International Journal of Engineering and Advanced Technology (IJEAT)
ISSN: 2249-8958 (Online), Volume-8 Issue-6S3, September 2019

Implementation of Logic Fault Detection Techniques using FPGA

N. MohanaSundaram, S. Arunkumar, A. Mahalingam

Abstract: ***The evolution of automation concept in industries and rising significance for digital mode of communication have in turn stresses the need for reliable, efficient and high performance digital system. One of the major problems faced in the digital transmission is the loss of data and hence an effective technique to detect and correct these faults is necessary. Conventionally, various techniques are used for this purpose. Among them, three of the techniques like Plain Majority Logic (ML) Decoder, Syndrome Fault Detector and Plain ML Detector Decoder (MLDD) are analyzed and compared in this paper. The purpose of this comparison is to identify the most suitable methodology based on its capability to track maximum number of faults in minimum number of cycles, with reduced memory read access time. All these logics are simulated using HDL code and analysis is done to implement it in FPGA. The results reveal that Plain MLDD is found to provide better solution for various digital circuits than other techniques.***

Index Terms: ***Majority Logic Decoder (MLD), Syndrome Fault Detector (SFD), Majority Logic Detector Decoder (MLDD)***

I. INTRODUCTION

Memories are the most common unit in all digital systems, where storage and retrieval of data are involved. Though the advancement in technologies helped in building advanced digital circuits, reliability of memory is getting affected especially due to Soft errors [1]. For that error correction codes have been in usage for many years. Single Error Correction and double error detection technique can correct one error and detect two errors [2]. They are good solution for low soft error rate. Because of integration densities, there is a need for high error correction capabilities [3,10].

Other commonly used techniques are triple modular redundancy and error correction codes. The former consumes more power when compared to the latter. Error correction codes meet the requirements of high error correction capability and low decoding complexity. Hence, error correction codes are best suitable to fix soft errors in memories [9].

Manuscript published on 30 September 2019.
* Correspondence Author (s)
N. MohanaSundaram, Assistant Professor, Kumaraguru College of Technology, Coimbatore, Tamilnadu, India.(Email: mohanasundaram.nms@gmail.com)
S. Arunkumar, Assistant Professor, Kumaraguru College of Technology, Coimbatore, Tamilnadu, India.(Email: akumar5989@gmail.com)
A. Mahalingam, Assistant Professor, Sri Shakthi Institute of Engineering and Technology, Coimbatore, Tamilnadu, India.(Email: mahalingamaruchamy@gmail.com)

The common multi error correction codes such as Reed-Solomon or Bose-Chaudhuri-Hocquenghem are not preferred due to its complexity. Difference-set cyclic codes wereproposed by Rudolph [4] and Weldon [5] and are based on the concept of a perfect difference set. Among them, Majority Logic (ML) decodable code is a simple and easier technique, which is a sub group of low density parity check codes. The main drawback of ML decoding is for a word of N bits it takes N cycles in the decoding process. This affects the system performance. To overcome this problem parallel encoders and decoders can be used. But it will increase the complexity and power consumption. Thus decoder works inefficiently most of the time.

Fault Detector Module triggers the correction mechanism after checking the error. It helps to reduce memory access delay. It is implemented by calculating syndrome, but it requires additional complex functional unit. This paper explores the idea of using best fault detection techniques by comparing ML decoding, syndrome fault detection and majority logic detector and decoder.

II. SIMPLE MEMORY SYSTEM

A common system of memory is shown in Fig. 1. First N bits are encoded and then it is stored in memory. The bits are then fed through decoder, after read from memory. The N bits coming out from the memory may suffer from soft errors. Hence detection logic is implemented in the decoder part of the normal operation. This detector will detect the faults present in N bits which are stored in memory and retrieved.

There are two types of encoders, one is of calculating the syndrome and another one is to correct current bit under decoding process. Calculation of syndrome is to identify the faulty bit from the N bits but will increase the area and complexity. The majority logic decoder is based on the number of parity check equations, which are orthogonal to each other. Each iteration and each code will participate in one parity check equation, except the very first bit which contributes to all equation. It will correct the current bit under decoding.

Logic behind the fault detection is based on XOR gate which gives high output for different inputs and low output for common inputs.

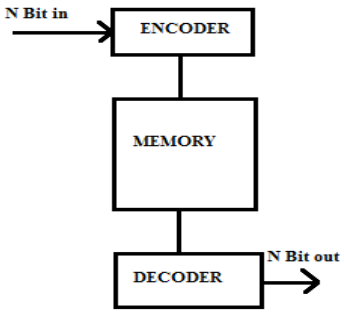

Fig. 1. Block diagram of Simple Memory System

III. LOGIC FAULT DETECTION TECHNIQUES

A. Normal Majority Logic (ML) Decoder

The fault present in the input can be corrected by ML decoder, a simple and efficient method. It works based on parity check equation, to correct multiple bit flip in the N code input bits stored in memory. ML decoder consists of four units namely, cyclic shift register, an XOR matrix, a majority gate and an XOR operation for correcting the codeword bit under decoding. The operation is explained below.

Input N bits are initially stored in cyclic shift register and shifted to all the units. An N bit input will take N cycles to produce the output signal. If the input bit has flip in it, then after processing the decoder will calculate parity check equation which is connected to XOR matrix. In next step, it will compare to majority gates for checking the correctness. If the number of 1s produced is greater than number of 0s then current bit under decoding should be corrected. This procedure is repeated for each bit in N bit input. For correct bit, parity checksum should be zero.

This method is useful only for simple bits, but will affect the system performance when applied for more number of bits.

B. Normal ML Decoder with Syndrome Fault Detector

To improve the performance of a normal ML Decoders, when operating with more number of bits, a new unit called Syndrome Fault Detector (SFD) is added. As most of the bits are error free in general, the performance will not be affected. Thus by adding this unit, average latency is reduced. Change in MLD and SFD will produce syndrome based parity check matrix. Faulty bit is detected by the syndrome equation. MLD will work normally until SFD activates the fault detection unit. As error free bit skips this step, error correction cycles are saved.

The disadvantage in this method is the complexity in design. This is because each bit will produce a syndrome equation, which will increase with increase in bits. Also SFD will consume more power and size of the system is little more.

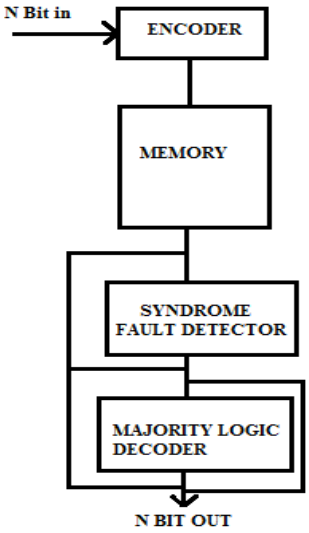

Fig. 2. ML Decoder with Syndrome Fault Detector

C. Majority Logic Detector and Decoder (MLDD)

Difference-set cyclic codes plays an important role in MLDD, which makes error detection simple. Majority Logic Detector and Decoder (MLDD) will correct large number of errors and occupies less area when compared to other two.

A technique is introduced to identify the bit which is not detected by parity check equation. This will identify up to five bit flips in it. When comparing MLD and SFD, MLDD will take less number of cycles to identify the bit flips.

Though the decoding process is same for all the three, in MLDD it will stop in third cycle. As in MLD, the intermediate values are analyzed and if number of 1s is more than number of 0s whole decoding process will takes place again.

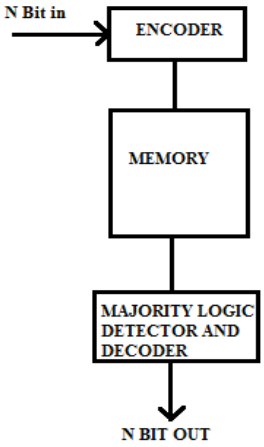

Fig. 3. Majority Logic Detector and Decoder (MLDD)

The detailed diagram of MLDD is shown in Fig. 4. In MLDD, in addition to four units of MLD, control logic and tristate buffer are added to improve system performance and efficiency.Control logic is the main unit which consists of two OR gates and two shift registers with a finite state machine. These are responsible for counting the number of cycles.

The tristate buffer is always in high impedance state until control unit triggers it. Finite state machine is activated only when the result is 0. Otherwise it will run normally for all bits.These extra units are added to separate first three cycles and last three cycles to find the faulty bits present in the input. Thus the structure of difference-set cyclic codes provides facility to identify faults in large number of bits in a simple manner.

IV. SIMULATION RESULTS AND ANALYSIS

The different logic fault detection techniques are simulated using VHDL and the simulation results are presented below. Fig. 5.shows the initial process of majority logic fault detection technique. The output of majority logic fault detection technique, syndrome fault detector and majority logic detector and decoder are shown in Fig. 6, 7 and 8.

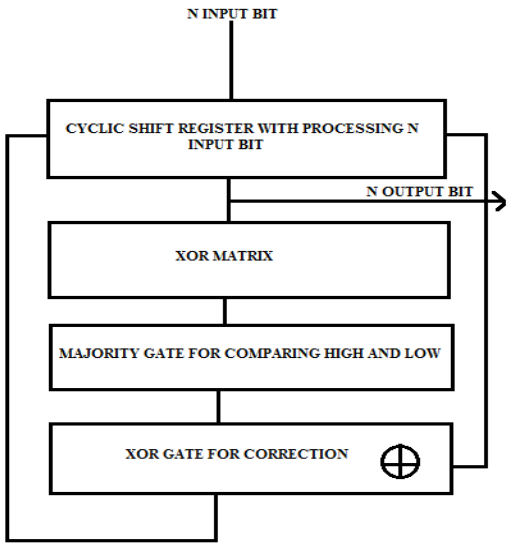

Fig. 4. Detailed representation of MLDD

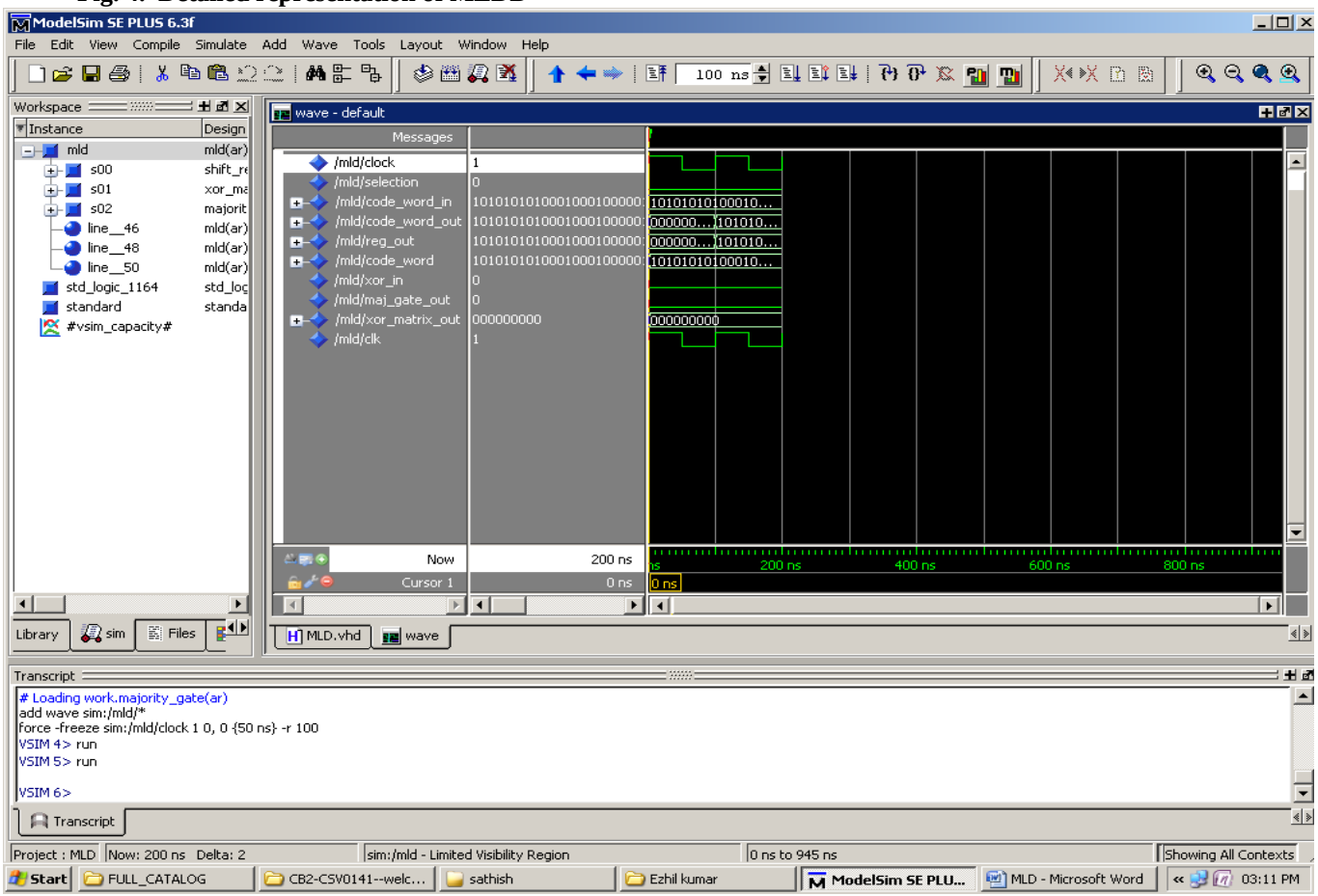

Fig. 5. Initial Input

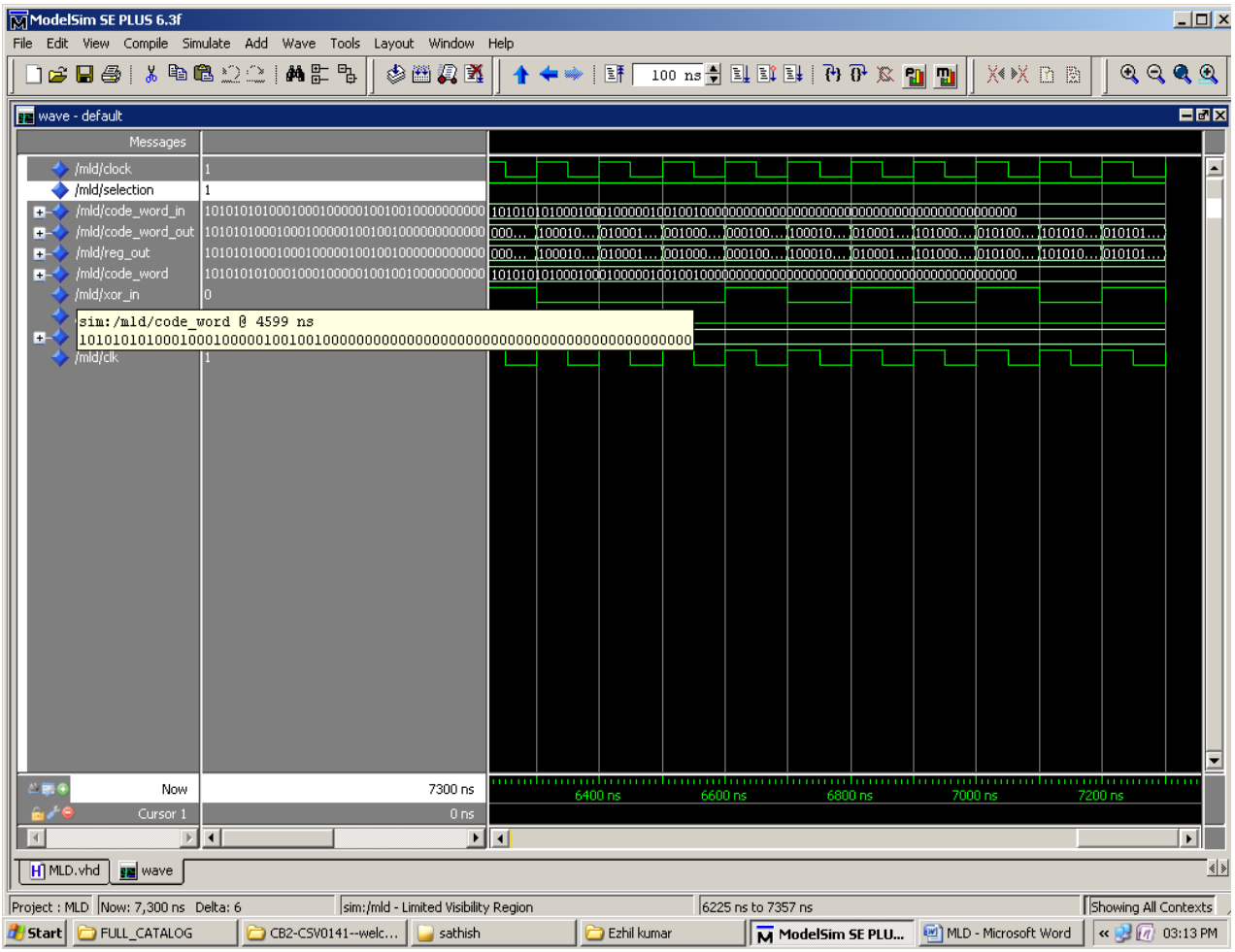

Fig. 6. Output of MLD

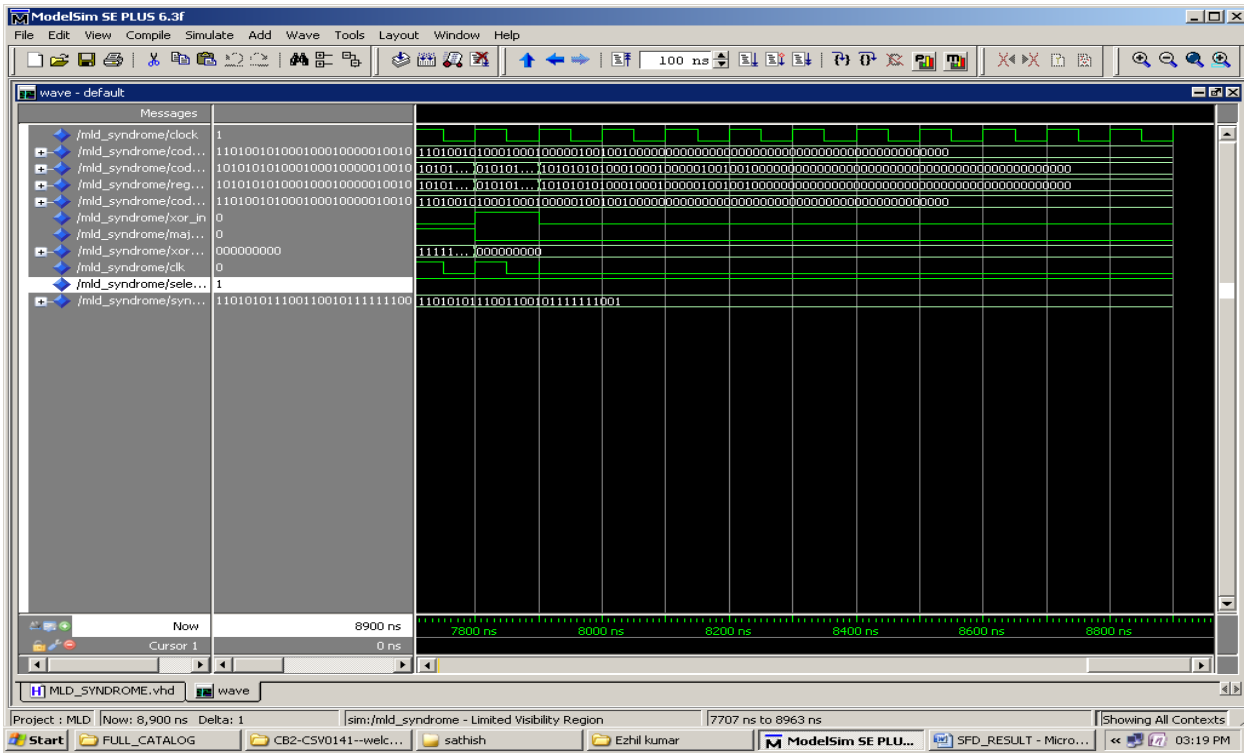

Fig. 7. Output of SFD

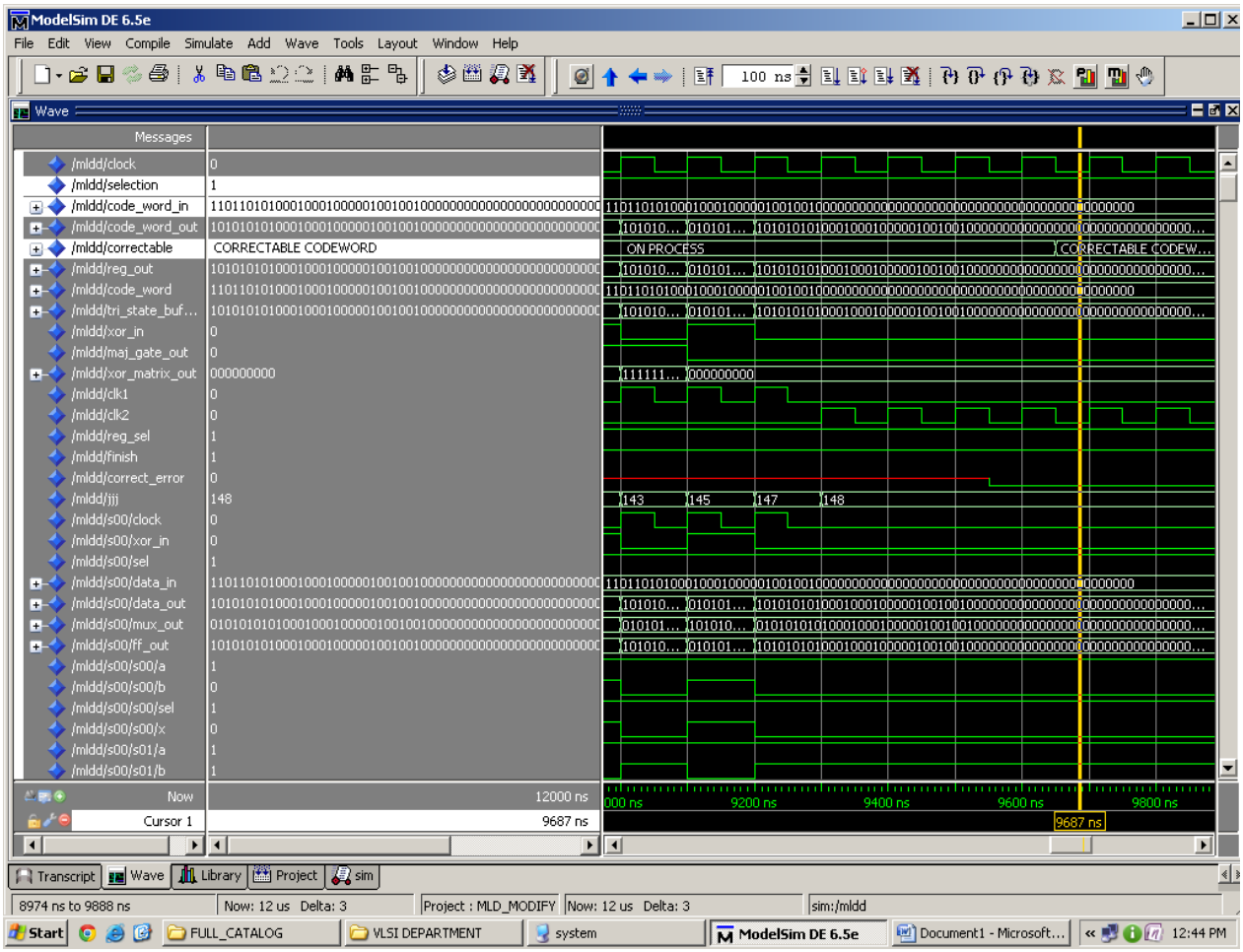

Fig. 8. Output of MLDD

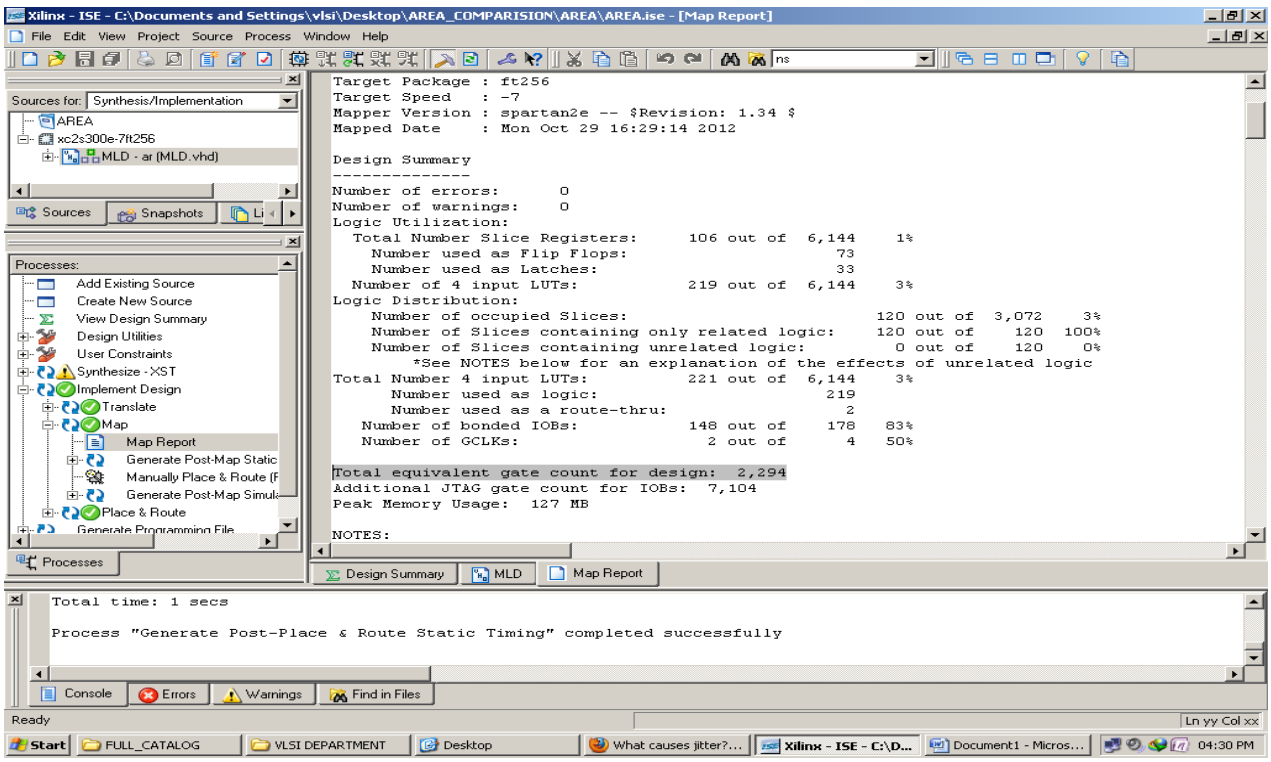

Fig. 9. Number of Gates for MLD

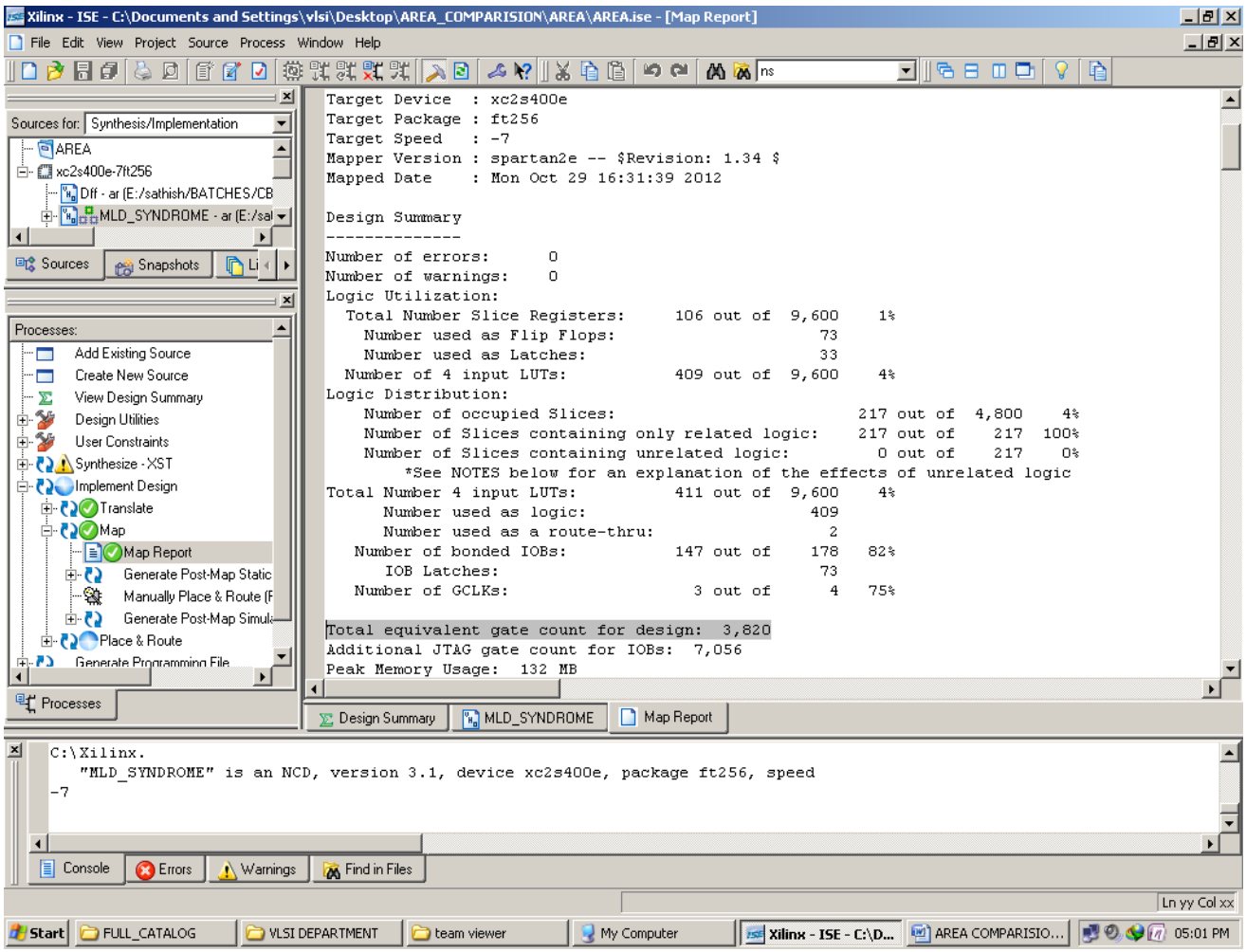

Fig. 10. Number of Gates for MLDD

Above results represents the number of gates occupied by logic fault detection techniques. By analyzing these three techniques, it is found that MLD cannot detect fault in minimum number of cycles and SFD adds complexity to thesystem. Among them, MLDD is found to occupy less area with improved performance.

V. CONCLUSION

In this paper, fault-detection mechanism like plain MLD, SFD and MLDD have been compared based on Difference-set cyclic codes. The mode of working of all the three techniques and their structure are analyzed and compared with the help of HDL code simulation and implemented using FPGA. The simulation test results show that the SFD needs less cycles to detect the fault. Plain MLD technique needs number of cycles which is equal to the number of input bits for error detection and correction. When no error is present memory access delay of SFD is minimum when compared to the plain ML decoder. Plain MLD occupies less area when compared to SFD. But MLDD will occupy minimum area when compared to SFD. Hence complexity is reduced. This comparison provides the solution to improve performance of the design with respect to the system size. Among the three techniques MLDD is found to be best based on number of working cycles, size and optimal gate requirement.

REFERENCES

1. *Sethuramalingam, T. K., and B. Nagaraj. "A comparative approach on PID controller tuning using soft computing techniques." International Journal of Innovations in Scientific and Engineering Research (IJISER) 1.12 (2014): 460-465.*
2. *R.C. Baumann, "Radiation-induced soft errors in advanced semiconductor technologies", IEEE Transactions on Device and Materials Reliability, vol. 5, no.3, pp. 301–316, Sep. 2005.*
3. *S. Ghosh and P.D. Lincoln, "Low-density parity check codes* for error correction in nanoscale memory", *SRI Computer Science Lab., MenloPark, CA, Tech. Rep. CSL-0703*, 2007.
4. H. Naeimi and A. DeHon, "Fault secure encoder and decoder for
5. NanoMemory applications", *IEEE Transactions on Very Large Scale Integration (VLSI) Systems.*, vol. 17, no. 4, pp. 473–486, Apr. 2009.
6. Y. Kato and T. Morita, "Error correction circuit using difference-set cyclic code," in *Proc. ASP-DAC*, 2003, pp. 585–586.
7. T. Shibuya and K. Sakaniwa, "Construction of cyclic codes suitable for iterative decoding via generating idempotents," *IEICE Trans. Fundamentals*, vol. E86-A, no. 4, pp. 928–939, 2003.

8. C. Tjhai, M. Tomlinson, M. Ambroze, and M. Ahmed, "Cyclotomic idempotent-based binary cyclic codes," *Electron. Lett.*, vol. 41, no. 6, Mar. 2005.
9. Agalya, A., and B. Nagaraj. "Certain investigation on concentration control of CSTR-A comparative approach." Int J Adv Soft ComputAppl 5.2 (2013): 1-14.
10. C.L. Chen and M.Y. Hsiao, "Error-correcting codes for semiconductor memory applications: A state-of-the-art review," *IBM Journal of Research and Development*, vol. 28, no. 2, pp. 124-134, Mar. 1984.
11. M. A. Bajura, Y. Boulghassoul, R. Naseer, S. Das Gupta, A. F. Witulski, J. Sondeen, S.D. Stansberry, J. Draper, L.W. Massengill, and J.N. Damoulakis, "Models and algorithimic limits for an ECC- based approach to hardening sub-100-nm SRAMs," *IEEE Transactions on Nuclear Science*, vol. 54, no. 4, pp. 935-945, Aug. 2007.
12. S Thilagam, P Karthigaikumar, "Implementation of adaptive noise canceller using FPGA for real-time applications", *2nd International Conference on Electronics and Communication Systems (ICECS), 2015*, ISBN: 978-1-4799-7225-8, DOI: 10.1109/ECS.2015.7124878.
13. Premalatha P, Amsaveni A, "Data hiding in a digital image with FPGA implementation", *Online International Conference on Green Engineering and Technologies (IC-GET), 2016*, ISBN: 978-1-5090-4556-3, DOI: 10.1109/GET.2016.7916641.