DIGITAL COMMUNICATIONS

PROJECT REPORT

Enhanced Detection of Double and Triple Adjacent Errors in Hamming Codes Through Selective Bit Placement

ELECTRICAL AND COMPUTER ENGINEERING COLLEGE OF ENGINEERING UIC

Submitted By:

Bhuvan Jain (UIN: 667704103)

TABLE OF CONTENTS

1. OVERVIEW OF THE PROJECT

Hamming codes that can correct one error per word are widely used to protect memories or registers from soft errors. As technology scales, radiation particles that create soft errors are more likely to affect more than 1 bit when they impact a memory or electronic circuit. This effect is known as a multiple cell upset (MCU), and the registers or memory cells affected by an MCU are physically close. To avoid an MCU from causing more than one error in a given word, interleaving is commonly used in memories. With interleaving, cells that belong to the same logical word are placed apart such that an MCU affects multiple bits but on different words. However, interleaving increases the complexity of the memory device and is not suitable for small memories or content-addressable memories. When interleaving is not used, MCUs can cause multiple errors in a word that may not even be detected by a Hamming code. In this paper, a technique to increase the probability of detecting double and triple adjacent errors when Hamming codes are used is presented. The enhanced detection is achieved by placing the bits of the word such that adjacent errors result in a syndrome that does not match that of any single error. Double and triple adjacent errors are precisely the types of errors that an MCU would likely cause, and therefore, the proposed scheme will be useful to provide error detection for MCUs in memory designs.

2. INTRODUCTION

Hamming codes were introduced more than 60 years ago [1] and they are still used in many applications. One example is the protection of memories or registers against radiation induced soft errors [2]. A soft error occurs when a radiation particle hits the device and changes the logical value of a memory cell or register. The data is encoded when it is written into the memory and decoded when it is read. Therefore, the encoding and decoding latency directly impact the memory access time. One example of codes for which decoding can be done with low delay is Single Error Correction (SEC) codes. SEC codes have a minimum distance of three and therefore a double error can be mistaken for a single error and erroneously corrected. To avoid this issue Single Error Correction Double Error Detection (SEC-DED) codes are preferred in memory applications [3], these codes have a minimum distance of four. There are different options for SEC-DED codes that reduce slightly the implementation cost [4] or the miscorrection probability for triple errors [5]. In any case, Hamming codes are attractive as they are simple to construct for any word length and the encoding and decoding can be done with low delay. Hamming codes are SEC codes and can be extended with a parity bit covering all bits to implement a SEC-DED code [1]. Therefore, they are suitable for memory applications and also to protect registers in digital circuits.

In this project, we have devised a technique to maximize the probability that a Hamming code detects double adjacent errors and that a parity extended Hamming code detects triple adjacent errors. This was achieved by placing the bits of the word such that those adjacent errors provoke a syndrome value that is different from those caused by single errors. The results show that the detection of such adjacent errors can be close to 100% in some cases. Therefore, the proposed technique can be useful to avoid SDC in the presence of MCUs.

2.1. HAMMING CODES

Hamming codes are linear block error-correcting codes that were proposed by R.W. Hamming [1]. They provide single error correction or double error detection. For any positive integer $m \geq 3$ they have the following parameters [2], $\boldsymbol{n = 2^m - 1, k = n - m, d_{min} = 3}$ where ***n* is the block size**, ***m* the parity check bits**, ***k* the number of information bits** and **d_{min}** the **minimum distance of the code**. Hamming codes are linear codes and, can be generated and decoded using the generator and parity-check matrices, respectively. A Hamming code can be used to correct single errors or, alternatively, to detect single and double errors. As the minimum distance between two words is three, it is not possible to distinguish between single and double errors. Hamming codes are Single Error Correction (SEC) codes and can be extended with a parity bit covering all bits to implement a Single Error Correction Double Error Detection (SEC-DED) code [1].

Hamming codes can be computed in linear algebra terms through matrices because Hamming codes are linear codes. For the purposes of Hamming codes, two Hamming matrices can be defined: the code generator matrix G and the parity-check matrix H:

$$\mathbf{G} := \begin{pmatrix} 1 & 1 & 0 & 1 \\ 1 & 0 & 1 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}, \qquad \mathbf{H} := \begin{pmatrix} 1 & 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 \end{pmatrix}.$$

[6]

Rows 1, 2, and 4 of **G** should look familiar as they map the data bits to their parity bits where p_1 covers d_1, d_2, d_4, p_2 covers d_1, d_3, d_4 and p_3 covers d_2, d_3, d_4. The remaining rows (3, 5, 6, 7) map the data to their position in encoded form and there is only 1 in that row, so it is an identical copy. In fact, these four rows are linearly independent and form the identity matrix.

Suppose we want to transmit this data (1011) over a noisy communications channel. Where, all source vectors are assumed to be equiprobable. We take the product of **G** and **p**, with entries modulo 2, to determine the transmitted codeword **x**:

$$\mathbf{x} = \mathbf{G}\mathbf{p} = \begin{pmatrix} 1 & 1 & 0 & 1 \\ 1 & 0 & 1 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 1 \\ 0 \\ 1 \\ 1 \end{pmatrix} = \begin{pmatrix} 2 \\ 3 \\ 1 \\ 2 \\ 0 \\ 1 \\ 1 \end{pmatrix} = \begin{pmatrix} 0 \\ 1 \\ 1 \\ 0 \\ 0 \\ 1 \\ 1 \end{pmatrix}$$

[6]

Now, after the matrix multiplication and modulo-2 operation, 0110011 will be transmitted instead of 1011.

If there aren't any error occurs during transmission, then the received codeword **r** is the same as the transmitted codeword **x.** The receiver multiplies **H** and **r** to obtain the **syndrome** vector **z**, which indicates whether an error has occurred, and if so, for which codeword bit. Performing this multiplication, we get,

$$\mathbf{z} = \mathbf{H}\mathbf{r} = \begin{pmatrix} 1 & 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 \end{pmatrix} \begin{pmatrix} 0 \\ 1 \\ 1 \\ 0 \\ 0 \\ 1 \\ 1 \end{pmatrix} = \begin{pmatrix} 2 \\ 4 \\ 2 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix}$$

[6]

In the example used above, as the syndrome vector is null, there is no error during transmission. If an error occurs at a bit, that bit in binary will be returned by the syndrome.

The parity bits are organized in special way so different incorrect bits produce different error results when decoding. Some possible values of the parameters are illustrated in Table I. For memory applications, the number of information bits *k* is commonly a power of two and Hamming codes are shortened to fit that word length as illustrated in Table II.

TABLE I
HAMMING CODES PARAMETERS

k	n
4	7
11	15
26	31
57	63
120	127
247	255
503	511

TABLE II
SHORTENED HAMMING CODES PARAMETERS

k	n
8	12
16	21
32	38
64	71
128	136
256	265

3. DOUBLE BIT ERROR DETECTION WITH SHORTENED HAMMING CODES

The shortened versions of Hamming codes have a special behavior. There are some double error combinations that are detected and not erroneously corrected. Additionally, some triple errors combinations do not generate a valid word and, therefore, can be detected as well and not miscorrected when using the extended Hamming code. This feature comes from the fact that these errors generate a new word which is 2 bits away from a valid code word. Coming back to the Lexicographic check matrix and the syndrome calculation and focusing on shortened Hamming codes, code words with errors can generate syndrome vectors which do not correspond to a code position due to shortening.

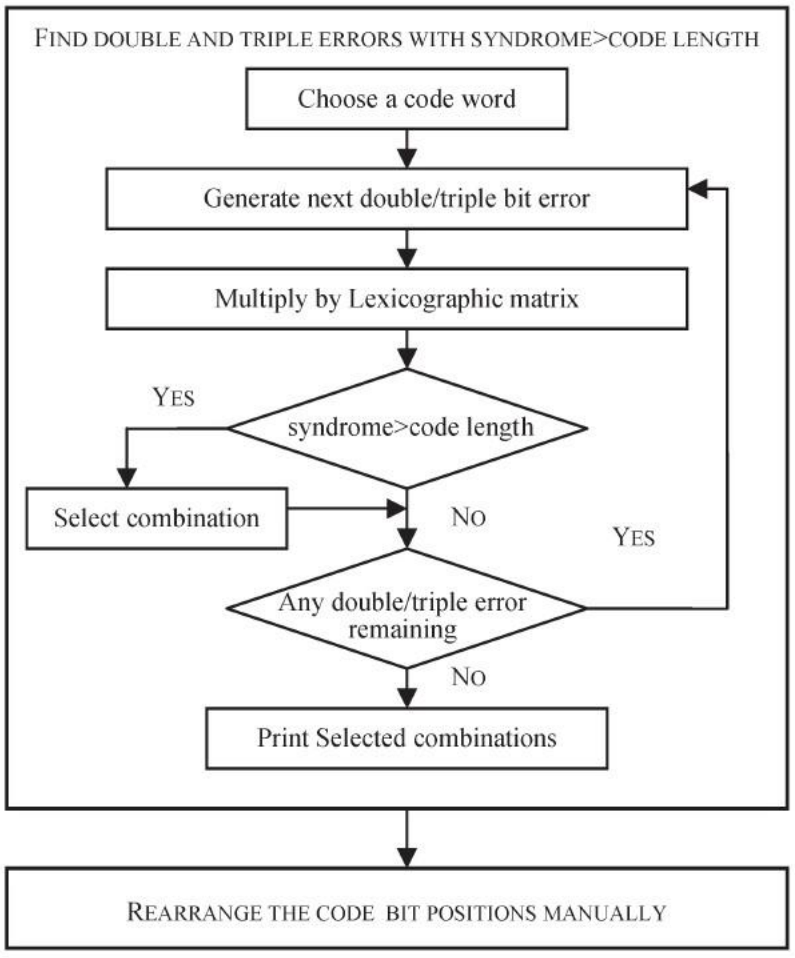
The syndrome vector in the Hamming (12, 8) code (000010110100) has 4 bits. So its maximum value is 15, but there are only 12 positions in the shortened code. If a syndrome of 13, 14, or 15 is found, the error is detected, but not corrected. For instance, when there is a double error in the code word (000010110100) affecting positions 1 and 12, the error word is (100010110101). When it is processed with the lexicographic check matrix, it results in a syndrome vector (1011), which is the binary value of position 13. It cannot be corrected as it does not match a valid code bit position. In Hamming (13, 8) a triple error in the code bits is usually detected as a single error and corrected erroneously. However, some triple error combinations are detected without correction. In the example, when changing bits 1, 7 and 11 in the original word (0000101101000) produces the erroneous word (1000100101100). If the parity bit is excluded and the lexicographic check matrix from the original Hamming code is used, the syndrome vector is (1011) instead of the null vector. It is not corrected as the syndrome value does not correspond to that of any single error, therefore there are at least two bits in error. These special combinations of double and triple bit errors are randomly distributed through the code word and only a small subset of them includes adjacent bits. As MCUs typically cause errors in adjacent bits, most of double and triple bit errors caused by them are miscorrected. From the above discussion it becomes apparent that by changing the bit placement, the probability of double or triple adjacent error detection can be maximized. For example, a simultaneous error in positions 1 and 12 in the Hamming (12, 8) code is detected and no correction is performed. So we can reorder the code word to place together these two bits. Similarly, for the extended Hamming (13, 8) code, errors in bits 1, 7 and 11 are also detected appropriately. Thus, we can place them next to each other in any possible way (e.g., 11, 1 and then 7).

4. SELECTIVE BIT PLACEMENT STRATEGY

The objective of our project is to reorder the bits of the code word to maximize the adjacency of the special combinations. To achieve this goal, the following procedure is followed:

• Using a MATLAB, we found all double and triple error combinations which produce a syndrome that does not match any of the ones caused by a single error.
• Then we reorder the word manually to maximize the number of adjacent bits matching the error combinations found in the previous stage.

The detailed procedure for selective bit placement can be seen in the flow chart below:



FIND DOUBLE AND TRIPLE ERRORS WITH SYNDROME>CODE LENGTH
Choose a code word
Generate next double/triple bit error
Multiply by Lexicographic matrix
YES
syndrome>code length
Select combination
NO
YES
Any double/triple error remaining
NO
Print Selected combinations
REARRANGE THE CODE BIT POSITIONS MANUALLY

[7]

5. SIMULATION RESULTS

5.1. Adjacent "2" error bit detection for HAMMING [12,8]

Code-word = 010101010111

Error bits= (1 5), (1 10), (2 3), (2 4), (2 6), (2 11), (3 10), (4 6), (5 10), (6 7), (7 9), (7 10), (9 10), (10 12)

Number of 2-bit error detected= 14

Adjacent bits in error= (2 3), (6 7), (9 10)

Only 3 out of 11 adjacent pairs are detected using normal order.

Bit Placement												Detection	
1	2	3	4	5	6	7	8	9	10	11	12	3/11	27.27%

After doing selective bit placement strategy, 8 out 11 adjacent pairs are detected, except (3 11), (9 12), (8 12)

Bit Placement												Detection	
1	5	10	3	11	2	4	6	7	9	12	8	8/11	72.72%

5.2. Adjacent "3" error bit detection for Extended HAMMING [13,8]

Code-word = 1010010010000

Error bits =

(1 2 4), (1 2 12), (1 3 8), (1 4 10), (1 5 8), (1 9 11), (1 9 12), (2 3 4), (2 3 12), (2 4 10), (2 5 7), (2 5 8), (2 6 10), (2 9 12), (3 4 12), (3 5 7), (3 6 10), (3 7 11), (3 9 10), (4 5 7), (4 6 10), (4 7 11), (4 9 10), (5 7 10), (5 10 12), (6 7 12), (6 8 9), (10 11 12)

Number of 3-bit error detected = 28

Adjacent bits in error= (2 3 4) (10 11 12)

Only 2 out of 11 adjacent pairs are detected using normal order.

Bit Placement													Detection	
1	2	3	4	5	6	7	8	9	10	11	12	p	2/11	18.18%

After doing selective bit placement strategy, 8 out 11 adjacent pairs are detected, except (7 10 11), (1 11 12), (2 4 6), (4 6 8), (8 9 p)

Bit Placement													Detection	
2	5	7	10	11	12	1	2	4	6	8	9	p	6/11	54.54%

5.3. Adjacent "2" error bit detection for HAMMING [21,16]

Code-word = 101011001100001111001

Error bits=

(1 6), (1 8), (1 10), (1 12), (1 14), (1 16), (1 17), (1 19), (2 13), (2 15), (3 6), (3 8), (3 10), (3 12), (3 14), (3 16), (3 17), (4 13), (4 15), (5 6), (5 8),(5 10), (5 12), (5 14), (5 16), (5 17), (6 12), (6 13), (6 14), (6 15), (7 17), (7 18), (7 20), (8 10), (8 13), (8 15), (9 17), (9 18), (9 20), (10 13), (10 15), (11 17), (11 18), (11 20), (12 13), (12 15), (12 16), (13 18), (13 19), (13 20), (13 21), (14 15), (14 16), (15 18), (15 19), (15 20), (15 21), (17 18), (17 20)

Number of 2-bit error detected = 59

Adjacent bits in error= (5 6), (12 13), (14 15), (17 18)

Only 4 out of 20 adjacent pairs are detected using normal order.

Bit Placement																					Detection	
1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	4/20	20%

After doing selective bit placement strategy, 17 out 20 adjacent pairs are detected, except (4,9), (2,4), (2,21)

Bit Placement																					Detection	
19	1	14	6	3	8	13	10	15	12	16	5	17	7	20	11	18	9	4	2	21	17/20	85%

6. CONCLUSION

In this project, a technique to maximize the probability of detecting adjacent errors in Hamming codes has been developed. The enhanced detection is achieved by selectively placing the bits in the memory such that adjacent errors produce a syndrome that does not match any of those that correspond to a single error. The proposed approach has been implemented and tested for different word sizes showing that it can be effective, achieving in some cases a large error detection rate for double and triple adjacent errors. The proposed scheme does not require any additional circuitry.

For our selective bit placement strategy code, we have increased our efficiency for adjacent 2 bit-error detections from 27.27% to 72.72% in Hamming [12,8] and 20% to 85% in Hamming [21,16]. For adjacent 3 bit-error detection we have increased the efficiency in Hamming [13,8] from 18.18% to 54.54%.

7. REFERENCES:

[1] R. W. Hamming, “Error detecting and error correcting codes,” *Bell Syst. Tech. J.*, vol. 29, no. 2, pp. 147–160, Apr. 1950.

[2] R. C. Baumann, “Radiation-induced soft errors in advanced semiconductor technologies,” *IEEE Trans. Device Mater. Rel.*, vol. 5, no. 3, pp. 301– 316, Sep. 2005.

[3] C. L. Chen and M. Y. Hsiao, “Error-correcting codes for semiconductor memory applications: A state-of-the-art review,” *IBM J. Res. Develop.*, vol. 28, no. 2, pp. 124–134, Mar. 1984.

[4] M. Y. Hsiao, “A class of optimal minimum odd-weight column SEC-DED codes,” *IBM J. Res. Develop.*, vol. 14, no. 4, pp. 395–401, Jul. 1970.

[5] M. Richter, K. Oberlaender, and M. Goessel, “New linear SEC-DED codes with reduced triple bit error miscorrection probability,” in *Proc. 14th IEEE IOLTS*, Jul. 2008, pp. 37–42.

[6] https://en.wikipedia.org/wiki/Hamming(7,4)

[7] Alfonso Sánchez-Macián, Pedro Reviriego and Juan Antonio Maestro “Enhanced Detection of Double and Triple Adjacent Errors in Hamming Codes through Selective Bit Placement”, *IEEE transactions on device and material reliability*, vol. 12, no. 2, pp. 357-362 June 2012.