Experimental study on Hamming and Hsiao Codes in the Context of Embedded Applications

G. Tshagharyan, G. Harutyunyan, S. Shoukourian, Y. Zorian
Synopsys
{grigor.tshagharyan, gurgen.harutyunyan, samvel.shoukourian, yervant.zorian}@synopsys.com

Abstract

Increasing soft error rate and decreasing technological nodes sizes pave a way for Error Correcting Codes (ECC) widespread use in embedded systems. Depending on application safety goals and acceptable performance and area overhead, different codes can be selected. The goal of this paper is to investigate the efficiency and expediency of two of the most prominent ECC codes, Hamming and Hsiao, in the context of embedded memories and provide practical guidance for their exploitation.

1. Introduction

The existence of soft errors in System on Chips (SoC), especially in embedded memories, is already well known for a long time as a result of interaction with charged particles or due to radiation. The main cause of such errors are alpha particles emitting in integrated circuits and cosmic rays coming from the outer space as recent studies show [1]-[3]. Soft errors, compared to hard errors, have transient nature, and do not cause permanent damage to memory cells. Over the last 50 to 60 years, this field was well investigated and a fair amount of research studies have been published. In the end, number of solutions were proposed to deal with these types of errors having different complexity, performance and area indices.

With the growing reliability concerns, detection and correction of soft errors becomes critical. This problem is especially compounded by the fact that over the recent years the modern technology continues to aggressively shrink and has already reached and, moreover, surpassed the boundary of 10nm feature sizes. This was made possible due to such technologies like FDSOI and FinFET. With each technology node reduction, the technology complexity and the packing density of the memory array constantly increases. Subsequently, the probability of occurrence of transient errors increases proportionally [2], [4]. This means that it should be considered as a serious menace for applications which specifically require high level of safety and reliability like automotive, biomedical, and to some extent also Internet of Things (IoT) and mobile industries.

Fortunately, there are already established solutions which were verified and proved to be viable over the years. ECC is a technique which is being commonly used for this purpose. It is based on the concept of using check bits which are determined by even parity checks over selected data bits and storing or transmitting them along with the data bits. Several variations of ECC codes exist most common of which are:

- Single error detection (SED)
- Single error correction (SEC)
- Single error correction & Double error detection (SEC-DED)

There are also other more complex types of ECC codes which provide multiple error detection and correction ([5]-[6]), but they usually imply high performance and area penalty and are therefore not practical for usage especially in embedded and real-time applications.

Despite the fact that coding theory has developed greatly over the recent years [5], [7], many of the first developed codes are still being used as robust and light-weight solutions. Particularly, Hamming SEC code and its extension to SEC-DED [8] designed by one of the originators of this theory are still the first-choice codes for many applications. Another ECC code designed by Hsiao in 1970 [9] is modified version of Hamming SEC-DED code which is also widely used in modern systems. Hamming and Hsiao codes provide different trade-offs between implementation of the code and the imposed overhead.

This paper aims to provide the comparative overview of both Hamming and Hsiao codes from the perspective of their usability for modern embedded memories. Experimental results are included to provide complexity, area, and performance numbers for practical application scenarios.

2. Coding Theory Basics

Coding theory is a broad concept and has many applications in various fields performing data compression, cryptographic, networking, data storage

978-1-5386-3299-4/17/$31.00 ©2017 IEEE

and many other functions. It was initially developed for communication systems working in the noisy environments. Shannon and Hamming are usually considered as the originators of this theory since they were first who investigated the problem of error-prone communication channels and developed first codes.

In this paper only the block codes are discussed, i.e., when the data space is divided into blocks of fixed number of bits, which are referred as data words and the code is constructed for each of the words separately. Assume that the original word consists of k data bits. This means that there are 2 to the power of k possible words in the data space considering that each data bit contains either 0 or 1 value. Hamming showed that each data word can be augmented with certain number of redundant bits that would allow to detect whether errors have occurred during the word transmission and eventually correct them. The function of the additional bits is to perform the parity checks over certain data bits and using the fail pattern detect the mismatch.

ECC is a two-step process consisting of encoding and decoding. During the encoding step all the necessary calculations are performed to generate required number of check bits. During the decoding step using check bits values it is determined whether the actual retrieved data contains errors or not and fix them if possible.

Before moving forward, some terminology must be defined first. The word comprised of data bits and check bits is called a codeword and thereby the code is defined as the set of all possible codewords. The code is called linear if a linear combination of any of its two codewords is also a codeword. The Hamming weight of the codeword is defined as the number of its non-zero elements. The number of positions two codewords differ, i.e., otherwise said, the Hamming weight of the XOR sum of two codewords is called the Hamming distance between two codewords. The minimum Hamming distance of a code is defined as the minimum of the distances between all pairs of its codewords, i.e., in case of linear codes it is the minimum weight of its nonzero codewords. This notion of distance allows to define the relationship between minimum distance of the code and the codes capability of error detection and/or correction. Table 1 shows the minimum Hamming distance requirements in the case of SED, SEC and SEC-DED codes respectively.

Table 1. Minimum Hamming Distance Criteria

ECC Type	*Minimum Hamming Distance*
SED	2
SEC	3
SEC-DED	4

For the convenience of calculations, linear code can be fully specified with its r x (k+r) parity-check matrix, where k is the number of data bits and r is the number of check bits. It determines which data bits are added to generate the check bits and has the following structure:

$$H=[H_{r,k}\,|\,C_r]=\begin{bmatrix} h_{0,0} & h_{0,1} & \dots & h_{0,k-1} & c_{0,0} & c_{0,1} & \dots & c_{0,r-1} \\ h_{1,0} & h_{1,1} & \dots & h_{1,k-1} & c_{1,0} & c_{1,1} & \dots & c_{1,r-1} \\ \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots \\ h_{r-1,0} & h_{r-1,1} & \dots & h_{r-1,k-1} & c_{r-1,0} & c_{r-1,1} & \dots & c_{r-1,r-1} \end{bmatrix} \quad (1)$$

H is also called a generator matrix. The matrix Cr does not play essential role for check bits' generation part but it is required during the decoding step where the obtained codeword is checked for the errors. If we denote the data bits as $d = (d_1, \dots, d_k)$ and check bits as $c = (c_1, \dots, c_r)$ then c is calculated in the following way:

$$c = H * (d,0) = H_{r,k} * d \quad (2)$$

Denote the resulting codeword comprised of data and check bits as w, i.e., w = (d, c). The important characteristic of the parity-check matrix is that for any error-free codeword w the following equation holds:

$$s = w * H^T = 0 \quad (3)$$

In equation (3), the computed s vector of length r is called an error syndrome. So this means that if for any word w, its syndrome s=0 then w is a codeword. Depending on the code's capability of error detection and correction, the syndrome allows to determine the nature of the error (whether it is single-bit error, double-bit or other) as well as to localize the error position.

3. Hamming Code

Hamming Code is one of the first ECC codes, designed by Hamming as early as in 1950. Hamming proposed a scheme to build a code which satisfied a minimum distance criterion for SEC. To achieve it, the parity-check matrix requires all columns to be unique and non-zero. The general concept lying behind the construction of Hamming parity-check matrix is using the binary representation of bits' positions in the word. For the Cr matrix the identity matrix is used, i.e., the binary representations of powers of 2's (1, 2, 4, …) since it has the advantage of making the parity check bits independent of each other. The columns of $H_{r,k}$ matrix are constructed using the remaining natural numbers' binary representations in the ascending order (3, 5, 6, 7, etc.). So the H matrix has the following structure:

$$H=[H_{r,k}\,|\,I_r]=\begin{bmatrix} 1 & 1 & 0 & \dots & 1 & 0 & \dots & 0 \\ 1 & 0 & 1 & \dots & 0 & 1 & \dots & 0 \\ 0 & 1 & 1 & \dots & 0 & 0 & \dots & 0 \\ \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots \\ 0 & 0 & 0 & \dots & 0 & 0 & \dots & 1 \end{bmatrix} \quad (4)$$

From equation (2) it follows that the first check bit checks the parity of those data bits whose corresponding column-vectors in H matrix contain 1 in the first position, i.e., are odd numbers in decimal representation (3, 5, 7, …). Similarly, second check bit checks those

data bits whose corresponding column-vectors contain 1 in the second position (3, 6, 7, …) and so forth. This coding scheme allows to make the implementation of Hamming code quite simple and easy to implement. Number of check bits should be selected such that the syndrome calculated via equation (3) can differentiate between error-free case (i.e., when s = (0, …, 0)) and each separate case when the error is situated in one of the k data bits or r check bits. In other words, syndrome should be able to differentiate between k+r+1 distinct cases, which is possible only when $2^r \geq k+r+1$. That means r should be selected as a minimum natural number for which

$$2^r - r - 1 \geq k \quad (5)$$

After constructing the syndrome, the following procedure is used to determine the existence of errors:

1. If s = 0, then the codeword is error-free.
2. If s ≠ 0 and the syndrome is identical to i-th column-vector in H, then the corresponding error is in i-th bit of the codeword (data or check bit). In order to correct the error, i-th bit of the word should be inversed.
3. Otherwise, two or greater number of errors are detected which is not possible to correct.

It is important to note that there is a certain probability that if codeword contains two or greater number of errors, it's syndrome may match one of the column-vectors in H, in that case it is treated as a single error and miscorrection is performed. Moreover, there is a small chance that erroneous word's syndrome is equal to 0, then the received codeword is treated as error-free.

Hamming also provided a way to extend Hamming SEC code into SEC-DED code. The main idea is to add overall parity check bit, which checks the parity of all data bits in a word. The parity-check matrix of the Extended Hamming code has the following structure:

$$H = \begin{bmatrix} 1 & 1 & 0 & \dots & 1 & 0 & \dots & 0 & 0 \\ 0 & 0 & 1 & \dots & 0 & 1 & \dots & 0 & 0 \\ 0 & 1 & 1 & \dots & 0 & 0 & \dots & 0 & 0 \\ \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots \\ 0 & 0 & 0 & \dots & 0 & 0 & \dots & 1 & 0 \\ 1 & 1 & 1 & \dots & 1 & 1 & \dots & 1 & 1 \end{bmatrix} \quad (6)$$

The number of check bits of Hamming SEC-DED code compared to SEC code is always one more, thus the equation (5) can be rewritten in the following way for Extended Hamming code:

$$2^{r-1} - r \geq k \quad (7)$$

The decoding procedure of Extended Hamming code is a little bit different compared to Hamming SEC code and has the additional step for double error detection:

1. If s = 0, then the codeword is error-free.
2. If s ≠ 0 and $s_r = 0$ (overall parity bit is 0), then double error (or even number of errors) is detected, which is not possible to correct.
3. If s ≠ 0 and the syndrome is identical to i-th column-vector in H, then the corresponding error is in i-th bit of the codeword (data or check bit). In order to correct the error, i-th bit of the word should be inversed.
4. Otherwise, three or greater odd number of errors is detected which is not possible to correct.

Similar to the case with SEC code there is a certain probability of miscorrection and four or greater even number of errors remaining undetected.

4. Hsiao Code

In 1970 Hsiao proposed a new code scheme which is an optimization over Hamming code. The idea is based on constructing the parity-check matrix consisting of distinct, odd-weight column-vectors. Since the columns are distinct and non-zero they are linearly independent thus Hamming distance 3 requirement is satisfied for SEC. Moreover, sum of any three distinct odd-weight vectors is also an odd-weight vector, which means that distance 4 requirement is also satisfied and the designed code is SEC-DED per Table 1.

The parity-check matrix for Hsiao code is constructed in a slightly different way. For C_r matrix all combinations of 1-out-of-r column-vectors are used. $H_{r,k}$ matrix is filled with combinations of 3-out-of-r column-vectors followed by 5-out-of-r, 7-out-of-r and so forth until all k columns get selected. The advantage over Hamming SEC-DED code is that overall parity check is not required and the number of 1's in the matrix is minimized resulting into faster calculations. If the odd-weight column-vectors are selected in a way that the number of 1's in each row is kept close to the average number of 1's in a row, then it is possible to reduce the number of required logic gate levels of the hardware.

The double error detection is possible since if there are even number of errors in the codeword the calculated syndrome will have an even weight. The number of required check bits should be enough to generate k distinct odd-weight columns thus it should be selected as the minimum natural number for which:

$$k \leq \sum_{\substack{i=1 \\ i=odd}}^{\leq r} \binom{r}{i} - r \quad (8)$$

Easy to show that this leads to the same number as in the case of Hamming SEC-DED code since

$$\sum_{\substack{i=1 \\ i=odd}}^{\leq r} \binom{r}{i} = \sum_{\substack{i=1 \\ i=even}}^{\leq r} \binom{r}{i} = 2^{r-1} \quad (9)$$

The decoding procedure for Hsiao is the following:

1. If s = 0, then the codeword is error-free.
2. If s ≠ 0 and overall parity (i.e., XOR sum) of all syndrome bits is equal to zero then double error (or even number of errors) is detected, which is not possible to correct.
3. If s ≠ 0 and the syndrome is identical to i-th column-vector in H, then the corresponding error is in i-th bit of the codeword (data or check bit). In order to correct the error, i-th bit of the word should be inversed.
4. Otherwise, three or greater odd number of errors is detected which is not possible to correct.

The same observations can be done also for cases when 3 or greater number of errors are not detected or miscorrected although the experimental results brought in [9] show that their percentage has decreased compared to Hamming code.

4. Experimental Results

Considered ECC codes have wide application in the scope of embedded applications. This especially refer to random access memories (RAMs) which span most of the space on system-on-chip (SOC). In this case the goal is to ensure that the stored data was not corrupted over the course of time while SoC is in mission mode. The encoder task is to calculate the values of check bits during the write operation to memory and store them along with the data. Meanwhile the decoder task is to check whether the same data is received during the read operation from memory. To evaluate and compare the performance of ECC codes for RAMs, several experiments were done for different configurations of memories. For experiments 16nm single port memory instances were used. The implementation of Hamming and Hsiao codes was done exactly the way they were described above. Tables 2 and 3 demonstrate the obtained area and logical gate levels numbers for both Hamming and Hsiao codes for different data lengths. As experiments show Hamming codes are preferable for smaller data lengths while the efficiency of Hsiao codes is becoming visible for bigger data lengths mainly due to improved decoding logic.

Table 2. Logical Gate Levels Numbers

ECC Type	*Number of Data Bits*	*Number of Check Bits*	*Encoder Logic Levels*	*Decoder Logic Levels*
Hamming SEC	8	4	2	6
	64	7	5	14
	310	9	12	17
Hamming SEC-DED	8	5	2	6
	64	8	6	15
	310	10	14	18
Hsiao SEC-DED	8	5	2	8
	64	8	5	13
	310	10	10	15

Table 3. Area numbers

ECC Type	*Number of Data Bits*	*Number of Check Bits*	*Encoder Area (um^2)*	*Decoder Area (um^2)*
Hamming SEC	8	4	7.78	16.38
	64	7	70.81	148.21
	310	9	350.28	573.92
Hamming SEC-DED	8	5	8.92	19.39
	64	8	82.06	152.57
	310	10	407.26	582.06
Hsiao SEC-DED	8	5	9.02	20.03
	64	8	80.92	137.49
	310	10	402.64	535.90

5. Summary

The aim of this paper is to conduct the research study on two of the most popular linear ECC codes, Hamming and ECC, and explore the efficiency of their application in embedded systems. As the results show Hamming codes are simpler to implement and display better efficiency for smaller data lengths while Hsiao code are more complex but have better performance especially for larger data sizes. This means there is an implementation and performance trade-off between Hamming and Hsiao codes which should be considered for each specific application scenario.

6. References

[1] Actel Corporation, "Understanding Soft and Firm Errors in Semiconductor Devices: Questions and Answers", http://www.actel.com, Dec. 2002.

[2] Tezzaron Semiconductor, "Soft Errors in Electronic Memory – A White Paper", Jan. 2004.

[3] R. C. Baumann, "Radiation-induced soft errors in advanced semiconductor technologies", IEEE Transactions on Device and Materials Reliability, Vol. 5, No. 3, Sep. 2005, pp. 305-316.

[4] S. Kaushik, "How to use ECC to protect embedded memories", May 2013.

[5] E. Fujiwara H.-W. Cheng, Y. Li, "Code Design for Dependable Systems: Theory and Practical Applications", New Jersey, USA, Wiley-Interscience Publication, 2006.

[6] M. Poolakkaparambil, J. Mathew, A. M. Jabir, S. P. Mohanty, "Low complexity cross parity codes for multiple and random bit error correction", International Symposium on Quality Electronic Design (ISQED), March 2012, pp. 57 – 62.

[7] M. Sudan, "Coding Theory: Tutorial and Survey", Foundations of Computer Science Symposium (FOCS 2001), Oct. 2001, pp. 1-18.

[8] R. W. Hamming, "Error Detecting and Error Correcting Codes", Bell System Technical Journal, Vol. 29, No. 2, April 1950, pp. 147-160.

[9] M.Y. Hsiao, "A Class of Optimal Minimum Odd-weight-column SEC-DED Codes", IBM Journal of Research and Development, Vol. 14, No. 4, July 1970, pp. 395-401.