High-speed decoding of extended Golay code

I.Boyarinov, I.Martin and B.Honary

Abstract: A simple high-speed decoding algorithm for the [24, 12, 8] extended Golay code suitable for implementation in combinational circuits is described. The proposed decoding algorithm corrects all patterns of three or fewer errors and detects quadruple errors using the Turyn construction of the extended Golay code. It is proved that the [24, 12, 8] Golay code can correct all patterns of three or fewer random errors as well as certain patterns of quadruple errors such as four-bit cyclic single-burst and two-dimensional burst errors.

1 Introduction

As encoders and decoders of computer semiconductor memories are realised in combinational circuits, there is need to construct simple and fast combinational encoders/decoders. In high-speed memories, single-bit error correcting and double-bit error detecting (SEC-DED) codes are most commonly used [1, 2]. A memory system with a large capacity or with high chip failure rates may use double-bit or triple-bit error correcting and triple-bit or quadruple-bit error detecting codes to meet its reliability requirements. A number of codes have been proposed for such an application (see survey in [3] and the references therein). Among these codes the [24, 12, 8] extended Golay code has the most advantageous properties owing to the optimality of its parameters.

The Golay code is probably the most important of all codes, for both practical and theoretical reasons [4]. Several methods for decoding the Golay code are known: algebraic decoding algorithms [4–15], bounded-distance decoding algorithms [16–18], trellis decoding [18–20] (see also the references in [4–20]). These algorithms are useful and efficient for many applications. However, these algorithms are not suitable for applications in computer semiconductor memories as they are relatively complex for implementation in combinatonal circuits and have a large decoding delay. We propose a simple high-speed decoding algorithm for the [24, 12, 8] extended Golay code, based on the $|a + x|b + x|a + b + x|$ Turyn construction [4]. The algorithm can be easily realised in combinational circuits. Futhermore we show that [24, 12, 8] Golay code can correct all patterns of three or fewer random errors as well as certain patterns of quadruple errors such as 4-bit cyclic single-burst and two-dimensional byte errors.

2 Construction of extended Golay code

Let C_1 be the binary cyclic [7, 4, 3] Hamming code with the generator polynomial $g_1(x) = x^3 + x + 1$ over $GF(2)$ and let C_2 be the binary cyclic [7, 4, 3] Hamming code with the generator polynomial $g_2(x) = x^3 + x^2 + 1$. It is easy to see that the code words of C_2 can be formed by reversing the code words of C_1. If α is a root of the polynomial $x^3 + x + 1$, the roots of the $g_1(x)$ and $g_2(x)$ are α, α^2, α^4 and α^3, α^5, α^6, respectively.

Let V_1 and V_2 be the [8, 4, 4] codes obtained by adding an overall parity check to C_1 and C_2. The codes V_1 and V_2 have two common words: (00000000) and (11111111). Construct the code C, consisting of all vectors

$$c = |a + x|b + x|a + b + x| \qquad (1)$$

where $a, b \in V_1$ and $x \in V_2$. The code C given by eqn. 1 is the [24, 12, 8] extended Golay code G_{24}. [4]. Let $z = c + e$ be a received word, c be a code word of the Golay code and e be an error word. Represent the words z and e as $z = |z_0|z_1|z_2|$ and $e = |e_0|e_1|e_2|$, where $z_0 = a + x + e_0$, $z_1 = b + x + e_1$, $z_2 = a + b + x + e_2$. Define

$$u_{01} = z_0 + z_1 = a + b + e_0 + e_1$$
$$u_{02} = z_0 + z_2 = b + e_0 + e_2$$
$$u_{12} = z_1 + z_2 = a + e_1 + e_2$$
$$u_{012} = z_0 + z_1 + z_2 = x + e_0 + e_1 + e_2$$

Consider u_{01}, u_{02}, u_{12} and u_{012} as words of codes V_1 and V_2, respectively. which possibly contain errors. Under decoding we need to calculate the syndromes of these words

$$s(u_{01}) = \boldsymbol{H}_1 u_{01}$$
$$s(u_{02}) = \boldsymbol{H}_1 u_{02}$$
$$s(u_{12}) = \boldsymbol{H}_1 u_{12}$$
$$s(u_{012}) = \boldsymbol{H}_2 u_{012}$$

where $\boldsymbol{H}_1$ is the parity check matrix of V_1 and $\boldsymbol{H}_2$ is the parity check matrix of V_2.

3 Decoding algorithm for random errors

The codeword $z = |z_0|z_1|z_2|$ can be represented graphicaly as shown in Fig. 1 where z_i is represented by the ith row.

Fig. 1 *Graphical representation of codeword*

© IEE, 2000

IEE Proceedings online no. 20000823

DOI: 10.1049/ip-com:20000823

Paper first received 9th August 1999 and in revised form 11th September 2000

I.Boyarinov is with the Institute for High-Performance Computer Systems, Russian Academy of Sciences, Nahimov ave. 36/1, 117872, Moscow, Russia

I.Martin and B.Honary are with the Communications Research Centre, Lancaster University, Lancaster LA1 4YR, UK

We now examine the syndrome characteristics for all the possible error patterns of weight $wt \le 4$.

3.1 Single error

A single error can only occur in one way as shown in Fig. 2. The parity check for the row containing the error λ_i is equal to one and the checks for the other two rows are equal to zero. Also the syndromes $s(u_{i+1,i+2}) = s(u_{i,i+1}) = S(u_{i,i+2}) = \alpha_j$ and $s(u_1\ 23) = \alpha_j^3$. The features of a single error are therefore

$$\lambda_i = 1, \quad \lambda_{i+1} = \lambda_{i+2} = 0$$
$$s(u_{012}) = s^3(u_{i,i+1}), \quad s(u_{i+1,i+2}) = 0 \tag{2}$$

when these conditions are met the syndrome $s(u_{i,i+1})$ can be used as an error locator.

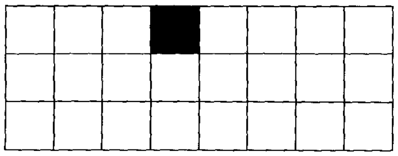

Fig. 2 *Single error*

3.2 Double errors

A double error may occur in one of two ways.

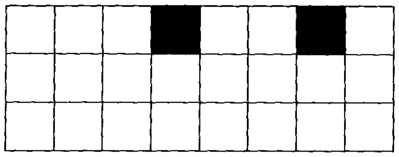

Fig. 3 *Double error, arrangement one*

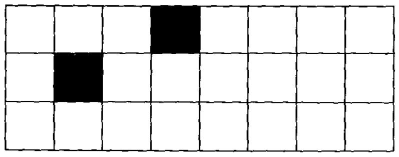

Fig. 4 *Double error, arrangement two*

3.2.1 Two errors in the same row:
The features of this type of error (Fig. 3) are

$$\lambda_0 = \lambda_1 = \lambda_2 = 0$$
$$s(u_{i+1,i+2}) = 0s(u_{012}) \neq 0$$
$$Tr(s^4(u_{i,i+1}s(u_{012}) + 1) = 0 \tag{3}$$

where $Tr(a)$ is the trace. The error locators for the word z_i are given by finding the roots of

$$t^2 + s(u_{i,i+1})t + s(u_{012})s^{-1}(u_{i,i+1}) + s^2(u_{i,i+1}) = 0 \tag{4}$$

3.2.2 One error in each of two rows:
The features of this error pattern shown in Fig. 4 are

$$\lambda_{i_1} = \lambda_{i_2} = 1, \quad \lambda_{i_3} = 0 \tag{5}$$

and

$$s(u_{i_1,i_3}) = s(u_{i_2,i_3}) \neq 0, \quad s(u_{012}) = 0 \tag{6}$$

or

$$s(u_{i_1,i_2}) \neq 0, \quad s(u_{012}) \neq 0$$
$$Tr(s^4(u_{i_1,i_2})s(u_{012}) + 1) = 0 \tag{7}$$

The error locator for the error in row i_1 is given by $(u_{i1,i3})$ and the error locator for the error in row i_2 is given by $s(u_{i2,i3})$.

3.3 Triple errors

A triple error may occur in one of three ways.

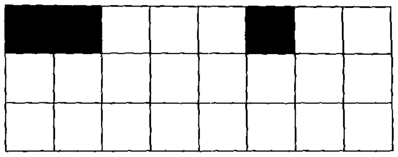

Fig. 5 *Triple error, arrangement one*

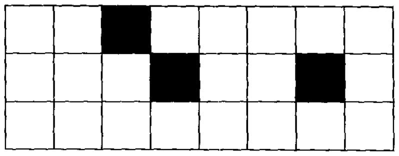

Fig. 6 *Triple error, arrangement two*

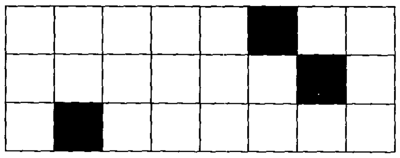

Fig. 7 *Triple error, arrangement three*

3.3.1 Three errors in the same row:
Features of this error pattern shown Fig. 5 are

$$\lambda_i = 1, \quad \lambda_{i+1} = \lambda_{i+2} = 0$$
$$s(u_{012}) \neq s^3(u_{i,i+1}), \quad s(u_{i+1,i+2}) = 0 \tag{8}$$

The error locators are then determined by finding the roots of the equation

$$t^3 + \sigma_1 t^2 + \sigma_2 t + \sigma_3 = 0 \tag{9}$$

where

$$\begin{aligned}
\sigma_1 &= s(u_{i,i+1}) \\
\sigma_2 &= \left(s^2(u_{i,i+1})s(u_{012}) + s^4(u_{012})\right) \\
&\quad \times \left(s^3(u_{i,i+1}) + s(u_{012})\right)^6 \\
\sigma_3 &= \left(s^6(u_{i,i+1}) + s^3(u_{i,i+1})s(u_{012})\right. \\
&\quad \left. + s(u_{i,i+1})s^4(u_{012}) + s^2(u_{012})\right) \\
&\quad \times \left(s^3(u_{i,i+1}) + s(u_{012})\right)^6
\end{aligned}$$

3.3.2 Two errors in one row and one in another row:
The features of such an error shown in Fig. 6 are

$$\lambda_{i_1} = 1, \quad \lambda_{i_2} = \lambda_{i_3} = 0$$
$$s(u_{i_2,i_3}) \neq 0$$
$$Tr\left(s^4(u_{i_1,i_2})\left(s(u_{012}) + s^3(u_{i_0,i_1})\right) + 1\right) = 0 \tag{10}$$

The error locator for the single error is given by $s(u_{i1,i3})$ and the error locators for the double errors are given by finding the roots of

$$t^2 + s(u_{i+1,i+2})t + \left(s(u_{012}) + s^3(u_{i,i+1})\right)$$
$$\times s^{-1}(u_{i+1,i+2}) + s^2(u_{i+1,i+2}) = 0 \tag{11}$$

3.3.3 Single error in each of three rows:
The features of such a triple error shown in Fig. 7 are

$$\lambda_0 = \lambda_1 = \lambda_2 = 1 \tag{12}$$

The error locators for the errors in each of the rows are obtained as follows:

$$\alpha_{j_1} = s(u_{12}) + (s(u_{012}) + s(u_{01})s(u_{02})s(u_{12}))^5$$
$$\alpha_{j_2} = s(u_{02}) + (s(u_{012}) + s(u_{01})s(u_{02})s(u_{12}))^5$$
$$\alpha_{j_3} = s(u_{01}) + (s(u_{012}) + s(u_{01})s(u_{02})s(u_{12}))^5 \quad (13)$$

3.4 Quadruple errors

Random quadruple errors cannot be corrected but since their characteristics are distinct from those of error patterns of weight three or less they can be detected.

3.5 Description of algorithm

Now we describe the decoding algorithm for random errors as follows:

(i) For the received word $z = |z_0|z_1|z_2|$ calculate the syndromes $s(u_{02})$, $s(_{012})$, $s(u_{012})$ and λ_0, λ_1, λ_2.

(ii) If $s(u_{02}) = s(u_{12}) = s(u_{012}) = 0$ and $\lambda_0 = \lambda_1 = \lambda_2 = 0$, there are no errors in the word z.

(iii) Calculate $s(u_{01})$.

(iv) Compare the syndrome and parity check values with the features of each type of error; if no match is found the received word contains an uncorrectable error pattern.

(v) Evaluate the error locators for the type of error found to determine the error pattern.

(vi) Perform the correction.

The correctness of the algorithm can easily be demonstrated by using a computer to test all error patterns of weight four or less

4 Decoder complexity

Table 1 shows the number of binary operations (and, or, exclusive or) which must be performed to correct errors of the given types. Note that it requires 80 operations to calculate the initial syndromes to determine there are no errors present. The worst-case complexity is that of pattern 3a which occurs when one row of the codeword contains three errors, 338 operations are required to correct this error pattern. The algorithm presented by Blaum and Bruck in [15] states a worst-case complexity of 968 operations with an average complexity of approximately 279 operations. The worst-case complexity of permutation decoding [9, 11] is 1078 binary operations with an average of 539.

Table 1: Operations required to correct all types of error patterns

Type of error	Number of operations
0	80
1	178
2a	276
2b	197
3a	338
3b	265
3c	225

To determine the average complexity of the proposed algorithm it is necessary to take in to account the channel bit error rate since this effects the probability of occurrence of each type of error. For a given error weight the probability of finding a given error pattern is independent of the bit error rate. We can therefore say that for errors of weight two, pattern 2b is twice as likely as pattern 2a and for errors of weight three, pattern 3b is twice as likely as pattern 3c which is, in turn, twice as likely as pattern 3a.

We now use these ratios to construct a table of average complexities for errors of each weight, Table 2. By weighting these averages with the probability of receiving an error with that weight we can calculate the average complexity for a number of different bit error rates (BER). This is shown in Table 3. It is clear from this table that the average complexity of the proposed decoder is significantly less than that of the decoder of [15].

Table 2: Average complexity for each error weight

Error weight	Average number of operations
0	80
1	178
2	220
3	264

Table 3: Average complexity for differing BER

BER	Average number of operations
0.01	102.5
0.001	82.34
0.00001	80.03

The decoder is particularly suitable for implementation as a combinational circuit. When implemented in this way 617 logic gates will be required for the complete design and a decoding delay of 32 gate delays would be required for decoding. The gate-count and delay are small due to the large amount of parallelism in the design. This simple, high-speed circuit is suitable for use with computer memory or disc storage devices and high capacity communications channels.

5 Correction of single-burst and two-dimensional byte errors

In some applications it is required that the memory array chips be packaged in a b-bit-per-chip organisation. A chip failure or a word-line failure in this case would result in byte-oriented errors that contain from 1 to b erroneous bits. Byte errors can also be caused by the failures of the supporting modules at the memory card level [2]. In other applications it is required that cyclic single-burst errors in code words are corrected [21]. We show that the [24, 12, 8] extended Golay code can correct certain patterns of quadruple errors such as 4-bit cyclic single-burst and two-dimensional byte errors simultaneously with all patterns of three or fewer errors. Note that the description of the [24, 12, 8] Golay code given in [19] allows the correction 4-bit byte errors simultaneosly with all patterns of three or fewer errors.

5.1 Cyclic burst errors

A cyclic burst is a diagonal arrangement of errors as shown in Fig. 8. The features of the four bit cyclic burst error are

$$\lambda_i = 0,\ \lambda_{i+1} = \lambda_{i+2} = 0$$
$$s(u_{i,i+1}) = 0,\ s(u_{i,i+2}) = s(u_{i+1,i+2}) \neq 0$$
$$s(u_{012}) \neq 0$$

or

$$s(u_{i,i+1}) = 0,\ s(u_{i,i+2}) \neq 0,\ s(u_{i+1,i+2}) \neq 0$$
$$s(u_{012}) \neq 0,\ Tr(s^4(u_{i+1,i+2})s(u_{012}) + 1) \neq 0$$

All errors of this type have different syndromes which differ from those of all error words $w \leq 3$. The position in the codeword of the first error in the block is defined by $s(u_{01})$, $s(u_{02})$, $s(u_{012})$ the location of the block can then be found using Table 4.

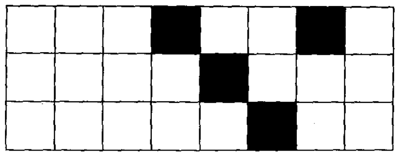

Fig. 8 *Four-bit cyclic burst error*

Table 4: Syndromes $s(u_{01})$, $s(u_{02})$, $s(u_{012})$ of the four-bit cyclic burst error

e_{ij}	$i = 0$	$i = 1$	$i = 2$
$j = 0$	$\alpha^6, \alpha^4, \alpha^5$	$\alpha^4, \alpha^3, \alpha^5$	$\alpha^3, \alpha^6, \alpha^5$
$j = 1$	$0, \alpha^4, \alpha^3$	$\alpha^4, \alpha^4, \alpha^3$	$\alpha^4, 0, \alpha^3$
$j = 2$	$0, \alpha^5, \alpha^6$	$\alpha^5, \alpha^5, \alpha^6$	$\alpha^5, 0, \alpha^6$
$j = 3$	$0, \alpha^6, \alpha^2$	$\alpha^6, \alpha^6, \alpha^2$	$\alpha^6, 0, \alpha^2$
$j = 4$	$0, 1, \alpha^5$	$1, 1, \alpha^5$	$1, 0, \alpha^5$
$j = 5$	$1, \alpha^3, \alpha^3$	$\alpha^3, \alpha, \alpha^3$	$\alpha, 1, \alpha^3$
$j = 6$	$\alpha^3, \alpha^4, \alpha^6$	$\alpha^4, \alpha^6, \alpha^6$	$\alpha^6, \alpha^3, \alpha^6$
$j = 7$	$\alpha^5, \alpha^4, \alpha^2$	$\alpha^4, 1, \alpha^2$	$1, \alpha^5, \alpha^2$

5.2.4 Four-bit two-dimensional error

A four-bit two-dimensional error is one where the errors are arranged in a square pattern as shown in Fig. 9. The features of the four-bit two-dimensional burst error are

$$\lambda_i = \lambda_{i+1} = \lambda_{i+2} = 0$$
$$s(u_{i,i+1}) = 0,\ s(u_{i,i+2}) = s(u_{i+1,i+2}) \neq 0$$
$$s(u_{012}) = 0$$

As in the case of the four-bit cyclic error the syndromes uniquely identify an error pattern which is distinct from all patterns $w \leq 3$ and all four-bit cyclic burst errors. The location of the first error in the burst can be found by looking up the syndromes in Table 5.

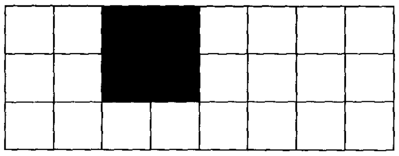

Fig. 9 *Four-bit two-dimensional burst error*

Table 5: Syndromes $s(u_{01})$, $s(u_{02})$, $s(u_{012})$ of the four-bit two-dimensional burst errors

e_{ij}	$i = 0$	$i = 1$	$i = 2$
$j = 0$	$0, 1, 0$	$1, 1, 0$	$1, 0, 0$
$j = 2$	$0, \alpha^4, 0$	$\alpha^4, \alpha^4, 0$	$\alpha^4, 0, 0$
$j = 4$	$0, \alpha^6, 0$	$\alpha^6, \alpha^6, 0$	$\alpha^6, 0, 0$
$j = 6$	$0, \alpha, 0$	$\alpha, \alpha, 0$	$\alpha, 0, 0$

6 Conclusion

A simple high-speed decoding algorithm for the [24, 12, 8] extended Golay code has been described. The proposed algorithm has a low complexity implementation and is suitable for applications in computer data storage systems. The complexity of the decoder is significantly less than that of other decoders for the same code. Furthermore, it has been proved that the [24, 12, 8] Golay code can correct simultaneously all patterns of three or fewer random errors as well as certain patterns of quadruple errors such as four-bit cyclic single-burst and two-dimensional block errors as defined.

7 Acknowledgment

The authors would like to thank The Royal Society, UK, in their support of Dr. I. Boyarinov during his stay with the Communications Research Centre at Lancaster University.

8 References

1 LIN, S., and COSTELLO, D.J. Jr.: 'Error control coding: fundamentals and applications' (Prentice-Hall, Englewood Cliffs, NJ, 1983), pp. 498–502
2 RAO, T.R.N., and FUJIWARA, E.: 'Error-control coding for computer systems' (Prentice-Hall, Englewood Cliffs, NJ, 1989), pp. 138–157
3 BOYARINOV, I.M.: 'Correction of multiple errors in computer semiconductor memories', *Vop. Kibernet.*, 1992, **174**, pp. 132–161 (in Russian)
4 MACWILLIAMS, F.J., and SLOANE, N.J.A.: 'The theory of error-correcting codes' (North-Holland, Amsterdam 1977), pp. 587–588
5 MACWILLIAMS, F.J.: 'Permutation decoding of systematic codes', *Bell. Syst. Tech. J.*, 1964, **43**, pp. 485–505
6 KASAMI, T.: 'A decoding procedure for multiple-error-correction cyclic codes', *IEEE Trans. Inf. Theory*, 1964, **IT-10**, pp. 134–139
7 CHIEN, R.T., and LUM, V.: 'On Golay's perfect codes and step-by-step decoding', *IEEE Trans. Inf. Theory*, 1966, **IT-12**, pp. 403–404
8 GOETHALS, J.M.: 'On the Golay perfect binary code', *J. Comb. Theory*, 1971, **11**, pp. 178–186
9 GORDON, D.M.: 'Minimal permutation sets for decoding the binary Golay codes', *IEEE Trans. Inf. Theory*, 1982, **IT-28**, pp. 541–543
10 SOLOMON, G., and SWEET, M.M.: 'A Golay puzzle', *IEEE Trans. Inf. Theory*, 1983, **IT-29**, pp. 174–175
11 WOLFMAN, J.: 'A permutation decoding of the (24,12,8) Golay code', *IEEE Trans. Inf. Theory*, 1983, **IT-29**, pp. 748–750
12 PLESS, V.: 'Decoding the Golay codes', *IEEE Trans. Inf. Theory*, 1986, **IT-32**, pp. 561-567
13 ELIA, M.: 'Algebraic decoding of the (23,12,7) Golay code', *IEEE Trans. Inf. Theory*, 1987, **IT-33**, pp. 150–151
14 WEI, S.W., and WEI, C.H.: 'On high-speed decoding of the (23,12,7) Golay code', *IEEE Trans. Inf. Theory*, 1990, **36**, pp. 692–695
15 BLAUM, M., and BRUCK, J.: 'Decoding of Golay code with Venn diagrams', *IEEE Trans. Inf. Theory*, 1990, **36**, pp. 906–910
16 SUN, F.W., and VAN TILBORG, H.C.A.: 'The Leech lattice, the Octacode, and decoding algorithms', *IEEE Trans. Inf. Theory*, 1995, **41**, pp. 1097–1106
17 VARDY, A.: 'Even more efficient bounded distance decoding of the hexacode, the Golay code and the Leech lattice', *IEEE Trans. Inf. Theory*, 1995, **41**, pp. 1495–1499
18 FORNEY, G.D.: 'Coset codes II: Binary lattices and related codes', *IEEE Trans. Inf. Theory*, 1988, **34**, pp. 1152–1187
19 DUMER, I.I., and ZINOVIEV, V.A.: 'Some new maximal codes over the Galois field GF(4)', *Probl. Inf. Trans.*, 1978, **14**, (3), pp. 24–34
20 HONARY, B., and MARKARIAN, G.: 'Trellis decoding of block codes. A practical approach' (Kluwer, Boston/Dordrecht/London 1997), pp. 25, 91
21 FARRELL, P.G.: 'A survey of array error control codes', *Eur. Trans. Telecommun. Related Techn. (ETT)*, 1992, **3**, (5), pp. 441-454