P MALLESWARI, et al , International Journal of Computers Electrical and Advanced Communication Engineering [IJCEACE]™
Volume 1 , Issue 6, pp: 18 - 22 , SEPTEMBER 2014.

DEVELOPMENT OF NANO MEMORY APPLICATIONS USING VLSI TECHNIQUES

P.Malleswari 1*, K.Naga Lakshmi 2*

1. *II.M.Tech (VLSI), Dept of ECE, AM Reddy Memorial College of Engineering & Technology, Petlurivaripalem.*
2. *Asst.Prof , HOD - Dept. of ECE, AM Reddy Memorial College of Engineering & Technology, Petlurivaripalem.*

ABSTRACT

A novel method is proposed to accelerate the majority of logical decoding the different set of low density parity check codes. And it is useful for majority of logic decoding; it can be implemented serially with a simple hardware but also requires a large decoding time. Proposed method detects whether a word has an errors occurred in first iterations of majority of logic decoding and there is no error the decoding processes terminates without completing the rest of the iterations. Most of the existing words are in a memory will be error-free and then averaged encoding time is effectively reduced. Application of this paper is a similar technique to a class of Euclidean geometry technique for Low Density Parity Check (EG-LDPC) codes that are one step for majority logic decodable. The proposed method is also effectively for EG-LDPC codes.

Keywords: *ECC, EG-LDPC, Memory.*

I. INTRODUCTION

Error correction codes are commonly used to protect memories from so-called soft errors which change the logical value of memory cells without damaging the circuit. Technology scales are memory devices become larger and more powerful error correction codes are needed. The use of more advanced codes has been recently proposed. These codes can correct a larger number of errors but generally require complex decoders. To avoid a high decoding complexity the use of one step majority logic decodable codes was first proposed in for memory applications. One step majority logic decoding can be implemented serially with very simple circuitry, but requires long decoding times. In a memory, this would increase the access time which is an important system parameter. Only a few classes of codes can be decoded using one step majority logic decoding. Among those is some Euclidean geometry low densities Parity check (EG-LDPC) codes which were used in, and difference set low density parity check (DS-LDPC) codes. A method was recently proposed in to accelerate a serial implementation of majority logic decoding of DS-LDPC codes. The idea behind the method is to use the first iterations of majority logic decoding to detect if the word being decoded contains errors. If there are no errors, then decoding can be stopped without completing the remaining iterations, therefore greatly reducing the decoding time.

II. PROPOSED SYSTEM

a) Majority of Logic Decoder

If errors can be detected in the first few iterations of MLD, then whenever no errors are detected in those iterations the decoding can be stopped without completing the rest of the

iterations. In the first iteration, errors will be detected when at least one of the check equations are affected by an odd number of bits in error. In the second iteration, as bits are cyclically shifted by one position, errors will affect other equations such that some errors undetected in the first iteration will be detected. As iterations advance, all detectable errors will eventually be detected.

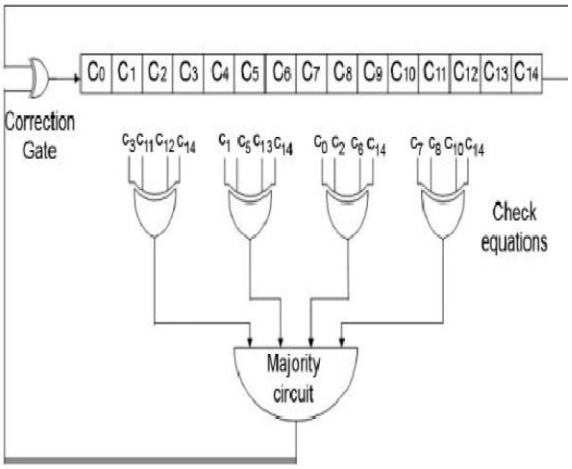


Fig: 1 serial one-step majority logic decoder for the (15,7)EG-LDPC code.

In the proposed approach, only the first three iterations are used to detect errors, thereby achieving a large speed Increase when N is large. Fig.1. shown that for DS-LDPC codes, all error combinations of up to five errors can be detected in the first three iterations. Errors affecting more than five bits were detected with a probability very close to one. The probability of undetected errors was also found to decrease as the code block length increased. For a billion error patterns only a few errors were undetected. This may be sufficient for some applications. Another advantage of the proposed method is that it requires very little additional circuitry as the decoding circuitry is also used for error detection. The additional area required to implement the scheme was only around 1% for large word sizes.

b) Corrector

One-step majority-logic correction is a fast and relatively compact error-correcting technique. There is a limited class of ECCs that are one-step-majority correctable which include type-I two-dimensional EG-LDPC.

ONE-STEP MAJORITY-LOGIC CORRECTOR

One-step majority logic correction is the procedure that identifies the correct value of an each bit in the codeword directly from the received codeword; this is in contrasting to the general message-passing error correction strategy which may demand multiple iterations of error diagnosis and trial correction. Avoiding iteration makes the correction latency both small and deterministic. This technique can be implemented serially to provide a compact implementation or in parallel to minimize correction latency. This method consists of two parts:

1) Generating a specific set of linear sums of the received vector bits

2) Finding the majority value of the computed linear sums.

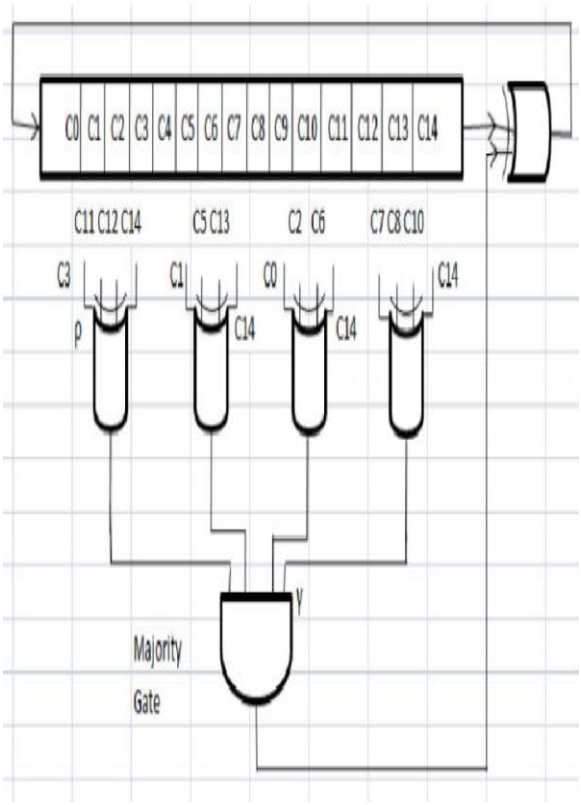


Fig: 2 Serial one-step majority logic corrector structure to correct last bit (bit 14th) of 15-bit (15, 7, 5) EG-LDPC code.

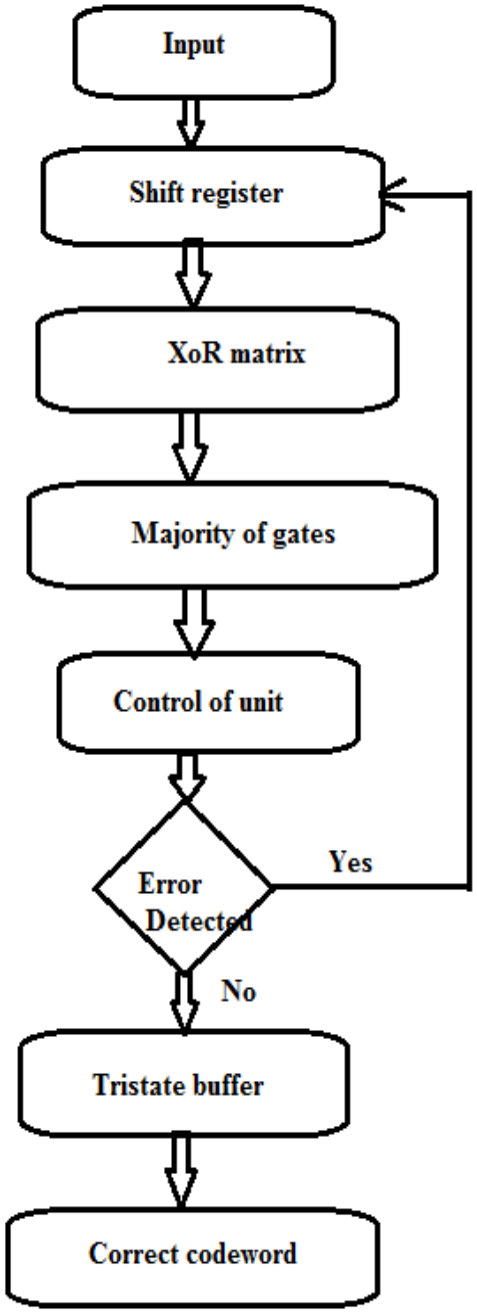
The majority value indicates the correctness of the code bit under consideration; if the majority value is 1, the bit is inverted, otherwise it is kept unchanged. The circuit implementing a serial one-step majority logic corrector for (15,7, 5) EG-LDPC code is shown in Fig 2

c) Majority of Logic Decoding Algorithm

The Modified MLDD algorithm performs the decoding as in the MLDD with some modifications..Modified MLDD algorithm requires additional logic compared to the MLDD algorithm. The corrections are performed during the first n iterations. A counter is used to determine if there have been more than t errors in those iterations and based on the result the corrected or the original register is send to the output. If the majority gate detect any error in codeword the iterations take place depends upon the length of the codeword's.

Below is the Flow diagram of the MLDD algorithm.

Modified MLDD detect more than five bit-flips and high efficiency compare to Majority logic decoder. The control unit manages the detection process. It uses a counter that counts up to three, which distinguishes the first three iterations of the ML decoding. Data bits stay in memory for a number of cycles and, during this period, each memory bit can be upset by a transient fault with certain probability. Therefore, transient errors accumulate in the memory words over time. The process stops after three iteration if the codeword length less than ten and six iteration for codeword less than twenty, if the codeword greater than 20 then nine iterations are performed. In order to avoid accumulation of too many errors in any memory word that surpasses the code correction capability, the system must perform memory scrubbing. Memory scrubbing is the process of periodically reading memory words from the memory, correcting any potential errors, and writing them back into the memory.



III. CONCLUSION

In this brief, the detection of errors during the first iterations of serial one step Majority Logic Decoding of EGLDPC codes has been studied. The objective was to reduce the decoding time by stopping the decoding process when no errors are detected. The simulation results show that all tested combinations of errors affecting up to four bits are detected in the first three iterations of decoding. These results extend the ones recently presented for DS-LDPC codes, making the modified one step majority logic decoding more attractive for memory applications. The designer now has a larger choice of word lengths and error correction capabilities. Future work includes extending the theoretical analysis to the cases of three and four errors. More generally, determining the required number of iterations to detect errors affecting a given number of bits seems to be an interesting problem. A general solution to that problem would enables a fine-grained tradeoff between decoding time and error detection capability.

REFERENCES

[1] 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.
[2] M. A. Bajura, Y. Boulghassoul, R. Naseer, S. DasGupta, A. F.Witulski,J. Sondeen, S. D. Stansberry, J. Draper, L. W. Massengill, and J. N.Damoulakis, "Models and algorithmic limits for an ECC-based approachto hardening sub-100-nm SRAMs," IEEE Trans. Nucl. Sci., vol.54, no. 4, pp. 935–945, Aug.
2007.
[3] R. Naseer and J. Draper, "DEC ECC design to improve memory reliabilityin sub-100 nm technologies," Proc. IEEE ICECS, pp. 586–589,2008.
[4] S. Ghosh and P. D. Lincoln, "Dynamic low-density parity check codesfor fault-tolerant nano-scale memory," presented

at the FoundationsNanosci. (FNANO), Snowbird, Utah, 2007.
[5] S. Ghosh and P. D. Lincoln, "Low-density parity check codes for errorcorrection in nanoscale memory," SRI Computer Science Lab., MenloPark, CA, Tech. Rep. CSL-0703, 2007.
[6] H. Naeimi and A. DeHon, "Fault secure encoder and decoder formemory applications," in Proc. IEEE Int. Symp. Defect Fault Toler.VLSI Syst., 2007, pp. 409–417.
[7] B. Vasic and S. K. Chilappagari, "An information theoretical frameworkfor analysis and design of nanoscale fault-tolerant memoriesbased on lowdensity parity-check codes," IEEE Trans. Circuits Syst.I, Reg. Papers, vol. 54, no. 11, pp. 2438–2446, Nov. 2007.