Elimination of Silent Data Corruption by Improved Error Detection using Difference Set Codes for Memories

S.Vaishnavi, R.Karthika, P.Suganya

Abstract: Now-a-days, the memory devices are susceptible to Single Event Upsets (SEU) which is one of the soft errors due to radiation effects. Though several error correcting codes (ECC) are available, the simplest and effective ECC for memory application is Difference Set Cyclic Codes (DSCC). These codes are Majority logic decodable codes. This method of error correction will lead to Silent Data Corruption (SDC) in which the additional errors beyond the code's correction capability are not detected. This SDC will provide faulty word after decoding which actually looks like error free corrected word. This is because the existing method assumes that the input word contains error which is less than the error correction capability and it tends to correct the additional errors, resulting in wrong output. The proposed algorithm will eliminate the SDC and detect the any number of additional errors. The proposed algorithm is coded in VHDL and simulated using Modelsim and Xilinx ISE simulator

Keywords: Single Event Upsets (SEU), Error Correction Codes (ECC), Difference Set(DS) Codes, Majority logic decoding.

I. INTRODUCTION

SEU are considered as soft errors which became a primary problem for commercial and terrestrial applications. Generally these errors are caused by radiation effects in semiconductor devices. The radiation effects can reverse or flip the data state of a memory cell, register, latch or flip-flops[1]. The three mechanism responsible for SEU is (a).Reaction of high energy neutrons with silicon and other device materials, (b).Reaction of low energy cosmic neutrons with the high concentration of ^{10}B in the device materials. In order to eliminate the SEU occurring in memory devices, there are several methods have been proposed. One of them is Triple Modular Redundancy (TMR). TMR is a special case of von Neumann method consisting of three versions of the design in parallel, with a majority vector selecting the correct output. The complexity overhead of this method is three times plus the complexity of the majority vector compared to the other method. Thus the power consumption of this method is larger leading to failure of this method. And also the convenience and the accuracy are very low in this method [2].

Manuscript published on 30 February 2013.
***** Correspondence Author (s)

Vaishnavi, Electronics and Communication Engineering, Avinashilingam Institute for Home Science and Higher Education for Women University, Coimbatore, India.

Karthika, Computer Science Engineering, PPG Institute of Technology, Coimbatore, India.

Suganya Electronics and Communication Engineering, Mohamed Sathak Engineering College, kilakarai, Ramanathapuram, India.

Another method which is used to correct the SEU is ECC. There are several codes has been introduced in ECC. They are Single Error Correcting-Double Error Detecting(SEC-DED), Double Error Correcting-Triple Error Detecting(DEC-TED), Repetitions, Parity checkers, Hamming Codes, Reed Muller and BCH codes, Reed Solomen Codes, Low Density Parity Check codes (LDPC), Convolutional Codes and Turbo Codes.[3][4]. SRAM failures have been generally corrected using ECC codes. One of them is SEC-DED method which detects two errors and able to correct only one error. This method is very simple, but when the codeword is of multi Bit Upsets (MUB), which is the major contributor of soft errors, this SEC-DED method may not be sufficient to meet the reliability goals [5].

DEC-TED is used in memory applications and it uses simple single cycle implementation instead of multi-cycle decoding in communication system [6]. Since there are multiple soft errors in memories, we need a special code which is capable of correcting multiple errors. One of such codes is Reed Muller codes [8]. The usual multi error correction codes such as Reed-Solomon(RS) Bose-Chaudhuri-Hocquenghem (BCH) are proposed later. These codes uses more sophisticated decoding algorithms like floating point operations, logarithms. Also it uses iterative algorithms. Hence both the codes seem to be very complex and increased computational costs [4]. Among many ECC codes they revised that the LDPC codes are the best codes which are used in many application for error correction [10]. Difference Set Cyclic Codes (DSCC) is one of the LDPC codes which is the very simplest and most suitable method for error correction in memory applications [16]. These codes are majority logic decidable codes. Majority logic decoding detection of errors generally requires 'n' clock cycles for 'n' bit word in memory. This is overcome by having only three clock cycles for error detection since the DSCC have that special property [17]. The DSCC can correct only specified no of SEU for all size word. When the error rate exceeds that limit then the existing Majority logic Decoding has failed to detect the additional error. This is called Silent Data Corruption (SDC). This paper concentrates on detecting the additional errors. The rest of this paper is organized as follows Section II provides an overview of the DSCC using the Majority Logic Decoding (MLD) method; Section III discusses about DSCC with prior Error detection; Section IV illustrates the improved error detection by avoiding SDC. Section V provides the simulation results and finally the Section VI provides the conclusion.

II.PRELIMINARIES

Difference set codes were introduced by Rudolph and Weldon which relies on the concept of perfect difference set. DSCCs are one-step ML decodable codes with high error-correction capability and are linear cyclic block codes.

1)Perfect Difference Set: DSCCs work on the difference-set concept for which a brief description follows. Given a set *P* and a difference of the elements *D*, we have

$$P = \{l_{0}, l_{1,....}q\}\ (0 \leq\ l_1 < l_2 \ldots < l_q \leq (q+1)) \qquad (1)$$

$$D = \{l_i - lj : i \neq j\} \qquad (2)$$

The perfect difference set must satisfy the three following Conditions,

1) All positive differences in *D* are distinct.
2) All negative differences in *D* are distinct.
3) If $l_i - lj$ is a negative difference in *D*, then q(q+1) + 1 + ($l_i - lj$) is not equal to any positive difference in *D*.

DSCC Construction: For a binary code, the perfect difference-set is constructed using the relationship

$$q = 2^s : s \in N \qquad (3)$$

Using the set elements as powers in the terms of the polynomial z(X)

$$z(X) = 1 + X^{l1} + X^{l2} + \ldots + X^{lq} \qquad (4)$$

and the syndrome polynomial h(X) for the difference-set, the cyclic code is given by the greatest common divisor of z(X) and $X^N + 1$

$$\begin{aligned} h(X) &= GCD\left\{z(X), X^N - 1\right\} \\ &= 1 + h_1X + h_2X^2 + \cdots + h_{k-1}X^{k-1} + X^k \end{aligned} \qquad (5)$$

Finally, the DSCC code is generated b

$$\begin{aligned} g(X) &= \frac{X^N - 1}{h(X)} \\ &= 1 + g_1X + g_2X^2 + \cdots + X^{N-k} \end{aligned} \qquad (6)$$

DSCC Parameters: Besides from the definitions and equations previously explained, the following parameters completely define the DSCC codes(see table I):

- Code length: $N = 2^{2s} + 2^s + 1$.
- Message bits: $k = 2^{2s} + 2^s - 3^s$.
- Parity-check bits: $(N-k) = 3^s + 1$.
- Minimum distance: $d = 2^s + 2$.

As $\{0 \leq l_1 \leq l_2 \leq \ldots\ldots. \leq l_q \leq q(q+1)$ is a perfect difference-set, not two polynomials $w_i(X)$ and $w_j(X)$,given by (7), can have any common term except X_{n-1}, for $i \neq j$:

$$\begin{aligned} w_i(X) = {} & X^{l_i - l_{i-1} - 1} + X^{l_i - l_{i-2} - 1} + \cdots + X^{l_i - l_1 - 1} + X^{l_i - 1} \\ & + X^{N-1-l_{2^s}+l_i} + X^{N-1-l_{2^s-1}+l_i} + \cdots + X^{N-1}. \end{aligned} \qquad (7)$$

Thus, $w_0(X)$, $w_1(X)$,………., $w_2{}^s(X)$ form a set of J = 2^s + 1polynomials orthogonal on the bit at position X^{N-1}.

This implies that there will be J = 2^s + 1 parity check-sums able to correct up to $t_{ML} = 2^{(s-1)}$ errors. These difference set cyclic codes are decoded using majority logic decoder. The ML decoder is a simple and powerful decoder, capable of correcting multiple bit-flips depending on the number of parity check equations. It consists of four parts:

1. cyclic shift register;
2. XOR matrix;
3. majority gate; and
4. XOR for correcting the codeword bit under decoding,

For example, the memory word of size (73, 45), there are 73 cyclic shift registers, 9 XOR gates, 1 majority gate circuit and 1 XOR gate for correction of errors.

Table 1. Difference Set Code Parameters.

Code length (n)	Message bit (k)	Correctable errors (t)
21	11	2
73	45	4
273	191	8
1057	813	16

The difference set cyclic codes for (73, 45) has **28,25,22,16,12,8,6,4,0** as generator polynomial and **45,42,36,29,25,10,2,0** as difference set polynomial. With the help of this polynomial, the ex-or inputs are derived. The input to Ex-or is put up in table 2 for (73, 45) input word.

II. MLD WITH PRIOR ERROR DETECTION

The drawback of the MLD is that it requires n clock cycles for n-bit word to detect the error. If suppose a (73, 45) word would require 73 clock cycles to complete its action even when there is no error in the word. Hence it is a waste of time even when there is no error. With the help of properties of DSCC, the prior error detection is carried out with just three clock cycles

Table 2 EX-OR Inputs using difference set codes for (73,45) input word

Difference-set polynomial	0	2	10	24	25	26	36	42	45
Difference		2	8	14	1	4	7	6	3
XOR_IN 1	72	70	62	48	47	43	36	30	27
XOR_IN 2	72	64	50	49	45	38	32	29	1
XOR_IN 3	72	58	57	43	46	40	37	9	7
XOR_IN 4	72	71	67	60	44	51	23	21	13
XOR_IN 5	72	68	61	55	52	24	22	14	0
XOR_IN 6	72	65	59	56	28	26	18	4	3
XOR_IN 7	72	66	63	35	33	25	11	10	6
XOR_IN 8	72	69	41	39	31	17	16	12	5
XOR_IN 9	72	44	42	34	20	19	15	8	2
XOR_IN 1	72	70	62	48	47	43	36	30	27

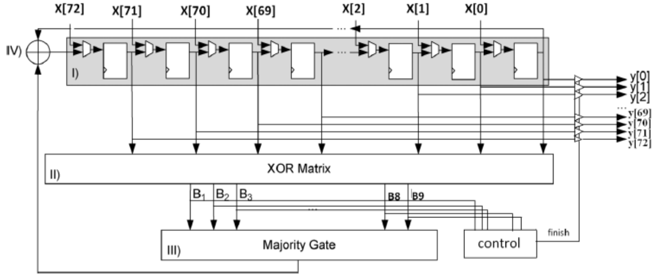

Fig 1. MLD structure with prior error detection circuit

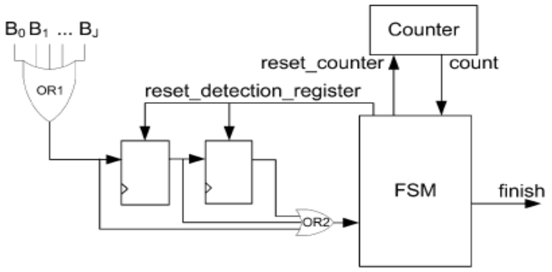

Fig 2. Control Circuit

. If there is no error, then the extra computation is avoided. The prior error detection is done with the help of prior error detection-control circuit shown in Fig.1. The control circuit is able to check error for three consecutive cycles which is shown in Fig.2.

IV. IMPROVED ERROR DETECTION MLD (IED-MLD) TO AVOID SDC

The existing method failed to detect the additional errors when the errors exceed the correctable errors (t). Hence the Silent Data Corruption occurs. To make clear idea about SDC, let us consider (73, 45) error prone input word which can be corrected only when the no of errors are 4. When there are more than 4 error occurs, the existing method assumes it as 4 errors and gives a wrong output which looks like a correctable code word. Hence a new detection sequence is introduced in order to detect the additional errors. Since the properties of difference set cyclic code is such a way that the error detection is possible in just three clock cycles. This paper explains about the avoidance of SDC using additional clock cycles. Once after the error correction is done then the input is fed to the shift register for additional error detection.

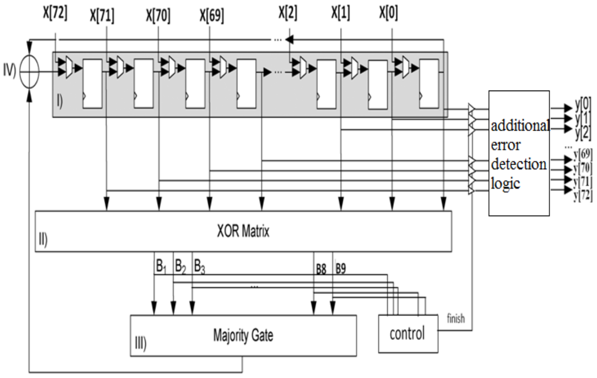

Fig 3 IED-MLD

There is also another method which is able to avoid SDC. In that method, the majority gate output is counted for error correction. If the majority gate output is more than correctable errors, then it is possible that the SDC is eliminated and can be reported as uncorrectable error.

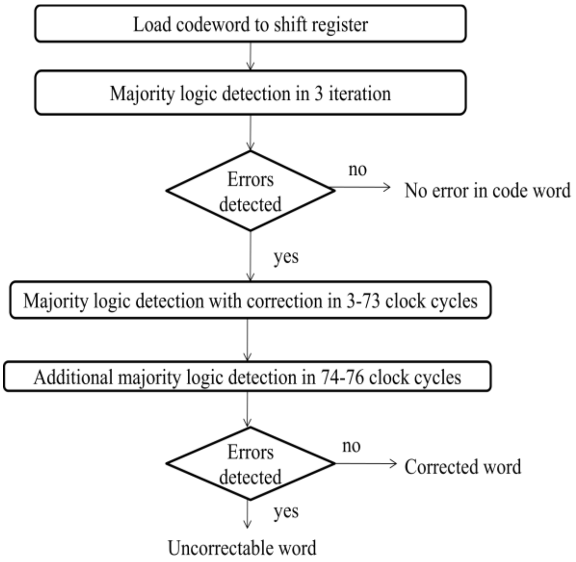
Load codeword to shift register
Majority logic detection in 3 iteration
Errors detected
no
No error in code word
yes
Majority logic detection with correction in 3-73 clock cycles
Additional majority logic detection in 74-76 clock cycles
Errors detected
no
Corrected word
yes
Uncorrectable word

Fig 4.Flow chart representing the algorithm for IED-MLD avoiding SDC

V. SIMULATION RESULTS

The modified algorithm and the existing MLD with prior error detection for the input word (73,45) using Difference Set Cyclic Codes is coded in VHDL and simulated using MENTOR GRAPHICS front end (Model sim 10.a) and Xilinx ISE simulator. Fig 4 shows the simulation result for the existing MLD representing that it is able to detect and correct only the fixed no of correctable errors. Fig 5 shows the simulation result for the proposed algorithm.

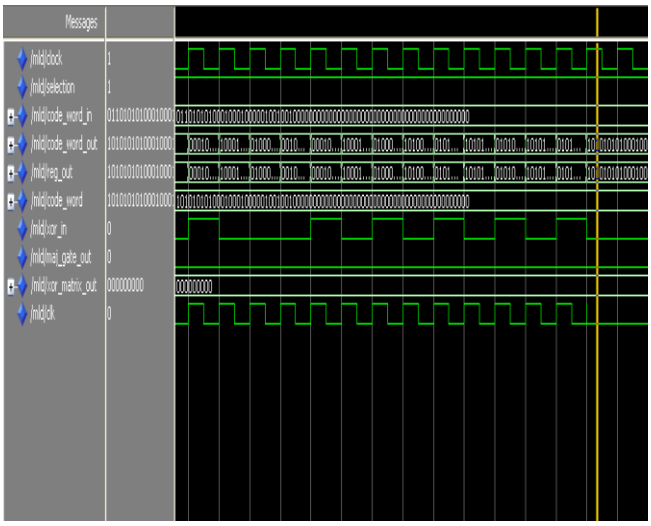
Messages
/mld/clock 1
/mld/selection 1
/mld/code_word_in 0110101010001000
/mld/code_word_out 1010101010001000
/mld/reg_out 1010101010001000
/mld/code_word 1010101010001000
/mld/xor_in 0
/mld/maj_gate_out 0
/mld/xor_matrix_out 000000000
/mld/clk 0

Fig 5. Simulation result for MLD with prior error detection.

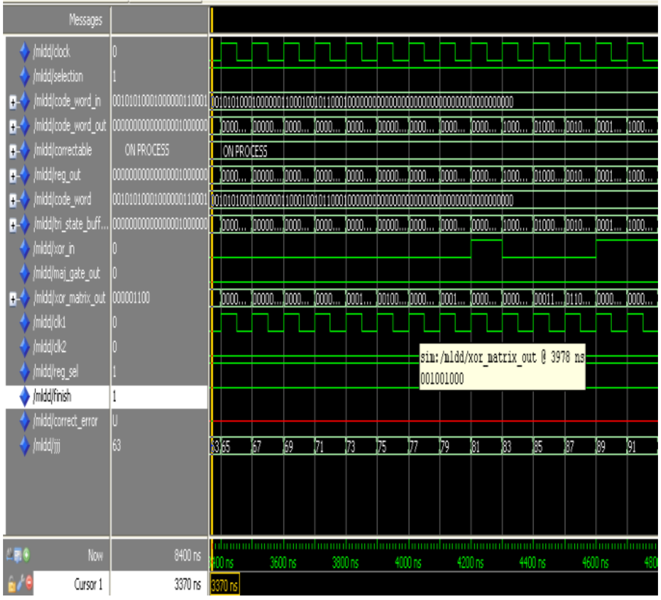
Messages
/mldd/clock 0
/mldd/selection 1
/mldd/code_word_in 0010101000100000110001
/mldd/code_word_out 0000000000000000100000
/mldd/correctable ON PROCESS
/mldd/reg_out 0000000000000000100000
/mldd/code_word 0010101000100000110001
/mldd/tri_state_buff... 0000000000000000100000
/mldd/xor_in 0
/mldd/maj_gate_out 0
/mldd/xor_matrix_out 000001100
/mldd/clk1 0
/mldd/clk2 0
/mldd/reg_sel 1
/mldd/finish 1
/mldd/correct_error U
/mldd/jjj 63
sim:/mldd/xor_matrix_out @ 3978 ns
001001000
Now 8400 ns
Cursor 1 3370 ns

Fig 6 Simulation result for IED-MLD detecting uncorrectable errors

VI. CONCLUSION

In this paper the elimination of Silent Data Corruption is avoided by using the newly presented algorithm. The Difference Set Cyclic Codes has the property that error detection is done in three clock cycles which makes this code to be widely used in memory application. The proposed schemes ensure that uncorrectable errors that exceed the error correction capability of the code are always detected.

The proposed method requires just three additional clock cycles to eliminate the SDC. This proposed method is finally coded in VHDL and simulated using Modelsim and ISE simulator.

REFERENCES

1. 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.
2. J. von Neumann, “Probabilistic logics and synthesis of reliable organisms from unreliable components,” *Automata Studies*, pp. 43–98, 1956.
3. Robin Schriebman, “Error Correcting Codes” *http://wwwmath.mit.edu/phase2/UJM/vol1/COOKE7FF.PDF*.Viewed, April 13, 2006.
4. S. Lin and D. J. Costello, “Error Control Coding”, *2nd ed. Englewood Cliffs, NJ: Prentice-Hall*, 2004.
5. M.Y. Hsiao, “A Class of Optimal Minimum Odd-weight-column SECDED Codes”, *IBM Journal of R & D* Vol. 14, July 1970, pp. 395-401.
6. 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.
7. R.W.Hamming, *Bell Systems Tech*.J.26, No.2, 147 (April 1950).
8. I. S. Reed, “A class of multiple-error-correcting codes and the decoding scheme,” *IRE Trans. Inf. Theory*, vol. IT-4, pp. 38–49, 1954.
9. V. Beiu, S. Aunet, J. Nyathi, R. R. Rydberg-III, and A. Djupdal. “The vanishing majority gate trading power and speed for reliability*”. In Proceedings of NanoArch*, 2005.
10. S.-C. Chae and Y.-O. Park. “Low complexity encoding of regular low density parity check codes”. *In Proceedings of VTC*, 2003.
11. S. Ghosh and P. D. Lincoln, “Low-density parity check codes for error correction in nanoscale memory,” *SRI Comput. Sci. Lab. Tech. Rep.* CSL-0703, 2007.
12. Bane Vasic, Shashi Kiran Chilappagari, “An Information Theoretical Framework for Analysis and Design of Nanoscale Fault-Tolerant Memories Based on Low-Density Parity-Check Codes”, *ieee transactions on circuits and systems*, vol. 54, no. 11, november 2007.
13. M. A. Bajura et al., “Models and algorithmic limits for an ECC-based approach to hardening sub-100-nm SRAMs*,” IEEE Trans. Nucl. Sci.*, vol. 54, no. 4, pp. 935–945, Aug. 2007.

15 C. W. Slayman, “Cache and memory error detection correction, and reduction techniques for terrestrial servers and workstations*,” IEEE Trans. Device Mater. Reliabil.*, vol. 5, no. 3, pp. 397–404, Sep. 2005.

16. 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.
17. E. J.Weldon, Jr., “Difference-set cyclic codes,” *Bell Syst. Tech. J.*, vol.45, pp. 1045–1055, 1966.
18. C. Tjhai, M. Tomlinson, M. Ambroze, and M. Ahmed, “Cyclotomic idempotent-based binary cyclic codes,” *Electron. Lett.*, vol. 41, no. 6, Mar. 2005.
19. 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

S.Vaishnavi received the B.E Degree in Electronics and communication Engineering from P.S.N.A college of Engineering and Technology, Dindigul., affiliated to Anna University, Trichirapalli. She presented a paper in national level conference and attended many workshops. Her area of interest are Digital electronics, VLSI and Digital Image Processing. She is currently pursuing her M.E Degree in VLSI Design in Avinashilingam Institute for Home Science and Higher Education for Women University, India.

R.Karthika received the B.E Degree in Computer Science Engineering from Senguthar Engineering College, affiliated to Anna University, Chennai. She presented a paper in national level conference and attended many workshops. Her area of interest is Cloud Computing. She received her M.E Degree in Computer Science Engineering, in Karpagam University, Coimbatore. She is currently working as Assistant Professor in PPG Institute of Technology, Coimbatore, India.

P.Suganya received her B.E Degree in P.S.N.A College of Engineering and Technology, Affiliated to Anna University, Trichirapalli. She presented a project on "Speed Checker For Highways" in Anna University Tiruchirapalli and ,Coimbatore Institute of Technology. She is currently pursuing her M.E Degree in applied electronics in Mohamed Sathak Engineering College, kilakarai, Ramanathapuram, India.