Efficient Implementation of Single Error Correction and Double Error Detection Code with Check Bit Pre-computation for Memories

Sanguhn Cha and Hongil Yoon

Abstract—**In this paper, efficient implementation of error correction code (ECC) processing circuits based on single error correction and double error detection (SEC-DED) code with check bit pre-computation is proposed for memories. During the write operation of memory, check bit pre-computation eliminates the overall bits computation required to detect a double error, thereby reducing the complexity of the ECC processing circuits. In order to implement the ECC processing circuits using the check bit pre-computation more efficiently, the proper SEC-DED codes are proposed. The H-matrix of the proposed SEC-DED code is the same as that of the odd-weight-column code during the write operation and is designed by replacing 0's with 1's at the last row of the H-matrix of the odd-weight-column code during the read operation. When compared with a conventional implementation utilizing the odd-weight-column code, the implementation based on the proposed SEC-DED code with check bit pre-computation achieves reductions in the number of gates, latency, and power consumption of the ECC processing circuits by up to 9.3%, 18.4%, and 14.1% for 64 data bits in a word.**

Index Terms—**Error correction code, single error correction and double error detection code, check bit pre-computation, extended Hamming code and memory**

Manuscript received Apr. 30, 2012; revised Jun. 29, 2012.
Sch. EE., Yonesi University, Seoul, Korea
E-mail : rustle88@yonsei.ac.kr and hyoon@yonsei.ac.kr

I. INTRODUCTION

With recent advances within semiconductor technology, memory cells have become more vulnerable to electrical, thermal, and mechanical stresses [1]. Additionally, transient errors due to cosmic neutrons, alpha particles, and radiation have emerged as key reliability concerns [2, 3]. Error correction code (ECC) techniques have been used to enhance memory reliability. Specifically, the extended Hamming and odd-weight-column codes in the category of single error correction and double error detection (SEC-DED) code are commonly used [4-6].

The extended Hamming code [4, 6] are first developed from the SEC-DED code by adding an overall check bit to the Hamming single error correction (SEC) code for double error detection. The extended Hamming code requires overall bits computation to detect a double error, thereby implementing a complicated check bit generator. To reduce the complexity of the check bit generator, odd-weight-column code [5, 6] is designed to not require an overall check bit. Therefore, the check bit generator becomes simple, but the error locator and double error detection method are more complicated than the extended Hamming code.

In this paper, the overall bits computation is eliminated by check bit pre-computation during the write operation of memory despite using the error locator and double error detection method, coinciding with those of the extended Hamming code. Moreover, a new SEC-DED code is selected from the shortened SEC codes in order to efficiently utilize check bit pre-computation. Details

regarding check bit pre-computation and the proposed SEC-DED code are discussed. Comparative analyses of implementation based on the odd-weight-column code and proposed SEC-DED code with check bit pre-computation are also performed.

II. CONVENTIONAL SEC-DED CODES

The parity check matrix of the ECC is called the H-matrix [4-9]. For a (r × n) H-matrix of the (n, k) SEC-DED code, r, k and n denote the number of check bits and data bits in a word and the length of a word, respectively. The number of data bits in a word is then k=n-r, and the number of data bits used in memory is typically $k=2^{r-2}$.

A (2^{r-2} + r, 2^{r-2}) extended Hamming code is designed by adding an overall check bit in a word for the (2^{r-2} + r-1, 2^{r-2}) Hamming code. That is, the H-matrix of the extended Hamming code is formed by adding a row with all 1's, and also by adding a 1-weight column with upper r-1 0's to the H-matrix of the Hamming code [4, 6]. Fig. 1 shows the H-matrix of the (13, 8) extended Hamming code, which is formed by adding a row and a column to the H-matrix of the (12, 8) Hamming code.

During the write operation of memory using the (13, 8) extended Hamming code, the written_check bits (C1-C5) are generated using the modulo-2 sums of the written_data bits (D1-D8) and written_check bits (C1-C4) as follows:

$$\begin{aligned} C1 &= D1 \oplus D2 \oplus D4 \oplus D5 \oplus D7, \\ C2 &= D1 \oplus D3 \oplus D4 \oplus D6 \oplus D7, \\ C3 &= D2 \oplus D3 \oplus D4 \oplus D8, \\ C4 &= D5 \oplus D6 \oplus D7 \oplus D8, \\ C5 &= D1 \oplus D2 \oplus D3 \oplus D4 \oplus D5 \oplus D6 \\ &\oplus D7 \oplus D8 \oplus C1 \oplus C2 \oplus C3 \oplus C4. \end{aligned} \tag{1}$$

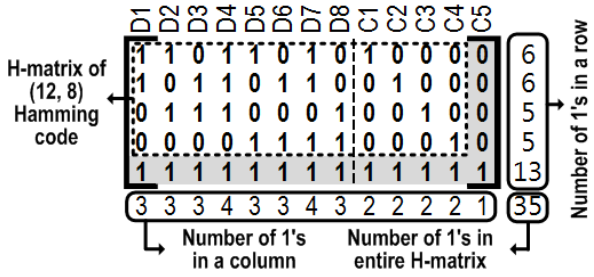

Fig. 1. H-matrix of the (13, 8) extended Hamming code.

During the read operation, the read_check bits (C1"-C5") are generated using the modulo-2 sums of the stored_data bits (D1'-D8') and stored_check bits (C1'-C4') as follows:

$$\begin{aligned} C1'' &= D1' \oplus D2' \oplus D4' \oplus D5' \oplus D7', \\ C2'' &= D1' \oplus D3' \oplus D4' \oplus D6' \oplus D7', \\ C3'' &= D2' \oplus D3' \oplus D4' \oplus D8', \\ C4'' &= D5' \oplus D6' \oplus D7' \oplus D8', \\ C5'' &= D1' \oplus D2' \oplus D3' \oplus D4' \oplus D5' \oplus D6' \\ &\oplus D7' \oplus D8' \oplus C1' \oplus C2' \oplus C3' \oplus C4'. \end{aligned} \tag{2}$$

The syndromes (S1-S5) are generated using the modulo-2 sums of the stored_check bits (C1'-C5') and read_check bits (C1"-C5"). In the extended Hamming code, there are modulo-2 sums of the overall bits in a word to generate the last written_check bit and syndrome, which are required to detect a double error.

In order to eliminate an overall check bit, odd-weight-column code is presented [5, 6]. Every column in the H-matrix of the odd-weight-column code contains an odd number of 1's. The modulo-2 sum of any odd number of odd-weight columns is always an odd-weight vector, and the modulo-2 sum of any even number of odd-weight columns is always an even-weight vector, including the 0-weight vector. Using this characteristic, the odd-weight-column code can detect a double error. Fig. 2 illustrates the H-matrix of the (13, 8) odd-weight-column code.

During the write operation, the written_check bits are generated using the modulo-2 sums of the written_data bits. During the read operation of memory, the read_check bits are generated using the stored_data bits and the syndromes are generated using the stored_check bits and read_check bits. There are no modulo-2 sums of the overall bits in a word to generate the written_check bits and syndromes.

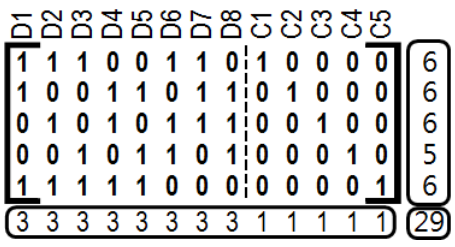

Fig. 2. H-matrix of the (13, 8) odd-weight-column code.

III. ECC PROCESSING CIRCUITS

The ECC processing circuits generally consist of the following four units: 1) a check bit generator, 2) a syndrome generator, 3) an error locator, and 4) a corrector [7-9]. In order to detect a double error, a double error detection signal (DEDS) generator is additionally required. The check bit generator and syndrome generator can be merged, resulting in reduction of one level of exclusive-or (XOR) gates [5, 6]. The parts that detect and correct the errors at the location assigned to the check bits positions can be omitted from the error locator and corrector, because only the corrected data bits are generally read out of the memory without the corrected check bits.

Fig. 3 shows the ECC processing circuits of the (13, 8) extended Hamming code. During the write operation, the written_check bits are generated through the merged check bit and syndrome generator (MCSG). The upper four written_check bits are generated using only the written_data bits. However, the fifth written_check bit is generated using the overall written_data bits and upper four written_check bits. In addition, the fifth written_check bit is generated after the upper four written_check bits are generated. Consequently, the fifth written_check bit is generated after the written_data bits pass the four levels of XOR gates of the MCSG. During the read operation, the fifth syndrome is generated using the overall stored_data bits and stored_check bits. The fifth syndrome can be generated without waiting for the generation of the upper four syndromes, because the stored_check bits are already stored in the memory cell

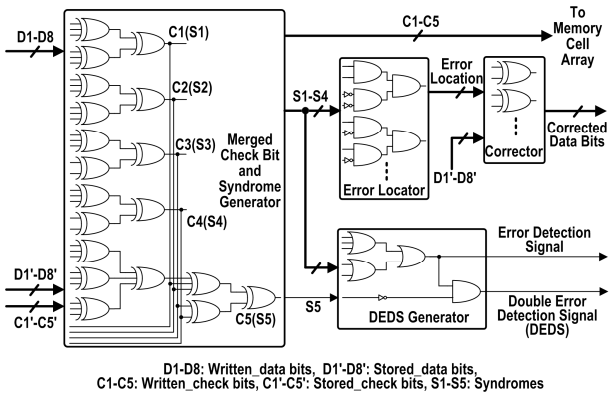

Fig. 3. ECC processing circuits of the (13, 8) extended Hamming code.

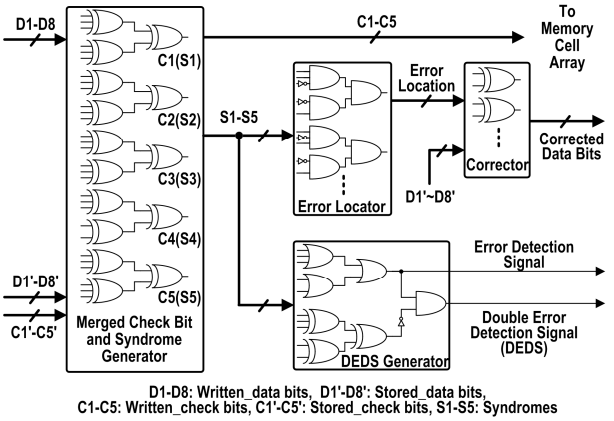

Fig. 4. ECC processing circuits of the (13, 8) odd-weight-column code.

array. Although the fifth syndrome can be generated after the stored_data bits and stored_check bits pass the three levels of XOR gates of the MCSG, the MCSG is composed of four levels of XOR gates due to the sharing the MCSG during the write and read operations.

Fig. 4 shows the ECC processing circuits of the (13, 8) odd-weight-column code. During the write operation, all written_check bits are generated without the modulo-2 sums of the overall written_data bits and written_check bits. The written_check bits are generated after the written_data bits pass the two levels of XOR gates of the MCSG. During the read operation, the syndromes are also generated after the stored_data bits and stored_check bits pass the two levels of XOR gates of the MCSG.

In the (13, 8) odd-weight-column code, all syndromes are required to find the error location. That is, the error locator based on the odd-weight-column code is implemented using the 5-input AND gates. However, only the upper four syndromes are required to find the error location in the extended Hamming code. Therefore, an error locator based on the extended Hamming code is implemented using the 4-input AND gates. In addition, the odd-weight-column code needs the additional XOR gates of the DEDS generator, compared with the extended Hamming code.

To summarize, the MCSG of the extended Hamming code is composed of larger numbers and levels of XOR gates than that of the odd-weight-column code. On the other hand, the error locator and DEDS generator based on the extended Hamming code are less complicated than counterparts based on the odd-weight-column code.

IV. CHECK BIT PRE-COMPUTATION

In the (13, 8) extended Hamming code, the equation related to the generation of the fifth written_check bit of Eq. (1) can be pre-computed as

$$C5 = D1 \oplus D2 \oplus D3 \oplus D5 \oplus D6 \oplus D8. \qquad (3)$$

During the write operation, this check bit pre-computation related to the fifth written_check bit is valid because the modulo-2 sum of any even number of the same values is zero, and the upper four written_check bits are computed using the written_data bits. Fig. 5 shows the H-matrix of the (13, 8) extended Hamming code with the check bit pre-computation. In the fifth row in the H-matrix of Fig. 1, 1's assigned in the even-weight columns are replaced by 0's. Consequently, the fifth written_check bit can simply be generated without requiring the modulo-2 sums of the overall written_data bits and written_check bits, and waiting for the generation of the upper four written_check bits, despite using the double error detection method of the extended Hamming code.

However, check bit pre-computation is unfortunately invalid for the generation of the fifth read_check bit during the read operation because the last read_check bit is generated using the stored_data bits and stored_check bits, and the upper four stored_check bits are not computed using the stored_data bits, but instead are read from the memory cell array. For example, if eight written_data bits are "10000000", five written_check bits become "11001" regardless of using check bit pre-computation during the write operation. If eight stored_data bits are "10011000" when there is a double error at the fourth and fifth stored_data bits, five read_check bits become "10111" using the H-matrix of Fig. 1. However, five read_check bits become "10110" using the H-matrix of Fig. 5. Therefore, check bit pre-computation is valid only during the write operation. In this paper, a new implementation method that uses the H-matrix with and without check bit pre-computation during the write operation and read operation respectively will be proposed to implement the ECC processing circuits based on the SEC-DED code efficiently.

D1	D2	D3	D4	D5	D6	D7	D8	C1	C2	C3	C4	C5	
1	1	0	1	1	0	1	0	1	0	0	0	0	6
1	0	1	1	0	1	1	0	0	1	0	0	0	6
0	1	1	1	0	0	0	1	0	0	1	0	0	5
0	0	0	0	1	1	1	1	0	0	0	1	0	5
1	1	1	0	1	1	0	1	0	0	0	0	1	7
3	3	3	3	3	3	3	3	1	1	1	1	1	29

Fig. 5. H-matrix of the (13, 8) extended SEC-DED code with check bit pre-computation.

V. PROPOSED SEC-DED CODE

In order to implement the ECC processing circuits efficiently, the proper $(2^{r-2}+r, 2^{r-2})$ SEC-DED code should be selected among the shortened SEC codes. The maximum length of a word of a binary SEC code is $n = 2^r-1$, when the number of check bits in a word is r. The code whose word length is less than the maximum length is called a shortened code [6]. The $(2^{r-2}+r, 2^{r-2})$ proposed SEC-DED code can be obtained by shortening a $(2^r-1, 2^r-1-r)$ SEC code. After check bit pre-computation, the H-matrix of the $(2^{r-2}+r, 2^{r-2})$ proposed SEC-DED code is subject to the following four constraints:

(1) Each column should be different from the other columns.

(2) The number of 1's in each column should be an odd number.

(3) The total number of 1's in the H-matrix should be minimal.

(4) The number of 1's in each row should be an integer as close to the ratio of the number of 1's in the H-matrix to the number of rows

The second constraint is required because every column become an odd-weight column after check bit pre-computation. The third constraint reduce the hardware overhead of the ECC processing circuits. Fewer 1's in the H-matrix means fewer modulo-2 additions and thus, faster check bit generation and error correction times are achieved. Also, a reduction in the number of 1's means fewer gates, leadings to reduced hardware overhead and increased hardware reliability [6-8]. The fourth constraint makes every row have a similar number of 1's, thereby reducing the latency [6].

The odd-weight-column code is the shortened SEC code satisfying four constraints. The H-matrix of the proposed SEC-DED code coincides with that of the odd-weight-column code during the write operation. During the read operation, the H-matrix of the proposed SEC-DED code can be designed by replacing 0's with 1's at the last row of the H-matrix of the odd-weight-column code. In other words, the proposed SEC-DED code is the odd-weight-column code during write operation and the modified version of the extended Hamming code during read operation. Therefore, it is verified that the minimum distance of the proposed SEC-DED code is at least four, because the extended Hamming code and odd-weight-column code are well known distance-4 codes.

Fig. 6 demonstrates the H-matrix of the (13, 8) proposed SEC-DED code. This H-matrix is formed by replacing 0's with 1's at the fifth row of the H-matrix of the (13, 8) odd-weight-column code in Fig. 2. The gray part is the part where 0's are replaced with 1's. During the write operation, the generation procedure of the written_check bits based on the proposed SEC-DED code using check bit pre-computation coincides exactly with the generation procedure of the written_check bits based on the odd-weight-column code.

Fig. 7 illustrates the ECC processing circuits of the (13, 8) proposed SEC-DED code. During the write operation, all written_check bits are generated using only the written_data bits. In addition, the fifth written_check bit can be generated without waiting for the generation of the upper four written_check bits. Consequently, all written_check bits are generated after the written_data bits pass the two levels of XOR gates of the MCSG, as in the odd-weight-column code. During the read operation, the fifth syndrome can also be generated without waiting for the generation of the upper four syndromes. Consequently, the fifth syndrome can be generated by

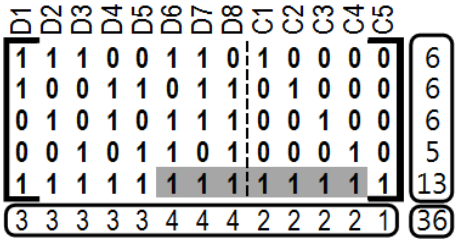

Fig. 6. H-matrix of the (13, 8) proposed SEC-DED code.

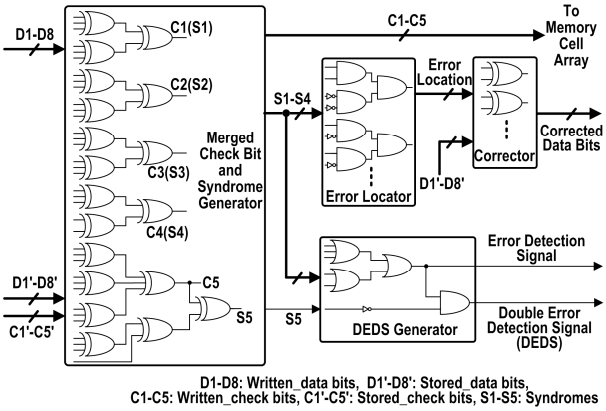

Fig. 7. ECC processing circuits of the (13, 8) proposed SEC-DED code.

passing only three levels of XOR gates, despite of the sharing the MCSG during the write and read operations. Additionally, the fifth syndrome is not concerned with finding the error location, similar to the extended Hamming code. Therefore, the error locator based on the proposed SEC-DED code is implemented using 4-input AND gates, similar to the error locator based on the extended Hamming code. The DEDS generator based on the proposed SEC-DED code coincides with that of the extended Hamming code.

The proposed SEC-DED code can simply be expanded to accommodate a large number of data bits. Figs. 8, 9, and 10 illustrate the H-matrices of the (39, 32), (72, 64), and (137, 128) proposed SEC-DED codes for 32, 64, and 128 data bits, respectively. The H-matrices are formed by replacing 0's with 1's at the seventh, eighth, and ninth rows of the H-matrices of the (39, 32), (72, 64), and (137, 128) odd-weight-column code, respectively.

VI. COMPARISON OF IMPLEMENTATION

In order to evaluate the advantages of the implementation of the ECC processing circuits based on the proposed SEC-DED codes with check bit pre-computation, we assume that 32, 64, and 128 data bits in a word are used. There have been efficient techniques to implement the check bit generator and error locator for the extended Hamming code and odd-weight-column code [10-12]. Also, these techniques can be adapted in implementation of the proposed SEC-DED code. However, only common source sharing technique using synthesis tool is used to clearly compare the

implementation based on the conventional and proposed SEC-DED codes in this paper. The ECC processing circuits of the proposed SEC-DED code and odd-weight-column code have been synthesized using Synopsis Design Vision synthesis tool and NanGate 45nm open cell library. For synthesizing the ECC processing circuits, H-matrices in Figs. 8, 9, and 10 are used as the proposed SEC-DED codes. Furthermore, the gray parts are excluded from the H-matrices in Figs. 8, 9, and 10 for the odd-weight-column codes. Table 1 tabulates the number of gates in the ECC processing circuits of the proposed SEC-DED and odd-weight-column codes for 32, 64, and 128 data bits in a word.

Compared with the odd-weight-column code, the proposed SEC-DED code brings about 8.5%, 9.3%, and 6.8% decreases in the number of gates for the ECC processing circuits for 32, 64, and 128 data bits in a word, respectively. These decreases are derived from a simple error locator and DEDS generator based on the proposed SEC-DED code. Moreover, most of the XOR gates related to the generation of the r-th syndrome are shared with the XOR gates related to the generation of the upper r-1 syndromes by synthesis.

Table 2 shows the latency of the ECC processing circuits of the proposed SEC-DED and odd-weight-column codes during the read operation. The proposed SEC-DED code results in decreases of 1.8%, 18.4%, and 22.5% in latency during the read operation, compared with the odd-weight-column code for 32, 64, and 128 data bits in a word, respectively.

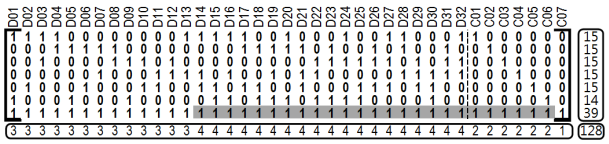

Fig. 8. H-matrix of the (39, 32) proposed SEC-DED code.

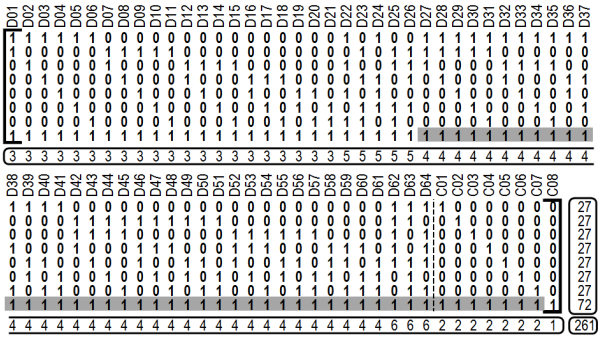

Fig. 9. H-matrix of the (72, 64) proposed SEC-DED code.

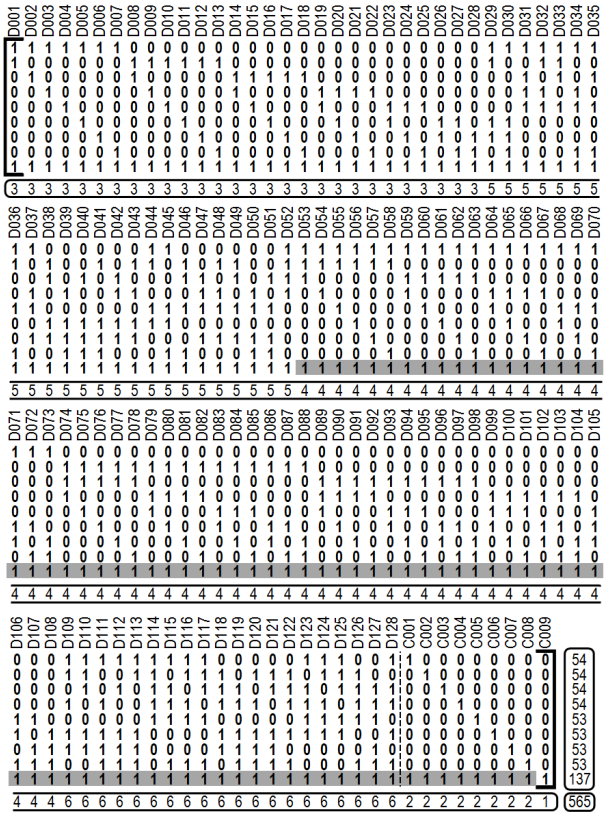

Fig. 10. H-matrix of the (137, 128) proposed SEC-DED code.

Table 1. Number of gates in the ECC processing circuits

Code	32 data bits	64 data bits	128 data bits
Odd-weight-column	199	397	768
Proposed	182	360	716

Table 2. Latency of the ECC processing circuits during the read operation (ns)

Code	32 data bits	64 data bits	128 data bits
Odd-weight-column	0.56	0.76	1.02
Proposed	0.55	0.62	0.79

Table 3. Power consumption of the ECC processing circuits during the read operation (μW)

Code	32 data bits	64 data bits	128 data bits
Odd-weight-column	420.9	870.5	1496.4
Proposed	391.9	747.5	1407.2

Table 3 tabulates the power consumption of the ECC processing circuits of the proposed SEC-DED and odd-weight-column codes during the read operation. Compared with the odd-weight-column code, the proposed SEC-DED code brings about 6.9%, 14.1%, and 6.0% decreases in power consumption of the ECC

Table 4. W(4) and the probability of miscorrected triple error (P3) and undetected quadruple error (P4)

	Code	32 data bits	64 data bits	128 data bits
W(4)	Odd-weight-column	1,363	8,397	56,449
	Proposed	1,363	8,397	56,449
P3 (%)	Both	59.7	56.3	53.9
P4 (%)	Both	1.7	0.8	0.4

processing circuits for 32, 64, and 128 data bits in a word, respectively.

If the triple error results in a syndrome vector coinciding with another column, the error locator is forced to find the normal bit location and arouse the additional fourth error [5, 6]. This is referred to as miscorrected triple error. The probability of miscorrected triple error (P3) can be computed as

$$P3 = \frac{4W(4)}{{}_nC_3}, \tag{4}$$

where W(4) is the number of cases in which the modulo-2 sum of four columns in the H-matrix is a zero vector. The SEC-DED code can actually detect even numbers of errors. If the quadruple error causes a syndromes vector to be a zero vector, the quadruple error is not detected. This is referred to as undetected quadruple error. The probability of undetected quadruple error (P4) can be computed as

$$P4 = \frac{W(4)}{{}_nC_4}, \tag{5}$$

Table 4 tabulates W(4) and the probabilities of miscorrected triple error and undetected quadruple error. The values of W(4) of the proposed SEC-DED codes are the same as those of the odd-weight-column codes, because the proposed SEC-DED codes are formed based on the odd-weight-column codes. Consequently, the miscorrected triple error and undetected quadruple error of the proposed SEC-DED codes match those of the odd-weight-column codes.

VII. CONCLUSIONS

The efficient implementation of the ECC processing circuits based on the proposed SEC-DED code is presented in this paper. During the write operation, the H-matrix of the proposed SEC-DED code becomes that of the odd-weight-column code using check bit pre-computation. Meanwhiles, the H-matrix is designed by replacing 0's with 1's at the last row of the H-matrix of the odd-weight-column code during the read operation. Consequently, the proposed SEC-DED code is the odd-weight-column code during write operation and the modified version of the extended Hamming code during read operation. Compared with the odd-weight-column code, the number of gates in the ECC processing circuits based on the proposed SEC-DED code can be reduced by up to 8.5%, 9.3%, and 6.8%, for 32, 64, and 128 data bits in a word, respectively. The proposed SEC-DED code results in decreases of 1.8%, 18.4%, and 22.5% in the latency during the read operation for 32, 64, and 128 data bits in a word, respectively. The proposed SEC-DED code brings about 6.9%, 14.1% and 6.0% decreases in the power consumption of the ECC processing circuits for 32, 64, and 128 data bits in a word, respectively. Furthermore, the probabilities of miscorrected triple error and undetected quadruple error in the proposed SEC-DED codes match those of the odd-weight-column codes. By implementing the ECC processing circuits based on the proposed SEC-DED code with check bit pre-computation, the costs of employing SEC-DED code in memories can be made more feasible.

ACKNOWLEDGMENTS

This work was supported by the National Research Foundation (NRF) grant funded by the Korean government (No. 2010-0026822).

REFERENCES

[1] Z. Al-Ars and A. J. van de Goor, "Soft faults and importance of stresses in memory testing," *Design, Automation and Test in Europe Conference and Exhibition*, pp. 1084–1089, Feb. 2004.

[2] R. C. Baumann, "Radiation-induced soft errors in advanced semiconductor technologies," *Device and Materials Reliability, IEEE Transactions on*, vol. 5, pp. 305–316, 2005.

[3] P. E. Dodd, M. R. Shaneyfelt, J. R. Schwank, and J. A. Felix, "Current and future challenges in

radiation effects on CMOS electronics," *Nuclear Science, IEEE Transactions on*, vol. 57, pp. 1747–1763, 2010.

[4] R. W. Hamming, "Error detecting and error correcting code," *Bell System Technical Journal,* vol. 26, pp. 147–160, Apr.1950.

[5] M. Y. Hsiao, "A class of optimal minimum odd-weight-column SEC-DED codes," *Research and Development, IBM Journal of,* vol. 14, Issue 4, pp. 395–401, Jul. 1970.

[6] Eiji Fujiwara, "Code design for dependable systems – Theory and practical applications," *Wiley-interscience*, 2006.

[7] S. Cha, Y. Lee, and H. Yoon, "A low-power ECC check bit generator implementation in DRAMs," *Journal of Semiconductor Technology and Science,* vol. 6, no. 4, pp. 252–256, Dec. 2006.

[8] P. K. Lala, P. Thenappan, and M. T. Anwar, "Single error correcting and double error detecting coding scheme," *IET Electronics Letters,* vol. 41, Issue 13, pp. 758–760, Feb. 2005.

[9] S. Cha and H. Yoon, "High speed, minimal area, and low power SEC code for DRAMs with large I/O data widths," *Circuits and Systems, 2007. ISCAS 2007. IEEE International Symposium on,* pp. 3026–3029, May 2007.

[10] B. Polianskikh and Z. Zilic, "Design and implementation of error detection and correction circuitry for multilevel memory protection," *Multiple-Valued Logic, 2002. ISMVL 2002. Proceedings 32nd IEEE International Symposium on,* pp. 89–95, 2002.

[11] W. Gao and S. Simmons, "A study on the VLSI implementation of ECC for embedded DRAM," *Electrical and Computer Engineering, 2003. IEEE CCECE 2003. Canadian Conference on,* vol. 1, pp. 203–206, 2003.

[12] M. Nicolaidis, T. Bonnoit, and N.-E. Zergainoh, "Eliminating speed penalty in ECC protected memories," *Design, Automation & Test in Europe Conference & Exhibition (DATE),* pp. 1–6, 2011.

Sanguhn Cha received B.S and M.S. degrees in Electrical and Electronic Engineering from Yonsei University, Seoul, Korea, in 2005 and 2007, respectively. He is currently working towards a Ph.D degree in the School of Electrical and Electronic Engineering, Yonsei University. His primary research interests include low-voltage memory circuits and technology, and error correction code for memories.

Hongil Yoon received the B.S. degree in electrical engineering and computer sciences from the University of California, Berkeley, in 1991 and the M.S. and Ph.D degrees in electrical engineering and computer science from the University of Michigan, Ann Arbor, in 1993 and 1996, respectively. From 1996 to 2002, he was with Samsung Electronics, Kiheung, Korea, being involved in the design of dynamic random access memory. Since 2002, he has been with the School of Electrical and Electronic Engineering, Yonsei University, Seoul, Korea. His research interests include low-voltage memory circuits and technology, high-frequency RF circuits and devices, and evolvable hardware design and test.