1-1-1979

Stuck error correcting codes for b-bit-per-card memory systems.

Takeo Kanai

STUCK ERROR CORRECTING CODES FOR B-BIT-PER-CARD MEMORY SYSTEMS

by

Takeo Kanai

A Thesis

Presented to the Graduate Committee

of Lehigh University

in Candidacy for the Degree of

Master of Science

in

Electrical Engineering

Lehigh University

1979

CERTIFICATE OF APPROVAL

This thesis is accepted and approved in partial fulfillment of the requirements for the degree of Master of Science in Electrical Engineering.

December 13, 1979
(date)

Professor in Charge

Chairman of Department

ACKNOWLEDGEMENTS

I wish to thank Professor Kenneth K. Tzeng for his helpful advice. I also wish to thank the National Science Foundation for support under Grant ENG-76-10758, and Lehigh University for support in the form of a scholarship and a fellowship.

Table of Contents

		Page
Abstract		1
1	Introduction	2
2.1	Random Block Error Correction Codes in Main Memory System	5
2.2	Channel Model For B-Bit-Per-Card Memory Systems	7
2.3	Subcodes of Fire Codes For Compound Channel	8
3.1	Reliability Improvement in Defective Main Memory Systems	14
3.2	Basic Idea of Error Correcting Code Is For Defective Memory Systems	17
3.3	Stuck Block Error Correcting Codes For Defective B-Bit-Per-Card Memory Systems	20
4.1	Modified Stuck Block Error Correcting Codes C_M	31
4.2	Encoding of C_M	36
4.3	Memory Check Routine	40
4.4	Decoding of C_M	40
4.5	Example of C_M	44
4.6	Example of Encoding and Decoding in Fault Free State	46
5	Conclusion	52
References		53
Vita		55

Abstract

Memory systems constructed from high-density memory devices have a tendency to yield stuck errors in addition to random errors. When the memory system employs b-bit-per-card configuration, these stuck errors become block errors. In this thesis, error correcting codes which correct both single stuck block errors and random bit errors are discussed. These codes are subcodes of additive codes and can correct both single stuck block errors and t or less random bit errors which may occur simultaneously. These codes have higher information rate than that of random error correcting codes.

Encoding and decoding procedures of these codes are also discussed with examples.

1. Introduction

One application of error correcting codes is the correction or detection of random errors in main memory systems for improving reliability. Another application is to enhance the yield of large scale integrated (LSI) memory devices.[1]-[5] In this application, error correcting codes are used for the correction of errors due to hardware defects in memory devices. Since it is inevitable that memory systems yield random errors, a class of additive code[4][5] has been proposed for the correction of random errors as well as errors due to hardware defects.

If a memory system is packaged in a b-bit-per-card basis where the number of bits in a memory word is equal to pb as shown in Fig. 1, hardware defects in one memory array card are likely to cause a block of b bits in error. In this thesis we assume that a memory system employs the b-bit-per-card configuration in which random error may occur bit by bit and errors due to hardware may occur in a single block of information readout. Furthermore, we assume that these single block errors are stuck-at errors.[1]-[5] One code shown here can correct both t or less random bit errors and a single block error due to hardware defects which may occur simultaneously. The other code shown here can correct t or less random bit errors and a single random block error which do not occur simultaneously, if there is no hardware

defects, and also can correct t or less random bit errors and single block errors due to hardware defects which may occur simultaneously. These two codes are referred to as C_S and C_M, respectively.

In the following section, we assume that memory systems are not defective and we discuss random block error correcting codes and a model of channel for memory systems with b-bit-per-card configuration. Also a class of codes suitable for this model is shown. In section 3, methods for improving reliability of defective memory systems, are discussed and the class of C_S is shown. In section 4, we discuss a case in which the memory system becomes defective when it is used and the class of C_M which can cope with this model is shown. Finally, encoding and decoding procedures for C_M are discussed with examples in section 5.

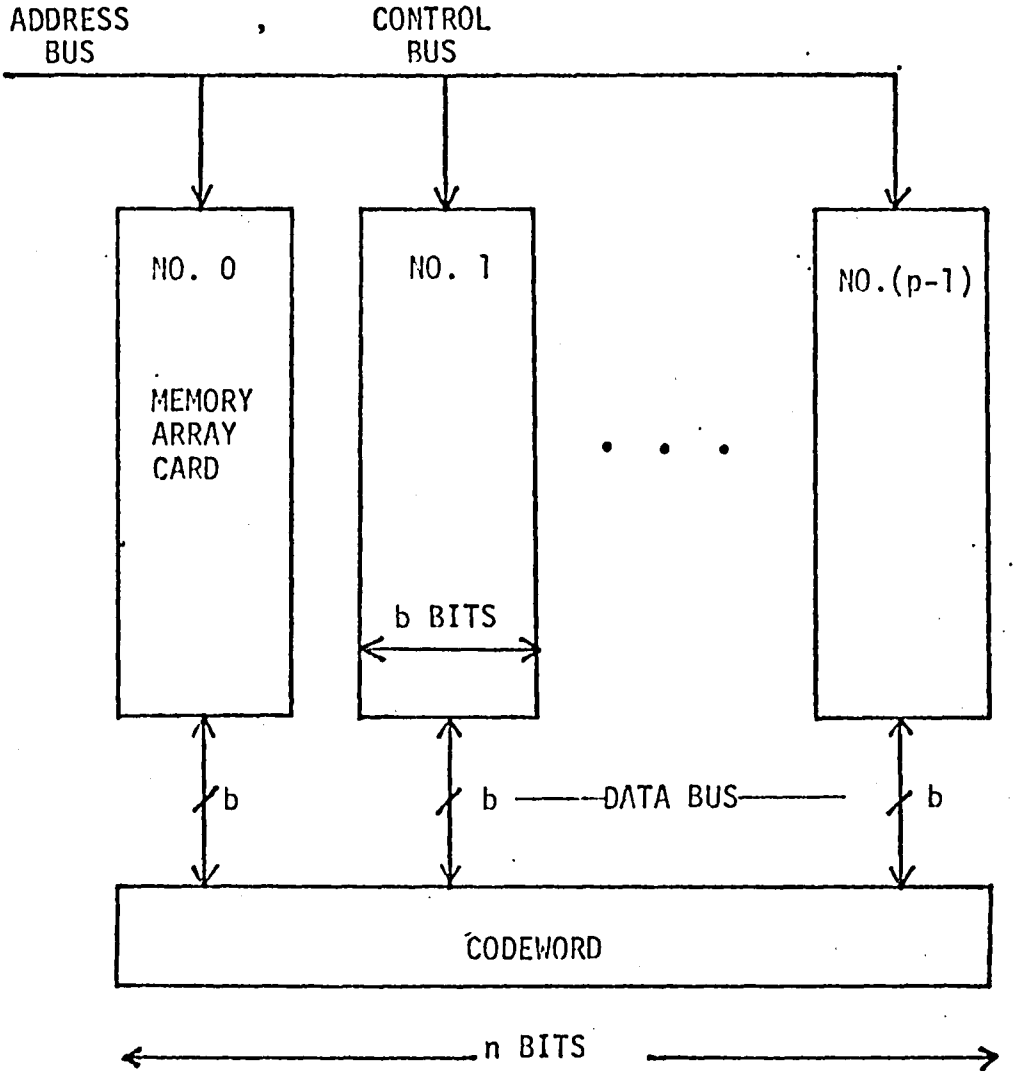

Fig. 1. B-bit-per-card memory configuration.

2.1 Random Block Error Correcting Codes in Main Memory Systems

The technology of large scale integration has been greatly developed in recent years. Semiconductor high-density memory devices have been manufactured and used in recent computer main memory systems. Since it is inevitable that such digital systems yield random errors due to random noises in the system, error correcting codes have been more popularly used in the main memory systems for the purpose of reliability improvement. At the present time, a single bit error correcting and double bit error detecting binary Hamming code[6][7] is actually used in the main memory systems of some commercially available general purpose computer systems[8] under the assumption that not more than double bit errors occur in a word in the main memory. This assumption is somewhat reasonable since integration of the memory devices is not extremely high so far and all the memory devices can be expected to be perfectly functional, namely there is no hardware defect in memory devices.

As observed in the market, high-density semiconductor memory devices have a tendency to expand in the bit-wise direction to increase the flexibility of memory allocation for users. For instance, sixteen 16K x 1 bit memory devices are required for a 16 bit memory system, but the same word length can be achieved with only two 2K x 8 bit memory devices. Suppose that such devices are used in a main memory system. If random noises

interfere control lines or address lines of some memory device, the resultant information readout from memory is likely to have a block of b bits in error, where b is the bit-wise length of the memory device. In such a memory system, a b-bit-per-card memory system configuration will show a good performance. For the b-bit-per-card memory system configuration, random block error correcting codes are proposed assuming that all errors occur block by block instead of bit by bit[9][10]. This class of codes is generally referred to as b-adjacent-bit error correcting codes[9], and is a subclass of Reed-Solomon codes or BCH codes over $GF(2^b)$[6][7]. In memory systems which employ the b-adjacent error correcting code, a word is divided into blocks of b bits and every block is assumed to be a symbol in $GF(2^b)$. Let b, n and m be block length, codeword length and check bit length, respectively. These parameters must satisfy the conditions,

$$n \leq (2^b-1)b$$

and

$$m \geq 2tb$$

where t is the number of correctable symbol errors in $GF(2^b)$. In the case of minimum distance d = 3, n can be extended to

$$n = (2^b+1)b$$[11]

If b is small, n may not satisfy the necessary length. For instance, there is no Reed-Solomon code and BCH code over $GF(2^2)$ applicable to a 16 bit memory system with b = 2. For a small b, modified Reed-Solomon codes have been proposed to achieve a larger codeword length[12][13].

2.2 Channel Model For B-Bit-Per-Card Memory Systems

The class of b-adjacent error correcting codes is very effective for the b-bit-per-card memory system configuration if we can assume that all errors are block errors since this class is a maximum separable code[6][7]. Unfortunately, this assumption implies that error correcting codes of this class require a large number of check bits for the correction of multiple random bit errors because individual random bit errors may not be confined in a fewer number of blocks.

Consider that a channel model of the memory system with b-bit-per-card configuration. In such a memory system, both random bit errors and random block errors are likely to occur: random block errors may occur when control lines or address lines of some memory array card are interfered and random bit errors are inevitable in any digital system. Thus the memory system can be assumed as a compound channel[6][7].

For random error correction in compound channels, a class of Fire codes[6][7][14] is well known. A general class of Fire codes given by the generator polynomial p(x) such that

$$p(x) = (x^{2b-1}-1)g(x)$$

can correct both t or less random bit errors and any burst of bit errors of length b, which do not occur simultaneously, where g(x) is a generator polynomial of a code with minimum distance d = 2t + 1 such that

$$\deg[g(x)] \geq b,$$

g(x) and $x^{2b-1}-1$ are relatively prime.

Since our model of compound channel for memory systems with b-bit-per-card memory systems will yield a block error instead of a burst error, we modify Fire codes so that t or less random bit error and a single random block error can be corrected with less redundant check bits.

2.3 Subcodes of Fire Codes For Compound Channel

To avoid ambiguity, we distinguish burst errors and block errors.

Definition 1: Burst errors are bursts of bit errors which locate any part of a word. Block errors are bursts of bit errors

which locate block by block in a word, that is, they are "phased" burst errors.

Theorem 1:

Let C be a $(n,n-m)$ t or less random bit error correcting cyclic code generated by $g(x)$ such that $\deg[g(x)] = m$ and $t \geq 1$. Any single burst error can be corrected if the error pattern of the burst error is known and if $b \leq m$.

Proof:

Let $E(x)$ be the error polynomial of a single burst error described by

$$E(x) = \sum_{i=0}^{b-1} e_i x^{i+j}, \quad j \in \{0,1,\ldots,n-b\}.$$

Let $S_1(x)$ be the corresponding syndrome given by

$$S_1(x) = E(x) \mod g(x)$$

$$= \sum_{i=0}^{b-1} e_i x^i (x^j \mod g(x)); \mod g(x).$$

Let $S_2(x)$ be the error pattern syndrome, which is known, such that

$$S_2(x) = \sum_{i=0}^{b-1} e_i x^i.$$

The start bit of the burst error is given by

$$\frac{S_1(x)}{S_2(x)} = x^j \mod g(x).$$

Suppose that another burst error $E'(x)$, $E'(x) \neq E(x)$, occurs such that

$$E'(x) = \sum_{i=0}^{b-1} e_i' x^{i+k} \qquad k \in \{0,1,\ldots,n-1\}.$$

The corresponding syndromes $S_1'(x)$ and $S_2'(x)$ define the start bit of the burst error $E'(x)$, that is

$$\frac{S_1'(x)}{S_2'(x)} = x^k \qquad \text{mod. } g(x).$$

If $k \neq j$

$$x^k \neq x^j \qquad \text{mod. } g(x)$$

since C can correct at least a single random bit error. If $k = j$ the error patterns $S_1(x)$ and $S_2(x)$ must be different from the assumption. Therefore $E(x)$ and $E'(x)$ can be distinguished and any single burst error of length b, $b \leq m$, can be corrected.

Let H be the parity check matrix of C described by

$$H = [h_0, h_1, \ldots, h_{n-1}]$$

where h_i is a m-tuple binary column vector, i=0,1,...,n-1. Some p sets of consecutive b columns which are not overlapped are chosen from H and a sub matrix H' is defined such that

$$H' = [h_{\phi_0} \; h_{\phi_0+1} \cdots h_{\phi_0+b-1} \; h_{\phi_1} \; h_{\phi_1+1} \cdots h_{\phi_1+b-1} \cdots \cdots h_{\phi_{p-1}} \; h_{\phi_{p-1}+1} \cdots h_{\phi_{p-1}+b-1}]$$

where ϕ_i satisfies Condition 1, i = 0,1,...,p-1.

Condition 1:

(1) $\phi_i \in \{0,1,\ldots,n-b-1\}$, $i = 0,1,\ldots,p-1$

(2) $\phi_i - \phi_{i-1} \geq b$.

Lemma 1-1

Let C' be a shortened code given by H'. A code C_b given by the parity check matrix H_b,

$$H_b = \begin{bmatrix} H' \\ H_p \end{bmatrix} ,$$

can correct any single random block error, where

$$H_p = [I_b \; I_b \ldots I_b],$$

I_b is a b x b identity matrix.

Proof:

Let

$$C'(x) = \sum_{i=0}^{pb-1} c_i \, x^i$$

be a codeword of C'. This codeword must be represented by a codeword of C since C' is a shortened code obtained from C. Let

$$C(x) \sum_{i=0}^{p-1} \sum_{j=0}^{b-1} C_{i+j} x^{\phi_i+j}$$

be the projected codeword of $C'(x)$ in the codeword space of C. Suppose that E(x) is the error polynomial of a single block error occurring in the k-th block, $k \in \{0,1,\ldots,p-1\}$, such that

$$E'(x) = \sum_{i=0}^{b-1} e_i \, x^{b(k-1)+i}$$

The equivalent error for the codeword C(x) is described by

$$E(x) = \sum_{i=0}^{b-1} e_i^{\phi_k+i}$$

That is, the syndrome of E' generated H' is equivalent to the syndrome of E by H. Therefore

$$\begin{aligned} S_1(x) &= C(x) + E(x) \quad ; \quad \text{mod.}[a(x)] \\ &= \sum_{i=0}^{b-1} e_i x^{\phi_k+i} \quad ; \quad \text{mod.}[g(x)] \\ &= \sum_{i=0}^{b-1} e_i x^i (x^{\phi_k} \quad ; \quad \text{mod.}[a(x)]) \; ; \; \text{mod.}[g(x)] \end{aligned}$$

The error pattern syndrome $S_2(x)$ is directly obtained by H_p, such that

$$S_2(x) = \sum_{i=0}^{b-1} e_i x^i$$

Since $S_1(x)$ and $S_2(x)$ are known, any single random block error can be corrected by Theorem 1.

Lemma 1-2

Suppose that C' is a systematic code. C_b can correct any single random block error if H_p is given by either

$$H_p = H_{p_{odd}} = \underbrace{[I_b \; I_b \ldots I_b]}_{p \; I_b\text{'s}}$$

or

$$H_p = H_{p_{even}} = [\underbrace{I_b\ I_b \ldots I_b\ I_b}_{(p-1)\ \ I_b\text{'s}}]$$

where O_b denotes an all zero b x b matrix.

Proof:

The case that $H_p = H_{p_{odd}}$ is obvious. Suppose that each block of a codeword of C' is named 0,1,...,(p-1)-th block from left to right and that the (p-1)-th block does not include information bits. If a random block error occurs in the k-th block, $k \neq p-1$, the block error can be corrected by Lemma 1-1. If a random block error occurs in the (p-1)-th block, the error pattern syndrome $S_2(x)$ is zero and no correction will be done. But since the information bits of the codeword have no error in this case, correct information can be obtained.

We remark that Lemma 1-1 and Lemma 1-2 describe shortened Reed-Solomon codes and extended-shortened Reed-Solomon codes with d = 3 in $GF(2^m)$, respectively, when a primitive polynomial in $GF(2^m)$ is chosen for g(x).

Theorem 2:

C' can correct both any single random block error and t or less random bit errors, which do not occur simultaneously.

Proof:

Suppose that $H_p = H_{p_{odd}}$. If t or less random bit errors occur, the weight of error pattern syndrome S_2, $w[S_2]$, is t or less. If a random block error E occurs such that w[E] is t or less, E is corrected as random bit errors. If w[E] is more than t, $w[S_2]$ is greater than t. Thus a single block error and t or less bit errors have disjoint syndromes.

Suppose that $H_p = H_{p_{even}}$. If a block error occurs in the (p-1)-th block, $w[S_2]$ is zero. In this case correct information bits can be obtained by Lemma 1-2.

Thus any single random block error and t or less random bit errors can be corrected.

3.1 Reliability Improvement in Defective Main Memory Systems

In Section 2, main memory systems are assumed to be perfectly functional and a subcode of the Fire codes, C_b, has been shown. In this section, some methods for reliability improvement in defective main memory systems are discussed.

Suppose that memory devices are extremely integrated. It is practically impossible to produce perfectly functional memory devices since impurities in a substrate may result in hardware defects in the memory device. This becomes an extremely serious problem in ultra-high-density memory devices such as full-wafer

memory devices. Since it is difficult to produce perfectly functional ultra-high-density memory devices, some methods have been proposed to enhance the yield[1][5]. That is, even memory devices containing hardware defects can be practically used by employing these methods by users. These methods are:

(1) skewing memory reconfiguration.

(2) application of error correcting codes.

The skewing memory reconfiguration is a method in which a defective memory area is replaced by a supplemental memory area[1] or spread out in a functional memory area[3]. In the second case, errors caused by the spread out defects are corrected by an ordinary random bit error correcting code since these errors may not occur in a burst if the defective memory area is properly spread out.

In general, the skewing methods require an address encoder so that a defective memory area can be manipulated as a function of virtual memory address. Fig. 2 shows a simple example of the skewing method. The virtual memory address is encoded to the corresponding physical memory address by the address encoder. In most cases, the virtual address bits to the address encoder are high-order bits so that the physical memory is properly segmented.

The application of error correcting codes is a method in which error correcting codes correct errors caused by defects.

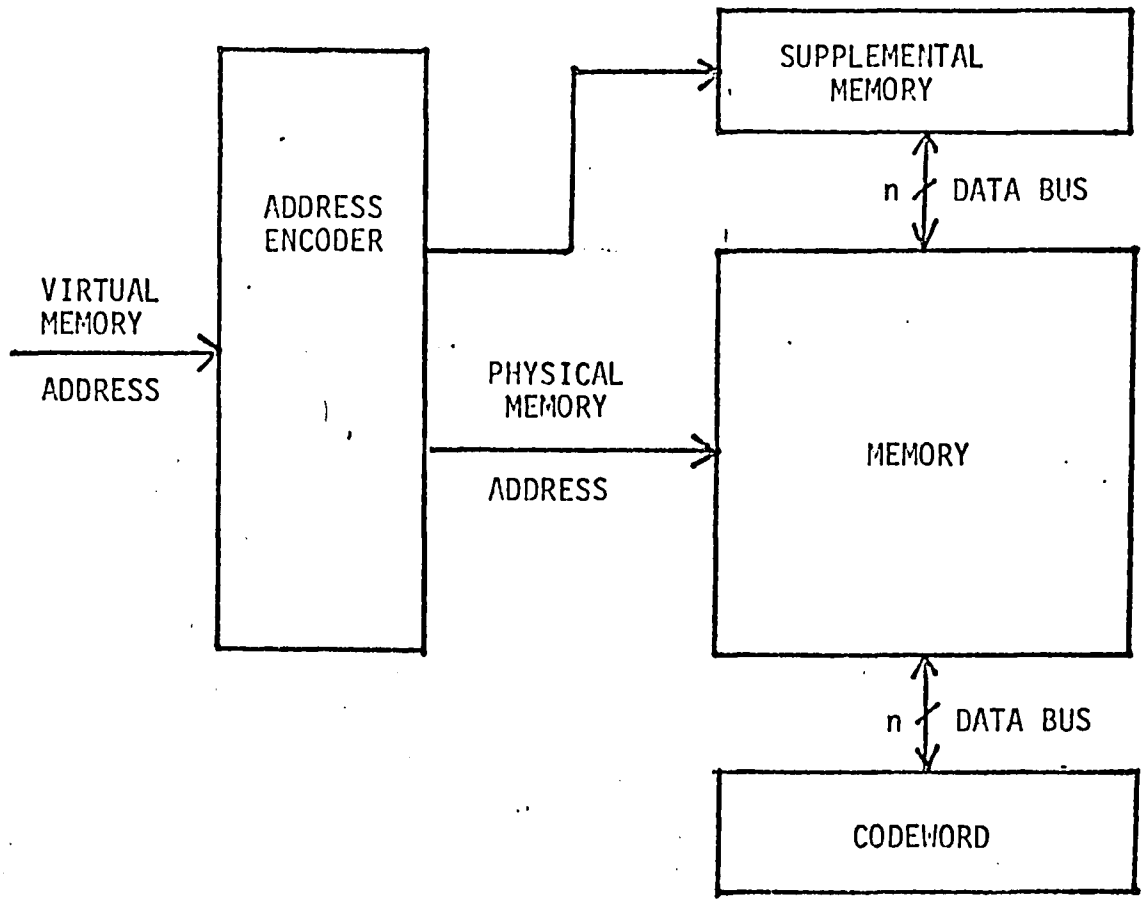

Fig. 2, An example of skewing reconfiguration.

Since defects are associated with hardware and will not change in a short period, information about defects could be available and used as error information. In the remainder of this thesis, the application of error correcting codes for reliability enhancement, that is, reliability improvement of defective memory systems is described. Thereafter, only stuck-at faults in memory devices are assumed to be possible hardware defects in the memory system.

3.2 Basic Idea of Error Correcting Code Is For Defective Memory Systems

First, we define some terminology.

Definition 2:

A stuck fault is a physical defect in a memory device and has a fixed logical value, either 0 or 1.

Definition 3:

The Stuck Fault Data (SFD) is bit-wise information which indicates the existence of stuck faults in the memory. The SFD may take three logical values, 0,1 and x in each bit position of the word. 0 and 1 denote "stuck-at-0" and "stuck-at'1", respectively and x denotes that the bit position is not affected by the stuck fault. If the SFD is all x at some memory address,

the write word and read word are consistent. This case is to be called "no stuck fault". If the SFD is not all x, the write word and read word may be inconsistent.

Definition 4:

A stuck error is the difference between the write word and read word due to stuck faults.

Definition 5:

A stuck fault is said to be masked if the write word and read word are consistent. Note that stuck errors are associated not only with the stuck fault but also with the write word. That is, if there is no stuck fault, the stuck error equals zero but the converse is not always true.

Example 1:

Let (1 x 0 x), (1 1 0 0) be the SFD and write word, respectively. The read word will be (1 1 0 0) and the corresponding stuck error is (0 0 0 0). That is, the stuck fault is masked.

First, we assume that all SFD's are known. As defined in Definition , the SFD contains information of erroneous bit positions which may result in stuck errors. This implies that the SFD can be used as error information in the decoding of random error correcting codes. Fig. 3 shows a basic encoder and decoder configuration of this method.

Let t, T and d be the number of correctable random errors, the number of correctable erasures and the minimum distance of a code, respectively. Then

$$d \geq 2t + T + 1$$

must be satisfied[6]. Suppose that τ is the number of errors which should be corrected. The required minimum distanced is

$$d \geq 2\tau + 1 \qquad \text{for random errors}$$
$$d \geq \tau + 1 \qquad \text{for erasures.}$$

if they are not erasures. Directly from this fact, Cater[2] showed a method of stuck error correction using a random bit error correcting code with d = 4 to correct 3 or less stuck bit errors. Although this method is quite simple, the decoder of the error correcting code must refer to an additional data base in the decoding procedure. This is quite a disadvantage since every read cycle requires the decoder to refer to the additional data base. Namely, each instruction cycle will be lengthened.

To avoid this disadvantage, another method has been proposed[4][5]. In this method, the decoder need not refer to the additional data base and high-speed decoding is possible. A basic idea of this method is to encode write words so that the encoded write words can mask stuck faults. The set of encoded write words is not a group and we have to define the set as

follows.

Definition 6:

A stuck error correcting code is a set of codewords which are encoded so that stuck faults are masked.

3.3 Stuck Block Error Correcting Codes For Defective B-Bit-Per-Card Memory Systems

As assumed before, a memory system with b-bit-per-card configuration is a compound channel in which both block errors and random bit errors are likely. If there is no stuck fault in the memory, the subcode of the Fire codes, C_b, will show a good performance. Consider the case that one defective memory device exists in the memory system. The defects may cause single stuck block errors. If there is no additional error, C_b corrects these single stuck block errors as single random block errors. But, in general, random errors and stuck errors are independent and additional random errors must be considered. Because of the nature of digital systems, we assume hereafter that random bit errors are more likely to occur than random block errors.

To correct both stuck errors and random errors, a class of additive codes[14][15] has been shown. This class has unique encoder and decoder configurations as shown in Fig. 4. The additive code is a subset of codewords of a random error correcting code, which mask stuck faults. Encoder I and Decoder I are for the

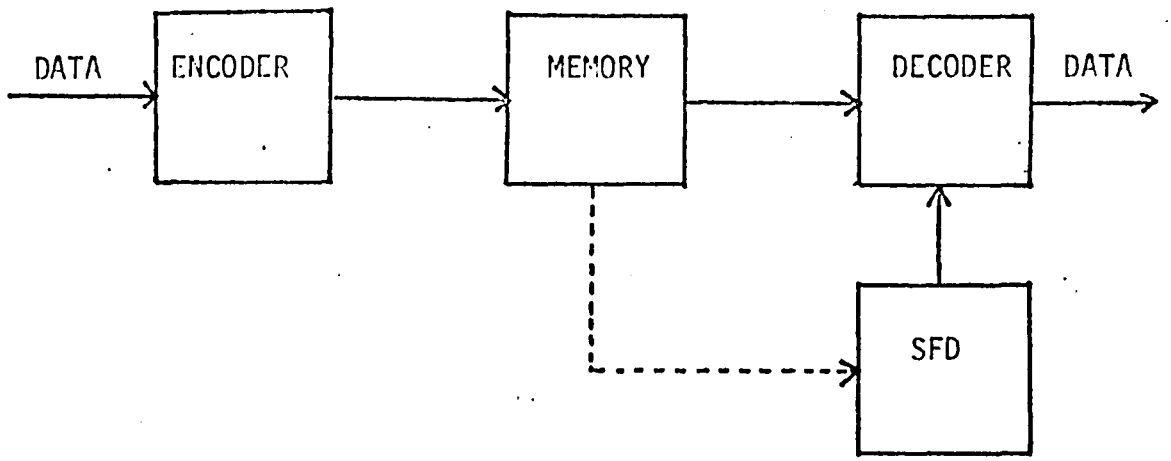

Fig. 3. Encoder and decoder configuration for erasure error correcting codes.

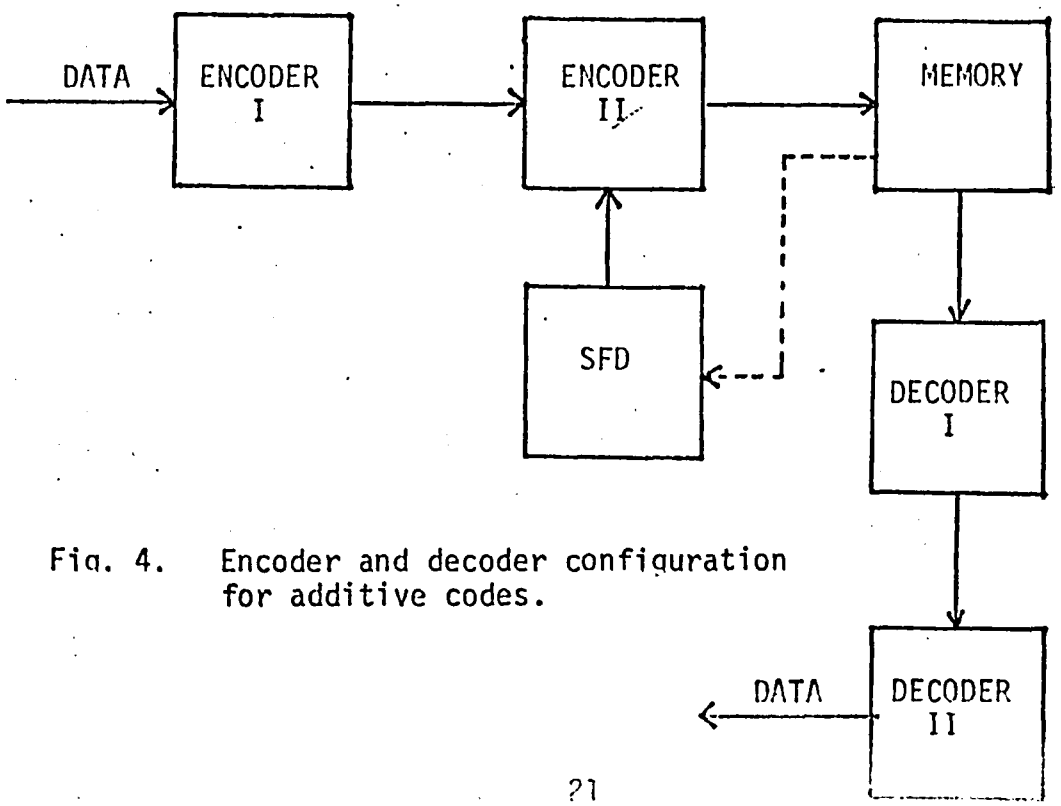

Fig. 4. Encoder and decoder configuration for additive codes.

correction of random errors. Encoder II and Decoder II are for the correction of stuck errors. All codewords output from Encoder II are still in the codeword space of the random error correcting code.

In this section a subcode of additive stuck error correcting codes is described. For simplicity, a subcode of additive stuck block error correcting codes which correct both any single stuck block error and random bit errors which may occur simultaneously, is referred to as simply "stuck block error correcting code".

Definition 7:

The binary operation $\alpha * \beta$ is defined by the truth table

α	β	$\alpha * \beta$
0	0	0
0	1	1
0	x	0
1	0	0
1	1	1
1	x	1

where $\alpha \in \{0,1\}$ and $\beta \in \{0,1,x\}$. The binary operation * is neither commutative nor associative. Assume for simplicity, that the word length n is equal to pb. Let $C_w = (C_{w_0}, C_{w_1}, \ldots, C_{w_{p-1}})$, $C_r = (C_{r_0}, C_{r_1}, \ldots, C_{r_{p-1}})$, $F = (F_0, F_1, \ldots, F_{p-1})$, and $E = (E_0, E_1, \ldots, E_{p-1})$ be write word, read word, SFD, and stuck error,

respectively. C_{w_i}, C_{r_i} and E_i are binary row vectors of length b and F_i is a trinary vector, i = 0,1,...,p-1. The read word and stuck error are given by

$$C_r = C_w * F$$

and

$$E = C_w + (C_w * F),$$

respectively. Suppose that there is a defective memory device in the k-th memory array card, possible stuck error is described by

$$E_k = C_{w_k} + (C_{w_k} * F_k)$$

and

$$E_i = C_{w_i} \text{ for } i \neq k;\ i = 0,1,\ldots,p-1.$$

Proposition 1:

Let $$C_w' = (C_{w_0} + E_k,\ C_{w_1} + E_k,\ldots,C_{w_{p-1}} + E_k).$$

If C_w' is written in the memory instead of C_w, the stuck error can be masked.

Proof:

For $i \neq k$,

$$C_{r_i} = C_{w_i}' * F_k$$

$$C_{r_i} = C'_{w_i}$$

since F_i is all x.

For i = k, since $C_{w_k} + E_k = C_{w_k} * F_k$, the erroneous k-th block of C_r is described by

$$\begin{aligned} C_{r_k} &= C'_{w_k} * F_k \\ &= (C_{w_k} + E_k) * F_k \\ &= (C_{w_k} * F_k) * F_k \\ &= C_{w_k} * F_k \\ &= C_{w_k} + E_k \\ &= C'_{w_k} \end{aligned}$$

Example 2:

Let [(1 x 0 x), (x x x x)] and [(1 1 1 1), (1 0 1 1)] be the SFD and write word, respectively. The read word and stuck error are [(1 1 0 1), (1 0 1 1)] and [(0 0 1 0), (0 0 0 0)], respectively. If [(1 1 0 1), (1 0 1 1)] is the write word, the read word and the SFD are [(1 1 0 1), (1 0 1 1)] and [(0 0 0 0), (0 0 0 0)], respectively. Hence the stuck fault is masked.

Let a matrix H,

$$H = [h_0, H_1, \ldots, h_{n-1}]$$

be a parity check matrix of an (n,n-m) code C, where h_i is a binary column vector of length m, i = 0,1,...,n-1, and n is equal to pb. Let $(C_0,C_1,\ldots,C_{p-1})$ be a codeword of C, where C_i is a binary row vector of length b.

Define the following condition.

Condition 2:

The column vectors of H satisfy the condition

$$\sum_{i=0}^{p-1} h_{bi+\ell} = 0 \quad \text{for } \ell = 0,1,\ldots,b-1.$$

Theorem 3:

Let C (E) be given by

$$C(E) = (C_0 + E, C_1 + E, \ldots, C_{p-1} + E)$$

where E is an arbitrary binary row vector of length b. Then C (E) is also a codeword of C only if H satisfies Condition 2.

Proof:

Since C (0) is a codeword of C,

$$C(0)\,H^T = 0$$

where T denotes transposition of a matrix. Then

$$C(E)\,H^T = (C_0 + E, E_1 + E, \ldots, C_{p-1} + E)H^T$$

$$= C(0)\,H^T + (E,E,\ldots,E)H^T$$

$$= E[(\sum_{i=0}^{p-1} h_{bi})\ (\sum_{i=0}^{p-1} h_{bi+1})\ldots(\sum_{i=0}^{p-1} h_{(b-1)i-\ell})]^T$$

$$= 0$$

Lemma 3-1:

Any (n,n-m) code C can generate a (bp, b(p-m)) code with the same error correcting capability, which satisfies Condition 2 if C has a codeword of weight p.

Proof:

Suppose that a binary vector $(a_0,a_1,\ldots,a_{n-1})$ is a codeword of C, whose weight is p. Namely,

$$\sum_{i=0}^{p-1} h_{\psi_i} = 0 .$$

where ψ_i is an integer, $\psi_i \in \{0,1,\ldots,n-1\}$ such that

$$a_{\psi_i} = 1,\ i = 0,1,\ldots,p-1.$$

Let H' be given by

$$H' = [h'_0,h'_1,\ldots,h'_{bp-1}]$$

where

$$h'_{bi+\ell} = [0\ 0\ \ldots\ 0\ (h_{\psi_i})^T\ 0\ 0\ \ldots\ 0]^T,\ i = 0,1,\ldots,p-1$$

$\overleftarrow{\qquad}$ $(b-\ell-1)m$ zeros; $\overleftarrow{\qquad}$ ℓm zeros; and $\ell=0,1,\ldots,b-1$.

Then H′ gives a shortened interleaved code with degree b, which satisfies Condition 2 , since

$$\sum_{i=0}^{p-1} h_{bi+\ell} = [\underbrace{0\ 0\ \ldots\ 0}_{(b-\ell-1)m \text{ zeros}}\ (\sum_{i=0}^{p-1} h_{\psi_i})^T\ \underbrace{0\ 0\ \ldots\ 0}_{\ell m \text{ zeros}}]^T$$

$$= 0 \qquad \text{for } \ell = 0,1,\ldots,b-1.$$

It is obvious that C′ retains the same error correcting capability of C.

Lemma 3-2:

Suppose that a(n,n-m) code C is cyclic. A shortened code C′ given by the parity check matrix H′,

$$H' = [h_{\psi_0}\ h_{\psi_0+1} \cdots h_{\psi_0+b+1}\ h_{\psi_1}\ h_{\psi_1+o} \cdots h_{\psi_{p-1}} \cdots \cdots h_{\psi_{p-1}+b-1}]$$

satisfies Condition 2 only if there exists a set Ψ,

$$\Psi = \{\psi_0,\psi_1,\ldots,\psi_{p-1}\},$$

whose elements satisfy the following condition.

Condition 3:

$$(1) \sum_{i=0}^{p-1} h_{\psi_i} = 0$$

(2) Condition 1

Proof:

Since

$$\sum_{i=0}^{p-1} h_{\Psi_i} = 0 ,$$

$(a_0,a_1,\ldots,a_{n-1})$ is a codeword of C where

$$a_i = 1 \qquad \text{if } i \in \Psi$$

and

$$a_i = 0 \qquad \text{otherwise}, i = 0,1,\ldots,n-1.$$

Since C is cyclic, the cyclically shifted vector $(a_{n-\ell},a_{n-\ell+1},\ldots,a_{n-1},a_0,\ldots,a_{n-\ell-1})$ is also a codeword of C for $\ell = 0,1,\ldots,n-2$. Thus

$$\sum_{i=0}^{p-1} h_{\Psi_i+\ell} = 0 \quad , \quad \ell = 0,1,\ldots,b-1.$$

Since $\Psi_{p-1} \leq n-b-1$ and $\Psi_{i+1} - \Psi_i \geq b$ from Condition 1, there exists a unique column $h_{\Psi_i+\ell}$ for $i = 0,1,\ldots,p-1$ and $\ell = 0,1,\ldots,b-1$. Namely, C' is a shortened code of C, which satisfies Condition 2.

Suppose that C' is a systematic code obtained from a t or less random bit error correcting code C and that C' satisfies

Condition 2. Let $C'\{0\}$ be a subset of codewords of C' whose j-th block is all zero and let $C'(0)$ be a codeword in $C'\{0\}$. If the SFD of the memory address in which $C'(0)$ will be written is not all x, some stuck error E may result in the read word C_r. Let E_k be a non-zero block of E such that

$$E = C'(0) + (C'(0) * F)$$

where F is the SFD at the memory address. Since $C'(E_k)$ is a codeword of C' from Theorem 3, any t or less random bit errors in C_r can be corrected. Furthermore, $C'(E_k)$ will mask the stuck fault and no stuck error will result. Fig. 5 illustrates the error correcting procedure in the case of j = 0.

A set of all codewords $C'(E_k)$ which mask stuck faults are referred to as C_s, a block stuck correcting code with t or less random bit error correcting capability.

The information rate R of C_s is given by

$$R = \frac{n'-m-b}{n'}$$

where n', m and b are the length of codeword, check bits and block of C', respectively.

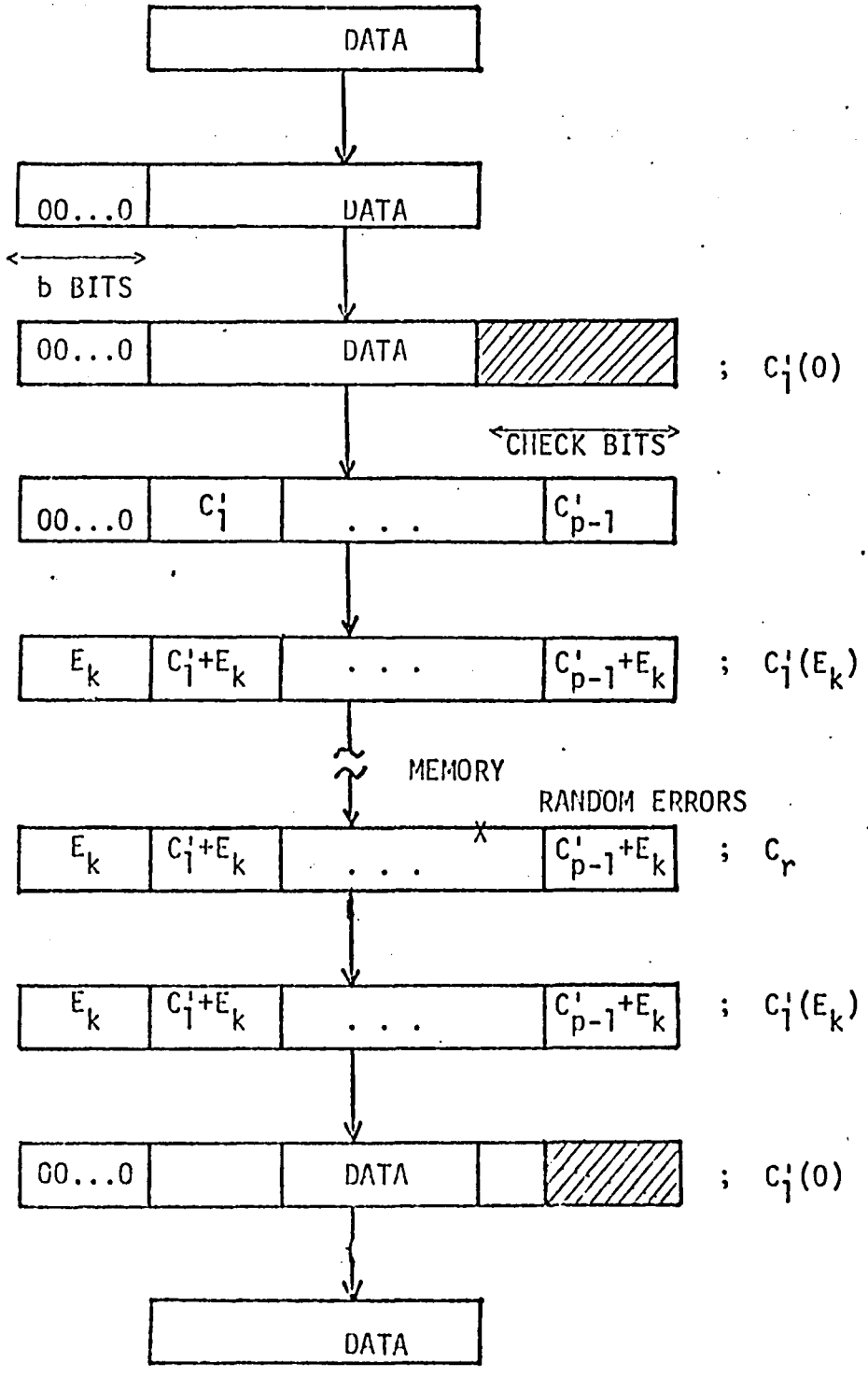

Fig. 5. Illustration of encoding and decoding procedures.

4.1 Modified Stuck Block Error Correcting Codes C_M

As described in the previous section, stuck error correcting codes are primarily proposed to enhance the yield of high-density memory devices. In this case, we can assume that all stuck faults have been found before encoding write data and all SFD's are available.

Consider the case that no memory device is defective in the initial period. Since it is impossible to predict stuck faults when write data are encoded, the set of all codewords is equivalent to $C'\{0\}$. So far as all devices are functional, only random bit errors are assumed to occur and will be corrected if the number of bit errors are not more than t.

Suppose that some weak device becomes defective. It may result in single stuck block errors. Since there is no information about the stuck fault, these single block errors must be corrected as random block errors.

In this section a modified stuck block error correcting code C_M is shown. All codewords of C_M are codewords of C_b, the subcode of the Fire codes, when the memory system has no stuck fault. If the memory system becomes defective, all codewords of C_M are considered to be codewords of C_s after re-encoding erroneous words. Although most codewords are left in the codeword

space of C_b, all read words are decoded as C_s.

There may exist alternative ways to correct single stuck block errors as random block errors. That is, all write data are encoded to codewords of C_b, when the memory system has no stuck fault. After some stuck faults are found, all codewords are re-encoded to codewords of C_s. This method is straightforward but re-encoding of all codewords takes a long time for large memory systems. Furthermore, if the memory system contains some Read Only Memory (ROM) area, it is impossible to re-encode the contents in the ROM. It is also a possible way to re-encode only codewords which yield stuck block errors with flags. But the decoder may require an additional data base for flags to distinguish classes, C_b and C_s, of input words read out.

The encoding-decoding procudure of C_M is as follows:

(1) Initially, all write data are encoded to codewords of C_b.

(2) If stuck errors are observed in some memory addresses, only the corresponding codewords are re-encoded to codewords of C_s so that stuck faults are masked.

(3) All codewords are decoded as codewords of C_s.

Suppose that C' is a t or less random bit error correcting systematic code which satisfies condition 2. Then a codeword of C_M is described by

$$C_M(E_k) = (P+E_k, C'_1+P+E_k, \ldots, C'_{p-1}+P+E_k)$$

where $(0_b, C'_2, C'_2, \ldots, C'_{p-1})$ is a codeword of C', C'(0). P is a parity check block such that

$$P = \sum_{i=1}^{p-u+1} C'_i$$

where

$u = 0$ for p odd

$u = 1$ for p even

E_k is the non-zero block of E given by

$$E = C'(P) + (C'(P) * F)$$

where F is the SFD of the memory address in which the write word is written.

Theorem 4:

C_M can correct any t or less random bit errors and single random block errors if all the SFD's are x when all the write data are encoded.

Proof:

Since C' satisfies Condition 2,

$$C_M(E_k) = C'(P+E_k)$$

$$\in C'$$

from Theorem 3. To correct single random block errors, the block

error pattern must be obtained. Suppose that a single random block error E_r occurs in the k-th block, $k \neq p-1$, the error pattern syndrome S_2 is obtained by

$$\begin{aligned} S_2 &= P + \sum_{i=1}^{p-u-1} (C_i' + P) + E_r \\ &= \sum_{i=1}^{p-u-1} P + \sum_{i-1}^{p-u-1} C_i' + P + E_r \\ &= (p-u-1)P + E_r \\ &= E_r \end{aligned}$$

since p-u-1 is always even. Thus the block error E_r can be corrected by Theorem 1. If $k = p-1$, $S_2 = 0$. But we can avoid using (p-1)-th block for information bits since C' is a systematic code.

Note that the parity check matrix of C_M for random block error correction is given by

$$H = \begin{bmatrix} H' \\ H_p \end{bmatrix}$$

which is equivalent to the parity check matrix of C_b, where H_p is given by

$$H_p = H_{p_{odd}} \quad \text{for} \quad p_{odd}$$

and

$$H_p = H_{p_{even}} \quad \text{for} \quad p_{even}$$

Namely, the set of $C_M(0)$ and C_b are equivalent, but codeword structures are different. Because the codeword of C_b is represented by

$$(P,C'_1,\ldots,C'_{p-1})$$

While, $C_M(0)$ is

$$(P,C'_1+P,\ldots,C'_{p-1}+P)$$

Fig. 6 illustrates the relations of codeword of C_b and $C_M(0)$.

Theorem 5:

C_M can correct any t or less random block errors and single stuck block error which may occur simultaneously.

Proof:

We assume that all codewords which had stuck block errors have been re-encoded using new SFD's. That is, all stuck faults are masked. Since all codewords in the memory can be represented by the form,

$$C'(A) = (A,C'_1+A,\ldots,C'_{p-1}+A) .$$

Since $C'(A) \in C'$ from Theorem 3, any additional t or less random bit errors can be corrected.

4.2 Encoding of C_M

Let us define a "fault free state" and "fault state" for the memory system free from defects and having defects, respectively. The decoding procedure of C_M is different for these two states.

Fault free state:

Fig. 7 shows a configuration of encoder in the fault free state. An all zero block is added to the input binary write data $(d_0,d_1,\ldots,d_{n'-m-b})$ where n', m are the length of codeword and check bits of C' and b is the block length. Then Encoder 2 encodes

$$(0,0,\ldots,0,d_0,d_1,\ldots,d_{n'-m-b})$$

$$\underbrace{\qquad\qquad}_{b\ 0's}$$

to a codeword

$$C'(0) = (0_b,C'_1,\ldots,C'_{p-1})$$

Encoder II generates the parity check block P given by

$$P = \sum_{i=1}^{p-u-1} C'_i$$

where

$u = 0$ for p odd

$n = 1$ for p even.

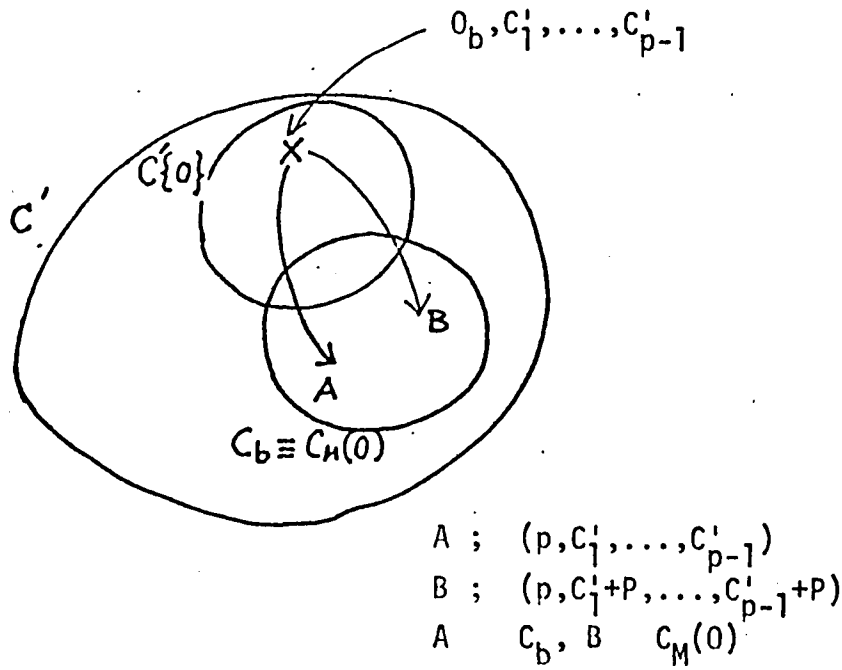

Fig. 6. Relation between C_b and $C_M(0)$.

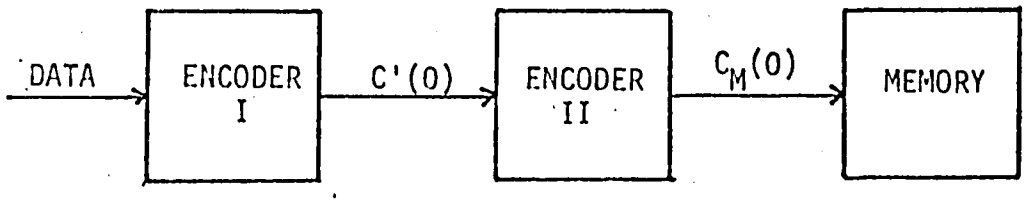

Fig. 7. Encoder configuration of C_M (Fault Free State).

P is added to all the block of $C'(0)$ and $C'(P)$, such that

$$C'(P) = C_M(0)$$

$$= (P, C'_1+P, \ldots, C'_{p-1}+P),$$

is written in the memory.

Fault State

In the fault state, write data must be encoded to codewords of C_s so that stuck faults are masked. Assume that operating systems in computer systems have some memory check routines which check the existence of stuck faults. The check routine will creat SFD's if there are stuck faults, and these SFD's can be used in the encoding. The input data $(d_0, d_1, \ldots, d_{n-m-b})$ is encoded to a codeword $C'(0)$ by Encoder I. Encoder II encodes $C'(0)$ to $C_M(0)$ such that

$$C_M(0) = C'(P)$$

Encoder III finds a stuck block error E given by

$$E = C_M(0) + (C_M(0)*F).$$

The non-zero block of E, E_k, is added to all the blocks of $C_M(0)$ in Decoder III and the output $C_M(E_k)$ is written in the memory. Fig. 8 shows the encoding procedure in the fault state.

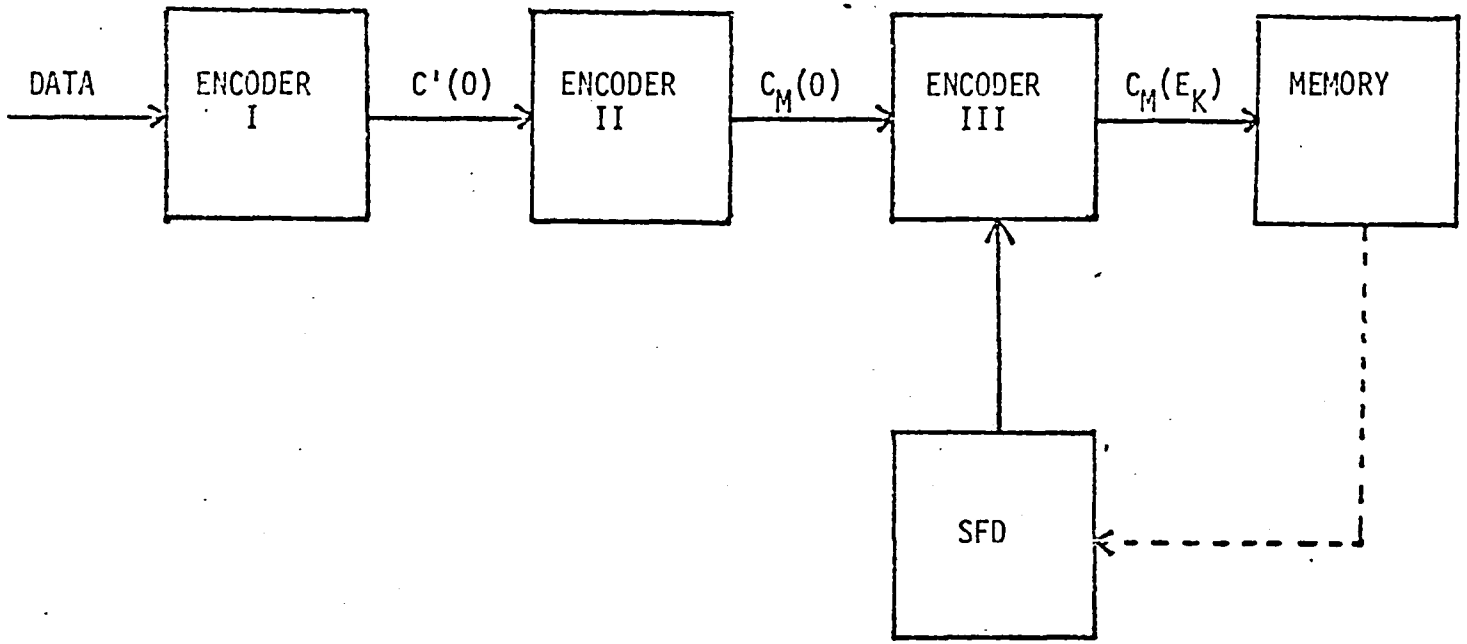

Fig. 8. Encoder configuration of C_M (Fault State).

4.3 Memory Check Routine

The memory check routine has two roles. It checks stuck faults in the memory and creates SFD's. Also it re-encodes contents of erroneous memory addresses. The memory check may be executed periodically. Since it may take a long time to check the whole memory, it is a good way to examine only memory addresses where block errors have been observed by the decoder. If some stuck faults are found, the contents of these memroy addresses are re-encoded by the same procedure of the decoding in the fault state. If any content is re-encoded, codewords are not considered as codewords of C_b but C_s. To let the decoder know this, the memory check routine sets a flag at the end of the routine.

Fig. 9 shows one example of the flowchart of a memory check routine.

4.4 Decoding of C_M

Fault Free State

If there is no stuck fault, all read data are decoded as codewords of C_b by Decoder I and any single random block error and t or less random bit errors can be corrected. Decoder I is followed by Decoder II so that codeword $C'(0)$ can be obtained from $C_M(0)$, the output of Decoder I. The input data is directly obtained from $C'(0)$ since C' is a systematic code. Fig. 10 shows

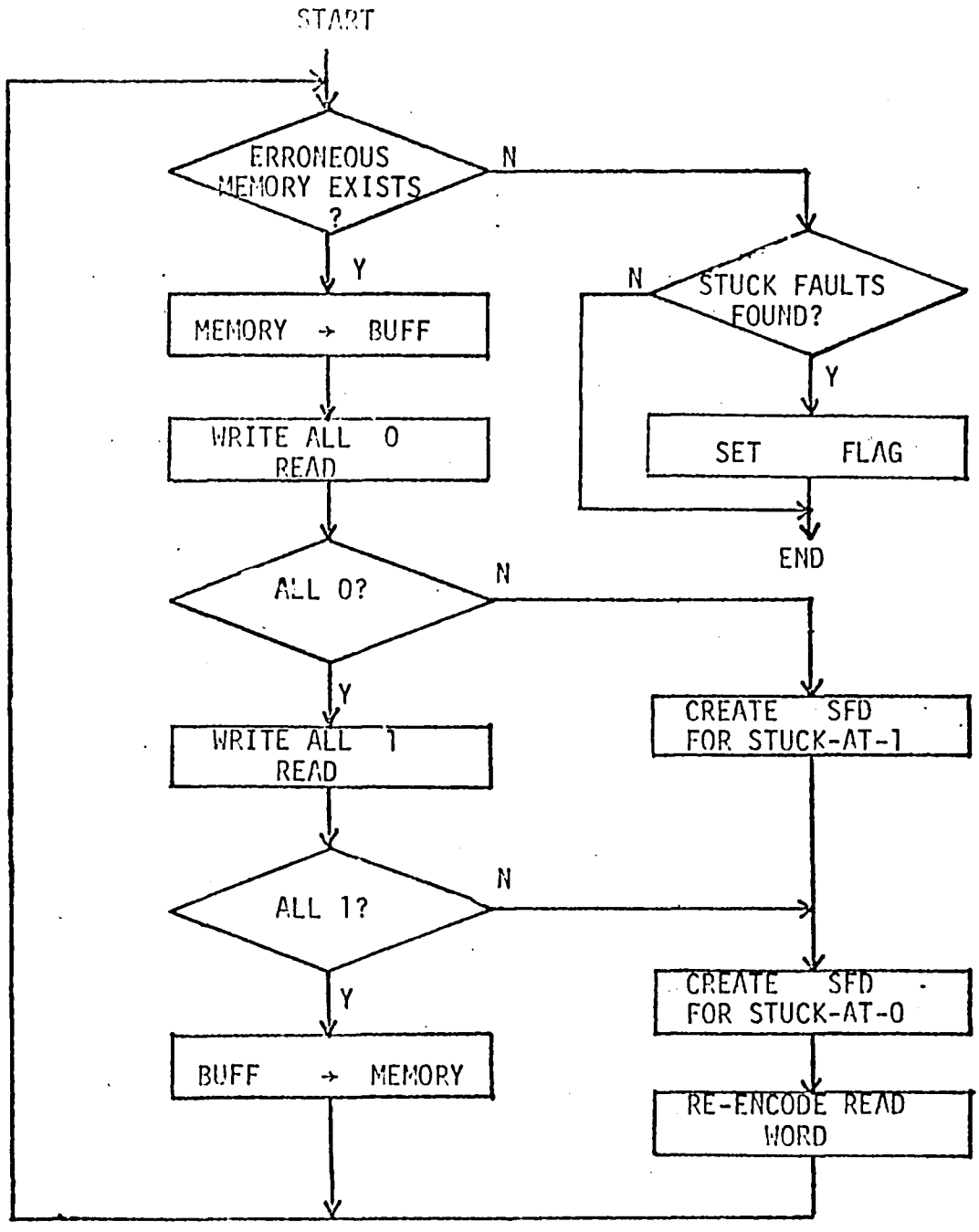

Fig. 9. Flowchart of memory check routine.

the decoder configuration of the fault free state. In Decoder I, error pattern syndrome S_2, which is given by

$$S_2 = C_r \cdot H_p^T,$$

is checked whether $W[S_2]$ is greater than t or not. If $W[S_2]$ is not greater than t, only syndrome S_1, which is given by

$$S_2 = C_r \cdot H'^T,$$

is used for correction of random bit errors. If $W[S_2]$ is greater than t, both S_1 and S_2 are used for correction of single random block error using Lemma 1-2. In the following Decoder II, the content of the 0-th block of $C_M(0)$ is added to all the blocks and $C'(0)$ is obtained.

Fault State

First, all read data are decoded as codewords of C' by Decoder III is followed by Decoder II so that $C'(0)$ is obtained. Since all stuck faults are assumed to have been masked, correct write data can be obtained. Fig. 11 shows the decoder configuration in the fault state, where E_k may or may not be zero. In Decoder III, only syndrome S_2, given by

$$S_2 = C_r \cdot H'^T$$

is used for t or less random bit error correction. In Decoder II, the content of the 0-th block of $C_M(E_k)$ is added to all the

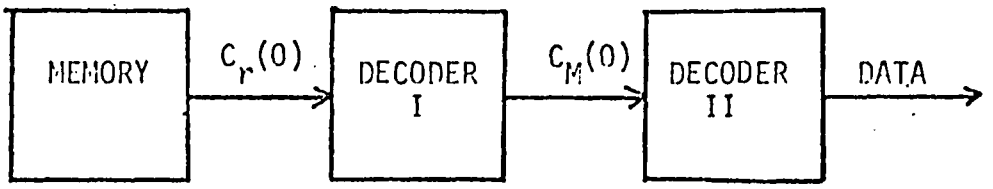

Fig. 10. Decoder configuration (Fault Free State).

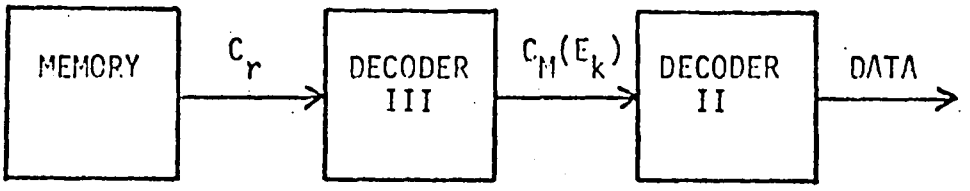

Fig. 11. Decoder configuration (Fault State).

block and $C'(0)$ is obtained.

Since Decoder I and Decoder III have many common parts, actual hardware can be minimized.

4.5 Example of C_M

An example of C_M is shown here. This code has parameters $n = 11$, $k = 6$, $b = 2$, and $t = 1$ where n, k and b are the length of codeword, information bits, and block, respectively, and t is the number of correctable random bit errors. Fig. 12 shows the parity check matrix of C generated by $g(x) = x^4 + x + 1$. Fig. 13 shows a partiy check matrix of C' which is shortened from C and satisfies Condition 2 since $b = 2$ and $p = 6$, the parity check matrix for C_b is given by

$$H = \begin{bmatrix} H' \\ H_{p_{even}} \end{bmatrix}$$

where H' is the parity check matrix of C', and $H_{p_{even}}$ is given by

$$H_{p_{even}} = [I_2\ I_2\ I_2\ I_2\ I_2\ 0_2]$$

Fig. 14 shows the parity check matrix H of C_b derived from H'. In the figures, $\overset{\downarrow i}{\alpha^i}$ denotes the binary column vector representation of α^i where α is the primitive root of $g(x)$ over $GF(2^4)$.

$$H = \begin{bmatrix} \alpha^{14} & \alpha^{13} & \alpha^{12} & \alpha^{11} & \alpha^{10} & \alpha^{9} & \alpha^{8} & \alpha^{7} & \alpha^{6} & \alpha^{5} & \alpha^{4} & \alpha^{3} & \alpha^{2} & \alpha^{1} & \alpha^{0} \end{bmatrix}$$

$$= \begin{bmatrix} 1 & 1 & 1 & 1 & 0 & 1 & 0 & 1 & 1 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 1 & 0 & 1 & 0 & 1 & 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 & 1 & 0 & 1 & 0 & 1 & 1 & 0 & 0 & 1 & 0 \\ 1 & 1 & 1 & 0 & 1 & 0 & 1 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 1 \end{bmatrix}$$

Fig. 12. The parity check matrix of (15,11,8) Haming code C.

$$H_1 = \begin{bmatrix} \alpha^{13} & \alpha^{12} & \alpha^{9} & \alpha^{8} & \alpha^{7} & \alpha^{6} & \alpha^{5} & \alpha^{4} & \alpha^{3} & \alpha^{2} & \alpha^{1} & \alpha^{0} \end{bmatrix}$$

$$= \begin{bmatrix} 1 & 1 & 1 & 0 & 1 & 1 & 0 & 0 & 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 1 & 0 & 1 & 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 & 1 & 0 & 1 & 1 & 0 & 0 & 1 & 0 \\ 1 & 1 & 0 & 1 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 1 \end{bmatrix}$$

Fig. 13. The parity check matrix of C'.

$$H_{m_f} = \begin{bmatrix} \alpha^{13} & \alpha^{12} & \alpha^{9} & \alpha^{8} & \alpha^{7} & \alpha^{6} & \alpha^{5} & \alpha^{4} & \alpha^{3} & \alpha^{2} & \alpha^{1} & \alpha^{0} \\ 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 0 \end{bmatrix}$$

$$= \begin{bmatrix} 1 & 1 & 1 & 0 & 1 & 1 & 0 & 0 & 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 1 & 0 & 1 & 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 & 1 & 0 & 1 & 1 & 0 & 0 & 1 & 0 \\ 1 & 1 & 0 & 1 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 1 \\ 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 0 \end{bmatrix}$$

Fig. 14. The parity check matrix of C_b.

4.6 Example of Encoding and Decoding in Fault Free State

Suppose that (1 0 1 0 1 1) be the write data. Then (0 0 | 1 0 1 0 1 1) is encoded to a codeword of C' by Encoder I, which has a generator matrix G' such that

$$G' H'^{T} = 0$$

The output codeword is (0 0 | 1 0 1 0 1 1 | 0 1 0 0). In Encoder II, parity check block P is generated. In our example, P is given by

$$P = (1\ 0) + (1\ 0) + (1\ 1) + (0\ 1) + (0\ 0) = (1\ 0).$$

Then P is added to the whole blocks and Encoder II outputs the codeword of $C_M(0)$ such that

$$C_M(0) = (1\ 0 \mid 0\ 0\ 0\ 0\ 0\ 1 \mid 1\ 1\ 1\ 0).$$

Since we assume that there is no stuck fault, $C_M(0)$ is written in the memory.

In the read cycle, suppose that

$$C_r = (1\ 0 \mid 1\ 0\ 0\ 0\ 0\ 1 \mid 1\ 1\ 1\ 0)$$

is read out from the memory. The parity check matrix H' and $H_{p_{even}}$ will generate syndromes S_1 and S_2, respectively, such that

$$S_1 = C_r \cdot H_1^T = (1\ 0\ 1\ 0)$$

and

$$S_2 = C_r \cdot H_{p_{even}}^T = (1\ 0)\ .$$

Since

$$w[S_2] = 1 \leq t,$$

a single random bit error will be corrected.

In our example,

$$S_1 = (1\ 0\ 1\ 0)$$

$$= (\alpha_1^{112})^T$$

Thus the rrror in the bit position of α^9 in H' is corrected. That is, the correct codeword

$$C_M(0) = (1\ 0 \mid 0\ 0\ 0\ 0\ 0\ 1 \mid 1\ 1\ 1\ 0)$$

is obtained. In Decoder II, the 0-th block (1 0) is added to the whole blocks. Then the output of Decoder II is (0 0 | 1 0 1 0 1 1 | 0 1 0 0) and the original write data (1 0 1 0 1 1) is obtained.

Suppose C_r has a single random block error. Let

$$C_r = (1\ 0 \mid 1\ 1\ 0\ 0\ 0\ 1 \mid 1\ 1\ 1\ 0)$$

be the read word. Then S_1 and S_2 are

$$S_1 = (1\ 1\ 1\ 1)$$

and

$$S_2 = (1\ 1),$$

respectively. Since

$$w[S_2] = 2 > t,$$

a random block error will be corrected. From Theorem 1, the start bit of the block error is given by

$$\frac{S_1(x)}{S_2(x)} \mod g(x) \ ; \ g(x) = x^4 + x + 1$$

where $S_1(x)$ and $S_2(x)$ are the polynomial expression of S_1 and S_2, respectively. Thus

$$\frac{x^3 + x^2 + x + 1}{x + 1} \mod g(x)$$

$$= x^2 + 1 \mod g(x) .$$

Since

$$x^2 + 1 \triangleq \alpha^8 ,$$

the block error starts at the bit position α^8 in H' towards the higher order of α, α is a primitive root of $g(x)$. On the other hand, the block error pattern is given by $S_2 = (1\ 1)$, Decoder I corrects the block error and outputs

$$C_M(0) = (1\ 0 \mid 0\ 0\ 0\ 0\ 0\ 1 \mid 1\ 1\ 1\ 0).$$

The rest part of the procedure is the same as the decoding in the fault free state.

4.6 Example of Encoding and Decoding in Fault State

Suppose that stuck faults are found in a memory check routine. Let F be the SFD at some memory address such that

$$F = (x\ x \mid 1\ 1\ x\ x\ x\ x \mid x\ x\ x\ x).$$

If the content of which is written in this memory is $C_M(0) = (1\ 0 \mid 0\ 0\ 0\ 0\ 0\ 1 \mid 1\ 1\ 1\ 0)$, then the read data C_r will be

$$C_r = (1\ 0 \mid 1\ 1\ 0\ 0\ 0\ 1 \mid 1\ 1\ 1\ 0)$$

and thus $C_M(0)$ will be re-encoded to mask the SFD.

First, $C_M(0)$ is encoded to $C_M(E_k)$ where E_k is the expected stuck block error such that

$$E_k = \sum_{i=0}^{p-1} (C_M(0)_i * F_i)$$

$$= (0\ 0) * (1\ 1)$$

$$= (1\ 1)$$

where $C_M(0)_i$ and F_i denote the i-th of $C_M(0)$ and F, respectively. Then, the re-encoded codeword.

$$C_M(E_k) = C_M\ (1\ 1)$$

$$= (0\ 1\ |\ 1\ 1\ 1\ 1\ 1\ 0\ |\ 0\ 0\ 0\ 1)$$

is written to the memory.

In the read cycle, there is no stuck error in the read word C_r since the stuck fault has been masked. Assume that no random bit error occurs in the read data C_r. Then C_r will be given by

$$C_r = C_r * F$$

$$= (0\ 1\ |\ 1\ 1\ 1\ 1\ 1\ 0\ |\ 0\ 0\ 0\ 1) *$$
$$(x\ x\ |\ 1\ 1\ x\ x\ x\ x\ |\ x\ x\ x\ x)$$

$$= (0\ 1\ |\ 1\ 1\ 1\ 1\ 1\ 0\ |\ 0\ 0\ 0\ 1)$$

$$= C_M(E_k)$$

The 0-th block (0 1) of $C_M(E_k)$ is added to the whole blocks in Decoder II and (0 0 | 1 0 1 0 1 1 | 0 1 0 0) is obtained. The original data (1 0 1 0 1 1) is directly obtained from the output of Decoder II.

If there is a single random bit error in C_r, Decoder III can correct this bit error because $C_M(E_k)$ are codewords of C'. Thus any single stuck block error and single random bit error can be corrected, even though they occur simultaneously.

5. Conclusion

Memory systems constructed from high-density memory devices invariably yield stuck errors due to hardware defects in a memory device, in addition to random errors. Although random error correcting codes can correct both types of error, the required redundancy is larger than that of stuck error correcting codes[1]-[5] since error information about stuck error may be available.

If the memory system employs the b-bit-per-card configuration, stuck errors will be block errors. The subcode of additive codes[4][5], C_s, is discussed for the correction of single stuck block errors and random bit errors, which may occur simultaneously.

Assume that no memory device has stuck faults in the initial period and that some device becomes defective. In this case, C_s will not show a good performance since error information must be known before write data are encoded. If no error information, is known, all stuck block errors must be corrected as random block errors. The modified stuck block error correcting code C_M discussed here can correct any single random block errors until stuck errors are found and has the same error capability of C_s after stuck block errors are observed and error information is obtained.

References

[1] S. A. Szygenda and M. J. Flynn, "Self-Diagnosis and Self-Repair in Memory: An Integrated System Approach", IEEE Trans. Periability. vol. R-22, pp 2-12, April 1973.

[2] W. C. Cater and C. E. McCarthy,"Implementation of an Experimental Fault-Tolerant Memory System", IEEE Trans. Comput., vol. C-25, pp 557-568, June 1976.

[3] M. Y. Hsiao and D. C. Bossen, "Orthogonal Latin Square Configuration For LSI Memory Yield and Reliability Enhancement", IEEE Trans. Comput., vol. C-34, pp 512-516, May 1975.

[4] Y. Toma and M. Imae, 1359, Proceedings of IECE Japan Conf., 1976.

[5] A. V. Kuznetsov, T. Kasami, and S. Yamamura, "A Scheme of Error Correction", IECE Japan Trans., AL77-11, pp 93-100, 1977.

[6] T. Kasami et al. Coding Theory. Corona Publishing, Tokyo, 1975.

[7] W. W. Peterson and E. J. Weldon, Jr., Error Correcting Codes. Second Edition. MIT Press. Cambridge. Mass., 1972.

[8] M. Y. Hsiao and J. T. Tou, "Application of Error-Correcting Code in Computer Reliability Studies",IEEE Trans. Periability, vol. R-18, pp 108-118, August 1969.

[9] D. C. Bossen, "b-Adjacent Error Correction", IBM J. Res. Develop., pp 402-408, July 1970.

[10] H. Imai and Y. Kamiyanagi, "On Type-B2 Burst-Error-Correcting Block Code", IECE Japan Trans., AL76-54, pp 27-36,1976.

[11] J. K. Wolf, "On an Extended Class of Error-Locating Codes", Inform. and Control, No. 8, pp 163-169, 1965.

[12] E. Fujiwara and S. Kaneda, "On SbEC-DbEC Codes For Main Strage", IECE Japan Trans., EC77-1, pp 1-12, 1977.

[13] E. Fujiwara, "A Modularized b-Adjacent Error Correction Memory Unit", IECE Japan Trans., E60-2, pp 69-76, 1977.

[14] H. T. Hsu, T. Kasami, and R. T. Chien, "Error-Correcting Codes For a Compound Channel", IEEE Trans. Information Theory, vol. IT-14, pp 135-139, January 1968.

Vita

Takeo Kanai was born in Hyogo, Japan on August 11, 1951. He graduated from Kyoto University in 1975 and received his Bachelor of Science in Electrical Engineering. From 1975 to 1978, he worked at Mitsubish Electric Corp., Japan.

He came to the United States in August of 1978 and entered the Lehigh University Graduate School in September of 1978.