Detection of Soft Errors in Majority Logic Decoding of Euclidean Geometry Low Density Parity Check (EG-LDPC) Codes

Prashant Puri Goswami[1], **Pankaj M Gulhane**[2]

[1]M. Tech Student, Disha Institute of Management and Technology, Raipur, India

[2]Assistant Professor, Disha Institute of Management and Technology, Raipur, India

Abstract: ***In modern scenario demands for nano-devices has been increased. But the problem with these devices is that they are more prone to soft errors, especially nano-memory devices. A fault secure memory system can be implemented by using majority logic decoder. This is beneficial because majority logic decoding can be implemented serially with simple hardware but decoding time is large, in memory application memory access time increases because of this. The suggested method checks whether a word under scrutiny has errors in the first three iteration of majority logic decoding and if there are no errors decoding ends without completing the rest of the iteration. Since most words in memory will be error free, the average decoding time is greatly reduced. In this paper this technique is applied to a class of Euclidean geometry low density parity check (EG-LDPC) codes that are one-step majority logic decodable. The results obtained shows that this technique can be applied in EG-LDPC effectctively.***

Keywords: LDPC, EG-LDPC, DS-LDPC, ECC, VHDL

1. Introduction

In these days larger error rates at various levels of digital communication system is of major concern. These larger error rates are result of variations in parameters and noise level which is caused by technology scaling and higher integration density. One of the main mottos in designing is to reduce the errors introduced at various stages of digital communication system. One of such error is soft error which affects the memories. Soft error changes the content of memory from „1‟to „0‟ or vice-versa. These error doesn‟t damages the circuits, only complements the particular bit or more. The soft error is caused due to radiation in the memory cell. When the intensity of radiation is very high, more than one bit could be affected by soft error [1].

This error can be detected and corrected by error control codes (ECC). In this paper we will see a technique for detection and correction of soft error. Majority logic decoding is one of technique to detect and correct this error. Among various ECC, LDPC codes are able to decode in majority logic decoding, subclass of which is EG-LDPC, DS-LDPC [4]. In our paper we will see how EG-LDPC code can be used to detect and correct more than one bit flip in memory cell caused by soft error. EG-LDPC majority logic decoding requires N iteration to check N bits of codeword, due to these memory access time increases. We will also see how we can reduce this memory access time by stopping after three iterations if no error is found. Section 2 gives the basic concepts required for understanding the proposed system, section 3 gives the details of proposed approach, and section 4 and 5 gives the result and conclusion respectively.

2. Preliminary

We will outline the concepts here that are necessary to understand our main design and error control code architecture for memory based on EG-LDPC codes.

2.1. LDPC codes

An LDPC code is the null space of a parity check matrix ***H***, which has the following properties [4]:

a) Each row has **ρ** number of 1‟s.
b) The number of 1‟s in each column is **γ**.
c) The number of 1‟s that are common between any two column (**λ**) is no greater than 1, i.e. **λ**=0 or 1.
d) Both **ρ** and **γ** are small compared to the length of the code and the number of rows in ***H***.

The number of 1‟s (**ρ**) in each row and the number of 1‟s (**γ**) in each column are very small than the length of the code and the number of rows in the matrix ***H***, so ***H*** has low density of 1‟s. Due to these reason ***H*** is said to be a low-density parity check matrix and the code defined by ***H*** is said to be a low-density parity check code.

There are many advantages of the LDPC codes. Some major advantages of LDPC codes are as follows [4]:

a) LDPC code has low density of the encoding matrix.
b) LDPC code provides easy iterative decoding.
c) LDPC code generates large codeword that can Shannon‟s limit of coding.

2.2. EG-LDPC code

A Euclidean geometry is defined as a finite geometry having ***n*** points and ***J*** lines such that [6]:

a) Each line contains **ρ** points.
b) Any two points are connected by exactly one line.
c) Each point is intersected by **γ** lines.
d) Any two lines are either intersects on only one point or they are parallel to each other.

Suppose ***H*** is a ***J*** X ***n*** binary matrix, where column corresponds to points and row corresponds to lines in the Euclidean geometry, where h_{ij} = 1 if and only if the i^{th} line

http://dx.doi.org/10.21275/v5i6.NOV164541

of Euclidean geometry contains j^{th} point of Euclidean geometry, $h_{ij} = 0$ otherwise [2]. The row of ***H*** represents the points on a specific line of Euclidean geometry having weight **ρ** while a column of ***H*** represents the lines that intersects at a specific point in Euclidean geometry having weight **γ**. In Euclidean geometry rows of ***H*** are referred to as incidence vectors of the lines in Euclidean geometry while the columns of ***H*** are referred as intersecting vectors of the points in Euclidean geometry. Because of these *H* is the incidence matrix of the lines in Euclidean geometry over the points in Euclidean geometry. This ***H*** is a LDPC matrix and the code generated by these ***H*** is an LDPC code.

3. Proposed System

In this section we will learn the concept of our proposed system. Fig. 1 shows the schematic of EG-LDPC code memory system. The implementation of the majority logic detector/decoder (MLDD) using EG-LDPC code is presented in this paper.

The main logic of proposed system is that if error can be detected in first few iterations of the majority logic decoding [2], then if no errors are detected in these iterations, we can stop decoding without completing the rest of iterations. If one or more check equations are affected by an odd number of bits in error, in the first iteration the error will be detected. When bits are cyclically shifted by one position in the second iteration, the error will affect other equations such that some errors undetected in the first iteration will be detected.

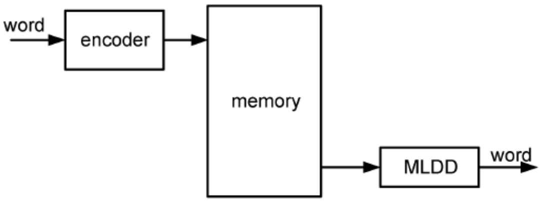

Figure 1: Block diagram of EG-LDPC code memory system

As the iteration advance all detectable errors will eventually be detected. In [5] it is clear that for DS-LDPC codes most error can be detected in the first three iterations of majority logic decoding. On the basis of simulation results and theoretical proof for the case of two errors in DS-LDPC, the following hypothesis was made [5]:

"Given a word read from memory protected with DS-LDPC codes and affected by up to five bit-flips, all error can be detected in only three decoding cycles."

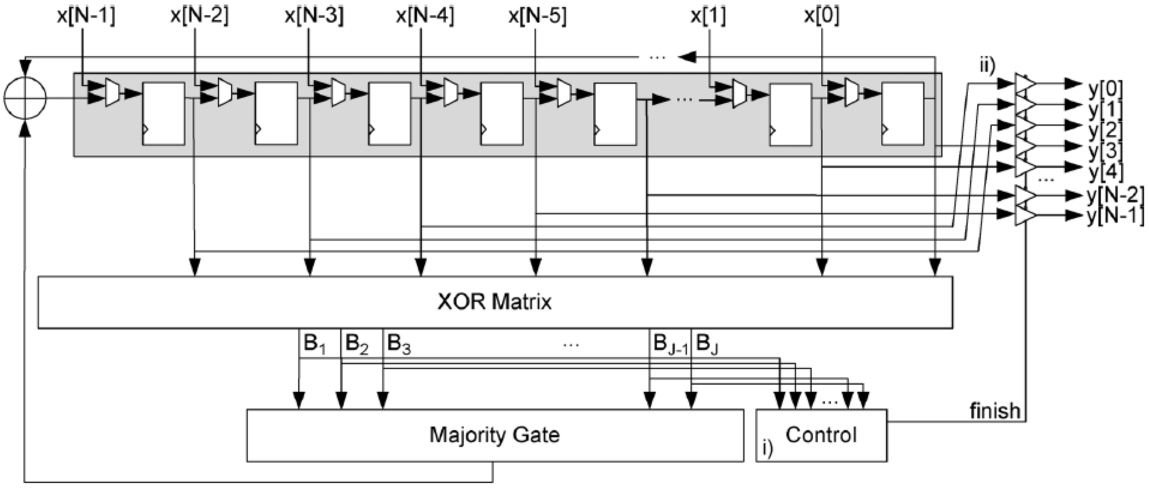

Figure 2: Block diagram of majority logic detector/decoder

The same technique is applied for the EG-LDPC codes. The detailed architecture of our proposed system is shown in fig. 2. In figure there is a majority logic detector/decoder with an N-tap shift register, an XOR array which calculates the orthogonal parity check sums, a majority gate which decides that whether the current bit under decoding is to be inverted or not. The control unit in this architecture triggers a finish flag if no errors are detected after the third cycle. The output tri-state buffer always remain in high impendence state until control unit sends the finish signal so that the current values of the shift register are forwarded to output. The whole detection process is managed by control unit. The detail schematic of control unit is shown in fig. 2. The control unit counts up to three by using counter, which distinguishes the first three iteration of the majority logic decoding. The control unit determines the value {$\mathbf{B_j}$} by combining them with $\mathrm{OR_A}$ function in the first three iterations. Then this value is fed into a three stage shift register where the result of last three iterations is stored. The $\mathrm{OR_B}$ gate evaluates the bits of the detection register in the third cycle. If the result is „0", FSM indicates that the processed word is error free by sending out finish signal. But if result is „1", the majority logic decoding process runs until the end. Fig. 3 shows the flow chart of this whole operation.

The proposed technique was implemented in VHDL and synthesized. The results obtained shows that the overhead is low with large block size codes, because the plane majority logic decoding circuitry is reused to perform error detection and only extra control logic is needed.

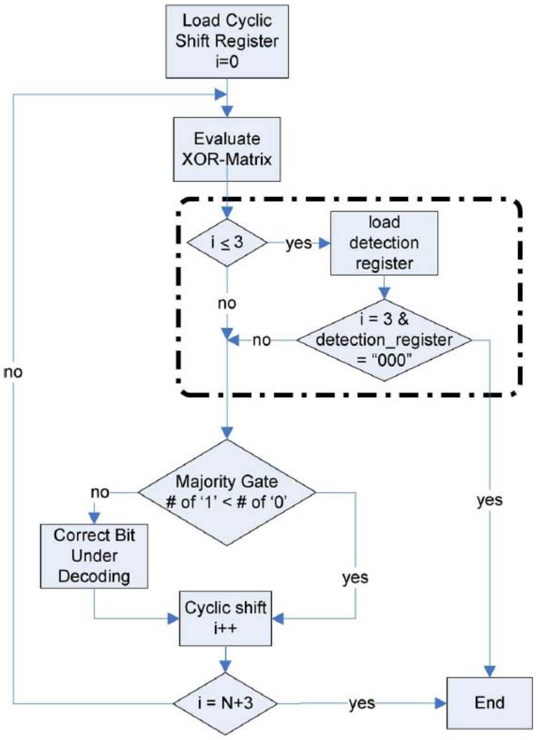
Load Cyclic Shift Register i=0
Evaluate XOR-Matrix
i ≤ 3
yes
load detection register
no
i = 3 & detection_register = "000"
no
no
Majority Gate # of '1' < # of '0'
no
yes
yes
Correct Bit Under Decoding
Cyclic shift i++
i = N+3
yes
End

Figure 3: Flow diagram of MLDD

4. Results and Discussion

The given approach have been synthesized and simulated on Xilinx 13.2 Isim simulator. Figure 4 shows the RTL view of the proposed MLDD.

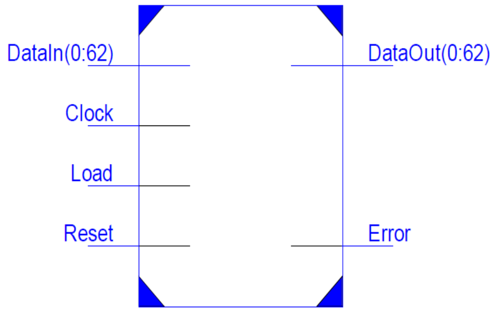

Figure 4: RTL view of EG-LDPC MLDD

In this system DataIn is 63 bits input. The clock input provides the clock to MLDD. When Load input is 1 then data are loaded on the shift register of MLDD. When the Reset input is 1, then all the shift register‟s contents get reset. In this system, when any error occurs in the first three iteration then the output Error bits set to 1. After the completion of three iterations, the corrected output will be available at the DataOut pin. Figure 5 shows the simulation result of MLDD.

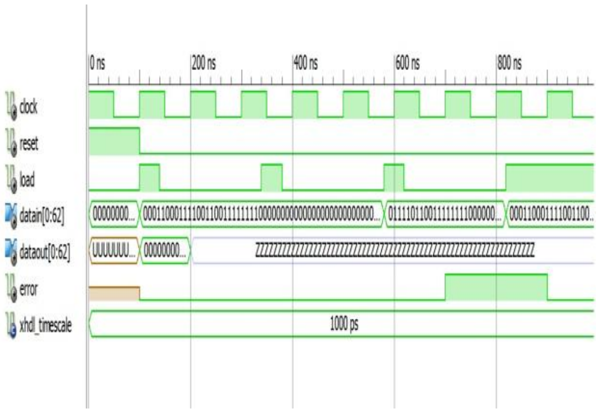

Figure 5: Simulation results of proposed MLDD

References

[1] Robert C. Baumann, Fellow, IEEE, "Radiation-Induced Soft Errors in Advanced Semiconductor Technologies", IEEE TRANSACTIONS ON DEVICE AND MATERIALS RELIABILITY, VOL. 5, NO. 3, SEPTEMBER 2005, pp. 305-316.

[2] S. Lin and D. J. Costello, "Error Control Coding", 2nd ed. Englewood Cliffs, NJ: Prentice-Hall, 2004.

[3] Shalini Ghosh and Patrick D. Lincoln, "Dynamic Low-Density Parity Check Codes for Fault-tolerant nano-scale Memory", presented at the Foundations Nanosci. (FNANO), Snowbird, Utah, 2007.

[4] S. Ghosh and P. D. Lincoln, "Low-density parity check codes for error correction in nanoscale memory," SRI Computer Science Lab., Menlo Park, CA, Tech. Rep. CSL-0703, 2007, pp. 1-22.

[5] S. Liu, P. Reviriego, and J. Maestro, "Efficient majority logic fault detection with difference-set codes for memory applications," IEEE Trans. Very Large Scale Integr. (VLSI) Syst., vol. 20, no. 1, pp. 148–156, Jan. 2012, pp. 148-156.

[6] H. Naeimi and A. DeHon, "Fault secure encoder and decoder for nanomemory applications," IEEE Trans. Very Large Scale Integr. (VLSI) Syst., vol. 17, no. 4, pp. 473–486, Apr. 2009, pp. 473-486.