Implementation of Decimal Matrix Code For Multiple Cell Upsets in Memory

Shwetha N¹, Shambhavi S²

¹, ² Department of E&C, Kalpataru Institute of Technology, Tiptur, Karnataka, India

Abstract: Transient multiple cell upsets (MCUs) are becoming major issues in the reliability of memories exposed to radiation environment. To prevent MCUs from causing data corruption, more complex error correction codes (ECCs) are widely used to protect memory, but the main problem is that they require higher delay overhead. Recently, matrix codes (MCs) have been proposed for memory protection. The main issue is that they are double error correction codes and the error correction capabilities are not improved in all cases. In this paper, novel decimal matrix code (DMC) based on divide-symbol is proposed to enhance memory reliability with lower delay overhead. The proposed DMC utilizes decimal algorithm to obtain the maximum error detection capability. Moreover, the encoder-reuse technique (ERT) is proposed to minimize the area overhead of extra circuits without disturbing the whole encoding and decoding processes. ERT uses DMC encoder itself to be part of the decoder. The proposed DMC is compared to well-known codes such as the existing Hamming, MCs, and punctured difference set (PDS) codes.

Keywords: Decimal algorithm, error correction codes (ECCs), mean time to failure (MTTF), memory, multiple cells upsets (MCUs).

1. Introduction

As CMOS technology scales down to nano scale and memories are combined with an increasing number of electronic systems, the soft error rate in memory cells is rapidly increasing, especially when memories operate in space environments due to ionizing effects of atmospheric neutron, alpha-particle, and cosmic rays[1].

Although single bit upset is a major concern about memory reliability, multiple cell upsets (MCUs) have become a serious reliability concern in some memory applications[2]. In order to make memory cells as fault-tolerant as possible, some error correction codes (ECCs) have been widely used to protect memories against soft errors for years [3]-[6]. For example, the Bose-Chaudhuri-Hocquenghem codes [7], Reed-Solomon codes[8] and punctured difference set (PDS) codes [9] have been used to deal with MCUs in memories. But these codes require more area, power, and delay overheads since the encoding and decoding circuits are more complex in these complicated codes. Interleaving technique has been used to restrain MCUS [10], which rearrage cells in the 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 structure [11], [12].

Built-in current sensors (BICS) are proposed to assist with error correction and double-error correction codes to provide protection against MCUs [13], [14]. However, this technique can only correct two errors in a word. More recently, in[15], 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 multipa columns in logical. The bits per row are protected by Hamming code, while parity code is added in each column. For the MC [15] 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. 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 utilizes decimal algorithm (decimal integer addition and decimal integer subtraction) to detect errors. The advantage of using decimal algorithm is that the error detection capability is maximized 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 process, because ERT uses DMC encoder itself to be part of the decoder. This paper is divide into the following sections. The proposed DMC is intruduced and its encoder and decoder circuits are present in Section II. This section also illustrates the limits of simple binary error detection and the advantage of decimal error detection with simple examples. The reliability and overheads analysis of the proposed code are analized in Section III.

![Figure 1: Proposed schematic of fault-tolerant memory protected with DMC.](image_url)

Finals, some conclusions of this paper are discussed and shared in section IV.

2. Proposed DMC

In this section, DMC is proposed to assure reliability in the presence overheads, and a 32-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. First, during the encoding (write) process information bits D are fed to the DMC encoder, and then the horizontal redundant bits H and vertical redundant bits V are obtained from the DMC encoder. When the encoding process is completed. The obtained DMC codeword is stored in the memory. If MCUs occur in the memory, these errors can be corrected in the decoding (read) process. Due to the advantage of decimal algorithm, the proposed DMC has higher fault-tolerant capability with lower performance overheads. In the proposed to reduce the area overhead of extra circuits and will be introduced in the following sections.

B. Proposed DMC Encoder

In the proposed DMC, first, the divide-symbol and arrange-matrix ideas are performed, i.e., the N-bit word is divided into k symbols of m bits (N=kxm), and these symbols are arranged in a k1xk2 2-D matrix (k=k1xk2, where the values of k1 and k2 represent the numbers of rows and columns in the logical matrix respectively). Second the horizontal 8, only 1-bit error can be corrected and the redundant bits H are produced by performing decimal integer. Third, the vertical redundant bits V are obtained by binary operation among the bits per column. It should be noted that both divide-symbol and arrange-matrix are implemented in logical instead of in physical. Therefore, the proposed DMC does not require changing the physical structure of the memory.

To explain the proposed DMC scheme, we take a 32-bit word as an example, as shown in Fig.2. The cells from D0 to D31 are information bits. This 32-bit word has been divided into eight symbols of 4-bit. K1=2 and k2=4 have been chosen simultaneously. H0-H19 are horizontal check bits; V0 through V15 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 chosen. Therefore, k and m should be carefully adjusted to maximize the correction capability and minimize the number of redundant bits. For example, in this case, when k=2x2 and m=4, only 1-bit error can be corrected and the number of redundant bits is 40. When k=4x4 and m=2, 3-bit errors can be corrected and the number of redundant bits is reduced to 32. However, when k=2x4, and m=4, the maximum correction capability is up to 5 bits and the number of redundant bits is 36. In this paper, in order to enhance the reliability of memory, the error correction capability is first considered, so k=2x4, and m=4, are utilized to construct DMC.

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

\[ H_0 + H_1 + H_2 + H_3 + \ldots + H_8 = D_0 + D_1 + D_2 + D_3 + \ldots + D_8 \] (1)

And similarly for the horizontal redundant bits

\[ H_{10} + H_{11} + H_{12} + H_{13} + H_{14} + H_{15} = D_{10} + D_{11} + D_{12} + D_{13} + D_{14} + D_{15} \] (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 multibit adders and XOR gates is shown in fig.3. In this figure, H0-H8 are horizontal redundant bits, V15-V0 are vertical redundant bits, and the remaining bits U11-U0 are the information bits which are directly copied from D31-D0. The enable signal En will be explained in the next section.

C. Proposed DMC decoder

To obtain a word being corrected, the decoding process is required. For example, first the received redundant bits H11-H12-H13-H14-V0-V15 are generated by the received information bits. The horizontal syndrome bits \(\Delta H_0, \Delta H_1, \Delta H_2, \Delta H_3, \ldots, \Delta H_8\) and the vertical syndrome bits S0-S9 can be calculated as follows:

\[ \Delta H_0 = H_0 - H_1 - H_2 - H_3 - \ldots - H_8 \] (5)

And similarly for the rest vertical syndrome bits, where “-” represents decimal integer subtraction. When \(\Delta H_0, \Delta H_1, \Delta H_2, \Delta H_3, \ldots, \Delta H_8\) and S0-S9 are equal to zero, the stored code word has original information bits in symbol 0 where no errors correct. When \(\Delta H_0, \Delta H_1, \Delta H_2, \Delta H_3, \ldots, \Delta H_8\) and S0-S9 are non zero, the induced errors (the number of errors is 4 in

![Figure 2: 32-bits DMC logical organization (k=2x4 and m=4), here, each symbol is regarded as a decimal integer](image-url)
This case) are detected and located in symbol 0, and then these errors can be corrected by

$$D_{\text{correct}} = D_0 \oplus S_0 \quad (7)$$

The proposed DMC decoder is depicted in Fig. 4, which is made up of the following submodules, and each executes a specific task in the decoding process: syndrome calculator, error locator, and error corrector. It can be observed from this figure that the redundant bits must be recomputed from the received information bits $D$ and compared 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. 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 by reusing its encoder. This is called the ERT. The ERT can reduce the area overhead of DMC without disturbing the whole encoding and decoding processes. From Fig. 4, it can be observed that the DMC encoder is also reused for obtaining the syndrome bits in DMC decoder. Therefore, the whole circuit area of DMC can be minimized as a result of using the existent circuits of encoder. Besides, this figure also shows the proposed decoder with an enable signal $E_n$ for deciding whether the encoder needs to be a part of the decoder. In other words, the $E_n$ signal is used for distinguishing the encoder from the decoder, and it is under the control of the write and read signals in memory. Therefore, in the encoding (write) process, the DMC encoder is only an encoder to execute the encoding operations. However, in the decoding (read) process, this encoder is employed for computing the syndrome bits in the decoder. These clearly show how the area overhead of extra circuits can be substantially reduced.

D. Limits Of Binary Error Detection

For the proposed binary error detection technique in [13], although it requires low redundant bits, its error detection capability is limited. The main reason for this is that its error detection mechanism is based on binary. We illustrate the limits of this simple binary error detection [13] using a simple example. Let us suppose that the bits $B_3$, $B_2$, $B_1$, and $B_0$ are original information bits and the bits $C_0$ and $C_1$ are redundant bits shown in Fig. 5. The bits $C_0$ and $C_1$ are obtained using the binary algorithm

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. 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 by reusing its encoder. This is called the ERT. The ERT can reduce the area overhead of DMC without disturbing the whole encoding and decoding processes. From Fig. 4, it can be observed that the DMC encoder is also reused for obtaining the syndrome bits in DMC decoder. Therefore, the whole circuit area of DMC can be minimized as a result of using the existent circuits of encoder. Besides, this figure also shows the proposed decoder with an enable signal $E_n$ for deciding whether the encoder needs to be a part of the decoder. In other words, the $E_n$ signal is used for distinguishing the encoder from the decoder, and it is under the control of the write and read signals in memory. Therefore, in the encoding (write) process, the DMC encoder is only an encoder to execute the encoding operations. However, in the decoding (read) process, this encoder is employed for computing the syndrome bits in the decoder. These clearly show how the area overhead of extra circuits can be substantially reduced.

D. Limits Of Binary Error Detection

For the proposed binary error detection technique in [13], although it requires low redundant bits, its error detection capability is limited. The main reason for this is that its error detection mechanism is based on binary. We illustrate the limits of this simple binary error detection [13] using a simple example. Let us suppose that the bits $B_3$, $B_2$, $B_1$, and $B_0$ are original information bits and the bits $C_0$ and $C_1$ are redundant bits shown in Fig. 5. The bits $C_0$ and $C_1$ are obtained using the binary algorithm

$$D_{\text{correct}} = D_0 \oplus S_0 \quad (7)$$

The proposed DMC decoder is depicted in Fig. 4, which is made up of the following submodules, and each executes a specific task in the decoding process: syndrome calculator, error locator, and error corrector. It can be observed from this figure that the redundant bits must be recomputed from the received information bits $D$ and compared 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. 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 by reusing its encoder. This is called the ERT. The ERT can reduce the area overhead of DMC without disturbing the whole encoding and decoding processes. From Fig. 4, it can be observed that the DMC encoder is also reused for obtaining the syndrome bits in DMC decoder. Therefore, the whole circuit area of DMC can be minimized as a result of using the existent circuits of encoder. Besides, this figure also shows the proposed decoder with an enable signal $E_n$ for deciding whether the encoder needs to be a part of the decoder. In other words, the $E_n$ signal is used for distinguishing the encoder from the decoder, and it is under the control of the write and read signals in memory. Therefore, in the encoding (write) process, the DMC encoder is only an encoder to execute the encoding operations. However, in the decoding (read) process, this encoder is employed for computing the syndrome bits in the decoder. These clearly show how the area overhead of extra circuits can be substantially reduced.

D. Limits Of Binary Error Detection

For the proposed binary error detection technique in [13], although it requires low redundant bits, its error detection capability is limited. The main reason for this is that its error detection mechanism is based on binary. We illustrate the limits of this simple binary error detection [13] using a simple example. Let us suppose that the bits $B_3$, $B_2$, $B_1$, and $B_0$ are original information bits and the bits $C_0$ and $C_1$ are redundant bits shown in Fig. 5. The bits $C_0$ and $C_1$ are obtained using the binary algorithm

In the proposed scheme, the circuit area of DMC is minimized by reusing its encoder. This is called the ERT. The ERT can reduce the area overhead of DMC without disturbing the whole encoding and decoding processes. From Fig. 4, it can be observed that the DMC encoder is also reused for obtaining the syndrome bits in DMC decoder. Therefore, the whole circuit area of DMC can be minimized as a result of using the existent circuits of encoder. Besides, this figure also shows the proposed decoder with an enable signal $E_n$ for deciding whether the encoder needs to be a part of the decoder. In other words, the $E_n$ signal is used for distinguishing the encoder from the decoder, and it is under the control of the write and read signals in memory. Therefore, in the encoding (write) process, the DMC encoder is only an encoder to execute the encoding operations. However, in the decoding (read) process, this encoder is employed for computing the syndrome bits in the decoder. These clearly show how the area overhead of extra circuits can be substantially reduced.

D. Limits Of Binary Error Detection

For the proposed binary error detection technique in [13], although it requires low redundant bits, its error detection capability is limited. The main reason for this is that its error detection mechanism is based on binary. We illustrate the limits of this simple binary error detection [13] using a simple example. Let us suppose that the bits $B_3$, $B_2$, $B_1$, and $B_0$ are original information bits and the bits $C_0$ and $C_1$ are redundant bits shown in Fig. 5. The bits $C_0$ and $C_1$ are obtained using the binary algorithm

In the proposed scheme, the circuit area of DMC is minimized by reusing its encoder. This is called the ERT. The ERT can reduce the area overhead of DMC without disturbing the whole encoding and decoding processes. From Fig. 4, it can be observed that the DMC encoder is also reused for obtaining the syndrome bits in DMC decoder. Therefore, the whole circuit area of DMC can be minimized as a result of using the existent circuits of encoder. Besides, this figure also shows the proposed decoder with an enable signal $E_n$ for deciding whether the encoder needs to be a part of the decoder. In other words, the $E_n$ signal is used for distinguishing the encoder from the decoder, and it is under the control of the write and read signals in memory. Therefore, in the encoding (write) process, the DMC encoder is only an encoder to execute the encoding operations. However, in the decoding (read) process, this encoder is employed for computing the syndrome bits in the decoder. These clearly show how the area overhead of extra circuits can be substantially reduced.

D. Limits Of Binary Error Detection

For the proposed binary error detection technique in [13], although it requires low redundant bits, its error detection capability is limited. The main reason for this is that its error detection mechanism is based on binary. We illustrate the limits of this simple binary error detection [13] using a simple example. Let us suppose that the bits $B_3$, $B_2$, $B_1$, and $B_0$ are original information bits and the bits $C_0$ and $C_1$ are redundant bits shown in Fig. 5. The bits $C_0$ and $C_1$ are obtained using the binary algorithm

In the proposed scheme, the circuit area of DMC is minimized by reusing its encoder. This is called the ERT. The ERT can reduce the area overhead of DMC without disturbing the whole encoding and decoding processes. From Fig. 4, it can be observed that the DMC encoder is also reused for obtaining the syndrome bits in DMC decoder. Therefore, the whole circuit area of DMC can be minimized as a result of using the existent circuits of encoder. Besides, this figure also shows the proposed decoder with an enable signal $E_n$ for deciding whether the encoder needs to be a part of the decoder. In other words, the $E_n$ signal is used for distinguishing the encoder from the decoder, and it is under the control of the write and read signals in memory. Therefore, in the encoding (write) process, the DMC encoder is only an encoder to execute the encoding operations. However, in the decoding (read) process, this encoder is employed for computing the syndrome bits in the decoder. These clearly show how the area overhead of extra circuits can be substantially reduced.
inconsecutive symbols, and type 5 is a consecutive error in four consecutive symbols.

\[ C_0 = B_0 \oplus B_1 = 1 \oplus 0 = 1 \]
\[ C_1 = B_1 \oplus B_2 = 0 \oplus 1 = 1. \]

Then assume now that MCUs occur in bits B_1, B_2, and B_0 (i.e., B_3=0, B_2=1, and B_0=0). The received redundant bits C_0 and C_1 are computed

\[ C_0 = B_0 \oplus B_2 = 0 \oplus 1 = 1 \]
\[ C_1 = B_1 \oplus B_3 = 0 \oplus 0 = 0. \]

In order to detect these errors, the syndrome bits S_0 and S_1 are obtained

\[ S_0 = C_0 \oplus C_1 = 1 \oplus 1 = 0 \]
\[ S_1 = C_1 \oplus C_0 = 0 \oplus 1 = 1. \]

These results mean that error bits B_2 and B_0 are wrongly regarded as the original bits so that these two error bits are not corrected. This example illustrates that for this simple binary operation [13], the number of even bit errors cannot be detected.

E. Advantage of Decimal Error Detection

In the previous discussion, it has been shown that error detection [13] based on binary algorithm can only detect a finite number of error. However, when the decimal algorithm is used to detect error, these errors can be detected so that the decoding error can be avoided. The reason is that the operation mechanism of decimal algorithm is different from that of binary. The detection procedure of decimal error detection using the proposed structure shown in Fig. 2 is fully described in Fig. 6. First of all, the horizontal redundant bits H_0, H_1, H_2, H_3, H_4, H_5, H_6, H_7, and H_8 are obtained from the original information bits in symbol 0 and 2 according to (1)

\[ H_0 = D_2 D_3 D_4 + D_1 D_0 D_9 D_8 = 1100 + 0110 = 10010. \]

When MCUs occur in symbol 0 and symbol 2, i.e., the bits in symbol 0 are upset to “1111” from “1100” (D_2D_3D_4D_5 = 1111) and the bits in symbol 2 are upset to “0111” from “0110” (D_1D_0D_9D_8 = 0111). During the decoding process, the received horizontal redundant bits H_0, H_1, H_2, H_3, H_4, H_5, H_6, H_7, and H_8 are first computed. As follows:

\[ H_0 = D_2 D_3 D_4 + D_1 D_0 D_9 D_8 = 0111 + 1111 = 10011 \]

Then, the horizontal syndrome bits \( \Delta H_0 \) of H_0 can be obtained using decimal integer subtraction

\[ \Delta H_0 = H_0 - H_0 = 10110 - 10010 = 00100. \]

The decimal value of \( \Delta H_0 \) is not “0” which represents that errors are detected and located in symbol 0 or symbol 2. Subsequently, the precise location of the bits that were flipped can be located by using the vertical syndrome bits S_1 and S_2. Finally, all these errors can be corrected by (7). Therefore, based on decimal algorithm, the proposed technique has higher tolerance capability for protecting memory against MCUs.

As a result, it is possible that all single and double errors and any types of multiple errors per row can be corrected by the proposed technique no matter whether these errors are consecutive or inconsecutive in Fig. 7. The proposed DMC

\[ \text{Figure 8: Error type cannot be corrected by our proposed DMC. The main reason is that } H_0, H_1, H_2, H_3, H_4, \text{ will be “}0\text{” (decimal). Note that even though 7-bit errors occur in symbols 0 and 2 simultaneously, the decoding error can be refused.} \]

Can easily correct upsets of type 1, 2, and 3, because these are the essential property of DMC: any types of single error and multiple error corrections in two consecutive symbols. Upsets of types 4 and 5 introduced in Fig. 7 are also corrected because the multiple errors per row can be detected by the horizontal syndrome bits (see Fig. 6). These show that the proposed technique is an attractive option to protect memories from large MCUs. However, for the upsets of type 4 and 5, it is important to recognize that it can result in decoding error when the following prerequisite factors are achieved simultaneously (this error is typical of its kind).

1. The decimal integer sum of information bits in symbols 0 and 2 is equal to \( 2^n-1 \).
2. All the bits in symbol 0 and 2 are upset.

The more detailed explanation is shown in Fig. 8. Assuming that these two factors have been achieved, according to the encoding and decoding processes of DMC, H_0, H_1, H_2, H_3, H_4, H_5, H_6, H_7, and H_8 are computed, as follows:

\[ H_0 = D_2 D_3 D_4 + D_1 D_0 D_9 D_8 = 0110 + 0101 = 1011 \]
\[ H_1 = D_2 D_3 D_4 + D_1 D_0 D_9 D_8 = 1001 + 0110 = 0111 \]
\[ \text{Then the horizontal syndrome bits } \Delta H_0, \Delta H_1, \Delta H_2, \Delta H_3, \Delta H_4, \Delta H_5, \Delta H_6, \Delta H_7, \text{ and } \Delta H_8 \text{ can be obtained, as follows:} \]

\[ \Delta H_0 = H_0 - H_0 = 10110 - 10010 = 00100. \]
\[ \Delta H_4 \Delta H_3 \Delta H_2 \Delta H_1 = H_4 \Delta H_3 \Delta H_2 H_1 \]
\[ = 01111-01111 = 00000 \]  
(19)

This result means that no errors occur in symbols 0 and 2 and memory will suffer a failure. However, this case is rare. For example, when \( m = 4 \), the probability of decoding errors is
\[ P \Delta H = 0 = 4 \times (1/2)^4 \times P_{MCU8} = 0.00001. \]  
(20)

If \( m = 8 \),
\[ P \Delta H = 0 = 4 \times (1/2)^4 \times P_{MCU8} \approx 0.000001, \]
(21)

Figure 9: \( J(S)s \) versus time of different protection codes (\( M = 32 \))

\( P_{MCU8} \) represents the probability of eight upsets in a given word, and similarly for \( P_{MCU16} \). Moreover, according to the radiation experiments in [1], [2], it can be obtained that the word in a memory usually has a limited number of consecutive errors and the interval of these errors is not more than three bits. Therefore, this should not be an issue.

3. Reliability and Overheads Analysis

In this section, the proposed DMC has been implemented in HDL, simulated with modelsim and tested for functionality by given various inputs. The encoder and decoder have been synthesized by the synopsys design compiler in the SMIC 0.18\( \mu \)m technology. The area, power, and critical path delay of extra circuits have been obtained. For fair comparisons, Hamming, PDS [9], and MC [15] are used for references. Here, the usage of (64, 45) PDS is a triple-error correction code [9] and its information bits is shorted to 32 bits from 45 bits.

A. Fault Injection

The correction coverage of PDS [9], MC [15], Hamming, and the proposed DMC codes are obtained from one million experiments. The results of coverage are shown in Table I. It can be seen that proposed DMC have superior protection level compared with other codes. These results show how our proposed technique provides single- and double-error correction, but can also provide effective tolerance capabilities against large MCUs that exceed the performance of other codes.

<table>
<thead>
<tr>
<th>ECC Codes</th>
<th>The Number of Errors in a Word</th>
</tr>
</thead>
<tbody>
<tr>
<td>DMC (%)</td>
<td>100 100 100 100 92.6 84.7 76.0 66.7 60.9 54.5 47.7 40.0 31.6 22.3 11.8</td>
</tr>
<tr>
<td>PDS [9] (%)</td>
<td>100 100 100 0.8 0 0 0 0 0 0 0 0 0 0 0</td>
</tr>
<tr>
<td>MC [15] (%)</td>
<td>100 100 76.4 54.3 35.1 14.2 6.7 0.6 0 0 0 0 0 0 0</td>
</tr>
<tr>
<td>Hamming (%)</td>
<td>100 0 0 0 0 0 0 0 0 0 0 0 0 0 0</td>
</tr>
</tbody>
</table>

B. Reliability Estimation

The reliability of our proposed code can be analyzed in terms of the mean time to failure (MTTF). It is assumed that MCUs arrive at memories following a poisson distribution [19]. For one word, the correctable probability \( R(S) \) after \( S \) radiation events can be given by [14], [15]
\[ R(S) = \sum_{i+j+\ldots+z \leq T} P_i \cdot P_j \cdot \ldots \cdot P_z^S \]  
(22)

Where \( T \) is the maximum number of errors and \( P_z^S \) is the correctable probability upon the reception of radiation event \( z \) which causes \( z \) errors. For a memory with \( M \) words, the correctable probability \( R(S) \) after \( S \) radiation events can be given by
\[ J(S) = \sum_{a+h+\ldots+c \leq S} C_M^a R_a^1 R_h^2 \ldots R_c^S \]  
(23)

Where \( x(x \leq S) \) is the number of words affected by radiation events. \( C_M^a \) is the selection of \( x \) from \( M \) words in memory. And \( R_x^S \) represents the correctable probability when \( S \) radiation events affect \( x \) words.

If we assume that the word number \( M \) is 32 and the correctable probability \( P_z^S \) can be obtained from Table I, the correctable probabilities \( J(S) \) of different protection codes have been shown in Fig. 9. It can be seen that the correctable probability \( J(S) \) of the proposed scheme is larger than other codes.

Then the MTTF can be given by (24), which is the integral of function (23)
\[ \text{MTTF} = \int_0^\infty J(t) dt. \]  
(24)

Table II shows MTTFs of different event arrival rate \( \gamma \). In this table, we can see that the proposed scheme has higher MTTF by more than 122.6% 154.6%, and 452.9% compared to PDS [9], MC [15], and Hamming, respectively.

In general cases, for the proposed technique, it can be inferred that the larger the word widths, the higher the tolerance capability and the better the reliability. For example, for a 64-bit word, when \( k=2\times4 \) and \( m=8 \) the correction capability of the proposed technique is up to 9 bits.
C. Overhead Analysis

For each protection code, area, power, and delay overheads of encoder and decoder have been shown in Table III. From Table III, we can observe that the proposed MC has a significant reduction compared with other codes. The area and power overheads of PDS are 1170.9%, 2047.2% of the proposed scheme, respectively. The delay overhead of DMC is 26.2%, 69.0%, and 73.1% of PDS [9], MC [15], and Hamming respectively. This indicates that the memory with the proposed scheme performs faster than other codes. Different decoding algorithms could result in different overheads. The decoding algorithms PDS [9] is more complex than that of other codes; thus, it has maximum area, power, and delay overheads. However, for the proposed DMC, its decoding algorithm is quite simple so that the overheads are minimal.

The issue is that the proposed technique requires more redundant bits compared with other codes. The redundant bits of these protection codes are shown in Table IV. Where a coding efficiency $\beta$ is used to evaluate the area overhead of memory cell

$$\beta = \frac{\text{Redundant bits}}{\text{Redundant bits} + \text{Information bits}}$$

(25)

If the value of $\beta$ is small, the code needs lower memory cell overheads. From this table, we can see that hamming code has the least $\beta$ value but its correction capability is a constant (1). For the MC [15], its correction capability is also a constant (2) due to the limits of error correction capability of Hamming code. PDS [9] has a lower $\beta$ value compared with the proposed scheme, but it requires higher delay overhead which would severely affect the access time of memory. The scaling down of CMOS technology has resulted in a dramatic increase in the number of MCUs. In 90-nm technology, more than three errors have been observed in radiation test [1], [2], so Hamming, MC, and PDS are not good choices for protecting memory. The proposed scheme needs higher $\beta$ value than other codes but it has higher correction capability. Therefore, designers should choose the optimal combination of k and m based on the radiation test to provide a good tradeoff between reliability and redundant bits. We have also used a metric to assess the efficiency of our proposed scheme compared to PDS, MC, and Hamming, which is called correction coverage per cost (CCC) and can be calculated as follow [15]:

$$\text{CCC} = \frac{\text{Correction Coverage}}{\text{Cost}}$$

(26)

where Cost is obtained by

$$\text{Cost} = \text{Area} \cdot \text{Power} \cdot \text{Delay} \cdot \text{Redundant bits}.$$  

(27)

Fig. 10 shows the values of this metric for different protection codes. From this figure, we can see that the proposed scheme has a higher CCC value than other codes except that only one MCU occurs. Therefore, based on the results in Table III and Fig. 10, the proposed scheme is quite suitable for high-speed memory applications. It should be mentioned that when the number of errors is more than two per word, Hamming and MC codes cannot correct any errors.

4. Conclusion

In this paper novel per-word DMC was proposed to assure the reliability of memory. The proposed protection code utilized decimal algorithm to detect errors, so that more errors were detected and corrected. The obtained results showed that the proposed scheme has a superior protection level against large MCUs in memory. Besides, the proposed decimal error detection technique is an attractive opinion to detect MCUs in CAM because it can be combined with BICS to provide an adequate level of immunity.

The only drawback of the proposed DMC is that more redundant bits are required to maintain higher reliability of
memory. So that a reasonable combination of k and m should be chosen to maximize memory reliability and minimize the number of redundant bits based on radiation experiments in actual implementation. Therefore, future work will be conducted for the reduction of the redundant bits and the maintenance of the reliability of the proposed technique.

References


