Detecting and Correcting Multiple Cell Upsets With 64-Bit Decimal Matrix Code in Memories

N.V. Satyanarayana¹, Ch.N.L. Sujatha², J.S.S. Ramaraju³
¹M.Tech ECE Pursuing, Bhimavaram Institute of Engineering & Technology, West Godavari, A.P, India
²Assistant Professor, Department of ECE, Bhimavaram Institute of Engineering & Technology, West Godavari, A.P, India
³Assistant Professor, Department of ECE, Bhimavaram Institute of Engineering & Technology (BVRM), West Godavari, A.P, India

ABSTRACT: The proposed security code utilizes decimal method to detect errors, so that more errors were detected and corrected. At the present days to maintain good level of reliability, it is required to protect memory cells using protection codes, for this purpose, various error detection and correction methods are being used. In the paper 64-bit Decimal Matrix Code was proposed to assure the dependability of memory. Here to detect and correct up to 50% errors. The results showed that the proposed scheme has a protection level against large MCUs in memory. To avoid MCUs from causing data corruption, more complex error correction codes (ECCs) are widely used to protect memory, but the main problem is that they would need higher delay overhead. Previously, matrix codes (MCs) based on Hamming codes include been proposed for memory safety. The main problem is that they are double error correction codes and the error correction capabilities are not enhanced in each case. Transient multiple cell upsets (MCUs) are appropriate major problems in the reliability of memories exposed to energy environment. In the new method we are implement 64-bit decimal matrix for error correction in memories. In the new method to increase the error correction rate compared to the 32-bit decimal matrix code. Moreover, the ERT (encoder-reuse technique) is proposed to decrease the area transparency of extra circuits exclusive of disturbing the total encoding and decoding processes. ERT (encoder re use technique) use DMC encoder itself to be part of in the decoder.

KEYWORDS: Error correction codes (ECCs), mean time to failure (MTTF), memory, Hamming code, multiple cells upsets (MCUs), memory.

I. INTRODUCTION

The general idea for achieving error detection and correction is to add some redundancy (i.e., some extra data) to a message, which receiver can use to check consistency of the delivered message, and to pick up data determined to be corrupt. Error-detection and correction scheme can be either systematic or non-systematic: In a systematic scheme, the transmitter sends the unique data, and attaches a fixed number of check bits (or parity data), which are derived from the data bits by some deterministic algorithm. If only the error detection is required, a receiver can simple apply the same algorithm to the received data bits and compare its output with the receive check bits; if the values do not match, an error has occurred at some point throughout the transmission. Error-correcting codes are regularly used in lower-layer communication, as well as for reliable storage in media such as CDs, DVDs, hard disks and RAM.

In a system to uses a non-systematic code, the unique message is transformed into an encoded message that has at least as many bits as the unique message. The aim of error detection and correction code is to provide against soft errors that manifest themselves as bit-flips in memory. Several techniques are used present to midi gate upsets in memories. For
example, the Bose–Chaudhuri–Hocquenghem codes [8], Reed–Solomon codes [9], punctured difference set (PDS) codes [10], and matrix codes has been used to contact with MCUs in memories. But the codes require more area, power, and delay overheads since the encoding and decoding circuits are more complex in these complicated codes. Reed–Muller code [14] is another protection code that is able to detect and correct additional error than a Hamming code. The main drawback of this protection code is its more area and power penalties.

Hamming Codes are more used to correct Single Error Upsets (SEU’s) in memory due to their ability to correct single errors through reduced area and performance overhead [13]. Though brilliant for correction of single errors in a data word, they cannot correct double bit errors caused by single event upset. An extension of the basic SEC-DED Hamming Code has been proposed to form a special class of codes known as Hsiao Codes to increase the speed, cost and reliability of the decoding logic [14].

One more class of SEC-DED codes known as Single-error-correcting, Double-error-detecting Single-byte-error-detecting SEC-DED-SBD codes be proposed to detect any number of errors disturbing a single byte. These codes are additional suitable than the conventional SEC-DED codes for protecting the byte-organized memories [15][16]. Though they operate through lesser overhead and are good for multiple error detection, they cannot correct multiple errors. There are additional codes such as the single-byte-error-correcting, double-byte-error-detecting (SBC-DBD) codes, double-error-correcting, triple-error-detecting (DEC-TED) codes that can correct multiple errors as discussed in [10].

The Single-error-correcting, Double-error-detecting and Double-adjacent-error-correcting (SEC-DED-DAEC) code provides a low cost ECC methodology to correct adjacent errors as proposed in [12]. The only drawback through this code is the possibility of miss-correction for a small subset of many errors.

AS CMOS technology scales down to nano-scale and memories are combined through an increasing number of electronic systems, the soft error rate in memory cells is rapidly increase, especially when memories operate in space environments due to ionizing effects of atmospheric neutron, alpha-particle, and cosmic rays.

Interleaving technique has been used to restrain MCUs, which rearrange cells in the physical arrangement to separate the bits in the same logical word into different physical words. However, interleaving technique may not be practically used in content-addressable memory (CAM), because of the tight coupling of hardware structures from both cells and comparison circuit structures.

Built-in current sensors (BICS) are proposed to assist with single-error correction and double-error detection codes to provide protection against MCUs. However, this technique can only correct two errors in a word.

More recently, in 2-D matrix codes (MCs) are proposed to efficiently correct MCUs per word with a low decoding delay, in which one word is divided into multiple rows and multiple columns in logical. The bits per row are protected by Hamming code, while parity code is added in each column. For the MC based on Hamming, when two errors are detected by Hamming, the vertical syndrome bits are activated so that these two errors can be corrected. As a result, MC is capable of correcting only two errors in all cases. In an approach that combines decimal algorithm with Hamming code has been conceived to be applied at software level. It uses addition of integer values to detect and correct soft errors. The results obtained have shown that this approach have a lower delay overhead over other codes.

In this paper, novel decimal matrix code (DMC) based on divide-symbol is proposed to provide enhanced memory reliability. The proposed DMC utilize decimal algorithm (decimal integer addition and decimal integer subtraction) to identify errors. The advantage of using decimal algorithm is that the error detection capability is maximize so that the reliability of memory is enhanced. Besides, the encoder-reuse technique (ERT) is proposed to minimize the area overhead of extra circuits (encoder and decoder) without disturbing the whole encoding and decoding processes, because ERT use DMC encoder itself to be part of the decoder.
The horizontal redundant bits of memory, the error correction capability is first measured to be 5 bits and the number of redundant bits is 72. In this paper, in order to enhance the reliability, the number of redundant bits is reduced to 32. However, when two error occurrences are simultaneously detected, the error correction capability and the number of redundant bits are different when the different values for k and m are selected. Therefore, k and m must be carefully adjusted to decrease the correction capability and minimize the number of redundant bits. For example, in this case, when \( k = 2 \times 2 \) and \( m = 8 \), only 1-bit error can be corrected and the number of redundant bits is 80. When \( k = 4 \times 4 \) and \( m = 2 \), 3-bit errors can be corrected and the number of redundant bits is reduced to 32. However, when \( k = 2 \times 4 \) and \( m = 4 \), the maximum correction capability is up to 5 bits and the number of redundant bits is 72. In this paper, in order to enhance the reliability of memory, the error correction capability is first measured, so \( k = 2 \times 8 \) and \( m = 4 \) are utilized to construct DMC.

The horizontal redundant bits H can be obtained by decimal integer addition as follow:

- For k = 2 and m = 8:
  \[ H_{0-39} = D_{20} \]

- For k = 4 and m = 2:
  \[ H_{0-39} = D_{20} \]

The proposed schematic of fault-tolerant memory protected with DMC is depicted in Fig. 1.

**II. PROPOSED DMC**

In this section, DMC is proposed to assure reliability in the presence of MCUs through reduced performance overheads, and a 64-bit word is encoded and decoded as an example based on the proposed techniques.

**A. Proposed Schematic of Fault-Tolerant Memory**

The proposed schematic of fault-tolerant memory is depicted in Fig. 1. During the encoding (write) process, information bits D are fed to the DMC encoder, and the horizontal redundant bits H and vertical redundant bits V are obtained from the DMC encoder. When encoding process is completed, the obtained DMC codeword is stored in the memory. If MCUs happen in the memory, these errors can be corrected in the decoding (read) method. Due to the advantage of decimal algorithm, the proposed DMC has high fault-tolerant capability with lower performance overheads. In the fault-tolerant memory, the ERT technique is proposed to decrease the area overhead of extra circuits and will be introduced in the following sections.

**B. Proposed DMC Encoder**

To explain the proposed DMC scheme, we take a 64-bit word as an example, as shown in Fig. 2. The cells from D0 to D63 are information bits. This 64-bit word has been divided into sixteen symbols of 4-bit, \( k = 2 \) and \( k = 4 \) have been selected simultaneously. \( H_0-H_{39} \) are horizontal check bits; \( V_0 \) through \( V_{31} \) are vertical check bits. However, it should be mentioned that the maximum correction capability (i.e., the maximum size of MCUs can be corrected) and the number of redundant bits are different when the different values for k and m are selected. Therefore, k and m must be carefully adjusted to decrease the correction capability and minimize the number of redundant bits. For example, in this case, when \( k = 2 \times 2 \) and \( m = 8 \), only 1-bit error can be corrected and the number of redundant bits is 80. When \( k = 4 \times 4 \) and \( m = 2 \), 3-bit errors can be corrected and the number of redundant bits is reduced to 32. However, when \( k = 2 \times 4 \) and \( m = 4 \), the maximum correction capability is up to 5 bits and the number of redundant bits is 72. In this paper, in order to enhance the reliability of memory, the error correction capability is first measured, so \( k = 2 \times 8 \) and \( m = 4 \) are utilized to construct DMC.

The horizontal redundant bits H can be obtained by decimal integer addition as follow:

1. \[ H_{4H3H2H1H0} = D_{3D2D1D0} + D_{19D18D17D16} \] (1)
2. \[ H_{9H8H7H6H5} = D_{7D6D5D4} + D_{23D22D21D20} \] (2)
and similarly for the horizontal redundant bits $H_{14}H_{13}H_{12}H_{11}H_{10}$, $H_{19}H_{18}H_{17}H_{16}H_{15}H_{14}$, $H_{24}H_{23}H_{22}H_{21}H_{20}$, $H_{29}H_{28}H_{27}H_{26}H_{25}$, $H_{34}H_{33}H_{32}H_{31}H_{30}$ and $H_{39}H_{38}H_{37}H_{36}H_{35}$ where “+” represents decimal integer addition.

For the vertical redundant bits $V$, we have

\[ V_0 = D_0 \oplus D_3 \]  
\[ V_1 = D_1 \oplus D_2 \]  
and similarly for the rest vertical redundant bits.

The encoding can be performed by decimal and binary addition operations from (1) to (4). The encoder that computes the redundant bits using multi-bit adders and XOR gates is shown in Figure. In this figure, $H_{39} - H_0$ are horizontal redundant bits, $V_{31} - V_0$ are vertical redundant bits, and the remain bits $U_{63} - U_0$ are the information bits which are directly copied from $D_{31}$ to $D_0$.

C. Proposed DMC Decoder

To obtain a word being corrected, the decoding process is required. For example, first, the received redundant bits $H_{4}H_{3}H_{2}H_{1}H_{0}'$ and $V_0' - V_3'$ are generated by the received information bits $D'$. Second, the horizontal syndrome bits $\Delta H_{4}H_{3}H_{2}H_{1}H_{0}$ and the vertical syndrome bits $S_3 - S_0$ can be calculated as follows:

\[ \Delta H_{4}H_{3}H_{2}H_{1}H_{0} = H_{4}H_{3}H_{2}H_{1}H_{0}' - H_{4}H_{3}H_{2}H_{1}H_{0} \]  
\[ S_0 = V_0' \oplus V_0 \]  
and alike for the rest vertical syndrome bits, where “−” represents decimal integer subtraction.

When $\Delta H_{4}H_{3}H_{2}H_{1}H_{0}$ and $S_3 - S_0$ are equal to zero, the stored codeword have original information bits in symbol 0 where no errors happen. When $\Delta H_{4}H_{3}H_{2}H_{1}H_{0}$ and $S_3 - S_0$ are nonzero, the induced errors (the quantity of errors is 4 in this case) are detected and located in symbol 0, and then the errors can be corrected by

\[ D_{0\text{correct}} = D_0 \oplus S_0 \]  

Fig 2: 64-bits DMC logical organization ($k = 2 \times 8$ and $m = 4$). Here, each symbol is regarded as a decimal integer

Fig 3: 64-bit DMC encoder structure using multi bit adders and XOR gates
The proposed DMC decoder is depicted in Fig, which is prepared up of the following sub modules, and each executes a particular task in the decoding process: syndrome calculator, error locator, and error corrector. It can be observed from the figure that the redundant bits must be recomputed from the received information bits $D'$ and compare to the original set of redundant bits in order to obtain the syndrome bits $\Delta H$ and $S$. Then error locator uses $\Delta H$ and $S$ to detect and locate which bits some errors occur in. Finally, in the error corrector, these errors can be corrected by inverting the values of error bits.
In the proposed scheme, the circuit area of DMC is minimized with reusing its encoder. This is calling the ERT. The ERT can decrease the area overhead of DMC without disturbing the entire encoding and decoding processes. From Fig. it can be practical that the DMC encoder is also reused for obtaining the syndrome bits in DMC decoder. Therefore, the entire circuit area of DMC can be minimized as a result of using the existent circuits of encoder. Besides, the figure shows the proposed decoder with an allow signal $\text{En}$ for deciding whether the encoder needs to be a part of the decoder. In other words, the $\text{En}$ signal is used for distinguishing the encoder from the decoder, and it is under manage of the write and read signals in memory. Therefore, in the encoding (write) mode, the DMC encoder is only an encoder to execute the encoding operations. However, in the decoding (read) mode, this encoder is employed for computing the syndrome bits in the decoder. These obviously show how the area overhead of extra circuits can be substantially decreased.

III. SIMULATION RESULTS

Block diagram
RTL Schematic diagram

Technology schematic
Design summary

Simulation output waveform
IV. CONCLUSION

In the proposed method we implemented the 64-bit decimal matrix code for detection and correction of errors in memories. To avoid MCUs from causing data corruption, more complex error correction codes (ECCs) are widely used to protect memory, but the main problem is that they would require higher delay overhead. newly, matrix codes (MCs) based on hamming codes have been proposed for memory protection. In proposed system increased error detection and correction rate compared to 32-bit decimal matrix code.

REFERENCES


