A Study On The Hardware Implementation Of EDAC

Li Hao, Lixin Yu
Beijing microelectronic technology institute
E-mail:haoliyou.cool@163.com

Abstract

This article proposes a method which reduces delay and area in EDAC circuits. A SEC-DED Hsiao code(39,32) and a DEC systematic (1 6, 8) code used for the hardware implementation of EDAC, are discussed and compared. Two codes are all proposed by the authors in earlier paper. In terms of parity-check matrix of these codes, this article presents a low-cost generation method of check bits. Simulation results show the delay and area of check bits generator based on 2-input XOR gates structure in the smic 180nm process case. EDAC for 32 bits data, SEC-DED Hsiao code(32,7) need less extra memory bits than DEC systematic (1 6, 8) code ,but SEC-DED Hsiao code(32,7) take more delay and area , Which show that different code and different implementation can affect the cost of EDAC circuitry and be used as a guide for selecting the proper Code for different application requirements.

Keywords: *Error detection and correction(EDAC) , Single error correction and double error detection(SEC-DED)code, double error correction （DEC）code, single error upset（SEU），hardware-implemention, memory fault tolerance.*

1. Introduction

Single event upsets (SEU) is a well-known problem in memory chips design，In many computer systems, the contents of memory are protected by an error detection and correction (EDAC) code. EDAC codes have been an effective solution to this problem,which is usually implemented in hardware using extra memory bits and EDAC circuitry. The codelength, code type will affect delay and area of EDAC circuitry based on hardware implementation. Each technique has some advantages and drawbacks, there is comparison of XOR-tree based and LUT-based on implementations for Four SEC-DED Hsiao codes: (39,32). (72,64), (137,128) and (266,256) in [8]. SEU is the major concern in space applications, double bit upsets start to be also a matter to be addressed in very deep submicron technologies. As the correction capability increasing of EDAC, the cost will be changing. In this paper, an SEC-DED Hsiao code(32,7) and a DEC systematic (1 6, 8) code proposed for hardware implementation of EDAC are discussed and compared.

This article is organized as follows: Section 2 shows the H matrix of Hsiao code(39,32) and DEC systematic (1 6, 8) code and the capability of the codes to cope with bits upsets. Section 3 presents the low-cost generation method of check bits for two codes. Simulation results show the delay and area of check bit generator based on 2-input XOR gates structure in the smic 180nm process case. Conclusions are discussed in Section 4.

2. Previous work

A coding scheme provides a mapping of input-data words to codewords. A codeword contains extra check bits that are used for error detection and correction. A distance-4 Hsiao code is an odd-weight-column SEC-DED code, invented by M.Y. Hsiao[8]. Compared with Hamming codes, it has a minimum number of 1's in the rows of the parity-check matrix permits fast generation of check bits and syndrome bits. This feature also permits minimizing the hardware. So it provides improvements in speed, cost and reliability in the encording and decording. The following is an H matrix of the hsiao code(39,32):

$$H=\begin{bmatrix}
1&1&1&1&1&1&1&1&0&0&0&0&0&1&1&0&0&0&0&1&0&0&1&0&1&0&0&0&0&0&1&1&1&0&0&0&0&0&0\\
0&0&0&0&1&0&0&1&1&1&1&1&1&0&1&1&0&0&1&0&0&1&0&0&1&0&0&0&0&1&0&0&0&1&0&0&0&0&0\\
0&0&0&1&0&0&0&0&0&0&0&1&0&0&0&0&1&1&1&1&1&1&1&1&0&0&1&1&0&1&1&0&0&0&1&0&0&0&0\\
0&0&1&0&0&0&1&0&0&0&1&0&0&1&0&1&1&0&0&0&0&0&0&0&1&1&1&1&1&1&1&1&0&0&0&1&0&0&0\\
0&1&1&0&0&1&0&1&0&1&0&0&1&0&0&1&1&0&0&0&1&1&1&1&0&1&1&0&1&0&0&0&0&0&0&0&1&0&0\\
1&0&0&0&0&1&1&0&1&0&0&0&1&1&1&0&0&1&1&1&1&0&0&0&0&0&0&0&1&0&0&0&0&0&0&0&0&1&0\\
1&1&0&1&1&0&0&0&1&1&1&1&0&0&0&0&0&1&0&0&0&0&0&1&0&1&0&1&0&0&0&1&0&0&0&0&0&0&1
\end{bmatrix}$$

A systematic (1 6, 8) code is a distance-5 code, which was presented for correcting double errors within 16 bit bytes,the H matrix of the (16,8) code is:

$$H=\begin{bmatrix}
0&0&0&1&1&1&0&1&1&0&0&0&0&0&0&0\\
1&0&0&0&1&1&1&0&0&1&0&0&0&0&0&0\\
0&1&0&0&0&1&1&1&0&0&1&0&0&0&0&0\\
1&0&1&0&0&0&1&1&0&0&0&1&0&0&0&0\\
1&1&0&1&0&0&0&1&0&0&0&0&1&0&0&0\\
1&1&1&0&1&0&0&0&0&0&0&0&0&1&0&0\\
0&1&1&1&0&1&0&0&0&0&0&0&0&0&1&0\\
0&0&1&1&1&0&1&0&0&0&0&0&0&0&0&1
\end{bmatrix}$$

3. Low-cost check bit generation method

978-0-7695-3407-7/08 $25.00 © 2008 IEEE
DOI 10.1109/ICCIT.2008.14

3.1 Calculating check bits

When the input data are given to the EDAC encoder or decoder, the check bits all need to be calculated. Traditionally, an XOR tree structure has been applied in the main part of EDAC for check bit generation, The generator delay in this structure is sensitive to the format of H matrix. The number of 1's in the rows of the H matrix can fix on the number of levels in the XOR tree, if we reduce the xor operation, we can gain the less area and delay of EDAC circuit.

In the study, we optimize EDAC by using uniform 2-input xor-gate cells repetitiously in different check bit calculated equation to reduce the delay and area. Eqs. (1) ~ (8) for calculating 8-bits check data of before-mentioned systematic code(16,8).

$$c[7]=m[4]\oplus m[3]\oplus m[2]\oplus m[0] \qquad (1)$$

$$c[6]=m[7]\oplus m[3]\oplus m[2]\oplus m[1] \qquad (2)$$

$$c[5]=m[6]\oplus m[2]\oplus m[1]\oplus m[0] \qquad (3)$$

$$c[4]=m[7]\oplus m[5]\oplus m[1]\oplus m[0] \qquad (4)$$

$$c[3]=m[7]\oplus m[6]\oplus m[4]\oplus m[0] \qquad (5)$$

$$c[2]=m[7]\oplus m[6]\oplus m[5]\oplus m[3] \qquad (6)$$

$$c[1]=m[6]\oplus m[5]\oplus m[4]\oplus m[2] \qquad (7)$$

$$c[0]=m[5]\oplus m[4]\oplus m[3]\oplus m[1] \qquad (8)$$

Using the calculate process, we need sixteen 2-input XORS . We observing Eqs. (1) ~ (8) ,we can find these cells are used to generate the check bit repeatedly,

$$m[1]\oplus m[0],$$

$$m[3]\oplus m[2],$$

$$m[5]\oplus m[4],$$

$$m[7]\oplus m[6],$$

$$m[6]\oplus m[2],$$

$$m[4]\oplus m[0]$$

These cells are used twice for once check bit generation, because xor operation follow the commutative law, we can generate these cells firstly, then generate the check bits, which can reduce the number of 2-input XOR gates. If we use the (16,8)code to protect 32 bits data, we need to generate 4 × 8 bits check data simultaneity, this method can reduce 24 2-input XOR gates, which showing in the table1, With the structure in Fig. 1.

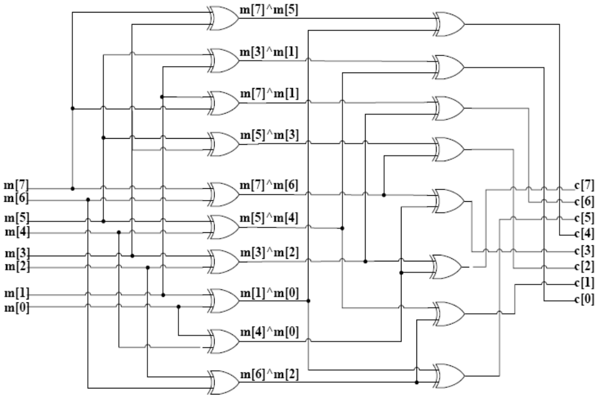

Fig 1

Table 1

Case Code	Original	Optimized	Reduced number
2-input xors for checking 8-bits data	24	18	6
2-input xors for checking 32-bit data	96	72	24

Using the same method to optimize the check bit generation of SEC-DED Hsiao code(39,32), we can calculate this cells:

$$m_{01} = m[0] \oplus m[1],$$

$$m_{23} = m[2] \oplus m[3],$$

$$m_{45} = m[4] \oplus m[5],$$

$$m_{67} = m[6] \oplus m[7],$$

$$m_{89} = m[8] \oplus m[9],$$

$$m_{1011} = m[10] \oplus m[11],$$

$$m_{1213} = m[12] \oplus m[13],$$

$$m_{1415} = m[14] \oplus m[15],$$

$$m_{1617} = m[16] \oplus m[17],$$

$$m_{1819} = m[18] \oplus m[19],$$

$$m_{2021} = m[20] \oplus m[21],$$

$$m_{2223} = m[22] \oplus m[23],$$

$$m_{2425} = m[24] \oplus m[25],$$

$$m_{2627} = m[26] \oplus m[27],$$

$$m_{2829} = m[28] \oplus m[29],$$

$$m_{3031} = m[30] \oplus m[31],$$

$$h_0 = m_{89} \oplus m_{1011},$$

$$h_1 = m_{1617} \oplus m_{1819},$$

$$h_2 = m_{2021} \oplus m_{2223}$$

Then, we calculate these Eqs.

$$\begin{aligned} c[6] = {} & m_{01} \oplus m[3] \oplus m[4] \oplus h_0 \oplus \\ & m[17] \oplus m[23] \oplus m[25] \oplus \\ & m[27] \oplus m[31] \end{aligned} \tag{9}$$

$$\begin{aligned} c[5] = {} & m[0] \oplus m[5] \oplus m[6] \oplus \\ & m[8] \oplus m_{1213} \oplus m[14] \oplus \\ & h_1 \oplus m[20] \oplus m[28] \end{aligned} \tag{10}$$

$$\begin{aligned} c[4] = {} & m[1] \oplus m[2] \oplus m[5] \oplus \\ & m[7] \oplus m[9] \oplus m[12] \oplus \\ & m[15] \oplus h_2 \oplus m[25] \oplus \\ & m[26] \oplus m[28] \end{aligned} \tag{11}$$

$$\begin{aligned} c[3] = {} & m[2] \oplus m[6] \oplus m[10] \oplus \\ & m[13] \oplus m[15] \oplus m[16] \oplus \\ & m_{2425} \oplus m_{2627} \oplus m_{2829} \oplus m_{3031} \end{aligned} \tag{12}$$

$$\begin{aligned} c[2] = {} & m[3] \oplus m[11] \oplus h_1 \oplus h_2 \oplus \\ & m_{2627} \oplus m[29] \oplus m[30] \end{aligned} \tag{13}$$

$$\begin{aligned} c[1] = {} & m[4] \oplus m[7] \oplus h_0 \oplus m_{1213} \oplus \\ & m_{1415} \oplus m[18] \oplus m[21] \oplus \\ & m[24] \oplus m[29] \end{aligned} \tag{14}$$

$$\begin{aligned} c[0] = {} & m_{01} \oplus m_{23} \oplus m_{45} \oplus m_{67} \oplus \\ & m[14] \oplus m[19] \oplus m[22] \oplus \\ & m[24] \oplus m_{3031} \end{aligned} \tag{15}$$

we can gain these results, which showing in the table2.

Table 2

Case / Code	Original of xors	Optimized xors	Reduced numeber	extra memory bits
Hsiaocode (39,32)	89	75	14	7
systematic code (16,8) for checking 32-bits data	96	72	24	32

3.2 Simulation result

Table 3 show the simulation result of the delay and area of check bit generator based on 2-input XOR gates structure in the smic 180nm process case. This result show two kinds of comparison, one for the same code transversely, the other between the two codes lengthways. EDAC for 32 bits data,the DEC systematic (1 6, 8) code with the better correction capability than EDAC for 8 bits data, the most high correction capability is 8-bits upset. Here, it need 32 extra memory bits, but SEC Hsiao code (39,32) need 7 extra memory bits.

Table 3

Code / Case	Hsiao code(39,32)	The (16,8)code	4--parallel-(16,8)code
timing	1.27ns	0.71ns	0.71ns
Optimized--timing	1.19ns	0.68ns	0.68ns
area	2541.369873	612.057617	2448.230468
Optimized-area	2195.424316	518.918335	2075.67334

4. Conclusion

In this paper, we optimized the check bit generator of the EDAC by using the uniform 2-input xor-gate cells repetitiously in different check bit calculated equation, which reduces the delay and area. For the SEC-DED Hsiao code(39,32) and DEC systematic (1 6, 8) code, we used the proposes method for reducing area and delay. Simulation results show the delay and area of check bits generator structure in the smic180nm process case. The optimized results are compared with the non- optimized results for the same code, which indicate that this method is effective to reduce the cost of EDAC circuit. A comparison of two codes show the difference of the delay and aera consumption. EDAC for 32 bits data, SEC-DED Hsiao code(32,7) need less extra memory bits than DEC systematic (1 6, 8) code ,but SEC-DED Hsiao code(32,7) takes more delay and area ,which shows that different code , different implementation can affect the cost of EDAC circuitry and be used as a guide for selecting the proper code for different application requirements

References

[1] Ronald Klein and Murali Varanasi, “A Systematic (16,8) Code for Correcting Double Errors, and Detecting Random Triple Errors”, 1996 IEEE.

[2] Craig Hafer ,Jonathan Mabra, Duane Slocum,T.S. Kalkur4 , “Next Generation Radiation-Hardened SRAM for Space Applications”,2006 IEEE.

[3] Davide Bertozzi,Luca Benini,Giovanni De Micheli, “Low power error resilient encoding for on-chip data buses”, 2002 IEEE.

[4] Heesung Lee, Joonkyung Sung and Euntai Kim, “Reducing Power in Error Correcting Code using Genetic Algorithm”,PWASET VOLUME 21 JANUARY 2007 ISSN 1307-6884.

[5] John R. Samson, Jr., Lou DeLa Torre, Paris Wiley, Thomas Stottlar, JeffRing “A Comprison Of Algorithm-Based Fault Tolerance And Traditional

Redundant Self-Checking For Seu Mitigation”,2001 IEEE.

[6] Shalini Ghosh', Sugato Bad, and Nur A. Touba, “Reducing Power Consumption in Memory ECC Checkers”,2004 IEEE.

[7] Philip P. Shirvani, Nirmal R. Saxena, and Edward J. McCluskey, “Software-Implemented EDAC Protection Against SEU”,IEEE TRANSACTIONS ON RELIABILITY, VOL. 49, NO. 3, SEPTEMBER 2000.

[8] W. Gao and S. Simmons“A study on the vlsi implemention of ecc for embedded dram”, Deparhnent of Electrical and Computer Engineering ,2003. IEEE.

[9] GUSTAVO NEUBERGER, “A Multiple Bit Upset Tolerant SRAM Memory”,ACM Transactions on Design Automation of Electronic Systems, Vol. 8, No. 4, October 2003, Pages 577–590.

[10] Hsiao,M.Y. “A class of optimal minimum odd-weight-column SECDED Codes”, IBM J. Res. Dev.,1970, 14, pp. 395–401.