International Journal of Computer Applications (0975 – 8887)
Volume 104 – No.8, October 2014

Simulation and Synthesis of Majority Logic Decoder/Detector for EG-LDPC Codes

J.Chinnababu
Assistant professor
Dept.of ECE, AITS
Rajampet, Kadapa

S.Prathyusha
M.Tech scholar
VLSI System Design, AITS
Rajampet, Kadapa

V.Usha Sree, Ph. D.
Head of Dept,
Dept of ECE, JBIET
Hyderabad

ABSTRACT

In this paper, a technique was proposed to protect memory cells, which are more susceptible to soft errors. These memory cells are to be protected with effective error correction codes. MLD codes are suitable for memory applications because of their ability to correct large number of errors. Conversely, they increase the average latency of the decoding process because it depends upon the code size that impacts memory performance. A method was proposed as majority logic decoder/detector of Euclidean geometry low density parity check codes(EG-LDPC).BUT this MLDD reduces the decoding time, memory access time and area utilization. In this brief, we obtain the application of MLDD to a class of EG-LDPC. The simulation results show that MLDD consumes less area and speed of execution is high for error detection and correction. On comparison with MLD, MLDD provides high speed of operation with reduced execution time, decreased area and high performance.

General Terms

Code word, Low density parity check codes (LDPC), Majority logic decoder (MLD), Majority logic decoder/detector (MLDD)

Keywords

Error correcting codes, Euclidean geometry low density parity check codes (EG-LDPC), majority logic decoder/detector (MLDD).

1. INTRODUCTION

The reliability and security of memories are essential considerations in the modern digital system design. Soft error occurs when a radiation event causes enough of a charge disturbance to Reverse or flip the data state of a memory cell, register. Memory size requirement increasing largely, and more powerful error correction and detecting codes are needed to protect memories from soft errors.

For reliable communication, errors must be detected and corrected. Some multi error bit correction codes are BCH codes, Reed Solomon codes, but in which the algorithm is very difficult. These codes can correct more number of errors, but need complex decoders. Among the error correction codes, cyclic block codes have higher error detection capability, low decoding complexity and that are majority logic (ML) decodable. A low-density parity-check (LDPC) code is known as a linear error correcting code, used to avoid a high decoding complexity**.** one specific type of low density parity check codes, namely Euclidean Geometry-LDPC codes are used due to their fault secure detector capability, higher reliability and lower area overhead.

Various error detection techniques are used to avoid the soft error [7]. One of the methods is one step majority logic decoder which used to detect and correct the error in simple way. This method uses the first iteration of MLD to detect the error present in the word. If there are no errors, then the decoding process can be resumed without continuing the remaining iterations [1]. The main reason for using Majority Logic Decoding (MLD) is that it is very easy to implement and has a low complexity [3]. The major drawback of this method is increase the average latency of the decoding process because it depends on the size of the code.

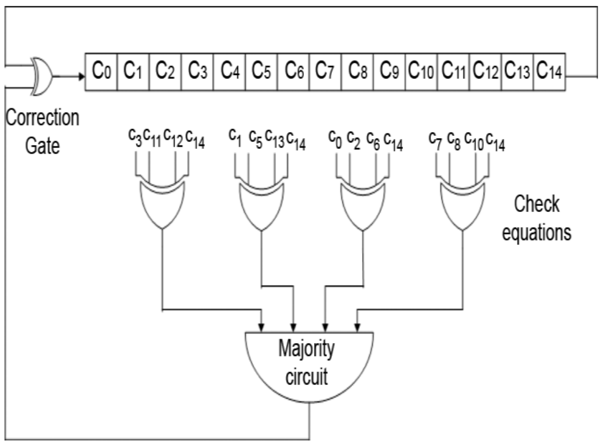
Correction Gate
C_0 C_1 C_2 C_3 C_4 C_5 C_6 C_7 C_8 C_9 C_{10} C_{11} C_{12} C_{13} C_{14}
$c_3 c_{11} c_{12} c_{14}$
$c_1 c_5 c_{13} c_{14}$
$c_0 c_2 c_6 c_{14}$
$c_7 c_8 c_{10} c_{14}$
Check equations
Majority circuit

Fig. 1.One-step MLD for (15,7) EG-LPDC code

As the size of the code is large then the memory access time also more and execution of codeword also takes more time. To overcome this drawback, we are implementing Majority Logic Decoder/Detector (MLDD) method which is used to detect the error in memory device itself .so the data corruption during processing has been eliminated easily to improve the system performance. The MLDD uses control unit for detecting the error. First, the data bits are encoded and then stored in memory. When memory is read, the code word is to be passed through the Majority Logic Decoder (MLD) before sent to the output. In this decoding process [2], the code word is corrected from all bit flips it might have suffered while being stored in the memory.

2. PRELIMINARIES

EG-LDPC codes are a subgroup of Low-Density Parity Check (LDPC) codes, which belongs to the family of the ML decoding codes. Euclidean Geometry codes are also called EG-LDPC codes, as they are Low-Density Parity Check (LDPC) codes [5]. LDPC codes have a limited number of 1‟s in each row and column of the matrix; this limit guarantees limited complexity in their associated detectors and correctors making them fast and light weight. Let EG be a Euclidean Geometry with n points and J lines. EG is a finite geometry that is to have the following fundamental structural properties is:

1) Every line consists of points
2) Any two points are connected by exactly one line
3) Every point is intersected by lines
4) Two lines intersect in exactly one point or they are parallel i.e., they do not intersect.

Let H be a J x n binary matrix, whose rows and columns corresponds to lines and points in an Euclidean geometry, respectively, where h_{ij} = 1 if and only if the line of EG contains the point of EG, and h_{ij}=0 otherwise. A row in H displays the points on a specific line and has weight ρ. A column in H displays the lines that intersect at a specific point in EG and has weight γ. The rows are called the incidence vectors [4] of the lines in EG, and the columns of H are called the intersecting vectors of the points in EG. Therefore, H is the incidence matrix of the lines over the points in EG. Hence, H is a LDPC matrix, and therefore the code is an LDPC code.

Table 1: Euclidean Geometry LDPC codes

N	Data bit	Parity bit
15	7	8
63	37	26
255	175	80
1023	781	242

The EG-LDPC codes are based on the structure of EG over a Galois field [8]. Among EG-LDPC codes there is a class of codes that is one step Majority Logic Decodable (MLD). When the size of code and the number of bits in codeword increases, it is difficult to continuously test all relative combinations.

In error detection and correction, majority logic decoding is a method to decode repetition codes, based on the assumption that the largest number of occurrences of a symbol was the transmitted symbol. Majority logic decoder is dependent on number of parity check equations which are orthogonal to each other [9]. So the correctness of the current bit under decoding is decided by the majority result of these parity check equations .If the codeword has been correctly decoded, then the parity check sum should be zero. In this process, only once each bit can be corrected. As a result, the decoding circuitry is simple, but if the code word is large then it requires a long decoding time. Thus, by using MLD the code is capable of correcting any error pattern with two errors. For example, for a code word of 15-bits, the decoding process would take 15 cycles, which would be excessive for most of the memory applications.

The proposed ML Decoder and Detector (MLDD) can be implemented using the Euclidean Geometry LDPC (EG-LDPC) and this has improved performance with reference to the ML decoder [7]. In Majority logic decoder/detector the majority logic decoder itself act as a fault detector. In general, the decoding algorithm of MLDD is still same as that of majority logic decoder. But, the major difference is that instead of decoding all codeword bits simultaneously, the MLDD method stops intermediately in the third decoding cycle, and can able to detect up to five bit flips in three decoding cycles. So the number of decoding cycles can be reduced to get improved performance.

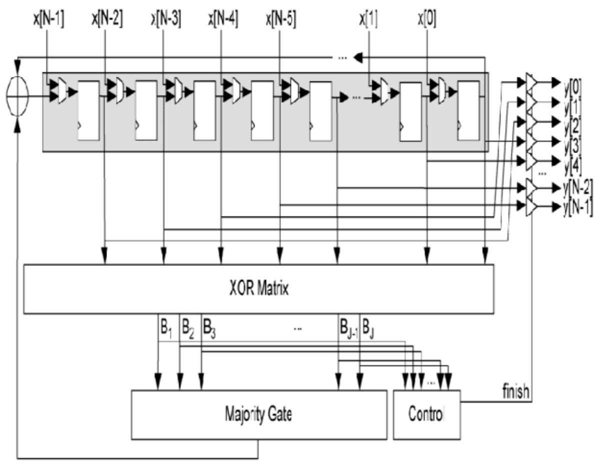

Fig 2: Schematic of Majority Logic Decoder/Detector (MLDD)

Initially the code word is loaded into the cyclic shift register and it shifted through all the taps. The intermediate values of each tap are given to the XOR matrix 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 it means that the current bit under decoding is wrong, so it move on the decoding process. Otherwise, the bit under decoding is to be correct and no extra operations would be needed on it. Decoding process [8] includes the operation as, the content of the registers is rotated and the same procedure is repeated and it stops intermediately in the third cycle. If the evaluation of XOR matrix for the first three cycles is “0”, then the code word is determined to be error-free and can be forwarded directly to the output. If error contains in any of the three cycles at least a “1,” then it would continue in the whole decoding process in order to eliminate the errors. If the code word has been correctly decoded, then the parity check sums should be zero. The additional hardware intended for fault detection are:

1) The control logic unit and
2) The tristate buffers.

The control logic unit is the most important unit in MLDD system which controls and manages the detection process. It uses a counter that counts up to three, which determines the first three iterations of the MLDD decoding. The control unit triggers a finish flag when errors are not detected in the data read. The output tristate buffers are always in high impedance state until the control unit sends the finish signal so that the current values are forwarded from the shift register to the output y.

Properties of the Majority Logic Decoder and Detector (MLDD) are

1) Capability to correct more number of errors.

2) Sparse encoding, decoding and checking and can be synthesizable into a simple hardware.

3) Encoder and decoder blocks that can allow an efficient hardware implementation.

4) Systematic code structure for partition of the information and code bits in the memory.

Suppose if a word is read from memory protected with EG-LDPC codes and then introduced to four bit-flips. Then only three decoding cycles are needed to detect and correct all the four errors. This is a good improvement over the simpler case, where N decoding cycles are needed to guarantee that errors are detected.

3. RESULTS

EG-LDPC codes along with correction of fault, suitable for memory applications, with reduced fault detection time. MLDD error detector is designed, so that it can be able to detect any error pattern of four bit-flips in the first three cycles of the decoded process. This will improve the performances of the design with respect to the one step MLD approach. Otherwise, the MLDD error detector has been designed in a way that it is independent of the codeword size. The proposed MLDD has comparatively less delay and can detect the presence of errors in just 3 cycles even for multiple bit flips. When comparing to the existing technique, a speed up is obtained when no errors are present in data read access for error detection and correction for codeword of 63. It's because the fault detection needs only three cycles and after the detection of an error free condition, the codeword is passed to the output without further corrections.

This is a great saving of time since most of the situations the memory read access does not make errors. Therefore there is a considerable reduction in the memory access time. The proposed MLDD have about 4% low power consumption than the existing MLD technique, since the proposed design detects the faults in just three cycles. Therefore a large no. of clock cycles are saved and hence considerable reduction in power is achieved. The performance of the proposed MLDD method is faster when compared to the traditional one step MLD.

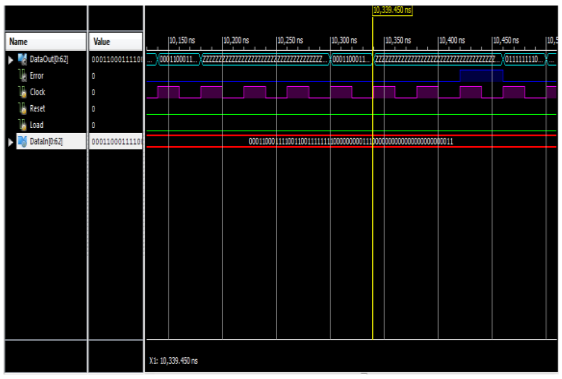

Fig 3: codeword without error

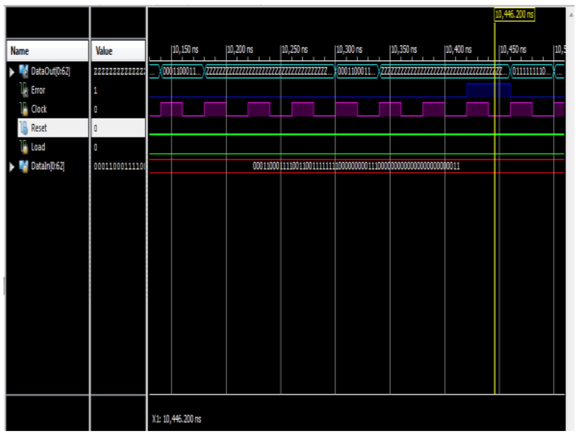

Fig 4: codeword with error

In the proposed MLDD method, the speed is increased when compared to MLD .The power consumption is less in MLDD method when compared to MLD. For the one step MLD, the memory read access delay is directly proportional to the code size, i.e., a code with length 15 needs 15 cycles etc. Then, for I/O two extra cycles needed. On the other hand, for the proposed MLDD the memory read access delay is only dependent on the word error rate (WER).

Table 2: Comparison of no. of cycles needed for One step MLD and MLDD designs

		CYCLES AT WHICH THE OUTPUT IS OBTAINED AFTER DETECTION PROCESS		
TECHINIQUE	I/O	ERROR DECTION	WITH ERROR	WITHOUT ERROR
One-step MLD	2	N	N+2	N+2
Proposed MLDD	2	3	3+2=5	N+5 (e.g. N=15+5)

4. CONCLUSION

In this brief, the detection of errors during the first three iterations of MLDD of EG-LDPC codes has been studied. In this project, the detection and correction of errors is done by using MLDD of EG-LDPC codes. The proposed method MLDD can detect up to five bit-flips and consumes less area of majority gate. The simulation results show that MLDD consumes less area and speed of execution is high for error detection and correction. On comparison with MLD, MLDD provides high speed of operation with reduced execution time, decreased area and high performance. Hence, memory access time decreases.

Future work includes extending the theoretical analysis to the cases of five and six errors. More generally, determining the required number of iterations to detect errors affecting a given number of bits seems to be an interesting problem. In future this project can be extended to higher code word length which would enable fine-grained tradeoffs between decoding time and error detection capability.

5. ACKNOWLEDGMENTS

Mr. J.ChinnaBabu has received his M.Tech degree in VLSI System Design. Currently, he is working as Assistant Professor in the Department of Electronics & Communication Engineering, AITS, Rajampet, Kadapa, A.P, and India. He has published a number of research papers in various National and International Journals and Conferences. His areas of interests are VLSI, Micro processor, Embedded Systems and Signals and Systems.

S.Prathyusha has received her B.Tech in ECE in 2011 from VIST, Madanapalli and currently, pursing M.Tech in VLSI System design from AITS, Rajampet under the guidance of Mr. J.Chinna Babu and her field of interests are Embedded Systems and VLSI design.

Dr.v.usha sree, currently she is working as HOD in the Department of Electronics & Communication Engineering, JBIET, Hyderabad, A.P, India.

6. REFERENCES

[1] Error Detection in Majority Logic Decoding of Euclidean Geometry Low Density Parity Check (EG-LDPC) Codes by Pedro Reviriego, Juan A. Maestro, and Mark F. Flanagan. IEEE transactions on very large scale integration (VLSI) systems, vol. 21, no. 1, January 2013.

[2] R. C. Baumann, “Radiation-induced soft errors in advanced semiconductor Technologies,” *IEEE Trans. Device Mater Reliab.* vol. 5, no. 3, pp. 301–316, Sep. 2005.

[3] S.Liu, P.Reviriego, J.A.Maestro, “Efficient Majority logic fault detection with difference-set codes f or memory applications,” IEEE Trans. Very Large Scale Integr. (VLSI) Syst., vol. 20, no. 1, pp. 148–156, Jan. 2012.

[4] Performance study of Non-binary LDPC Codes over GF (q) V.S. Ganepola1, R.A. Carrasco1, I. J. Wassell2 and S. Le Goff1 School of Electrical, Electronic and Computer Engineering, University of Newcastle Computer Laboratory, University of Cambridge.

[5] Robert G. Gallager (1963). *Low Density Parity Check Codes*. Monograph, M.I.T. Press. Retrieved August 7, 2013.

[6] R. M. Tanner, A recursive approach to low-complexity codes, IEEE Transactions on Information Theory, vol. 27, no. 5, pp. 533-547, 1981.

[7] R.C.Baumann,“Radiation-induced soft errors in advanced semiconductor technologies,” *IEEE Trans. Device Mater. Reliabil,* vol. 5, no.3, pp. 301–316, Sep. 2005.

[8] R. G. Gallager, Low-density parity-check codes, IRE Transactions on Information Theory, vol. 8, no. 1, pp. 21-28, 1962.

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