fault Tolerance in VLSI Systems,” vol. 14, 1992.