SELF-CHECKING APPROACH FOR REDUCING SOFT ERRORS IN STATES OF FSM

B.Adichenchaiah[1], S.Arunamastani[2]

[1]*M.tech, ECE Department, JNTUACEA, Ananthapur, Andhra Pradesh, India*
[2]*Assistant Professor, ECE Department, JNTUACEA, Ananthapur, Andhra Pradesh, India*

Abstract

Due to reduction in device feature size and supply voltages the probability of soft-errors in Finite State Machines' (FSMs) states has increased dramatically, and the protection against both Single Event Upset (SEU) and Multiple Bit Upsets (MBUs) soft-errors demand for design of fault tolerant FSMs that detect and correct more than one error. Redundancy has been mostly preferred methodology for Error Detection and Correction (EDAC), however selection of one EDAC method is a trade-off between performance and hardware overhead. In this paper, we present an SEU/MEU hardening approach for FSMs' states through 'binary-gray' code for state encoding and a self-checking process that can detect and correct the soft errors in FSM states. Here 8 bit register is used to store the FSM states using 'binary-gray' code, this approach can detect errors until the integer value of binary is not equal to integer value of gray for error state, which is a sparse situation. The little overhead of hardware provided by the self checking block implemented in FSM gives 100% error correction. The Experimentation is performed on a 16 state FSM through bit flip fault injection. The simulation results of bit-flip injection into the FSMs' state registers are analyzed and compared with the existing one-hot × m & self-checking method [1].

***Keywords** - FSMs, SEU, MBUs, EDAC*

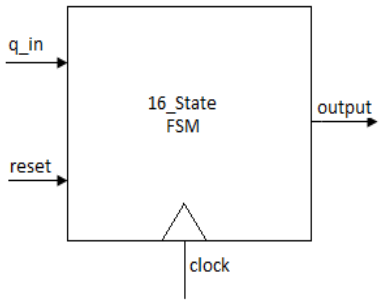
---***---

1. INTRODUCTION

The operation of FSM's produced in nanometre technology and supply voltages decreases. These causes the decrease of cut off charge, which makes the storage cells (flip-flops, memories etc.,) be more likely to get flipped due to Single Event Upsets (SEUs) [2]-[3]. Multiple Bit Upsets (MBUs) induces in adjacent storage cells becomes higher as circuit density [4]-[5]. So, it is important to design hardened FSMs against soft-errors.
Sequential circuit is the circuit that depends on the past behaviour of the circuit, as well as on present values of the inputs. Sequential circuits are also called as FINITE STATE MACHINES (FSMs). Generally FSMs are classified into two types
1. Mealy machines,
2. Moore machines.

A Mealy FSM is a state machine where the outputs are function of present state and inputs and a Moore FSM is a state machine where the outputs are only function of inputs. For both these types the output logic and next state logic are combinational circuits and the state registers are composed of several flip-flops to store the current state.

The soft errors occurred in state register may make an FSM fall into un- defined state [6] and then FSM goes to sudden reset. So it is important to concentrate on FSM states against soft-errors propagation has to be solved otherwise technology will be blocked soon. In present technology 16-state FSMs are large state machines in digital circuits. So here we are taking a 16-stae FSM for the experimentation against soft-errors. This FSM is vending that has only one input coin slot that accepts only one rupee coins as inputs and after accepting a sixteen one rupee coins it will give the output that may be a any product. The block diagram of the FSM and its state table as shown in Fig. 1



(a)

Table 1: State Table

Present-state	**Next_State q-in=0**	**q-in=1**	**Output**
S0	So	S1	0
S1	S1	S2	0
S2	S2	S3	0
S3	S3	S4	0
S4	S4	S5	0
S5	S5	S6	0
S6	S6	S7	0
S7	S7	S8	0
S8	S8	S9	0
S9	S9	S10	0
S10	S10	S11	0
S11	S11	S12	0
S12	S12	S13	0
S13	S13	S14	0
S14	S14	S15	0
S15	S15	S0	1

(b)

Fig 1 (a) Block diagram (b) State table, for 16-state FSM

To protect the FSM's states against soft-errors some approaches are implemented based on redundancy techniques such as EDAC methods. As probability of SEU and MBUs increases design a fault tolerant FSMs requires more than single error correction and detection. In the One-Hot×m &self-checking method, Here an soft-errors hardening approach for FSMs states through replicating of one hot code 3 times (one-hot×3) for state encoding and self-checking. So for an FSM containing 16 states, a state register of size 16×3 that is 48 flip-flops are needed. This approach has able correct only less than 3 bit-flip faults occurred in state register of size 48 flip-flops per cycle.

Here we propose a new approach to enhance the protection of FSM's states against soft-errors. In this approach a new encoding technique is called 'binary-gray' code is used to represent the states of FSM. So for an FSM containing 16 states, only a state register with size 2×4 that is 8 flip-flops are needed and for error detecting and correcting in state register a additional self-checking combinational circuit is involved to retrieve the correct next state logic for error state. This approach can detect errors until the integer value of binary is not equal to integer value of gray for error state and gives 100% error correction for error state.

This section provides an overview of the work presented in this paper. The rest of this paper is organized as follows. Section 2 briefly describes the One-Hot × M & self-checking method and its limitation. Section 3 describes the proposed method. Section 4 deals simulation results and discussions followed by conclusions.

2. ONE-HOT×M & SELF-CHECKING METHOD

The replication of one-hot code method [1] presents an SEU/MBUs hardening approach for FSMs' states. In this approach a replication of One-Hot code is used to represent the states of FSM and a combinational logic is involved for Self-Checking of SEU/MBUs in states of FSM.

2.1 One-Hot×3 Code

To protect the FSM states against SEU/MBUs a replication of One-Hot code is used for state encoding. A three times replication of One-Hot code (One-Hot×3) for S1 state for 16-state FSM is represented in Fig. 2. It requires 48 bits to represent the 16 state FSM.

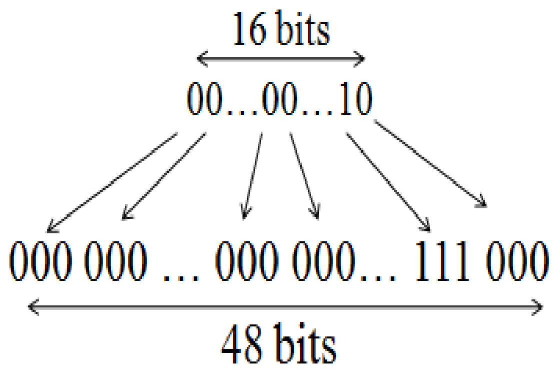


Fig 2 The 3 times replication of One-Hot code (One-Hot ×3)

Each bit of the original One-Hot code is copied 3 times and expanded to a group.

These 48 bits are divided into 16 groups, one ALL-HIGH group and 15 ALL-LOW group. In the hardware implementation, it requires one flip-flop to store one bit of each group. Hence a state register containing 16×3 that is 48 flip-flops are needed for 16-state FSM. In general for M times replication for 16-state FSM it requires a state register that containing 16×M flip-flops.

2.2 Self-Checking Circuit

In addition a Self-Checking combinational logic is involved within in the FSM to correct the SEU/MBUs in the states of FSM. This Self-Checking block is designed for identifying which group contains most bits HIGH, and this result is used for generating the hardened One-Hot state value for the output logic and next state logic. One-hot×3 and its self-checking block for 16-state FSM is in Fig.1. is illustrated in Fig.3.

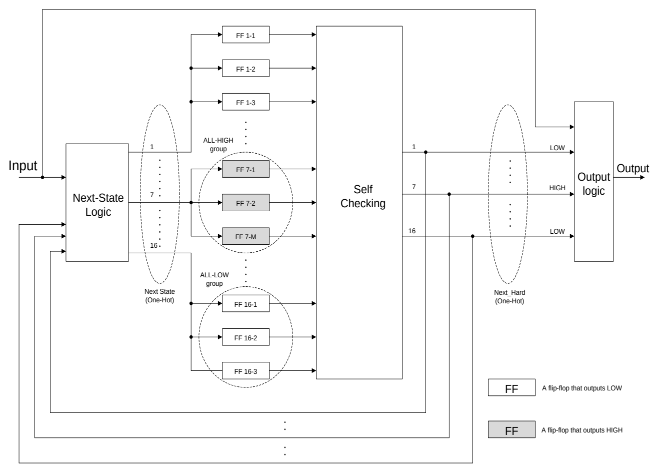


Fig 3 The ordinary structure applying ONE-HOT state encoding of an FSM. For the FSM hardened through One-hot×3, its state register contains 16×3 flip-flops .these flip-flops are divided into 16 groups. One ALL- HIGH group and 16-1 ALL LOW groups. The self-checking module is used to identify which group possesses the most bits HIGH and assert the corresponding bit of its output state_hard.

As shown in Fig. 3. The state register contains 16 groups. One group has3 bits HIGH (ALL-HIGH group) and 16 - 1 that is 15 groups each containing 3 bits LOW (ALL LOW group) for the 16 state FSM. A self-checking module is used to identify which group contains most bits HIGH (ALL-HIGH GROUP) and then generate the hardened One-Hot state value state-hard for the next state logic and output logic. If the group 7 possesses the most bits HIGH, the bit 7 will be asserted as state-hard that is a high state.

This approach fails when several groups possess the same maximum number of bits high respectively. This approach has the ability to correct less than 3 bit-flip faults occurred in the state register of 48 flip-flops per cycle. If the replications are increased to M times but it has able to correct less than M bit-flip faults occurred in the state register. At maximum this approach corrects only 2 bit flip-faults in state register of size 48 flip-flops for 16-state FSM. As states of FSM increases it requires large size of state register. The improved solutions of one-hot ×m with self-checking method called state reforming are also presented. State reforming means combining several short One-Hot codes for state encoding and replicating them. In One-Hot×3 (4×4) the output of the next state logic is replaced by a two 4-bits one hot codes and replicating three times, and two self-checking modules designed to implement the SEU/MBUs hardening for two replications respectively. Similarly in One-Hot× 3 (2×2×2×2) the next state logic is replaced by four 2-bits one hot codes and replicating. These state reformed solutions reduces the state register size to 24 with simple self-checking circuits

Even state reformed solutions requires 24 flip-flop state register and also has less probability of detection. To overcome the drawback of large state register size and to improve the probability of detection Here we present an state register with 8 flip-flops using a new encoding logic which is a combination of binary code and gray code that is a ’binary-gray’ code, and a special combinational circuit is presented with in the FSM for error correction and detection.

3. PROPOSED METHOD

In present technology states of FSM increases continuously, so it is required to choose minimum size of state register and to protect the FSM’s states against SEU and MBUs. In the proposed method to protect the FSM’s states against SEU and MBUs using 8 bit state register, we present a new encoding technique ’binary-gray’ code which decreases the state register size and a combinational circuit has LUT and gray code comparator for self-checking.

3.1 ’Binary-Gray’ Code

Table 2: ‘Binary-gray’ code for 16 State FSM

state	**‘Binary-gray’ code**
S0	00000000
S1	00010001
S2	00100011
S3	00110010
S4	01000110
S5	01010111
S6	01100101
S7	01110100
S8	10001100
S9	10011101
S10	10101111
S11	10111110
S12	11001010
S13	11011011
S14	11101001
S15	11111000

In the hardware implementation, one flip-flop requires to store one bit of each group. Hence a state register containing 2×4 that is 8 flip-flops needs for 16-state FSM. These 8 flip-flops are divided into two groups, one group represents the binary code and second group represents the gray code.

3.2 Self-Checking Circuit

In addition a self-checking circuit is used within the FSM for error detecting and correcting the states and to retrieve the correct state value when error occurred in current state. In this approach all states FSM have to be stored in memory. This approach requires only 16×8 that is 256 bits of memory for 16 state FSM, here LUT is taken to store the states of FSM and a comparator is used to compare the binary values and gray values.

3.2.1 Look up Table

A look up table (LUT) is a memory with a one-bit output that essentially implements a truth table where each input combination generates a certain logic output. The input combination is referred to as an address. The HDL synthesizer implements an AND gate or other simple logic function by programming the stored elements in a LUT. The LUT used here to retrieve the next-state logic for error state.

3.2.2 Binary-Gray Code Comparator

The binary-gray code comparator compares binary value of the first half bits in state register (current state) with gray value of the next half bits in state register. This comparator used here for the purpose of detection of soft-errors in state register.

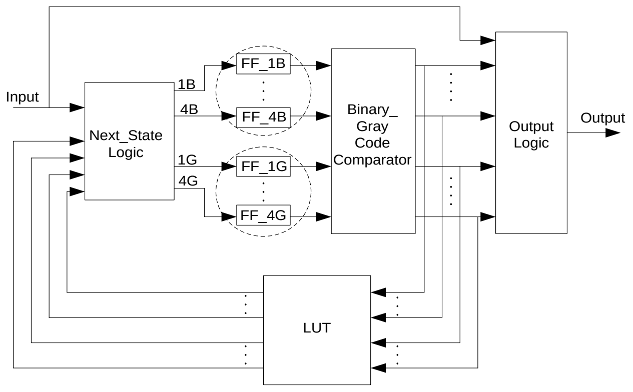


Fig. 4 Memory based Binary-Gray' encoded FSM.

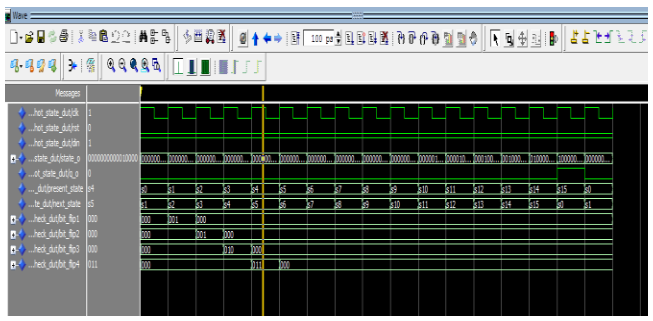
The 'binary-gray' code with LUT finite state machine (FSM) is illustrated in Fig.4 (Mealy type). As shown in Fig. 4.To represent the 16(=2^4) state FSM it takes 2×4 bits. The first 4 bits represent the binary code and next 4 bits represent the gray code. In the hardware implementation, each bit of each group is stored by a flip-flop. Hence a state register containing 2×4 that is 8 flip-flops are needed in this approach.

As shown in Fig. 4 binary-gray code comparator is used to compares the first four bits of binary value with next four bits of gray value. If the both values are not equal that means the error occurred in state register, the next state is taken from the LUT which is stored by all states of FSM from the FSM logic. Otherwise if both values equal that means there is no error occurred in state register in that there is no need to go to the LUT to retrieve the next state logic the FSM execution goes normally which is description vhdl.

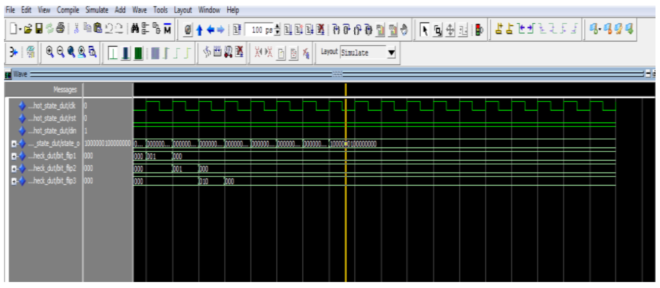
This approach fails when both binary value and gray value are equal for error state that is error occurred in present state. Fig.4, shows that no encoding logic is needed for hardware implementation of this approach. This approach can detect errors until the integer value of binary is equal to integer value of gray for error state. With little overhead of hardware provided by the self checking block implemented in FSM gives 100% error correction.

4. RESULTS AND DISCUSSIONS

The simulation results are analysed for both one one-hot×m &self-checking method and proposed method through bit-flip injection in state register. Simulation results for one hot method are shown in Fig. 5. Here we are injecting one bit-flip at s3 state and two bit-flips at s4 states respectively and this method recovers the next state (a) and this approach fails with 3 bit-flip injections at 16th Position of s9 state and it will change like 0000000100000000->1000000100000000 this is the undefined state and FSM stops execution (b). Simulation results for proposed method are shown in in Fig. 6.without bit-flip injection (a) and injecting a bit –flip at s3 state position and it recovers correct s4 state after one clock cycle (b).

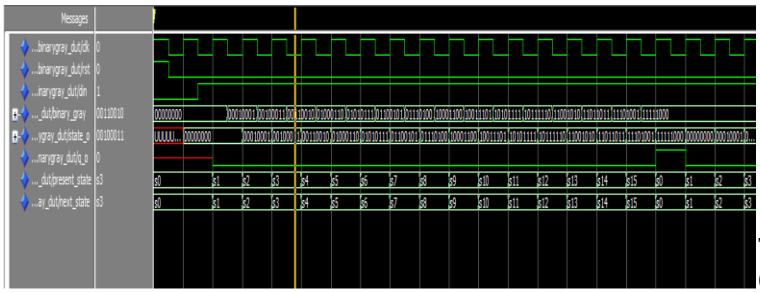


(a)

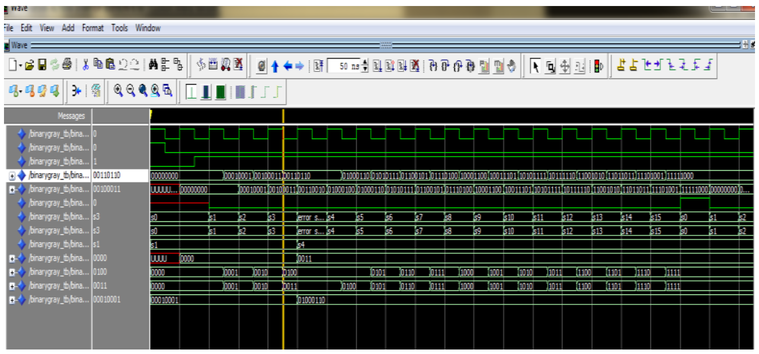


(b)

Fig 5: Simulation results for One-Hot × m & self-checking through bit-flip injection in state register, (a) FSM recovers next-state logic for less than 3 bit-flip faults.FSM fails to recover the next-state logic because two groups of one hot code possess the same maximum number bits HIGH.



(a)



(b)

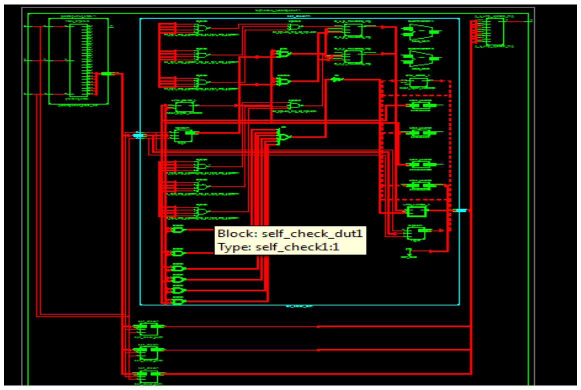
Fig 6: Simulation results for proposed method C) without bit-flip injection in state register D) with bit-flip injecting at S3 state recovers the next state that is S4 state with one clock period delay.

The synthesis results for state reformed solutions one-hot×m (2×2×2×2) of one-hot method and proposed method are shown in Fig. 7 (a) & (b) respectively. From the synthesis results observed, the proposed method has little hardware overhead because of using LUTs. If LUT is used for storing the Xilinx selects automatically unnecessary gates that is MUX, AND gates this will increases the hardware complexity. The hardware complexity can be reduced by using block_ram and this uses less number of gates for storing 256 bits of memory. The proposed method detects 2 bit-flip faults in 8-bit state register and one-hot method detects 2 bit-flip faults in 48-bit state register so the probability detection of proposed method is high. The proposed method has 100% error correction with little hardware over head when error occurred in present state.

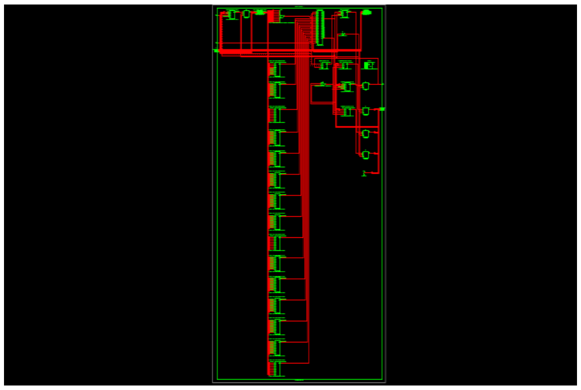
Table 3: Comparisons among the hardware implementations of one-hot×3(16), its state reformed solutions and proposed method.

Encoding technique	**No. of flip-flops needed to store current state**	**No.of inputs of each self-checking block**
One-Hot×3(16)	48	48
One-Hot×3 (4×4)	4	12
One-Hot×3 (2×2×2×2)	24	06
Proposed 'binary-gray'	08	08

The synthesis results for state reformed solutions one-hot×m (2×2×2×2) of one-hot method and proposed method are shown in Fig. 7 (a) & (b) respectively. From the synthesis results observed, the proposed method has little hardware overhead because of using LUTs. If LUT is used for storing the Xilinx selects automatically unnecessary gates that is MUX, AND gates this will increases the hardware complexity. The hardware complexity can be reduced by using block_ram and this uses less number of gates for storing 256 bits of memory. The proposed method detects 2 bit-flip faults in 8-bit state register and one-hot method detects 2 bit-flip faults in 48-bit state register so the probability detection of proposed method is high. The proposed method has 100% error correction with little hardware over head when error occurred in present state.



(a)



(b)

Fig. 7 Synthesis results for (a) one-hot×m (2×2×2×2) (b) proposed method.

5. CONCLUSIONS

In this paper a self checking approach to enhance the SEU/MBUs immunity of FSMs' states has been discussed. This approach uses 'binary-gray' code for state encoding and has ability to detect errors until the integer value of binary is not equal to integer value of gray for error state, which is a sparse situation and with little overhead of hardware provide by the self checking block implemented in FSM gives 100% error correction for error states. This approach gives higher reliabilities by reducing the number of flip-flops needed for storing state values. Simulation results of bit-flip injections further supported and verified the conclusions of theoretical analysis above.

This work has been successfully designed using VHDL and simulated using Model Sim, synthesized using Xilinx tool.

FUTURE SCOPE

Actually 16×8 that is a 256 bits of memory required to store the states of 16-state FSM in memory. By using LUTs for storing the un necessary gates like MUX gates, AND gates are selected this increases gate count for this design. There is a chance to reducing the hardware complexity of FSM by using block ram to store the states instead of LUTs.

REFERENCES

[1]. Li Yuanqing, Yao Suying, Xu Jiangtao and Gao Jing, "A Self-Checking Approach for SEU/MBUs-Hardened FSMs Design Based on the Replication of One-Hot Code" IEEE TRANSACTIONS ON NUCLEAR SCIENCE, VOL. 59, NO. 5, OCTOBER 2012.
[2]. M.J.Gadlage,R.D.Schrimpf,B.Narasimham,B.L.Bhuva,P.H. Eaton, and J. M. Benedetto, "Effect of voltage fluctuations on the single event transient response of deep submicron digital circuits," IEEE Trans. Nucl. Sci., vol. 54, no. 6, pp. 2495–2499, Dec. 2007.
[3]. S. M. Jahinuzzaman, M. Sharifkhani, and M. Sachdev, "An analytical model for soft error critical charge of nanometric SRAMs,"IEEE Trans. Very Large Scale Integr. (VLSI) Syst., vol. 17, no. 9, pp.1187–1195, Sep. 2009.
[4]. S.-F. Liu, G. Sorrenti, P. Reviriego, F. Casini, J. A. Maestro, and M. Alderighi, "Increasing reliability of FPGA-based adaptive equalizers in the presence of single event upsets,"IEEE Trans. Nucl. Sci.,vol.58, no. 3, pp. 1072–1077, Jun. 2011.
[5]. M. Zhu, L.-Y. Xiao, C. Liu, and J. W. Zhang, "Reliability of memories protected by multibit error correction codes against MBUs,"IEEE Trans. Nucl. Sci., vol. 58, no. 1, pp. 289–295, Feb. 2011. [6] Y. Yahagi, H. Yamaguchi, E. Ibe, H. Kameyama, M. Sato, T. Akioka, and S. Yamamoto, "A novel feature of neutron-induced multi-cell upsets in 130 and 180 nm SRAMs,"IEEE Trans. Nucl. Sci., vol. 54, no. 4, pp. 1030–1036, Aug. 2007.
[6]. Y. Yahagi, H. Yamaguchi, E. Ibe, H. Kameyama, M. Sato, T. Akioka, and S. Yamamoto, "A novel feature of neutroninduced multi-cell upsets in 130 and 180 nm SRAMs,"IEEE Trans. Nucl. Sci., vol. 54, no. 4, pp. 1030–1036, Aug. 2007.