

Construction Techniques for Systematic SEC-DED Codes with Single Byte Error Detection and Partial Correction Capability for Computer Memory Systems

Luca Penzo, Donatella Sciuto, *Member, IEEE*,
and Cristina Silvano, *Member, IEEE*

Abstract—This correspondence proposes three new techniques to construct a class of codes that extends the protection provided by previous single error correcting (SEC)–double error detecting (DED)–single byte error detecting (SBD) codes. The proposed codes are systematic odd-weight-column SEC-DED-SBD codes providing also the correction of any odd number of erroneous bits per byte, where a byte represents a cluster of b bits of the codeword that are fed by the same memory chip or card. These codes are useful for practical applications to enhance the reliability and the data integrity of byte-organized computer memory systems against transient, intermittent, and permanent failures. In particular, they represent a good tradeoff between the overhead in terms of additional check bits and the reliability improvement, due to the capability to correct at least 50% of the multiple errors per byte.

Index Terms—Binary linear block codes, byte errors, computer memory systems, error control codes, memory reliability, SEC-DED-SBD codes.

I. INTRODUCTION

The requirements of higher levels of system reliability and data integrity in computer memory systems have increased the use of error correcting codes (ECC's). Codes represent an effective means of providing protection against transient, intermittent, and permanent failures ([1]–[4]), thus permitting the design of fault-tolerant memory subsystems.

The single error correcting–double error detecting codes (SEC-DED) have been widely used to increase reliability in memories organized in one-bit-per-chip fashion ([5]–[9], [17]). Such memory organization is characterized by the fact that each bit of a codeword is stored in a different memory chip. Thus any fault in a memory chip, regardless of the physical defect mode, can affect, at most, a single bit of the codeword.

Unfortunately, the technology trend in standard memory devices moves toward higher levels of integration, thus implying bigger and denser memory chips in which multiple-bit per-chip organization is necessary, mainly to support system memory granularity ([10]). In this case, memory failures can corrupt the chip in different modes thus generating different errors. For instance, a single memory cell failure can generate a single bit error, while other types of faults can affect multiple bits. The distribution of failure modes represents a major factor in the definition of the error control codes necessary to effectively protect a memory subsystem in order to reach the required levels of reliability.

More recent computer systems adopt a b -bit-per-chip memory organization, where $b \geq 2$ ([17], [22]). In these systems, a chip failure can generate a one-to- b bit error or “byte error,” depending

on the type of failure of the chip: cell, word-line, bit-line, partial chip, or total chip. The conventional SEC-DED codes are not suitable for providing effective reliability in b -bit-per-chip organizations, since multiple bit errors are not correctable, and worse in some cases they can be miscorrected, thus losing data integrity. The uncorrectable error rates could become very high in case of a memory chip failure distribution resulting in a large number of multiple bit errors ([10]). In general, there is a tradeoff between error control capability and efficiency: the higher the reliability required, the larger is the necessary redundancy.

To increase data integrity and reduce uncorrectable error rates, several codes with byte error control capabilities have been proposed in literature ([10]–[16], [18]–[20]). Single error correcting–single byte error detecting (SEC-SBD) codes (indicated also as SEC-S b ED where b is the byte length) have been introduced in [11], [12], [16]. Errors are corrected if they are single random errors, but the class of detectable errors includes also double random errors (SEC-DED-SBD) in [13] and, for byte lengths b greater than or equal to 5, in [12]. SEC-DED-SBD codes have been proposed also in [20] for $b = 4$, in [18] for $b \geq 5$, in [16] for $b \geq 7$, and in [19] for even byte length. Other approaches ([10], [14], [15]) have presented single byte error correcting–double byte error detecting (SBC-DBD) codes (indicated also as S b EC-D b ED). However, the construction of optimal SEC-DED-SBD codes or SBC-DBD codes for the general case with the minimum number of check bits is still an open and challenging problem.

The present correspondence introduces an extension to previous published papers on SEC-DED-SBD codes. The proposed approach constructs systematic odd-weight-column SEC-DED-SBD codes in which the class of correctable errors includes also any odd number of erroneous bits per byte. This additional capability increases the level of reliability of a computer memory system with respect to those systems employing the conventional SEC-DED-SBD code.

However, to support such possibility, an overhead in terms of the number of check bits is required. Table I provides a first comparison among the code proposed in the rest of this correspondence and the SEC-DED-SBD codes presented in [12], [13], [18], [20] and the SBC-DBD codes reported in [9], [10], [14], [17], and [22]. Each table entry represents the check bits lengths (r) for some practical values of byte lengths (b) and data bit lengths (k) for the specified class of codes. In particular for SBC-DBD codes each entry is the minimum check bit length required by the corresponding codes. The mathematical formulas for computing the check bit lengths for the proposed codes are fully derived in the following sections. As shown in Table I, a few additional check bits (at most four) are required for the proposed code with respect to SEC-DED-SBD codes in order to correct at least 50% of the possible multiple errors per byte. On the other hand, such overhead is reasonable if compared with the requirements in terms of check bits of SBC-DBD codes (almost half for $b > 8$ for the data in Table I).

The proposed codes have been designed to be applied in a multiprocessor computer system to achieve high reliability in the communication between processors and the memory subsystem. Therefore such codes should allow, in addition to high reliability, the implementation of VLSI encoding/decoding circuits with high performances in terms of speed/area ratio together with a short design turn-around time. These goals have been achieved by constructing codes belonging to the class of systematic odd-weight-column codes with modular structure. The use of systematic codes/separable codes allows the encoding/decoding and the data processing to be performed in parallel, thus speeding up the data exchange between memories

Manuscript received March 23, 1994; revised September 19, 1994. The material in this correspondence was presented in part at the Eighth International Conference on VLSI Design, New Delhi, India, January 4–7, 1995.

L. Penzo is with Politecnico di Milano, Dipartimento di Elettronica, 20133 Milano, Italy, and Bull HN Information Systems, Italia, Technology and Hardware Methodology Department, 20010 Pregnana M. (Milano), Italy.

D. Sciuto is with Politecnico di Milano, Dipartimento di Elettronica, 20133 Milano, Italy.

C. Silvano is with Bull HN Information Systems Italia, Technology and Hardware Methodology Department, 20010 Pregnana M. (Milano), Italy.

IEEE Log Number 9408645.

TABLE I
COMPARISON AMONG CHECK BIT LENGTHS OF SOME CLASSES OF BYTE ERROR CONTROL CODES WITH DATA BIT LENGTHS $k = 16, 32, 64, 128$ AND 256

k \ b	16					32					64					128					256				
	A	B	C	D	E	A	B	C	D	E	A	B	C	D	E	A	B	C	D	E	A	B	C	D	E
3	5	6	-	6	9	6	7	-	8	12	7	8	-	8	12	8	9	-	9	12	9	10	-	10	15
4	6	6	-	8	12	7	7	-	8	12	8	8	-	9	14	9	9	-	10	16	10	10	-	11	16
5	7	7	7	9	15	8	8	8	9	15	8	9	8	10	15	9	9	9	10	15	10	10	10	12	20
6	8	8	8	9	18	8	8	8	10	18	9	9	9	11	18	10	10	10	12	18	11	11	11	12	18
7	9	9	9	10	21	9	9	9	11	21	10	10	10	12	21	11	11	10	13	21	12	11	11	14	21
8	10	10	9	11	24	10	10	10	12	24	11	10	10	13	24	12	11	11	14	24	13	12	12	15	24
9	10	11	10	12	27	11	11	11	13	27	12	11	11	14	27	13	12	12	14	27	13	13	12	15	27
10	11	12	11	13	30	12	12	12	14	30	13	12	12	14	30	13	13	13	15	30	14	14	13	16	30
11	12	13	12	14	33	13	13	13	14	33	13	13	13	15	33	14	13	13	16	33	15	14	14	17	33
12	13	14	13	15	36	14	14	14	15	36	14	14	14	16	36	15	14	14	17	36	16	15	15	18	36
13	14	15	14	16	39	15	15	15	16	39	15	15	15	17	39	16	15	15	18	39	17	16	16	19	39
14	15	16	15	17	42	16	16	16	17	42	16	16	16	18	42	17	16	16	19	42	18	17	17	20	42
15	16	17	16	18	45	17	17	17	18	45	17	17	17	19	45	18	17	17	20	45	19	18	18	21	45
16	17	18	17	18	48	18	18	18	19	48	18	18	18	20	48	19	18	18	21	48	20	19	18	22	48

A: SEC-SBD codes for $b=3, 4$ and SEC-DED-SBD codes for $b \geq 5$ proposed in [12];
 B: SEC-DED-SBD codes proposed in [13] and also in [20] for $b=4$;
 C: SEC-DED-SBD codes for $b \geq 5$ proposed in [18];
 D: Codes proposed in this correspondence;
 E: SBC-DBD codes reported in [9], [10], [14], [17], and [22].

and processors, since it eliminates the need for data extraction after decoding and the need for encoding during write operations to the memory. The code modularity allows the organization of the encoding/decoding circuits as two or more identical logic modules thus simplifying the process of automatic generation of the circuits and their physical design, reducing both design time and chip area. Finally, the complexity of the parity check circuits required by the given codes can be roughly estimated by examining the structure of the parity check matrix H . The fastest generation of check and syndrome bits can be obtained by the systematic odd-weight-column SEC-DED codes with the additional byte errors correction/detection capabilities ([6], [21]) thus allowing to satisfy the performance constraints with a low overhead, as shown in Table I.

This correspondence is organized as follows. Section II gives some preliminary notations along with a few necessary definitions. In particular, necessary and sufficient conditions to be satisfied by the class of linear block codes, which includes SEC-DED-SBD codes, are introduced.

Section III presents the techniques defined for the construction of the parity check matrix used to describe the code. In particular, three construction techniques, indicated as C_1 , C_2 , and C_3 are proposed and the relation that gives the code length as a function of the number of bits per byte and the number of check bits has been formally derived for the three techniques. Section IV shows some illustrative examples, significant to computer applications. Finally, applications considerations and future developments conclude the correspondence.

II. PRELIMINARIES

In this section some known concepts and notations from coding theory are introduced ([7]). Note that, in the remainder of the correspondence, all vectors and matrices have entries from the binary field $GF(2)$ and all operations are performed over this field, unless otherwise specified.

A binary linear block code C can be described as the null space of a binary vector space generated by the row vectors of an $(r \times n)$ matrix called parity check matrix and indicated as $H_{(r \times n)}$, with n being the length of the codeword composed of k data bits and r check bits. A n -bit row vector X is a codeword in C if and only if

$$HX^T = 0 \tag{1}$$

where X^T denotes the transpose of X . When the H matrix is expressed as

$$H = [BI_r] \tag{2}$$

where B is an $(r \times k)$ matrix and I_r the $(r \times r)$ identity matrix, then the code is called systematic. In this case the first k bits of the codeword can be designated as the data bits, and the last r bits as the check bits. The encoding process of systematic codes can be easily performed by deriving the i th check bit from the i th equation of the r equations (1).

A code C is said to be an odd-weight-column code ([6]) if there exist a parity check matrix for C composed entirely of column vectors of odd weight, where the weight of a vector is the number of its nonzero bits.

Let $S_{N^{(r \times 1)}} = HX'^T$ be the syndrome vector related to the n -bit row vector X' , where $X' = X \oplus E$ is a codeword read out of memory, X being the original codeword, E the error vector, and \oplus the bit by bit module 2 sum operator. The i th position of X' is in error if and only if the i th element of E is a 1.

Some necessary and sufficient conditions on the parity check matrix of a binary linear block code are given in the next theorems ([13], [12]).

Theorem 1: Let C be a binary linear block code of length n defined by the check matrix $H_{(r \times n)}$. The code C is an SEC-DED code if and only if the column vectors h_i ($i = 1, 2, \dots, n$) of H satisfy the following conditions:

- 1) $h_i \neq 0$
- 2) $h_i + h_j \neq 0$
- 3) $h_i + h_j + h_k \neq 0$

where i, j , and k are distinct integers of the set $[1, 2, \dots, n]$.

More generally, Theorem 2 gives the necessary and sufficient conditions to be satisfied by a binary linear block code to simultaneously correct all error patterns in a set E_1 and detect all error patterns in a set E_2 , where $E_1 \cap E_2 = \{0\}$.

Theorem 2: Let C be a binary linear block code of length n defined by the check matrix $H_{(r \times n)}$ and E_1, E_2 be two sets of error patterns where $E_1 \cap E_2 = \{0\}$. The code C simultaneously corrects all errors in E_1 and detects all errors in E_2 if and only if the parity check matrix H satisfies the following conditions:

- 1) $HE^T \neq 0 \forall E \in \{E_1 \cup E_2\}$
- 2) $HE_i^T \neq HE_j^T \forall E_i, E_j \in E_1$
- 3) $\forall E_j \in E_2$, there does not exist an $E_i \in E_1$ such that $HE_j^T = HE_i^T$.

III. CODE CONSTRUCTION TECHNIQUES

In this section, three code construction techniques are reported to obtain a class of systematic odd-weight-column SEC-DED-SBD codes which can also correct any odd number of erroneous bit per byte. Being SEC-DED codes, it is meaningless to consider a byte length b equal to 1 or 2; therefore, b greater than 2 is considered.

The parity check matrix $H_{(r \times n)}$ is in systematic form as in (2) with matrix $B_{(r \times k)}$ composed of a set of K matrices as

$$B_{(r \times k)} = [B_1 \ B_2 \ \dots \ B_i \ \dots \ B_K] \quad (3)$$

where the matrices B_i are nonzero distinct $(r \times b)$ matrices. By construction, the number K of nonzero distinct matrices B_i is equal to the number of data bytes in the codeword ($K = k/b$, where k is supposed to be a multiple of b , without any loss of generality).

The generic matrix B_i is composed of two submatrices.

The first submatrix is a $(b \times b)$ matrix having a single element equal to 1 for every row and column. In the following, this matrix is supposed to be the $(b \times b)$ identity matrix I_b , without any loss of generality. This structure allows us to unequivocally identify the single bit within the byte.

The second submatrix of B_i is a $(r - b \times b)$ matrix, indicated as H_i , with the property that its column vectors are equal to each other and the generic column is stated to be a nonzero $(r - b)$ -tuple of even weight over GF(2). A set of nonzero distinct matrices H_i can be obtained defining its generic column vector as one of all the possible distinct nonzero $(r - b)$ -tuples of even weight over GF(2). This part of B_i allows us to unequivocally identify the single byte.

The two submatrices composing B_i , indicated as I_b and H_i , have to be placed vertically in B_i following three different techniques depending on $(r - b)$ being equal, greater or less than b . In all cases, the code is an odd-weight-column code ([6]), in fact the column vectors corresponding to the check bits are one-weight columns and the column vectors corresponding to the data bits are odd-weight columns, being the sum of the generic even-weight column of H_i plus the generic one-weight column of I_b .

Furthermore, the code requires a number of check bits $r \geq b + 2$, where r is the number of rows in B , b the number of rows in I_b , and the number of rows in H_i is at least 2. Globally, the three proposed techniques allow to construct codes with the given properties for arbitrary code length (n) and byte length (b) within the range $r \geq b + 2$ and $b > 2$. In particular, the first technique (called

C_1) is applied for $r = 2b$, the second technique (C_2) for $r > 2b$, and the third technique (C_3) for $b + 2 \leq r < 2b$.

In C_1 , a first set of matrices B_i is obtained by placing all the possible H_i above a matrix I_b , while a second set of B_i is obtained by placing all the possible H_i below the I_b . In this way, the two sets of matrices are composed of the same number of matrices B_i .

In C_2 , to obtain a class of codes with the same properties as in C_1 , the two sets of matrices B_i are composed of a different number of matrices. A first set of matrices B_i is obtained by placing all the possible H_i above a matrix I_b , while a second set of B_i is defined by placing a subset of H_i below the I_b , choosing the subset of H_i such that the resulting column vectors of the first set of B_i are different from the column vectors of the second set of B_i .

Finally, to maintain the same properties in C_3 as in the previous cases, only a single set of B_i is obtained by placing the generic H_i above I_b or below I_b .

Once the set of matrices B_i is obtained as shown in the three constructions, a set of codes with the same properties can be simply derived by changing the position of the submatrices B_i in the matrix B defined by (3).

A detailed description of the structure of the matrices B_i for $i \in \{1, 2, \dots, K\}$ is proposed hereafter for the three constructions. The maximum number of data bytes in the codeword K as a function of (r, b) is derived, from which the maximum length of proposed codes can be easily derived as: $n(r, b) = bK(r, b) + r$. After each construction, a theorem states the properties of the proposed codes. These properties, in terms of error correction and detection, are the same in all three cases, while the maximum values of K as a function of (r, b) are different and given by

$$K(r, b) = 2^b - 2, \quad \text{for } C_1 \quad (4)$$

$$K(r, b) = 2^{r-b-1} + 2^{r-b-2} - 2, \quad \text{for } C_2 \quad (5)$$

$$K(r, b) = 2^{r-b-1} - 1, \quad \text{for } C_3. \quad (6)$$

Construction C_1

Assume $r = 2b$, let the matrix $B_{(r \times k)}$ in the (2) be defined as

$$B_{(r \times k)} = [B_1 \ B_2 \ \dots \ B_i \ \dots \ B_{K_0} \ B'_1 \ B'_2 \ \dots \ B'_i \ \dots \ B'_{K_0}]$$

where

$$B_i = \begin{bmatrix} H_i \\ I_b \end{bmatrix}, \quad \text{for } i \in [1, 2, \dots, K_0] \quad (7)$$

$$B'_i = \begin{bmatrix} I_b \\ H_i \end{bmatrix}, \quad \text{for } i \in [1, 2, \dots, K_0] \quad (8)$$

and I_b denotes the $(b \times b)$ identity matrix, H_i is a $(b \times b)$ matrix with its column vectors equal to each other and $K = 2K_0$.

Let c_i be the generic b -bit column vector of H_i and S_{b-1} the $(b - 1)$ -dimensional subspace of the finite field GF(2^b) composed of the 2^{b-1} distinct b -tuples of even weight over GF(2), it is possible to define a set of $(2^{b-1} - 1)$ vectors c_i and consequently a set of $(2^{b-1} - 1)$ matrices H_i corresponding to the $(2^{b-1} - 1)$ distinct nonzero b -tuples of S_{b-1} . Hence the function $K(r, b)$ is given by (4).

TABLE II
WEIGHT OF THE SYNDROME VECTOR CORRESPONDING TO THE SET OF ERROR PATTERN E_1 FOR CONSTRUCTION C_1

Error type	Syndrome vectors s_i	Weight of top b -rows of s_i	Weight of bottom b -rows of s_i	Notes
Single error check bits	s_{1A}	1	0	
Same	same	0	1	
Single error data bits given by (7)	s_{1B}	p	1	$p \in P$
Single error data bits given by (8)	s_{1C}	1	p	$p \in P$
Odd (d) error check byte	s_{0A}	d	0	$d \in D$
Same	same	0	d	$d \in D$
Odd (d) error data byte given by (7)	s_{0B}	p	d	$p \in P, d \in D$
Odd (d) error data byte given by (8)	s_{0C}	d	p	$p \in P, d \in D$

The subsets of integers P and D are defined in (9) and (10) respectively.

Hereafter the possible matrices H_i are reported for $b = 3$ and $b = 4$, where the index i has been chosen arbitrarily

$$\begin{aligned}
 & b = 3 \rightarrow K_0 = 3 \\
 & H_2 = \begin{bmatrix} 1 & 1 & 1 \\ 0 & 0 & 0 \\ 1 & 1 & 1 \end{bmatrix} \\
 & H_1 = \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \\
 & H_3 = \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix} \\
 & b = 4 \rightarrow K_0 = 7 \\
 & H_2 = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \end{bmatrix} \\
 & H_4 = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \end{bmatrix} \\
 & H_6 = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \end{bmatrix} \\
 & H_1 = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \end{bmatrix} \\
 & H_3 = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 \end{bmatrix} \\
 & H_5 = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \end{bmatrix} \\
 & H_7 = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \end{bmatrix}
 \end{aligned}$$

Theorem C_1 : For $r = 2b$ the codes constructed by C_1 are systematic odd-weight-column SEC-DED-SBD codes capable of correcting any odd number of erroneous bits per byte and $K(r, b)$ is given by (4).

Proof of Theorem C_1 : In order to prove that the proposed codes have the declared properties, it is necessary to apply Theorem 2, where the set E_1 is composed of all possible single-bit error patterns and all possible odd-bit per byte error patterns, while E_2 is composed of all possible double-bit error patterns and all possible even-bit per byte error patterns. To apply Theorem 2, the syndromes corresponding to the two sets of error patterns E_1 and E_2 have to be derived.

Considering E_1 , each r -bit syndrome vector s_1 associated with a single-bit error pattern is equal to the corresponding column vector h_i of the parity check matrix H and each syndrome vector s_0 associated with an odd number of erroneous bits per byte is equal to the bit by bit module 2 sum of the column vectors h_i of H corresponding to the erroneous bits per byte.

The set S_1 of the syndrome vectors s_1 can be subdivided into three subsets as follows:

$$S_1 = S_{1A} \cup S_{1B} \cup S_{1C}$$

where S_{1A} is composed of the nonzero distinct vectors s_{1A} corresponding to a single error in the check bits of the codeword; S_{1B} is composed of the nonzero distinct vectors s_{1B} corresponding to a single error in the data bits of the codeword given by (7); S_{1C} is composed of the nonzero distinct vectors s_{1C} corresponding to a single error in data bits of the codeword given by (8).

The set S_0 of the syndrome vectors s_0 can be partitioned into three subsets as follows:

$$S_0 = S_{0A} \cup S_{0B} \cup S_{0C}$$

where S_{0A} is composed of the nonzero distinct vectors s_{0A} corresponding to odd (≥ 3) numbers of erroneous bits per check byte of the codeword; S_{0B} is composed of the nonzero distinct vectors s_{0B} corresponding to odd (≥ 3) numbers of erroneous bits per data byte of the codeword given by (7); S_{0C} is composed of the nonzero distinct vectors s_{0C} corresponding to odd (≥ 3) number of erroneous bits per data byte of the codeword obtained from (8).

Let P be the subset of the even positive integers in the range from 2 to $2\lfloor b/2 \rfloor$ (where $\lfloor x \rfloor$ denotes the integer part of x and sums and products are integer operations) defined as

$$P = \{p = 2y | y \in [1, 2, \dots, \lfloor b/2 \rfloor]\} \quad (9)$$

and D be the subset of the odd positive integers defined as

$$D = \{d = 2y + 1 | y \in [1, 2, \dots, \lfloor (b-1)/2 \rfloor]\}. \quad (10)$$

The weight of the syndrome vectors (i.e., the number of nonzero bits in the syndrome vectors) belonging to the set $S_1 \cup S_0$, can be schematically represented as shown in Table II. In particular the table gives separately the weight of the top and bottom b -rows of the syndrome vectors corresponding to each type of error patterns in E_1 , in order to show that each syndrome vector is a nonzero vector that differs not only from the other vectors of the same set, but also from the vectors of the other syndrome sets. Globally the syndrome vectors corresponding to E_1 are nonzero and distinct to each other, hence the properties 1) and 2) of Theorem 2 are satisfied for E_1 .

Considering the error patterns belonging to the set E_2 , note that the corresponding syndrome vectors are nonzero even-weight r -tuples, being the sum of an even positive integer number of nonzero distinct odd-weight r -tuples. Thus comparing the syndromes corresponding

to the sets E_1 and E_2 , it is easy to verify that property 3) of Theorem 2 is also satisfied. Q.E.D.

Construction C_2

This construction technique assumes $r > 2b$ and determines the structure of the corresponding matrix $B_{(r \times k)}$ defined as follows:

$$B_{(r \times k)} = [B_1 \ B_2 \ \cdots \ B_i \ \cdots \ B_{K_0} \ B'_1 \ B'_2 \ \cdots \ B'_j \ \cdots \ B'_{K_1}]$$

where

$$B_i = \begin{bmatrix} H_i \\ I_b \end{bmatrix}, \quad \text{for } i \in [1, 2, \dots, K_0] \quad (11)$$

$$B'_j = \begin{bmatrix} I_b \\ H_j \end{bmatrix}, \quad \text{for } j \in [1, 2, \dots, K_1] \quad (12)$$

and I_b denotes the $(b \times b)$ identity matrix, H_i and H_j are $(r-b \times b)$ matrices with their column vectors equal to each other and $K = K_0 + K_1$.

Let c_i be the generic $(r-b)$ -bit column vector of H_i and S_{r-b-1} the subspace of $\text{GF}(2^{r-b})$ composed of the 2^{r-b-1} distinct $(r-b)$ -tuples of even weight over $\text{GF}(2)$, it is possible to define a set of $(2^{r-b-1} - 1)$ vectors c_i and consequently a set of $(2^{r-b-1} - 1)$ matrices H_i corresponding to the $(2^{r-b-1} - 1)$ distinct nonzero $(r-b)$ -tuples of S_{r-b-1} , hence the function $K_0(r, b)$ is given by

$$K_0(r, b) = 2^{r-b-1} - 1.$$

The generic matrix $H_j(r-b \times b)$ has the following structure:

$$H_j = \begin{bmatrix} M_m \\ N_n \end{bmatrix} \begin{matrix} r-1 \\ \vdots \\ b \\ \vdots \\ 1 \end{matrix} 2b$$

where M_m and N_n are an $(r-2b \times b)$ matrix and a $(b \times b)$ matrix, respectively, and each with its column vectors equal to each other.

As before, it is possible to define two sets of distinct matrices M_m and N_n in the following way. Let c_m be the generic $(r-2b)$ -bit column vector of M_m and S_{r-2b-1} the subspace of $\text{GF}(2^{r-2b})$ composed of the 2^{r-2b-1} distinct $(r-2b)$ -tuples of even weight over $\text{GF}(2)$, it is possible to define a set of 2^{r-2b-1} vectors c_m and, consequently, a set of 2^{r-2b-1} matrices M_m corresponding to the 2^{r-2b-1} distinct $(r-2b)$ -tuples of S_{r-2b-1} , including the all-0's $(r-2b)$ -tuple.

A set of 2^{b-1} matrices N_n corresponding to the 2^{b-1} distinct b -tuples of S_{b-1} , including the all-0's b -tuple, can be derived in a likewise fashion.

Now let H_j be defined considering all possible pairs of M_m , N_n except for the pair generating the all-0's matrix. The number of distinct matrices H_j obtainable following this construction is $(2^{r-2b-1} 2^{b-1} - 1)$, hence the function $K_1(r, b)$ can be presented as

$$K_1(r, b) = 2^{r-b-2} - 1.$$

In conclusion, the function $K(r, b)$ is given by (5).

Theorem C_2 : For $r > 2b$ the codes constructed in C_2 are systematic odd-weight-column SEC-DED-SBD codes capable of correcting any odd number of erroneous bits per byte and $K(r, b)$ is given by (5).

The proof of Theorem C_2 is found in the Appendix.

Note C_2 : Construction C_2 requires $r > 2b$ and, when r is not a multiple b , we have $\lfloor r/b \rfloor$ check bytes of b -bit length and one check byte of $(r - \lfloor r/b \rfloor b)$ -bit length. Without loss of generality, it is possible to assume that the first and the last b -bit of the check bits belong to a check byte of length b -bit.

Construction C_3

This construction method assumes $b+2 \leq r < 2b$ and derives the matrices $B_{i(r \times b)}$ for the corresponding code. Let the matrices $B_{i(r \times b)}$ in (3) be defined as

$$B_i = \begin{bmatrix} I_b \\ H_i \end{bmatrix}, \quad \text{for } i \in [1, 2, \dots, K] \quad (13)$$

or

$$B_i = \begin{bmatrix} H_i \\ I_b \end{bmatrix}, \quad \text{for } i \in [1, 2, \dots, K] \quad (14)$$

where I_b denotes the $(b \times b)$ identity matrix and H_i is an $(r-b \times b)$ matrix with its column vectors equal to each other.

Let c_i be the generic $(r-b)$ -bit column vector of H_i and S_{r-b-1} the subspace of $\text{GF}(2^{r-b})$ composed of the 2^{r-b-1} distinct $(r-b)$ -tuples of even weight over $\text{GF}(2)$, it is possible to define a set of $(2^{r-b-1} - 1)$ vectors c_i and consequently a set of $(2^{r-b-1} - 1)$ matrices H_i corresponding to the $(2^{r-b-1} - 1)$ distinct nonzero $(r-b)$ -tuples of S_{r-b-1} . Therefore, the function $K(r, b)$ can be represented as defined in (6).

Theorem C_3 : For $b+2 \leq r < 2b$ the codes constructed in C_3 are systematic odd-weight-column SEC-DED-SBD codes capable of correcting any odd number of erroneous bits per byte and $K(r, b)$ is given by (6).

The proof of Theorem C_3 is given in the Appendix.

Note C_3 : Construction C_3 requires $b+2 \leq r < 2b$, thus one check byte has b -bit length and one check byte has $(r-b)$ -bit length. Without any loss of generality, it is possible to assume that the first (last) b -bit of the check bits belong to the check byte of length b -bit if the set of matrices B_i has been chosen as in (13) (in (14)).

IV. ILLUSTRATIVE EXAMPLES

In this section the construction techniques shown above have been applied to generate some examples of parity check matrices.

Example 1: From Theorem C_1 a SEC-DED-SBD code, that can also correct any 3-tuples of erroneous bits per byte, can be constructed with $b=4$, $r=8$, and $K=14$. Construction C_1 can be applied to yield the following parity check matrix $H_{1(8 \times 64)}$ of the code:

$$H_1 = \begin{bmatrix} 1111 & 1111 & 1111 & 0000 & 0000 & 0000 & 1111 & 1000 \\ 1111 & 0000 & 0000 & 1111 & 1111 & 0000 & 1111 & 0100 \\ 0000 & 1111 & 0000 & 1111 & 0000 & 1111 & 1111 & 0010 \\ 0000 & 0000 & 1111 & 0000 & 1111 & 1111 & 1111 & 0001 \\ 1000 & 1000 & 1000 & 1000 & 1000 & 1000 & 1000 & 1111 \\ 0100 & 0100 & 0100 & 0100 & 0100 & 0100 & 0100 & 1111 \\ 0010 & 0010 & 0010 & 0010 & 0010 & 0010 & 0010 & 0000 \\ 0001 & 0001 & 0001 & 0001 & 0001 & 0001 & 0001 & 0000 \\ 1000 & 1000 & 1000 & 1000 & 1000 & 1000 & 1000 & 0000 \\ 0100 & 0100 & 0100 & 0100 & 0100 & 0100 & 0100 & 0000 \\ 0010 & 0010 & 0010 & 0010 & 0010 & 0010 & 0010 & 0000 \\ 0001 & 0001 & 0001 & 0001 & 0001 & 0001 & 0001 & 0000 \\ 1111 & 1111 & 0000 & 0000 & 0000 & 1111 & 0000 & 1000 \\ 0000 & 0000 & 1111 & 1111 & 0000 & 1111 & 0000 & 0100 \\ 1111 & 0000 & 1111 & 0000 & 1111 & 1111 & 0000 & 0010 \\ 0000 & 1111 & 0000 & 1111 & 1111 & 1111 & 0000 & 0001 \end{bmatrix}$$

Given $b=4$ and $r=8$, this code controls the maximum number of data bits (56). From this code, some other codes, controlling a number of data bits less than 56, can be simply derived by discarding some of the submatrices $B_{i(8 \times 4)}$. The choice of which one of these matrices is to be discarded follows the criterion of having the fastest encoding and error detection in the syndrome decoding process. In order to obtain the fastest encoding/decoding processes, it is necessary to minimize the number of 1's in each row of the H matrix. In fact,

TABLE III
WEIGHT OF THE SYNDROME VECTOR CORRESPONDING TO THE SET OF ERROR PATTERN E_1 FOR CONSTRUCTION C_2

Error type	Syndrome vectors s_i	Weight of top b -rows of s_i	Weight of middle $(r-2b)$ -rows of s_i	Weight of bottom b -rows of s_i	Notes
Single error check bits	s_{1A}	1	0	0	
Same	same	0	1	0	
Same	same	0	0	1	
Single error data bits given by (11)	s_{1B}	p''	p'	1	$p' \in P'$ $p'' \in P''$ $p'+p'' \geq 2$
Same	same	d_1	d'_1	1	$d'_1 \in D'_1$ $d_1 \in D_1$ $d'_1+d_1 \geq 2$
Single error data bits given by (12)	s_{1C}	1	p'	p''	$p' \in P'$ $p'' \in P''$ $p'+p'' \geq 2$
Odd (d) error check byte*	s_{0A}	d	0	0	$d \in D$
Same	same	0	d'	0	$d' \in D'$
Same	same	0	0	d	$d \in D$
Odd (d) error data byte given by (11)	s_{0B}	p''	p'	d	$p' \in P'$ $p'' \in P''$ $p'+p'' \geq 2$ $d \in D$
Same	same	d_1	d'_1	d	$d'_1 \in D'_1$ $d_1 \in D_1$ $d'_1+d_1 \geq 2$ $d \in D$
Odd (d) error data byte given by (12)	s_{0C}	d	p'	p''	$p' \in P'$ $p'' \in P''$ $p'+p'' \geq 2$ $d \in D$

The subsets of integers $D, P, P', P'', D', D'_1, D_1$ are defined in (10) and (A1) - (A6).
* See note C_2 .

the total number of 1's in each row of H is related to the number of logic levels required to generate the check bits and syndrome bits of that row ([6]).

Following this criterion, a set of matrices $H_{(8 \times 40)}$ can be derived from H_1 , having 32 data bits and a number of 1's in the rows of H equal to 13, that corresponds also to the minimum number given by the optimal minimum odd-weight-column SEC-DED codes ([6]). One of these matrices is the following $H_{2(8 \times 40)}$ matrix, that has already been reported in [9], as an example of SEC-DED-SBD code for $b = 4$ and $k = 32$. This code was derived in [9] from a SEC-DED code through the application of the column reconfiguration method to detect also single byte errors.

$$H_2 = \begin{bmatrix} 1111 & 0000 & 1111 & 0000 & 1000 & 1000 & 1000 \\ 0000 & 1111 & 0000 & 1111 & 0100 & 0100 & 0100 \\ 1111 & 1111 & 0000 & 0000 & 0010 & 0010 & 0010 \\ 0000 & 0000 & 1111 & 1111 & 0001 & 0001 & 0001 \\ 1000 & 1000 & 1000 & 1000 & 1111 & 1111 & 0000 \\ 0100 & 0100 & 0100 & 0100 & 0000 & 0000 & 1111 \\ 0010 & 0010 & 0010 & 0010 & 1111 & 0000 & 1111 \\ 0001 & 0001 & 0001 & 0001 & 0000 & 1111 & 0000 \\ 1000 & 1000 & 0000 & & & & \\ 0100 & 0100 & 0000 & & & & \\ 0010 & 0010 & 0000 & & & & \\ 0001 & 0001 & 0000 & & & & \\ 0000 & 0000 & 1000 & & & & \\ 1111 & 0000 & 0100 & & & & \\ 0000 & 0000 & 0010 & & & & \\ 1111 & 0000 & 0001 & & & & \end{bmatrix}$$

Example 2: From Theorem C_2 a SEC-DED-SBD code, that can also correct any 3-tuples of erroneous bits per byte, can be constructed with $b = 3$, $r = 8$, and $K = 22$. Construction C_2 can be applied to yield a parity check matrix $H_{(8 \times 74)}$ of the code:

$$H_3 = \begin{bmatrix} 111 & 111 & 111 & 111 & 000 & 000 & 000 & 000 & 000 \\ 111 & 000 & 000 & 000 & 111 & 111 & 111 & 000 & 000 \\ 000 & 111 & 000 & 000 & 111 & 000 & 000 & 111 & 111 \\ 000 & 000 & 111 & 000 & 000 & 000 & 111 & 000 & 111 & 000 \\ 000 & 000 & 000 & 111 & 000 & 000 & 000 & 111 & 000 & 111 \\ 100 & 100 & 100 & 100 & 100 & 100 & 100 & 100 & 100 & 100 \\ 010 & 010 & 010 & 010 & 010 & 010 & 010 & 010 & 010 & 010 \\ 001 & 001 & 001 & 001 & 001 & 001 & 001 & 001 & 001 & 001 \\ 000 & 111 & 111 & 111 & 111 & 000 & 100 & 100 & 100 & 100 \\ 000 & 111 & 111 & 111 & 000 & 111 & 010 & 010 & 010 & 010 \\ 000 & 111 & 111 & 000 & 111 & 111 & 001 & 001 & 001 & 001 \\ 111 & 111 & 000 & 111 & 111 & 111 & 000 & 000 & 000 & 000 \\ 111 & 000 & 111 & 111 & 111 & 111 & 000 & 000 & 000 & 000 \\ 100 & 100 & 100 & 100 & 100 & 100 & 111 & 111 & 111 & 000 \\ 010 & 010 & 010 & 010 & 010 & 010 & 010 & 111 & 000 & 111 \\ 001 & 001 & 001 & 001 & 001 & 001 & 000 & 111 & 111 & 111 \\ 100 & 100 & 100 & 100 & 100 & 00 & 000 & & & \\ 010 & 010 & 010 & 010 & 010 & 00 & 000 & & & \\ 001 & 001 & 001 & 001 & 001 & 001 & 00 & 000 & & \\ 111 & 111 & 111 & 111 & 000 & 10 & 000 & & & \\ 111 & 111 & 111 & 111 & 000 & 01 & 000 & & & \\ 000 & 111 & 111 & 000 & 000 & 00 & 100 & & & \\ 000 & 111 & 000 & 111 & 000 & 00 & 010 & & & \\ 000 & 000 & 111 & 111 & 000 & 00 & 001 & & & \end{bmatrix}$$

TABLE IV
WEIGHT OF THE SYNDROME VECTOR CORRESPONDING TO THE SET OF ERROR PATTERN E_1 FOR CONSTRUCTION C_3

Error type	Syndrome vectors s_i	Weight of top b -rows of s_i	Weight of bottom $(r-b)$ -rows of s_i	Notes
Single error check bits	s_{1A}	1	0	
Same	same	0	1	
Single error data bits given by (13)	s_{1B}	1	p	$p \in P$
Odd (d) error check byte*	s_{0A}	d	0	$d \in D$
Same	same	0	d'	$d' \in D'$
Odd (d) error data byte given by (13)	s_{0B}	d	p	$p \in P, d \in D$

The subsets of integers D , P and D' are defined in (10), (A1) and (A7), respectively
* See Note C_3

Example 3: From Theorem C_3 a SEC-DED-SBD code, that can also correct any odd number of erroneous bits per byte, can be constructed with $b = 8$, $r = 12$, and $K = 7$. Construction C_3 can be applied to obtain a parity check matrix $H_{(12 \times 68)}$. Using the same criterion as that in Example 1, it is possible to extract from this matrix a set of matrices $H_{(12 \times 44)}$, having 32 data bits in the codeword. One of the set of submatrices is

$$H_4 = \begin{bmatrix} 11111111 & 11111111 & 00000000 & 00000000 \\ 11111111 & 00000000 & 11111111 & 00000000 \\ 00000000 & 00000000 & 11111111 & 11111111 \\ 00000000 & 11111111 & 00000000 & 11111111 \\ 10000000 & 10000000 & 10000000 & 10000000 \\ 01000000 & 01000000 & 01000000 & 01000000 \\ 00100000 & 00100000 & 00100000 & 00100000 \\ 00010000 & 00010000 & 00010000 & 00010000 \\ 00001000 & 00001000 & 00001000 & 00001000 \\ 00000100 & 00000100 & 00000100 & 00000100 \\ 00000010 & 00000010 & 00000010 & 00000010 \\ 00000001 & 00000001 & 00000001 & 00000001 \\ 100000000000 \\ 010000000000 \\ 001000000000 \\ 000100000000 \\ 000010000000 \\ 000001000000 \\ 000000100000 \\ 000000010000 \\ 000000001000 \\ 000000000100 \\ 000000000010 \\ 000000000001 \end{bmatrix}$$

V. APPLICATION RESULTS AND CONCLUDING REMARKS

The proposed techniques allow us to generate a class of codes extending the protection provided by previous SEC-DED-SBD codes by constructing systematic odd-weight-column SEC-DED-SBD codes giving also the correction when the byte error pattern has odd weight.

Extension techniques are under development to increase the maximum length of the proposed codes. A first technique adds some submatrices to the parity check matrix corresponding to some extra data byte. The codes obtained with this extension technique are still systematic SEC-DED codes, but the property SBD is preserved only for the data bytes and not for the check bytes. This technique also intends to minimize the number of 1's in each row of the parity check matrix, to allow fast generation of check bits and syndrome

bits. Another technique extends a code generated with the proposed techniques to form a new code with the same error control capabilities but covering a double number of information bits using one more check bits.

Furthermore, the proposed codes are suitable for high performances VLSI implementations in computer applications, being systematic odd-weight-column and having modular structure ([21]). The proposed construction techniques have been effectively applied to design a 64 information bit-error control code integrated in a VLSI circuit developed by the R&D Labs of Bull HN Information Systems Italia for a multiprocessor computer system. This device interfaces the main memory and five data channels to processors and I/O's. Its complexity is about 55 000 equivalent gates with 11 500 internal nets and 202 logic pins. It was fabricated using the 0.7- μ m CMOS sea-of-gates technology supplied by LSI Logic Corporation.

APPENDIX

Proof of correctness for Theorem C_2 and C_3 are contained in this Appendix.

Proof of Theorem C_2 : The complete proof of Theorem C_2 is omitted, since it is quite similar to the proof of Theorem C_1 . Given the following subsets:

$$P = \{p = 2y | y \in [1, 2, \dots, \lfloor (r-b)/2 \rfloor]\} \quad (A1)$$

$$P' = \{p' = 2x | x \in [0, 1, 2, \dots, \lfloor (r-2b)/2 \rfloor]\} \quad (A2)$$

$$P'' = \{p'' = 2y | y \in [0, 1, 2, \dots, \lfloor b/2 \rfloor]\} \quad (A3)$$

D as defined in (10).

$$D' = \{d' = 2y + 1 | y \in [1, 2, \dots, \lfloor (r-2b-1)/2 \rfloor]\} \quad (A4)$$

$$D'_1 = D' \cup \{1\} \quad (A5)$$

$$D_1 = D \cup \{1\} \quad (A6)$$

the weight of the syndrome vectors, corresponding to the set E_1 , can be schematically represented as shown in Table III. In particular the table gives the weight of each set of syndrome vectors, corresponding to each type of error patterns in E_1 ; the global weight of the syndrome vectors is determined separately for its top b -rows, middle $(r-2b)$ -rows, and bottom b -rows. Q.E.D.

Proof of Theorem C_3 : The complete proof of Theorem C_3 is omitted, since it is quite similar to the proof of Theorem C_1 . Table IV reports the weights of the syndrome vectors corresponding to the set E_1 , P and D being the subsets defined in (10) and (A1), respectively, and

$$D' = \{d' = 2y + 1 | y \in [1, 2, \dots, \lfloor (r-b-1)/2 \rfloor]\}. \quad (A7)$$

Q.E.D.

REFERENCES

- [1] H. Kalter *et al.*, "A 50 ns 16 Mb DRAM with a 10 ns data rate," in *ISSCC Dig. Tech. Papers*, vol. 33, Feb. 1990, pp. 232-233.
- [2] H. L. Kalter, "A 50-ns 16-Mb DRAM with a 10-ns data rate and on-chip ECC," *IEEE J. Solid-State Circuits*, vol. 25, pp. 1118-1128, Oct. 1990.
- [3] K. Furutani *et al.*, "A built-in Hamming code ECC circuit for DRAM's," *IEEE J. Solid-State Circuits*, vol. 24, pp. 50-56, Feb. 1989.
- [4] T. Yamada *et al.*, "A 4-Mbit DRAM with 16-bit concurrent ECC," *IEEE J. Solid-State Circuits*, vol. 23, pp. 20-26, Feb. 1988.
- [5] R. W. Hamming, "Error detecting and error correcting codes," *Bell Syst. Tech. J.*, vol. 26, pp. 147-160, Apr. 1950.
- [6] M. Y. Hsiao, "A class of optimal minimum odd-weight-column SEC-DED codes," *IBM J. Res. Develop.*, pp. 395-401, July 1970.
- [7] S. Lin and D. J. Costello Jr., *Error Control Coding: Fundamentals and Applications*. Englewood Cliffs, NJ: Prentice Hall, 1983.
- [8] D. K. Pradhan, *Fault-Tolerant Computing: Theory and Techniques*, vol. 1. Englewood Cliffs, NJ: Prentice-Hall, 1986.
- [9] C. L. Chen and M. Y. Hsiao, "Error-correcting codes for semiconductor memory applications: A state-of-the-art review," *IBM J. Res. Develop.*, vol. 28, pp. 124-134, Mar. 1984.
- [10] C. L. Chen, "Error-correcting codes for byte-organized memory systems," *IEEE Trans. Inform. Theory*, vol. IT-32, Mar. 1986.
- [11] D. C. Bossen *et al.*, "Measurement and generation of error correcting codes for package failures," *IEEE Trans. Comput.*, vol. C-27, pp. 201-204, Mar. 1978.
- [12] S. M. Reddy, "A class of linear codes for error control in byte-per-card organized digital systems," *IEEE Trans. Comput.*, vol. C-27, pp. 455-459, May 1978.
- [13] C. L. Chen, "Error-correcting codes with byte error-detection capability," *IEEE Trans. Comput.*, vol. C-32, pp. 615-621, July 1983.
- [14] S. Kaneda and E. Fujiwara, "Single byte error correcting-double byte error detecting codes for memory systems," *IEEE Trans. Comput.*, vol. C-31, pp. 569-602, July 1982.
- [15] C. L. Chen, "Symbol error-correcting codes for computer memory systems," *IEEE Trans. Comput.*, vol. 41, pp. 252-256, Feb. 1992.
- [16] L. A. Dunning and M. R. Varanasi, "Code constructions for error control in byte organized memory systems," *IEEE Trans. Comput.*, vol. C-32, pp. 535-542, June 1983.
- [17] E. Fujiwara and D. K. Pradhan, "Error-control coding in computers," *Computer*, pp. 63-72, July 1990.
- [18] L. A. Dunning, "SEC-BED-DED codes for error control in byte-organized memory systems," *IEEE Trans. Comput.*, vol. C-34, pp. 557-562, June 1985.
- [19] W. E. Clark *et al.*, "The construction of some bit and byte error control codes using partial Steiner systems," *IEEE Trans. Inform. Theory*, vol. 35, pp. 1305-1310, Nov. 1989.
- [20] S. Kaneda, "A class of odd-weight-column SEC-DED-SbED Codes for memory system applications," *IEEE Trans. Comput.*, vol. C-33, pp. 737-739, Aug. 1984.
- [21] L. Penzo *et al.*, "VLSI design of systematic odd-weight-column byte error detecting SEC-DED codes," in *Proc. 8th Int. Conf. on VLSI Design* (New Delhi, India, 1995).
- [22] T. N. Rao and E. Fujiwara, *Error Control Coding for Computer Systems*. Englewood Cliffs, NJ: Prentice-Hall, 1989.

Bounded Minimum Distance Decoding of Unit Memory Codes

Uwe Dettmar and Ulrich K. Sorger

Abstract—In this paper we propose an algorithm for bounded minimum distance decoding of (partial) unit memory codes up to half the "designed" extended row distance. It makes use of a reduced trellis with the nodes found by bounded minimum distance decoding of the block codes used in the unit memory code. The results can be extended to general multimemory codes. The complexity of this algorithm is upper bounded by $2(\bar{d}_1 - 2d_\infty)$ times the complexity of the bounded minimum distance decoder of the block codes in the unit memory code. Here d_∞ is the linear increase of the designed extended row distance \bar{d}_1 .

Index Terms—(Partial) unit memory codes, bounded minimum distance decoding, extended row distance, convolutional codes.

I. INTRODUCTION

What is the meaning of a bounded minimum distance (BMD) decoder for (P)UM codes [1], [2] or convolutional codes? Naturally, a correct decoding up to half the free distance d_∞ must be guaranteed by such a decoder. However, that is not sufficient as a maximum-likelihood decoder is able to correct many error patterns of larger weight [3], as well. A BMD decoder should also correct some of these error patterns. There is another desirable property of such a BMD decoder: If an error event has occurred at a certain time t , the decoder must return to producing correct decisions after a relatively error poor sequence in the received word.

In [3], a BMD decoding algorithm for unit memory codes is given. This algorithm is based on a finite-state machine which represents all decodable error sequences as state transitions. This is possible as the number of decodable error patterns of minimum bounded weight is finite. Nondecodable sequences lead the decoder to a so-called dead state from which it returns after receiving a sequence with a small number of errors. To map the possible error patterns to the state transitions a syndrome former together with a table lookup is used. The machine starts at the beginning of the received sequence in state zero and sequentially uses the next received block to determine the next state transition. However, the application of the algorithm to codes with larger distances and to soft-decision decoding seems to be complicated.

The approach to BMD decoding proposed here uses block BMD decoders for the block codes that build the (P)UM code. This approach seems to be natural as most (P)UM codes are also constructed from block codes. We assume that BMD decoding of (P)UM codes is not far from the performance of maximum-likelihood decoding if the block length is short. In this case, the decoding domain of the BMD decoder should fill up a large fraction of the code space. Suboptimal decoders of low complexity are of great interest as some good (P)UM codes can be easily constructed [3]-[5] but not practically decoded by a maximum-likelihood decoder. The correspondence is organized as follows: in Section II, we recall the definitions of (P)UM codes and the extended row distance. In Section III, a BMD decoding algorithm for (P)UM codes that guarantees correct decoding up to half the

Manuscript received October 29, 1992; revised April 1, 1993. The material in this paper was presented at the 6th Swedish-Russian International Workshop on Information Theory, Moelle, Sweden, 1993.

The authors are with the Institut für Netzwerk- und Signaltheorie, Technische Hochschule Darmstadt, D-64283 Darmstadt, Germany.

IEEE Log Number 9408639.