ISSN: 2319-5967
ISO 9001:2008 Certified
International Journal of Engineering Science and Innovative Technology (IJESIT)
Volume 2, Issue 3, May 2013

An Efficient Design of VLSI Architecture Based ML Decoder/Detector in Fault Detection

[1]T.Sri Lakshmi Priya, [2]K.Karuppanasamy
[1] PG/VLSI Design, K.S.R College of Engineering, Thiruchengode
[2] Asst. professor, Dept. of ECE, K.S.R College of Engineering, Thiruchengode

Abstract-To prevent soft errors from causing data corruption, memories are typically protected with error correction codes. An advanced error correction codes are used when an additional protection is needed. The majority logic decoder/detector codes are used for memory application because of correcting large number of soft errors, less decoding time, area consumption. The differences set cyclic codes are suitable for error correction using majority logic decoder/detector. Because the difference set cyclic codes are small, powerful and easily implemented in terms of decoding latency and complexity and this design achieving very high data rate while minimizing complexity. The proposed improved majority logic decoder/detector to perform silent data error detection in simple way using additional error detection technique and also reducing the area of the majority gate using sorting network. Hence the decoding process uses less number of cycles, reduces the area and also reducing the power consumption.

I. INTRODUCTION

Soft error occurs when a radiation event causes enough of a charge disturbance to reverse or flip the data state of a memory cell. The soft error is also often referred to as a Single Event Upset (SEU). If the radiation event is of a very high energy, more than a single bit maybe affected, creating Multi Bit Upset (MBU). For reliable communication, errors must be detected and corrected. Error detection is the way to find out that is a data is correct or incorrect. Reed-Muller is one of the methods of multiple error detection in blocks for digital communications signals [2], [10]. Soft error reliability is to employ Error Detection and Correction (EDAC) techniques or Error Correction Codes (ECC) is employed. Various error detection techniques are used to avoid the soft error. One of the methods is majority logic decoder which used to detect and correct the error in simple way. The drawback of this method is increase the average latency of the decoding process because it depends on the size of the code. Another method is syndrome fault detector which increase the power consumption because it is complex module. Majority Logic Decoder/Detector (MLDD) is used for avoiding those drawbacks of existing methods [11].

MLDD is a method to decode the repetition codes. Repetition code is one of the most basic error-correcting codes. The idea of the repetition code is to just repeat the message several times. The hope is that the channel corrupts only a minority of these repetitions. This way the receiver will notice that a transmission error occurred. Since the received data stream is not the repetition of a single message and moreover the receiver can recover the original message by looking at the received message in the data stream that occurs most often [9].

One specific type of LDPC codes, namely the Difference-Set Cyclic Codes (DSCCs) is based on the construction of a perfect difference set. Cyclic codes are a class of linear block codes that have convenient algebraic structures for efficient error detection and correction. Thus, we can find generator matrix (G) and parity check matrix (H). The reason is that they can be easily implemented with externally cost effective. Electronic circuit and the output show that the properties of DSCC-LDPC enable efficient fault detection [8], [12].

Various error detection techniques are used to avoid the soft error. One of the methods is majority logic decoder which used to detect and correct the error in simple way. The drawback of this method is increase the average latency of the decoding process because it depends on the size of the code. Another method is syndrome fault detector which increase the power consumption because it is complex module [4]. Majority Logic Decoder/Detector (MLDD) is used for avoiding those drawbacks of existing methods. The error correction codes that meet the requirements of higher error correction capability and low decoding complexity, cyclic block codes have been identified as good candidates, due to their property of being Majority Logic Decoder (MLD). Majority logic decoding is a method to decode repetition codes. Repetition code is one of the most basic error-correcting codes [6].

The proposed method is used to detect the silent data error (SDE) in the MLDD by using an additional error detection technique. This addition logic is used to detect the error which is not detected by the first three iteration of the MLDD which is used to produce the accurate result of the MLDD. The area of the majority gate will reduced by using sorting network in the majority gate because it is used to reduce the number of gates which is used in the majority gate.

II. MLDD

Initially the input is stored into the cyclic shift register and it shifted through all the taps. The intermediate values in each tap are given to the XOR matrix which is used to perform the checksum equations. The resulting sums are then forwarded to the majority gate for evaluating its correctness. If the number of 1's received is greater than the number of 0's which would mean that the current bit under decoding is wrong, so it move on the decoding process of t It is used to produce the accurate result of the MLDD. Otherwise, the bit under decoding would be correct and no extra operations would be needed on it. Decoding process involving the operation of the content of the registers is rotated and the above procedure is repeated and it stops intermediately in the third cycle. If in the first three cycles of the decoding process, the evaluation of the XOR matrix for all is "0," the code word is determined to be error-free and forwarded directly to the output. If the error contains in any of the three cycles at least a "1," it would continue the whole decoding process in order to eliminate the errors.

Finally, the parity check sums should be zero if the code word has been correctly decoded. Finally the MLDD method is used to detect the five bit errors and correct four bit errors effectively. More than five bit errors it produces the output but it did not show the errors presented in the input. This type of error is called the silent data error. Drawback of this method is did not detecting the silent data error and it consuming the area of the majority gate. Overall operation of the MLDD is illustrated in figure 2.

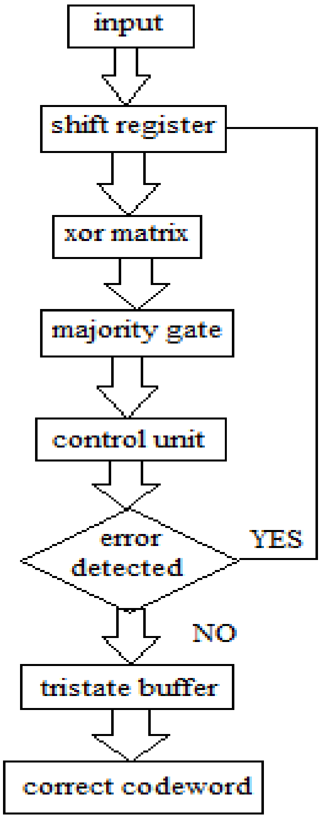

Fig 1. MLDD algorithm

III. IMPROVED MLDD

Figure 3(a) shows the data words are initially encoded and it producing codeword. Then the codeword is stored in the memory. When the memory is read, the codeword is then fed through the improved MLDD before sent to the output for further processing. The code word is the n bit encoded block of bits. It contains message bits and parity or redundant bits. The code efficiency is defined as the ratio of message bits to the number of transmitted bits per block.

The silent data error detection using MLDD algorithm performs the decoding as in the MLDD with some modifications. When the MLDD having more than 5 errors will be detected and corrected by the improved MLDD method. The MLDD is used the control unit for detecting the error. If it has any error in this iteration it will be perform with the modified algorithm is illustrated in Figure 4. It is used to avoid silent data corruption of the MLDD output. This would increase the error detection capabilities at the expense of the error-correction capabilities. In this algorithm up to four errors will be done as in the MLDD algorithm. If it has more than four errors will detected by after third iteration. Then correction will be done by after nth iteration.

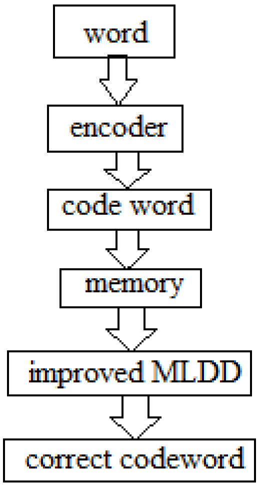

Fig 2. Memory schematic of improved MLDD

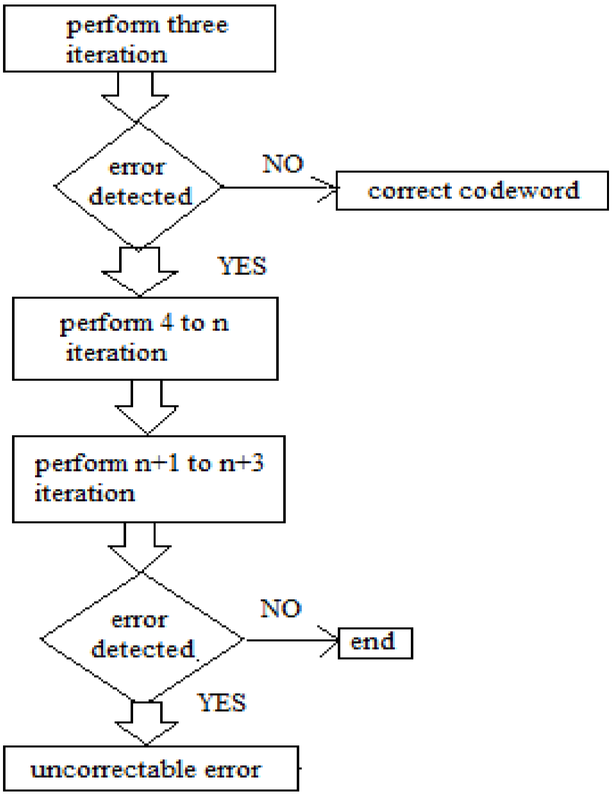

Fig 3(a). Improved MLDD algorithm

A. sorting network

A sorting network is an abstract mathematical model of a network of wires and comparator modules that is used to sort a sequence of numbers. Each comparator connects two wires and sorts the values by outputting the smaller value to one wire, and the larger to the other. The main difference between sorting networks and comparison sorting algorithms is that with a sorting network the sequence of comparisons is set in advance, regardless of the outcome of previous comparisons. This independence of comparison sequences is useful for parallel execution of the algorithms.

Sorting network consists of two items: comparators and wires. Each wire carries with it a value, and each comparator takes two wires as input and output. When two values enter a comparator, the comparator emits the lower value from the top wire, and the higher value from the bottom wire. A network of wires and comparators that will correctly sort all possible inputs into ascending order is called a sorting network. Using sorting network number of gates reduced in the majority gate. Initially it compares the inputs using comparator circuit. Comparator consist of AND gate and then OR gate for selecting maximum and minimum value. shown in figure 3 (b). OR gate producing maximum value will be placed in top of the wire and the AND gate producing minimum value will be placed in bottom of the wire in the comparator circuit. So it used to reducing the gates and their interconnections of the majority gate. Vertical line shows the comparison of two input which using the combination for reducing number of gates. Upper wire stores the maximum value and lower wire store the minimum value. Those value given to the AND gate for getting the minimum value and given to the OR gate for selecting the maximum value. Shown in figure 3 (c).

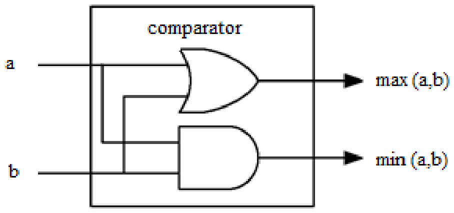

Fig 3(b).Comparator Structure

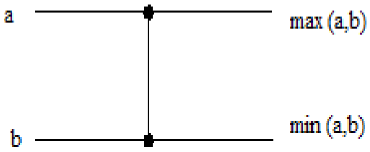

Fig 3(c). Two Bit Sorters

Initially the nine inputs from the XOR matrix are given to the comparator circuit. Then the comparator compares the inputs and it stores the higher value to the upper wire and lower wire to the lower wire of the comparator. Output of the comparator is given to the AND gate for producing minimum value and it will give to the OR gate for selecting the maximum value. Using sorting network in the majority gate is used to reducing the number of gates and wires compared with ordinary comparing method. So it used to reducing area and latency of the majority gate in the MLDD method.

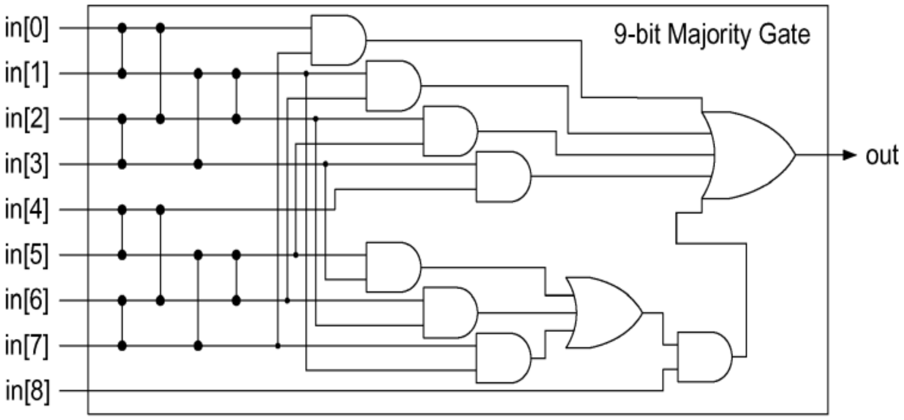

Fig 3(d). Nine input Majority Gate using Sorting Network

IV. SIMULATION RESULTS

The design is simulated using modelsim and xilinx. The power consumed and area utilization is calculated. The improved MLDD with sorting network shows reduced power consumption and area compared existing method. Then the performance of MLDD and improved MLDD has shown below. Using modelsim shows the output waveform of MLDD and improved MLDD with and without silent data error and then the memory access time of both MLDD and improved MLDD were shown. Using Xilinx get the power consumption and area utilization of both MLDD and improved MLDD.

V. CONCLUSION

The improved MLDD technique is used to detect any pattern of more than five bit-flips is designed. The comparison between the MLDD to detect any pattern up to five bit-flips and improved MLDD to detect more than five bit-flips were done. They can be simulated by using modelsim 6.4C and Xilinx 13.2 software and their performances were analyzed. Thus the performance of the proposed method was improved compared to the MLDD approach. Finally, the decoding cycles for the detection and correction of errors is slightly increased comparing to MLDD approach. The area and power consumed by improved MLDD is less compared to the MLDD approach by the use of sorting network.

TABLE 1: PERFORMANCE RESULTS FOR MLDD AND IMPROVED MLDD

MODEL	NO OF CYCLES FOR I/O	NO OF CYCLES FOR ERROR DETECTION	NO OF CYCLES FOR NO ERROR	NO OF CYCLES FOR ERROR PRESENTED	NO OF CYCLES FOR SILENT DATA ERROR DETECTION
MLDD	2	3	5	78	Not detected
IMPROVED MLDD	2	3	5	78	78

TABLE 2.COMPARISONS OF POWER AND AREA UTILIZATION

Design	Power (in watts)	Area (in number of slices)
MLDD	1.776	131
Improved MLDD using detection logic	0.823	162
Improved MLDD using sorting network	0.823	115

REFERENCES

[1] M.A. Bajura et al., Aug. 2007, “Models and algorithmic limits for an ECC-based approach to hardening sub-100-nm SRAMs,” IEEE Transaction Nuclear Science., volume. 54, no. 4, pp. 935–945.

[2] R.C. Baumann, Sep. 2005, “Radiation-induced soft errors in advanced semiconductor technologies.” IEEE Transaction Device Material Reliability., volume.5, no.3, pp. 301–316.

[3] S. Ghosh and P. D. Lincoln, 2007, “Low-density parity check codes for error correction in Nano scale memory,” SRI Computer Science Laboratory Tech. Rep.CSL-0703.

[4] H. Naeimi and A. DeHon, Apr. 2009, “Fault secure encoder and decoder for Nano Memory applications,” IEEE Trans. Very Large Scale Integration (VLSI) System, volume. 17, no. 4, pp. 473–486.

[5] R. Naseer and J. Draper, (2008) “DEC ECC design to improve memory reliability in sub-100 nm technologies,” in Proceedings IEEE ICECS, pp.586–589.

[6] Shih-Fu Liu, Pedro Reviriego, Member, IEEE, and Juan Antonio Maestro, Member, IEEE, Jan 2012, “Efficient Majority Logic Fault Detection With Difference-Set Codes for Memory Applications”, IEEE Transaction on VLSI Systems volume 20,no 1.

[7] J. von Neumann, 1956, “Probabilistic logics and synthesis of reliable organisms from unreliable components,” Automata Studies, pp. 43–98.

[8] E.J. Weldon, Jr., (1966) “Difference-set cyclic codes,” Bell System Tech J., volume 45, pp. 1045–1055.

[9] P. Ankolekar, S. Rosner, R. Isaac, and J. W. Bredow, “Multi-bit error correction methods for latency-constrained flash memory systems,” IEEE Trans. Device Mater. Rel., vol. 10, no. 1, pp. 33–39, Mar. 2010.

[10] C. L. Chen and M. Y. Hsiao, “Error-correcting codes for semiconductor memory applications: A state-of-the-art review,” IBM J. Res. Develop., vol. 28, no. 2, pp. 124–134, Mar. 1984.

[11] C.W.Slayman, 2005, “Cache and memory error detection, correction, and reduction techniques for terrestrial servers and workstations,” IEEE Trans. Device Mater. Reliable, Vol. 5, No. 3, pp. 397–404.

[12] Y.Kato and T.Morita, 2003, “Error correction circuit using difference-set cyclic code,” in Proc. ASP-DAC, pp. 585–586.