

# VLSI Design of Systematic Odd-Weight-Column Byte Error Detecting SEC-DED Codes

Luca Penzo<sup>^\*</sup>, Donatella Sciuto<sup>^</sup>, Cristina Silvano<sup>\*</sup>

<sup>^</sup> Politecnico di Milano, Dipart. di Elettronica, P.za Leonardo da Vinci 32, 20133 Milano, Italy

<sup>\*</sup> Bull HN Information Systems Italia, R&D Lab., 20010 Pregnana M. (Milano), Italy

## Abstract

*The paper introduces an extension to previous single error correcting (SEC) - double error detecting (DED) - single byte error detecting (SBD) codes. The proposed approach constructs systematic odd-weight-column SEC-DED-SBD codes whose corrigible errors also include any odd number of erroneous bits per byte.*

*Main purpose of this paper is to show that the proposed codes are suitable for high performances VLSI implementations in computer applications, using high speed encoding/decoding circuits and parallel data manipulation. Furthermore, the paper will show how such codes can be easily designed from the specifications using a software tool, which generates the VHDL (VHSIC Hardware Description Language) description of the circuits.*

## 1. Introduction

In the actual computer applications, the memory sub-systems are commonly organized as  $b$ -bit-per-chip or  $b$ -bit-per-card. If a failure occurs, the resulting information read out of the memory is likely to have up to  $b$  erroneous bits within a byte, where a byte represents the cluster of  $b$  data bits that are fed by the same memory chip or card ([4]). In byte organized memory sub-systems, error control codes (ECCs) capable of correcting/detecting bit errors as well as byte errors are suitable to increase reliability and data integrity.

Several codes with byte error control capabilities have been proposed in literature ([5], [6], [7], [8], [9], [10], [11], [13], [14], [15]). 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 [6], [7], [11]. Errors are corrected if they are single random errors, but the class of detectable errors includes also double random errors (SEC-DED-SBD) in [8] and, for byte lengths  $b$  greater than or equal to 5, in [7]. SEC-DED-SBD codes have been proposed also in [15] for  $b=4$ , in [13] for  $b \geq 5$ , in [11] for  $b \geq 7$  and in [14] for even byte length. Other approaches ([5], [9], [10]) have presented single byte error correcting - double byte error detecting (SBC-DBD) codes (indicated also as S $b$ EC-DbED). 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 paper introduces an extension to previous SEC-DED-SBD codes. The proposed techniques construct systematic odd-weight-column SEC-DED-SBD codes in which the

class of correctable errors also includes 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 codes.

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 codes proposed in the rest of this paper and the SEC-DED-SBD codes presented in [7], [8], [13], [15] and the SBC-DBD codes reported in [4], [5], [9], [12] and [17]. 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 of check bits lengths required by the referred codes. As shown in Table I, a few additional check bits (at most four) are required to the proposed codes 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 sub-system. In addition to high reliability, the proposed codes are systematic. Thus the information bits are maintained separated from the check bits, so that the encoding/decoding and the data processing can be performed in parallel.

The main purpose of this paper is to demonstrate that the proposed codes are suitable for high performance VLSI implementations in computer applications, using high speed encoding/decoding circuitries and parallel data processing. These characteristics are mainly due to the fact that the codes belong to the class of systematic odd-weight-column codes with a modular structure. The paper will also show how this code can be easily designed from the specifications using a new software tool.

The paper is organized as follows. First Section II briefly describes the new codes construction techniques, then Section III shows the main advantages offered by the codes and how these codes can be implemented in VLSI circuits. Finally the automatic code generation tool is proposed in Section IV, while some application results are given in Section V.

## 2. New code construction techniques

A general description of the code construction techniques and the error detection/correction capabilities of the codes are

b \ k	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

**Table I: Comparison among check bit lengths of some class of byte error control codes with data bit lengths k = 16, 32, 64, 128 and 256.**

- A: SEC-SDB codes for b=3, 4 and SEC-DED-SBD codes for b ≥ 5 proposed in [7];  
 B: SEC-DED-SBD codes proposed in [8] and also in [15] for b = 4;

- C: SEC-DED-SBD codes for b ≥ 5 proposed in [13];  
 D: Codes proposed in this paper;  
 E: SBC-DBD codes reported in [4], [5], [9], [12] and [17].

described in this Section, after a brief introduction of the notations used. Note that, in the rest of the paper, 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 x n) matrix called parity check matrix and indicated as H (r x 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:

$$H X^T = 0 \quad (2.1)$$

where X<sup>T</sup> denotes the transpose of X. When the H matrix is expressed as:

$$H = [B \ I_r] \quad (2.2)$$

being B an (r x k) matrix and I<sub>r</sub> the (r x 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.

As mentioned before, the proposed codes are basically SEC-DED, so 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 x n) is in systematic form as in (2.2) with the matrix B(r x k) composed of a set of K matrices as:

$$B_{(r \times k)} = [B_1 \ B_2 \ \dots \ B_i \ \dots \ B_K] \quad (2.3)$$

where the matrices B<sub>i</sub> are non zero distinct (r x b) binary matrices. By construction, the number K of non zero distinct matrices B<sub>i</sub> 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 structure of the generic matrix B<sub>i</sub> is composed of two sub-matrices. The first sub-matrix is a (b x b) identity matrix I<sub>b</sub>, allowing to uniquely identify the single bit within the byte. The second sub-matrix of B<sub>i</sub> is a (r-b x b) matrix, indicated as H<sub>i</sub>, having the property that its column vectors are equal to each other and the generic column is stated to be a nonzero (r-b)-tu-

ple of even weight over GF(2). A set of nonzero distinct matrices H<sub>i</sub> 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<sub>i</sub> allows to uniquely identify the single byte.

The two sub-matrices composing B<sub>i</sub>, indicated as I<sub>b</sub> and H<sub>i</sub>, have to be placed vertically in B<sub>i</sub> 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 ([1]), 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<sub>i</sub> plus the generic one-weight column of I<sub>b</sub>.

Furthermore the code requires a number of check bits r ≥ b + 2, being the number of rows of B (r) the sum of the number of rows of I<sub>b</sub> (b) and the number of rows of H<sub>i</sub> (at least 2). Globally codes with the given properties for arbitrary code length (n) and byte length (b) within the range r ≥ b + 2 and b > 2 can be constructed using the three proposed techniques. In particular the first technique (called C<sub>1</sub>) requires r = 2b, the second technique (C<sub>2</sub>) requires r > 2b and the third technique (C<sub>3</sub>) requires b + 2 ≤ r < 2b.

The maximum number of data bytes in the codeword (K) as a function of (r, b) is reported hereafter for the three constructions and it is fully derived in [16]:

$$K(r, b) = 2^b - 2 \quad \text{for } C_1 \quad (2.4)$$

$$K(r, b) = 2^{r-b-1} + 2^{r-b-2} - 2 \quad \text{for } C_2 \quad (2.5)$$

$$K(r, b) = 2^{r-b-1} - 1 \quad \text{for } C_3 \quad (2.6)$$

From these values, the maximum length of the proposed codes can be easily derived as: n(r, b) = b K(r, b) + r.

For C<sub>1</sub> a detailed description of the structure of the matrices B<sub>i</sub> for i ∈ {1, 2, ..., K} is proposed, K as a function of (r, b) is derived and Theorem C<sub>1</sub>, reporting the properties of the proposed codes, is enounced hereafter. The proof of Theorem C<sub>1</sub>,



where the  $I_b$  denotes the  $(b \times b)$  identity matrix and the matrices  $H_i$  ( $b \times b$ ) have been defined in  $C_1$ , it can be considered as composed of two modules, as follows:

$$H = \begin{bmatrix} H_1 & \dots & H_i & \dots & H_{K_0} & 0_b & I_b & \dots & I_b & \dots & I_b & I_b \\ I_b & \dots & I_b & \dots & I_b & I_b & H_1 & \dots & H_i & \dots & H_{K_0} & 0_b \end{bmatrix}$$

|<-----  $M_0$  ----->|<-----  $M_1$  ----->|

keeping in mind that the  $(K_0 + 1)$ -th byte and the  $2(K_0 + 1)$ -th byte are the check bytes. The first  $b$  rows of  $M_0$  are equal to the last  $b$  rows of  $M_1$  and vice versa, thus the  $H$  matrix defined by  $C_1$  has a 2-modularized organization.

For example the above  $H(8 \times 64)$  matrix can be seen as composed of two modules  $M_0$  and  $M_1$ :

$$H_M = \begin{bmatrix} H_1 & H_2 & H_3 & H_4 & H_5 & H_6 & H_7 & 0_4 & I_4 & I_4 & I_4 & I_4 & I_4 & I_4 & I_4 & I_4 \\ I_4 & I_4 & I_4 & I_4 & I_4 & I_4 & I_4 & I_4 & H_1 & H_2 & H_3 & H_4 & H_5 & H_6 & H_7 & 0_4 \end{bmatrix}$$

|<-----  $M_0$  ----->|<-----  $M_1$  ----->|

The proposed high-speed parallel encoding-decoding circuits consist of four main blocks: check bit generator, syndrome generator, syndrome decoder and error corrector. The check bit generator and syndrome generator blocks are constituted by trees of Exclusive-OR gates. The syndrome decoder is constituted by two main blocks. The first block, called SYNDEC, decodes the syndromes to generate the correction patterns for the single-bit errors and the odd-bit-per-byte errors. The second block, called SYNCNT, decodes the syndromes for the detection of double-bit errors and even-bit-per-byte errors.

Due to the 2-modularized structure of the  $H$  matrix, the SYNDEC block can be described instancing the same logic block twice. The first instance ( $I_0$ ) receives the  $r$  syndrome bits as input and outputs the Bit Error Pointers related to  $M_0$ , while the second instance ( $I_1$ ) receives the  $r$  syndrome bits as input and outputs the Bit Error Pointers related to  $M_1$ . The Bit Error Pointers are used by the error corrector block, that simply executes a bit per bit Exclusive-OR between each Bit Error Pointer and the related bit read out of the memory.

The  $r$  syndromes received by  $I_0/I_1$  are used to decode data and check byte errors (Byte Error Pointers), the single-bit errors (Single-bit Error Pointers) and the odd-bit-per-byte errors (Odd-bit-per-byte Error Pointers). Receiving the related pointers as inputs, the same logic structure, called BITDEC, is used to compute the Bit Error Pointers for each data or check bits.

Fig. 1 shows an example of the SYNDEC block for the above  $H_M$  matrix. The gate amount of the proposed code represents about six percent decrease compared to a conventional decoding logic of a  $(64, 56)$  minimum odd-weight-column SEC-DED code able to correct just single errors. The increment required by the proposed code, in terms of propagation delay, is just one gate level, using a standard VLSI cell library.

In addition to the SYNDEC block, the SYNCNT block receives as input the syndromes and recognizes the number of asserted bits in the syndromes. An example of the logic structure of the SYNCNT block having four syndromes as input is in Fig. 2.

#### 4. Automatic generation of VHDL description

The design of a code with the characteristics described in Section II can be automated using a software tool, called GeCo (Generator of Codes). GeCo is a more general system which generates ECCs, in fact not only it can generate the class of codes described in this paper, but the main classes of codes in literature as well. Furthermore, its modularity allows an easy introduction of new classes of codes.

The aim behind the use of the GeCo tool is to provide the system designer with an easy-to-use platform to develop rapidly ECCs from specifications and to evaluate the intermediate results of the interaction between the user and the system. In fact, chosen the desired class of codes, the user can define the code specifications in terms of requested parameters: number of data bits ( $k$ ), number of check bits ( $r$ ) and, eventually, byte length ( $b$ ). During the interactive dialogue, the user can vary a parameter and GeCo automatically re-computes the values of

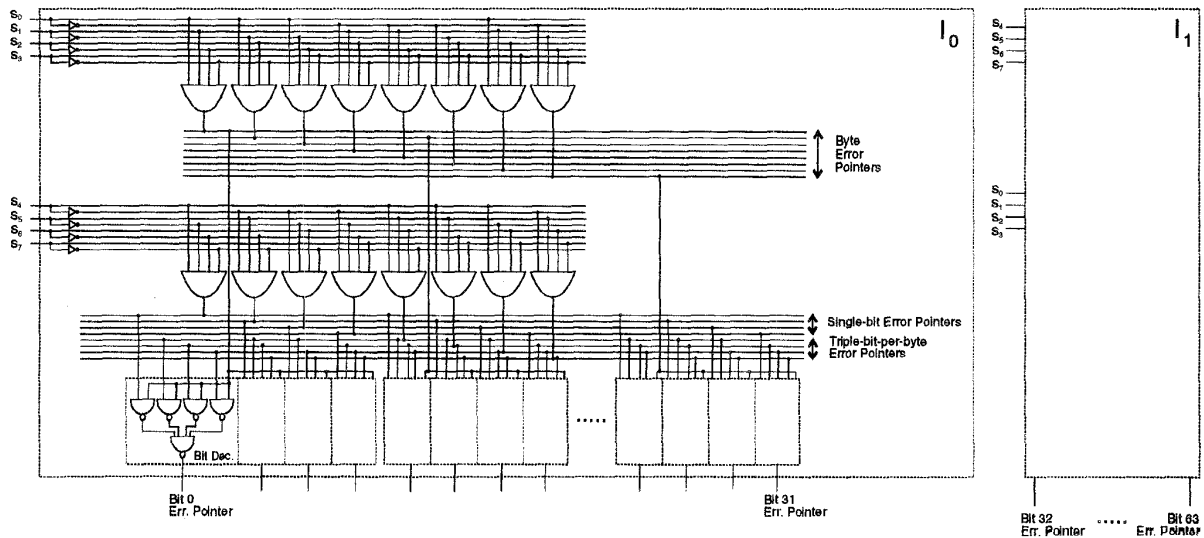


Fig. 1: The SYNDEC logic block for the  $H_M$  matrix.

the other parameters but, depending on the constraints existing among them, it prevents him from an illegal parameter assignment proposing only acceptable variations. After the dialogue led to an acceptable parameters configuration, the parity check matrix  $H$  can be automatically generated along with a hierarchical description of the main logic blocks implementing the code using the VHDL description language. GeCo computes also the global gate amount and the estimated propagation delay.

GeCo has been developed in the Microsoft Windows framework using the object oriented approach and the C++ language. It is composed of four main modules: the User Interface, the Controller, the Generator and the Translator.

## 5. Application results and concluding remarks

The proposed techniques allow the generation of a class of codes which extends the protection provided by previous SEC-DED-SBD codes by constructing systematic odd-weight-column SEC-DED-SBD codes in which the class of corrigible errors is enlarged to include any odd number of erroneous bits per byte. The proposed techniques are suitable for high performance computer applications and they are supported by an automatic tool.

This tool has been used to design a 64 data bit error control code inserted in an ASIC developed by Bull in the R&D Labs of Pregnana M. This ASIC interfaces the main memory, four data channels to processors and the I/O channel and it has been completely described using the VHDL language. Its main characteristics are reported in Fig. 3.

## References

[1] M.Y. Hsiao, "A Class of Optimal Minimum Odd-weight-column SEC-DED Codes", IBM J. Res. Develop., pp. 395-401, July 1970.  
 [2] S. Lin and D.J. Costello Jr., "Error Control Coding: Fundamentals and Applications, Prentice-Hall, Inc., Englewood Cliffs, NJ, 1983.

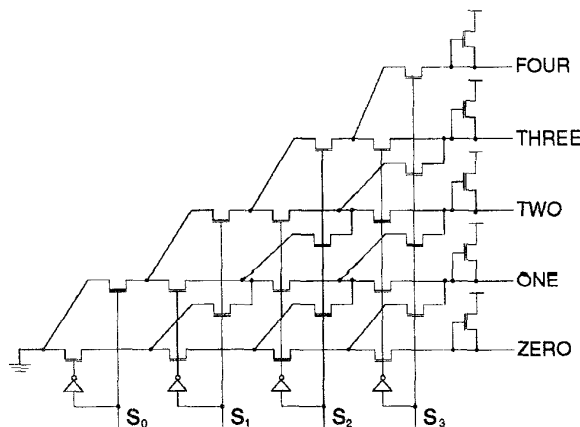


Fig. 2: The SYNCNT logic block with four syndromes as inputs.

[3] D.K. Pradhan, "Fault-Tolerant Computing: Theory and Techniques", vol.I, Prentice-Hall, Inc., Englewood Cliffs, NJ, 1986.  
 [4] 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.  
 [5] C.L. Chen, "Error-Correcting Codes for Byte-Organized Memory Systems", IEEE Trans. Information Theory, vol. IT-32, Mar. 1986.  
 [6] D.C. Bossen *et al.*, "Measurement and Generation of Error Correcting Codes for Package Failures", IEEE Trans. Computers, vol. C-27, pp. 201-204, Mar. 78.  
 [7] S.M. Reddy, "A Class of Linear Codes for Error Control in Byte-per-Card Organized Digital Systems", IEEE Trans. Computers, vol. C-27, pp. 455-459, May 1978.  
 [8] C.L. Chen, "Error-Correcting Codes with Byte Error-Detection Capability", IEEE Trans. Computers, vol. C-32, pp. 615-621, July 1983.  
 [9] S. Kaneda and E. Fujiwara, "Single Byte Error Correcting-Double Byte Error Detecting Codes for Memory Systems", IEEE Trans. Computers, vol. C-31, pp. 596-602, July 1982.  
 [10] C.L. Chen, "Symbol Error-Correcting Codes for Computer Memory Systems", IEEE Trans. Computers, vol. 41, pp. 252-256, Feb. 1992.  
 [11] L.A. Dunning and M.R. Varanasi, "Code Constructions for Error Control in Byte Organized Memory Systems", IEEE Trans. Computers, vol. C-32, pp. 535-542, June 1983.  
 [12] E. Fujiwara and D.K. Pradhan, "Error-Control Coding in Computers", Computer, July 1990, pp. 63-72.  
 [13] L.A. Dunning, "SEC-BED-DED Codes for Error Control in Byte-Organized Memory Systems", IEEE Trans. Computers, vol. C-34, pp. 557-562, June 1985.  
 [14] W.E. Clark *et al.*, "The Construction of Some Bit and Byte Error Control Codes Using Partial Steiner Systems", IEEE Trans. Information Theory, vol. 35, pp. 1305-1310, Nov. 1989.  
 [15] S. Kaneda, "A Class of Odd-weight-column SEC-DED-SBED Codes for Memory System Applications", IEEE Trans. Computers, vol. C-33, pp. 737-739, August 1984.  
 [16] L. Penzo *et al.*, "Construction Techniques for Systematic SEC-DED Codes with Single Byte Error Detection and Partial Correction Capability for Computer Memory Systems", Accepted for Publication by IEEE Trans. Information Theory.  
 [17] T.N. Rao and E. Fujiwara, "Error Control Coding for Computer Systems", Prentice-Hall, 1989.

Process	0.7 $\mu$ m DLM CMOS
Dimensions	11.82 mm x 11.82 mm
Total Equivalent Gates	55000
Internal Nets	11500
Average Pin Per Net	3.9
Max. Operating Frequency	75 MHz
Max. Power Dissipation	4 W @ 75 MHz
Package	299 CPGA
Logic Pins	202

Fig. 3: Main characteristics of the ASIC