

Majority-Logic Decoding for Cyclic codes

張德威 蔡志侃

1. Introduction

● Majority-Logic decoding algorithm was devised in 1954 by Reed for a class of multiple-error-correcting codes discovered by Muller.

● Most majority-logic decodable codes found so far are cyclic codes.

2. One-Step Majority-Logic Decoding

● Consider an (n, k) cyclic code C with parity-check matrix H . The row space of H is an $(n, n-k)$ cyclic code of C_d . Then for V in C and W in C_d

$$W \cdot V = w_0v_0 + w_1v_1 + \cdots + w_{n-1}v_{n-1} = 0 \quad (1)$$

The equality is called a parity-check equation.

- Suppose the receive vector $r = v + e$ where

$$e = (e_0, e_1, \dots, e_{n-1})$$

- For any vector w in the dual code C_d , we can form the following linear sum of the receive digits:

$$A = w \cdot r = w_0 r_0 + w_1 r_1 + \dots + w_{n-1} r_{n-1} \quad (2)$$

This is called a parity-check sum or simply check sum.

- If r is a code vector in C , this parity-check sum, A , must be zero; however, if r is not a code vector in C , then A may not be zero. Combine (1) and (2), we will have:

$$A = w \cdot r = w_0 e_0 + w_1 e_1 + \dots + w_{n-1} e_{n-1} \quad (3)$$

An error digit e_i is said to be checked by the check sum A if the coefficient $w_i = 1$.

- Suppose that there exist J vectors in the dual code C_d ,

$$w_1 = (w_{10}, w_{11}, \dots, w_{1,n-1})$$

$$W_2 = (W_{20}, W_{21}, \dots, W_{2,n-1})$$

•

•

•

$$W_J = (W_{J0}, W_{J1}, \dots, W_{J,n-1})$$

which have the following properties:

1. The (n-1)th component of each vector is a “1,” that is,

$$W_{1,n-1} = W_{2,n-1} = W_{3,n-1} = \dots = W_{J,n-1} = 1.$$

2. For $i \neq n-1$, there is at most one vector whose i th

component is a “1”; for example, if $w_{1,i} = 1$, then

$$w_{2,i} = w_{3,i} = \dots = w_{J,i} = 0.$$

These J vectors are said to be orthogonal on the (n-1)th digit position. We call them orthogonal vectors.

- Form (2) and using J vectors described above, we can

get:

$$A_1 = w_1 \cdot r = w_{10}r_0 + w_{11}r_1 + \dots + w_{1,n-1}r_{n-1}$$

$$A_2 = w_2 \cdot r = w_{20}r_0 + w_{21}r_1 + \dots + w_{2,n-1}r_{n-1}$$

$$\begin{matrix} \cdot \\ \cdot \\ \cdot \end{matrix} \tag{4}$$

$$A_J = w_J \cdot r = w_{J0}r_0 + w_{J1}r_1 + \cdot \cdot \cdot + w_{J,n-1}r_{n-1}$$

Since $w_{1,n-1} = w_{2,n-1} = w_{3,n-1} = \dots = w_{J,n-1} = 1$, these J check sums are related to the error digits in the following manner:

$$A_1 = w_{10}e_0 + w_{11}e_1 + \cdot \cdot \cdot + e_{n-1}$$

$$A_2 = w_{20}e_0 + w_{21}e_1 + \cdot \cdot \cdot + e_{n-1}$$

$$\begin{matrix} \cdot \\ \cdot \\ \cdot \end{matrix} \tag{5}$$

$$A_J = w_{J0}e_0 + w_{J1}e_1 + \cdot \cdot \cdot + e_{n-1}$$

● We see that the error digit e_{n-1} is checked by all the check sums above. Form the second property of the orthogonal vectors, $w_1, w_2, \cdot \cdot \cdot, w_J$, any error digit other than e_{n-1} is checked by at most one check sum. These J check sums are said to be orthogonal on the

error digit e_{n-1} .

- Each of the foregoing check sums orthogonal on e_{n-1} is of the form:

$$A_j = e_{n-1} + \sum_{i \neq n-1} e_i$$

If all the error digits in the sum A_j are zero for $i \neq n-1$, the value of e_{n-1} is equal to A_j . Base on this fact, the parity-check sums orthogonal on e_{n-1} can be used to estimate e_{n-1} , or to decode the received digit r_{n-1} .

- Suppose that there are $\lfloor J/2 \rfloor$ or fewer errors in the vector $e=(e_0, e_1, \dots, e_{n-1})$. If $e_{n-1} = 1$, the other nonzero error digits can distribute among at most $\lfloor J/2 \rfloor - 1$ check sums orthogonal on e_{n-1} . Hence at least $J - \lfloor J/2 \rfloor + 1$, or more than one-half of the check sums orthogonal on e_{n-1} , are equal to $e_{n-1} = 1$.

- If $e_{n-1} = 0$, the nonzero error digits can distribute among at most $\lfloor J/2 \rfloor$ check sums. Hence, at least $J - \lfloor J/2 \rfloor$ or at least one-half of the check sums orthogonal on e_{n-1} , are equal to $e_{n-1} = 0$.

● Base on the faces above, an algorithm for decoding e_{n-1} can be formulated as follows:

The error digit e_{n-1} is decoded as 1 if a clear majority of the parity-check sums orthogonal on e_{n-1} is 1; otherwise, e_{n-1} is decoded as 0.

● Thus, the value of e_{n-1} is equal to the value assumed by a clear majority of the parity-check sums orthogonal on e_{n-1} .

● The decoding algorithm described above is called one-stop majority-logic decoding. If J is the maximum number of parity-check sums orthogonal on e_{n-1} that can be formed, then, by one-step majority-logic decoding, any error pattern of $\lfloor J/2 \rfloor$ or fewer errors can be corrected.

Let d_{\min} be the minimum distance of the code. The one-step majority-logic decoding is effective for this

code only of $t_{ML} = \lfloor J/2 \rfloor$ is equal to or close to the error-correcting capability $t = \lfloor (d_{\min} - 1)/2 \rfloor$; in other words, J should be equal to or close to $d_{\min} - 1$.

● Definition: A cyclic code with minimum distance d_{\min} is said to be completely orthogonalizable in one step if and only if it is possible to form $J = d_{\min} - 1$ parity-check sums orthogonal on an error digit.

Example 1 : Consider a (15,7) cyclic code with the generator polynomial: $g(x)=1+X^4+ X^6+ X^7+ X^8$

The parity-check matrix of this code is found as follows:

$$H = \begin{matrix} & \begin{bmatrix} h_0 \\ h_1 \\ h_2 \\ h_3 \\ h_4 \\ h_5 \\ h_6 \\ h_7 \end{bmatrix} & = & \begin{bmatrix} 100000001101000 \\ 010000000110100 \\ 001000000011010 \\ 000100000001101 \\ 000010001101110 \\ 000001000110111 \\ 000000101110011 \\ 000000011010001 \end{bmatrix} \end{matrix}$$

Digit positions:

$$w_1 = \quad \quad \quad h_3 = e_3 + e_{11} + e_{12} + e_{14}$$

$$w_1 = \quad \quad h_1 + h_5 = e_1 + e_5 + e_{13} + e_{14}$$

$$w_1 = h_0 + h_2 + h_6 = e_0 + e_2 + e_6 + e_{14}$$

$$w_1 = \quad \quad \quad h_7 = e_7 + e_8 + e_{10} + e_{14}$$

We see that e_{14} is checked by all four check sums and no other error digit is checked by more than one check sum.

If $e_{14} = 1$ and if there is one or no error occurring among

the other 14 digit positions, then at least three (majority) of the four check sums A_1 , A_2 , A_3 and A_4 are equal to 1.

If $e_{14} = 0$ and if there are two or fewer errors occurring among the other 14 digit positions, then at least two of the four check sums are equal to $e_{14} = 0$.

Hence, if there are two or fewer errors in e , the one-step majority-logic decoding always results in correct decoding of e_{14} .

Consider an error pattern with three errors $e_0 = e_3 = e_8 = 1$. Substitute into four parity-check sums, we have $A_1 = 1$, $A_2 = 0$, $A_3 = 1$, and $A_4 = 1$. Since the majority of the four sums is 1. So e_{14} is decoded as 1. This results in an incorrect decoding.

Thus, by one-step majority-logic decoding, the code is capable of correcting any error pattern with two or fewer

errors.

- Given an (n, k) cyclic code C for which J parity-check sums orthogonal on an error digit can be formed, the one-step majority-logic decoding of the code can be implemented as follows:

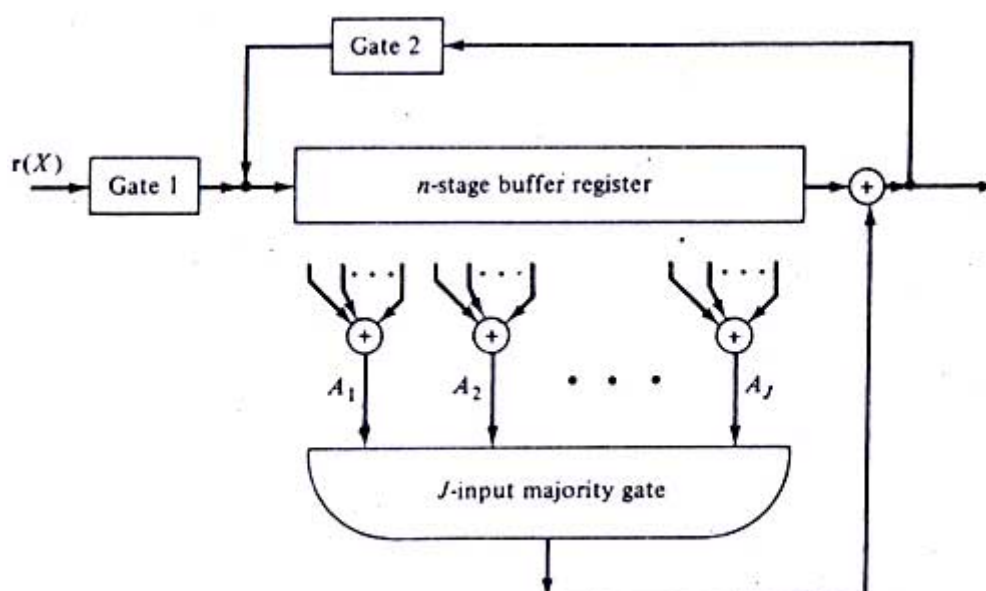


Figure 7.1 General type II one-step majority-logic decoder.

Step1. with gate turned on and gate turned off, the received vector r is read into the buffer register.

Step2. The J parity-check sums orthogonal on e_{n-1} are formed by summing the appropriate received digits.

Step3. The J orthogonal check sums are fed into a majority-logic gate. The first received digit r_{n-1} is read out of the buffer and is corrected by the output of the majority-logic gate.

Step4. At the end of step3, the buffer register has been shifted one place to the right with gate 2 on. Now the second received digit is in the rightmost stage of the buffer register and is corrected in exactly the same manner as the first received digit was. The decoder repeats steps 2 and 3.

Step5. The received vector is decoded digit by digit in the same manner above until a total of n shifts.

- If the received vector r contains $\lfloor J/2 \rfloor$ or fewer errors, the buffer register should contain the transmitted code vector and the inputs to the majority gate should be all

zero at the completion of the decoding operation.

If not all the inputs to the majority gate are zero, an uncorrectable error pattern has been detected.

The picture shown above is the type II one-step majority-logic decoder of Example 7.1.

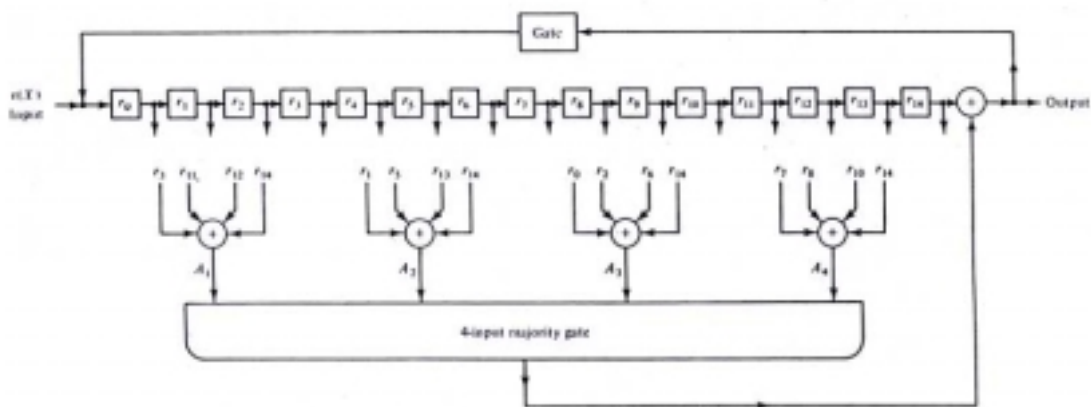


Figure 7.2 Type II one-step majority-logic decoder for the (15, 7) BCH code.

● The parity-check sums orthogonal on an error digit can also be formed from the syndrome digits. Let

$$\mathbf{H} = \begin{bmatrix} h_0 \\ h_1 \\ h_2 \\ h_3 \\ h_4 \\ h_5 \\ h_6 \\ h_7 \end{bmatrix} = \begin{bmatrix} 10000000 p_{00} p_{01} \dots p_{0,k-1} \\ 01000000 p_{10} p_{11} \dots p_{1,k-1} \\ 00100000 p_{20} p_{21} \dots p_{2,k-1} \\ \dots \\ \dots \\ \dots \\ 00000001 p_{n-k-1,0} p_{n-k-1,1} \dots p_{n-k-1,k-1} \end{bmatrix}$$

be the parity-check matrix for an (n, k) cyclic code C in systematic form. Since the orthogonal vectors w_1, w_2, \dots, w_J are vectors in the row space of H , that are linear combinations of rows of H . Let

$$\begin{aligned} w_j &= (w_{j0}, w_{j1}, \dots, w_{j,n-1}) \\ &= a_{j0}h_0 + a_{j1}h_1 + \dots + a_{j,n-k-1}h_{n-k-1}. \end{aligned}$$

Because of the systematic structure of H , we see that

$$w_{j0} = a_{j0}, w_{j1} = a_{j1}, \dots, w_{j,n-k-1} = a_{j,n-k-1}. \quad (7)$$

Let $r = (r_0, r_1, \dots, r_{n-1})$ be the received vector. Then the syndrome of r is:

$$s = (s_0, s_1, \dots, s_{n-k-1}) = r \cdot \mathbf{H}^T, \quad (8)$$

where the i th syndrome digit is:

$$s_i = r \cdot h_i \quad \text{for } 0 \leq i < n - k.$$

$$\begin{aligned}
A_j &= w_j \cdot r \\
&= (a_{j0}h_0 + a_{j1}h_1 + \cdot \cdot \cdot + a_{j,n-k-1}h_{n-k-1}) \cdot r \\
&= a_{j0}r \cdot h_0 + a_{j1} r \cdot h_1 + \cdot \cdot \cdot + a_{j,n-k-1} r \cdot h_{n-k-1} . \quad (9)
\end{aligned}$$

Form (7), (8), (9), we obtain

$$A_j = a_{j0}s_0 + a_{j1}s_1 + \cdot \cdot \cdot + a_{j,n-k-1}s_{n-k-1}. \quad (10)$$

Thus, the check sum A_j is simply a linear sum of the syndrome digits with coefficients being the first $n-k$ digits of the orthogonal vector w_j .

Based on (10), we obtain a different implementation of the one-step majority-logic decoding shown above.

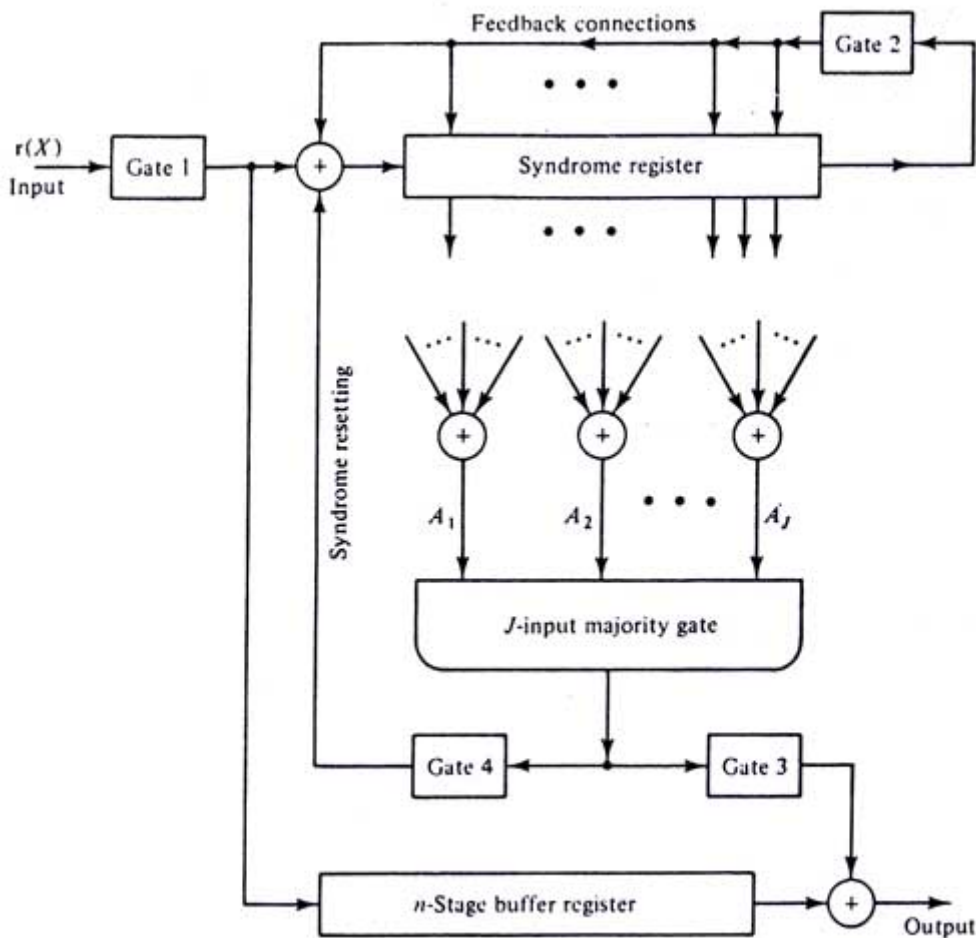


Figure 7.3 General type-I one-step majority-logic decoder.

● The correcting procedure is described as follows:

Step1. The syndrome is computed as usual by shifting the received polynomial $r(x)$ into the syndrome register.

Step2. The J parity-check sums orthogonal on e_{n-1} are formed by taking proper sums of the syndrome digits.

These J check sums are fed into a J -input majority-logic gate.

Step3. The first received digit is read out of the buffer register and is corrected by the output of the majority gate. At the same time the syndrome register is also shifted once (with gate 2 on) and the effect of e_{n-1} on the syndrome is removed (with gate 4 on). The new contents in the syndrome register form the syndrome of the altered received vector cyclically shifted one place to the right.

Step4. The new syndrome formed in step3 is used to decode the next received digit r_{n-2} . The decoder repeats steps 2 and 3. the received digit r_{n-2} is corrected in exactly the same manner as the first received digit r_{n-1} was corrected.

Step5. The decoder decodes the received vector r digit by digit in the manner shown above until a total

of n shifts of the buffer and the syndrome registers.

- The buffer register should contain the transmitted code vector and the inputs to the majority gate should be all zero at the completion of the decoding operation.

If not all the inputs to the majority gate are zero, an uncorrectable error pattern has been detected.

Example 2:

Consider the (15,7) BCH code given in Example 2. From the vectors $w_1, w_2, w_3,$ and $w_4,$ that are orthogonal on the digit position 14, we find that the parity-check sums orthogonal on e_{14} are equal to the following sums of syndrome digits:

$$A_1 = s_3 \quad , \quad A_1 = s_1 + s_5 \quad , \quad A_3 = s_0 + s_2 + s_6 \quad , \quad A_1 = s_7.$$

Hence we can construct type I one-step majority-logic

decoder in Figure 4.

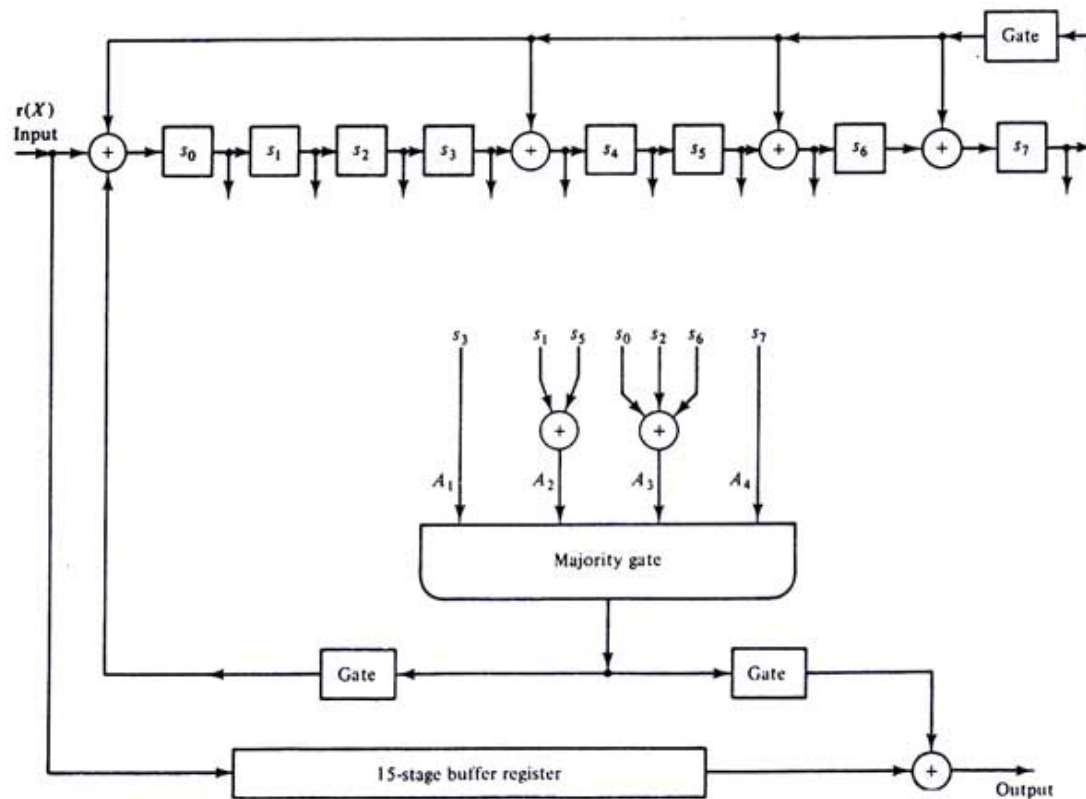


Figure 7.4 Type I one-step majority-logic decoder for (15, 7) BCH code.

Suppose that the all-zero code vector $(0,0,\dots,0)$ is transmitted and $r(X)=X^{13} + X^{14}$ is received, that is there are two errors in X^{13}, X^{14} . After the entire received polynomial has entered the syndrome register, we have $s=(0,0,1,1,1,0,0,1)$. The four parity-check sums will be $A_1=1, A_2=0, A_3=1, A_4=1$.

Since the majority of these four sums is 1, the output of the majority-logic gate is 1, which is the value of e_{14} .

Simultaneously shift the buffer and syndrome registers

once; the highest-order received digit $r_{14} = 1$ is then corrected by the output of the majority-logic gate and the new contents in the syndrome register are

$(0,0,0,1,0,1,1,1)$. The new parity-check sums are now

$$A'_1=1, \quad A'_2=1, \quad A'_3=1, \quad A'_4=1.$$

Again the output of the majority-logic gate is 1, which is the value of e_{14} . Shift both the buffer and syndrome registers once more; the received digit r_{13} would be corrected and the syndrome register would contain only zeros. At this point, both errors have been corrected and the next 13 received digits are error-free.

- One-step majority-logic decoding is most efficient for codes that are completely orthogonalizable, or for codes with larger J compared to $d_{\min}-1$. When J is small compared to $d_{\min}-1$, one-step majority-logic decoding becomes very inefficient, and much of the error-correcting capability of the code is sacrificed.

- The maximum number of parity-check sums

orthogonal on an error digit that can be formed are:

Theorem 1:

Let C be an (n, k) cyclic code whose dual code C_d has minimum distance δ . Then the number of parity-check sums orthogonal on an error digit that can be formed, J , is upper bounded by

$$J \leq \left\lfloor \frac{n-1}{\delta-1} \right\rfloor$$

● The dual code of the $(15,7)$ BCH code has minimum

distance 4. There $J \leq \left\lfloor \frac{n-1}{\delta-1} \right\rfloor = \left\lfloor \frac{14}{3} \right\rfloor = 4$ This

proves our claim in Example 1 that $J=4$ is the

maximum number of parity-check sums orthogonal on

an error digit that can be formed for the $(15,7)$ BCH

code.

● If it is possible to form J parity-check sums orthogonal

on an error digit for a cyclic code, the code has

minimum distance at least $J+1$.

3. Class of One step Majority-Logic Decodable Codes

- Let C be an (n, k) cyclic code generated by $g(x)$, where $n=2^m-1$. We may extend each vector $v=(v_0, v_1, \dots, v_{n-1})$ in C by adding an overall parity-check digit, denoted by v_∞ , to its left. The overall parity-check digit v_∞ is defined as the modulo-2 sum of all the digits of v ($v_\infty = v_0 + v_1 + \dots + v_{n-1}$). Adding v_∞ to v

results in the following vector fo $n+1 = 2^m$ components:

$$v_e = (v_\infty, v_0, v_1, \dots, v_{n-1}) .$$

- Consider a permutation that carries the component of v_e at the location Y to the location $Z = aY + b$, where a and b are elements form the field $GF(2^m)$ and $a \neq 0$.

This permutation is called an affine permutation.

Application of an affine permutation to a vector of 2^m components results in another vector of 2^m components.

Example 3:

Consider the following vector of 16 components, which are numbered with the elements of $GF(2^4)$:

$$\alpha^\infty \quad \alpha^0 \quad \alpha^1 \quad \alpha^2 \quad \alpha^3 \quad \alpha^4 \quad \alpha^5 \quad \alpha^6 \quad \alpha^7 \quad \alpha^8 \quad \alpha^9 \quad \alpha^{10} \quad \alpha^{11} \quad \alpha^{12} \quad \alpha^{13} \quad \alpha^{14}$$

$$(1 \quad 1 \quad 0 \quad 0 \quad 1 \quad 0 \quad 0 \quad 1 \quad 0 \quad 0 \quad 1 \quad 0 \quad 0 \quad 1 \quad 0 \quad 0)$$

Now, we apply the affine permutation

$$Z = \alpha Y + \alpha^{14} .$$

To the components of the vector above. The resultant

vector is

$$\alpha^\infty \quad \alpha^0 \quad \alpha^1 \quad \alpha^2 \quad \alpha^3 \quad \alpha^4 \quad \alpha^5 \quad \alpha^6 \quad \alpha^7 \quad \alpha^8 \quad \alpha^9 \quad \alpha^{10} \quad \alpha^{11} \quad \alpha^{12} \quad \alpha^{13} \quad \alpha^{14}$$

$$(0 \quad 0 \quad 1 \quad 1 \quad 0 \quad 0 \quad 0 \quad 0 \quad 1 \quad 0 \quad 1 \quad 0 \quad 1 \quad 0 \quad 0 \quad 1 \quad)$$

For example, the component at te location $Y = \alpha^8$ is

carried to the location:

$$Z = \alpha \cdot \alpha^8 + \alpha^{14} = \alpha^9 + \alpha^{14} = \alpha^4$$

- An extended cyclic code C_e of length 2^m is said to be invariant under the group of affine permutations if every affine permutation carries every code vector in C_e into another code vector in C_e .

● Definition:

Let h be a nonnegative integer less than 2^m . The radix 2 (binary) expansion of h is:

$$h = \delta_0 + \delta_1 2 + \delta_2 2^2 + \dots + \delta_{m-1} 2^{m-1}$$

where $\delta_i = 0$ or 1 for $0 \leq i < m$. Let h' be another nonnegative integer less than 2^m whose radix 2 expansion is:

$$h' = \delta'_0 + \delta'_1 2 + \delta'_2 2^2 + \dots + \delta'_{m-1} 2^{m-1}$$

The integer h' is said to be a descendant of h if

$$\delta'_i \leq \delta_i \text{ for } 0 \leq i < m.$$

Let $\Delta(h)$ denote the set of all nonzero proper descendants of h .

- The necessary and sufficient condition for an extended cyclic code C_e of length 2^m to be invariant under the affine permutations is:

Theorem 2

Let C be a cyclic code of length $n=2^m - 1$ generated by $g(X)$. Let C_e be the extended code obtained from C by appending an overall parity-check digit. Let α be a primitive element of the Galois field $GF(2^m)$. then the

extended code C_e is invariant under the affine permutations if and only if for every α^h that is a root of the generator polynomial $g(X)$ of C and for every h' in $\Delta(h)$, $\alpha^{h'}$ is also a root of a codeword C_e and $\alpha^0=1$ is not a root of $g(X)$.

● A cyclic code whose generator polynomial satisfies the conditions stated above is said to have the doubly transitive invariant (DTI) property.

● Let $J \cdot L = 2^m - 1 = n$. Obviously, both J and L are odd.

Then

$$X^{(2^m-1)+1} = (1+X^J)(1+X^J+X^{2J}+\dots+X^{(L-1)J}).$$

$$\text{Let } \pi(X) = 1+X^J+X^{2J}+\dots+X^{(L-1)J}$$

● From the theorem of Chapter 2, we know that the 2^m-1 nonzero elements of $GF(2^m)$ form the 2^m-1 roots of $X^{(2^m-1)+1}$. Let α be a primitive element of the Galois field $GF(2^m)$. Since $(\alpha^L)^J = \alpha^{(2^m-1)} = 1$, the

polynomial $1+X^J$ has $1, \alpha^L, \alpha^{2L}, \dots, \alpha^{(J-1)L}$ as all its roots. Therefore $\pi(X)$ has α^h as a root if and only if h is not a multiple of L and $0 < h < 2^m - 1$.

● $H(X)$ is form by:

$H(X)$ has α^h as a root if and only if (1) α^h is a root of $\pi(X)$ and (2) for every h' in $\Delta(h)$, $\alpha^{h'}$ is also a root of $\pi(X)$.

Let α^i be a root of $H(X)$ and ϕ_i be the minimal polynomial of α^i . Then

$H(X) = \text{LCM}\{\text{minimal polynomials } \phi_i \text{ of the roots of } H(X)\}$.

● From theorem 2, C' which is generated by $H(X)$ is invariant under the group of affine permutations. And the dual code C have corresponding generating polynomial

$$g(X) = X^{2^m - k - 1} G(X^{-1}).$$

where $G(X)H(X) = X^{2^m-1} + 1$

Example 4:

Let $m=5$. The polynomial $X^{2^m-1} + 1 = X^{15} + 1$ can be factored as $(X^{15} + 1) = (1 + X^5)(1 + X^5 + X^{10})$.

Thus, $J=5, L=3$, and $\pi(X) = 1 + X^5 + X^{10}$. Let α be a primitive element in $GF(2^4)$ whose minimal polynomial is $\phi_1(X) = 1 + X + X^4$.

The polynomial $1 + X^5$ has $1, \alpha^3, \alpha^6, \alpha^9, \alpha^{12}$ as

all its roots.

The polynomial $\pi(X)$ has $\alpha, \alpha^2, \alpha^4, \alpha^5, \alpha^7, \alpha^8, \alpha^{10}, \alpha^{11}, \alpha^{13}, \alpha^{14}$ as all its roots.

Thus $H(X)$ has $\alpha, \alpha^2, \alpha^4, \alpha^5, \alpha^8, \alpha^{10}$ as its roots.

The roots $(\alpha, \alpha^2, \alpha^4, \alpha^8)$ and (α^5, α^{10}) are conjugates. Hence,

$$\begin{aligned} H(X) &= \phi_1(X) \cdot \phi_5(X) = (1 + X + X^4)(1 + X + X^2) \\ &= 1 + X^3 + X^4 + X^5 + X^6. \end{aligned}$$

The dual code of C', C , is generated by

$$g(X) = X^{(2^m - k - 1)} G(X^{-1}) = X^9 G(X^{-1}) =$$

$1 + X + X^4 + X^5 + X^6 + X^9$. Thus C is a (15,6) cyclic code.

Decoding procedure:

The vectors corresponding to $\pi(X), X\pi(X), X^2\pi(X), X^3\pi(X),$ and $X^4\pi(X)$ are shown on next page.

	Location Numbers														
	α^0	α^1	α^2	α^3	α^4	α^5	α^6	α^7	α^8	α^9	α^{10}	α^{11}	α^{12}	α^{13}	α^{14}
$v_0 =$	(1	0	0	0	0	1	0	0	0	0	1	0	0	0	0)
$v_1 =$	(0	1	0	0	0	0	1	0	0	0	0	1	0	0	0)
$v_2 =$	(0	0	1	0	0	0	0	1	0	0	0	0	1	0	0)
$v_3 =$	(0	0	0	1	0	0	0	0	1	0	0	0	0	1	0)
$v_4 =$	(0	0	0	0	1	0	0	0	0	1	0	0	0	0	1)

which are code vectors in C' . Adding an overall parity-check digit to these vectors, we obtain the following vectors:

	Location Numbers															
	α^{∞}	α^0	α^1	α^2	α^3	α^4	α^5	α^6	α^7	α^8	α^9	α^{10}	α^{11}	α^{12}	α^{13}	α^{14}
$u_0 =$	(1	1	0	0	0	0	1	0	0	0	0	1	0	0	0	0)
$u_1 =$	(1	0	1	0	0	0	0	1	0	0	0	0	1	0	0	0)
$u_2 =$	(1	0	0	1	0	0	0	0	1	0	0	0	0	1	0	0)
$u_3 =$	(1	0	0	0	1	0	0	0	0	1	0	0	0	0	1	0)
$u_4 =$	(1	0	0	0	0	1	0	0	0	0	1	0	0	0	0	1)

which are vectors in C'_e (the extension of C'). Now, we apply the affine permutation $Z = \alpha Y + \alpha^{14}$ to permute the components of $u_0, u_1, u_2, u_3,$ and u_4 . The permutation results in the following vectors:

	Location Numbers															
	α^{∞}	α^0	α^1	α^2	α^3	α^4	α^5	α^6	α^7	α^8	α^9	α^{10}	α^{11}	α^{12}	α^{13}	α^{14}
$z_0 =$	(0	0	0	0	0	0	0	0	1	1	0	1	0	0	0	1)
$z_1 =$	(0	0	1	0	0	0	1	0	0	0	0	0	0	0	1	1)
$z_2 =$	(0	1	0	1	0	0	0	1	0	0	0	0	0	0	0	1)
$z_3 =$	(1	0	0	0	0	1	0	0	0	0	1	0	0	0	0	1)
$z_4 =$	(0	0	0	0	1	0	0	0	0	0	0	0	1	1	0	1)

which are also in C'_e . Deleting the overall parity-check digits from these vectors, we obtain the following vectors in C' :

	Location Numbers														
	α^0	α^1	α^2	α^3	α^4	α^5	α^6	α^7	α^8	α^9	α^{10}	α^{11}	α^{12}	α^{13}	α^{14}
$w_0 =$	(0	0	0	0	0	0	0	1	1	0	1	0	0	0	1)
$w_1 =$	(0	1	0	0	0	1	0	0	0	0	0	0	0	1	1)
$w_2 =$	(1	0	1	0	0	0	1	0	0	0	0	0	0	0	1)
$w_3 =$	(0	0	0	0	1	0	0	0	0	1	0	0	0	0	1)
$w_4 =$	(0	0	0	1	0	0	0	0	0	0	0	1	1	0	1)

We see that these vectors are orthogonal on the digit at location α^{14} .

Let $\mathbf{r} = (r_0, r_1, r_2, r_3, r_4, r_5, r_6, r_7, r_8, r_9, r_{10}, r_{11}, r_{12}, r_{13}, r_{14})$ be the received vector. Then the parity-check sums orthogonal on e_{14} are

$$A_0 = w_0 \cdot r = r_7 + r_8 + r_{10} + r_{14}$$

$$A_1 = w_1 \cdot r = r_1 + r_5 + r_{13} + r_{14}$$

$$A_2 = w_2 \cdot r = r_0 + r_2 + r_6 + r_{14}$$

$$A_3 = w_3 \cdot r = r_4 + r_9 + r_{14}$$

$$A_4 = w_4 \cdot r = r_3 + r_{11} + r_{12} + r_{14}$$

Hence the (15,6) cyclic code C generated by

$$g(X) = 1 + X + X^4 + X^5 + X^6 + X^9 \text{ is one-step}$$

majority-logic decodable and is capable of correcting

$$t_{ML} = \left\lfloor \frac{5}{2} \right\rfloor = 2 \text{ or fewer errors. The code has minimum}$$

distance at least $J + 1 = 5 + 1 = 6$. However, the

generator polynomial has weight exactly 6. Thus, the

minimum distance of the code is exactly 6. Hence, the

code is completely orthogonalizable.

- $\pi(X)$ has α^h as a root if and only if h is not a multiple of L and $0 < h < 2^m - 1$. So $\pi(X)$ has the following consecutive powers of α as roots: $\alpha, \alpha^2, \dots, \alpha^{L-1}$.

● Consequently, the polynomial $H(X)$ has $\alpha, \alpha^2, \dots, \alpha^{L-1}$ as roots. This proves that the minimum distance of the code generated by $H(X)$ is at least L . However, since $\pi(X)$ is a code polynomial of weight L in C' , the minimum distance of C' is exactly L . From Theorem 1, the number of parity-check sums orthogonal on an error digit that can be formed for C is upper bounded by $\left\lfloor \frac{2^m - 2}{L - 1} \right\rfloor$. However, $J = \frac{2^m - 2}{L}$.

Therefore, for large L , J is either equal to or close to the upper bound of $\left\lfloor \frac{2^m - 2}{L - 1} \right\rfloor$.

● The code structure stated above is called type 0 DTI codes. In general, it is not known whether the type 0 DTI codes are completely orthogonalizable. There are a number of special cases for which we can prove that the codes are completely orthogonalizable.

● If the type 0 DTI codes can be modified by:

$$H_1(X) = H(X) (X+1)$$

So that the resultant codes are also one-step majority-logic decodable and $J-1$ parity-check sum orthogonal on an error digit can formed. This is called type 1 DTI code.

It can be shown easily that the number of parity-check sums orthogonal on an error digit that can be formed for a type 1 DTI code are equal to its upper bound.

- A list of one-step majority-logic decodable type 1 DTI codes is given in Table 1.

TABLE 7.1 SOME ONE-STEP MAJORITY-LOGIC DECODABLE TYPE 1 DTI CODES

n	k	t_{ML}	n	k	t_{ML}
15	9	1	2047	1211	11
	7	2		573	44
63	49	1	4095	3969	1
	37	4		3871	2
	13	10		3753	4
255	225	1		3611	6
	207	2		3367	32
	175	8		2707	17
	37	25		2262	19
	21	42		2074	22
511	343	3		1649	45
	139	36		1393	52
1023	961	1		1377	136
	833	5		406	292
	781	16		101	409
	151	46		43	682
	30	170			

- For short length, DTI codes are comparable with BCH codes in efficiency.
- For example, there exists a (63,37) one-step majority-logic decodable type 1 DTI code which is capable of correcting four or fewer errors. The

corresponding four-error-correcting BCH code of the same length is a (63,39) code that has two information digits more than the (63,37) type-1 DTI code.

However, the decoding circuit for the (63,39) BCH code is much more complex than the (63,37) DTI code.

- But for large block length, the DTI codes are much less efficient than the BCH codes of the same length and the same error-correcting capability.

4. Other One-Step Majority-Logic Decodable Codes

Maximum-Length Code

- For any integer $m \geq 3$, there exists a nontrivial maximum-length code with the following parameters:

$$\text{Block length : } n = 2^m - 1$$

$$\text{Number of information digits : } k = m$$

$$\text{Minimum distance : } d = 2^m - 1$$

- The generator polynomial is :

$$g(X) = \frac{X^n + 1}{p(X)}$$

where $p(X)$ is a primitive polynomial of degree m .

- The dual code of the maximum-length code is a $(2^m - 1, 2^m - m - 1)$ cyclic code generated by the reciprocal of the parity polynomial $p(X)$,

$$p^*(X) = X^m p(X^{-1})$$

Since $p^*(X)$ is also a primitive polynomial of degree m , the dual code is thus Hamming code. Therefore the

minimum weight is 3.

- Consider the following set of distinct code

$$\text{polynomial : } Q = \{w(X) = X^i + X^j + X^{n-1} \mid 0 \leq i < j \leq n-1\}$$

in the Hamming code generated by $p^*(X)$.

- Therefore, the set Q contains polynomials orthogonal on the highest-order digits position X^{n-1} . To find $w(X)$, we start with a polynomial $X^{n-1} + X^j$ for $0 \leq j < n-1$, and then determine X^i such that $X^{n-1} + X^j + X^i$ is divisible by $p^*(X)$. This can carry out as follows. Divide $X^{n-1} + X^j$ by $p^*(X)$ step by step by long division until a single term X^i appears at the end of a certain step. Then $w(X) = X^{n-1} + X^j + X^i$ is a polynomial orthogonal on digit position X^{n-1} . Clearly, if we start with $X^{n-1} + X^i$, we would obtain the same polynomial $w(X)$. Thus, we can find $(n-1)/2 = 2^{m-1} - 1$ polynomial orthogonal on digit position X^{n-1} . That is, $J = 2^{m-1} - 1$ parity-check sums orthogonal on e_{n-1} can be formed.

Example 5 :

Consider the maximum-length code with $m=4$ and parity polynomial $p(X)=1+X+X^4$. This code has block length $n=15$ and minimum distance $d=8$. The generator polynomial of this code is

$$g(X) = \frac{X^{15} + 1}{p(X)}$$

$$= 1 + X + X^2 + X^3 + X^5 + X^7 + X^8 + X^{11}$$

$$p^*(X) = X^4 p(X^{-1}) = X^4 + X^3 + 1$$

Divides $X^{14} + X^{13}$ by $p^*(X) = X^4 + X^3 + 1$ with long division as shown in the following :

$$\begin{array}{r} X^{10} \\ X^4 + X^3 + 1 \overline{) X^{14} + X^{13}} \\ \underline{X^{14} + X^{13} + X^{10}} \\ X^{10} \leftarrow \text{Stop} \end{array}$$

Thus $w_1(X) = X^{14} + X^{13} + X^{10}$

As the same algorithm we can find :

$$w_2(X) = X^{14} + X^{12} + X^6, \quad w_3(X) = X^{14} + X^{11} + 1$$

$$w_4(X)=X^{14}+X^9+X^4 \quad , \quad w_5(X)=X^{14}+X^8+X$$

$$w_6(X)=X^{14}+X^7+X^5 \quad , \quad w_7(X)=X^{14}+X^3+X^2$$

From the set of polynomials orthogonal on X^{14} , we obtain the following seven parity-check sums orthogonal on e_{14} :

$$A_1 = e_{10} + e_{13} + e_{14}$$

$$A_2 = e_6 + e_{12} + e_{14}$$

$$A_3 = e_0 + e_{11} + e_{14}$$

$$A_4 = e_4 + e_9 + e_{14}$$

$$A_5 = e_1 + e_8 + e_{14}$$

$$A_6 = e_5 + e_7 + e_{14}$$

$$A_7 = e_2 + e_3 + e_{14}$$

In terms of syndrome bits, we have $A_1=s_{10}$, $A_2=s_6$, $A_3=s_0$, $A_4=s_4+s_9$, $A_5=s_1+s_8$, $A_6=s_5+s_7$, $A_7=s_5+s_7$. The code is capable of correcting three or fewer errors by one-step majority –logic decoding. The type I and type II one-step majority-logic decoders for this code

are shown in below

type I one-step majority-logic decoders

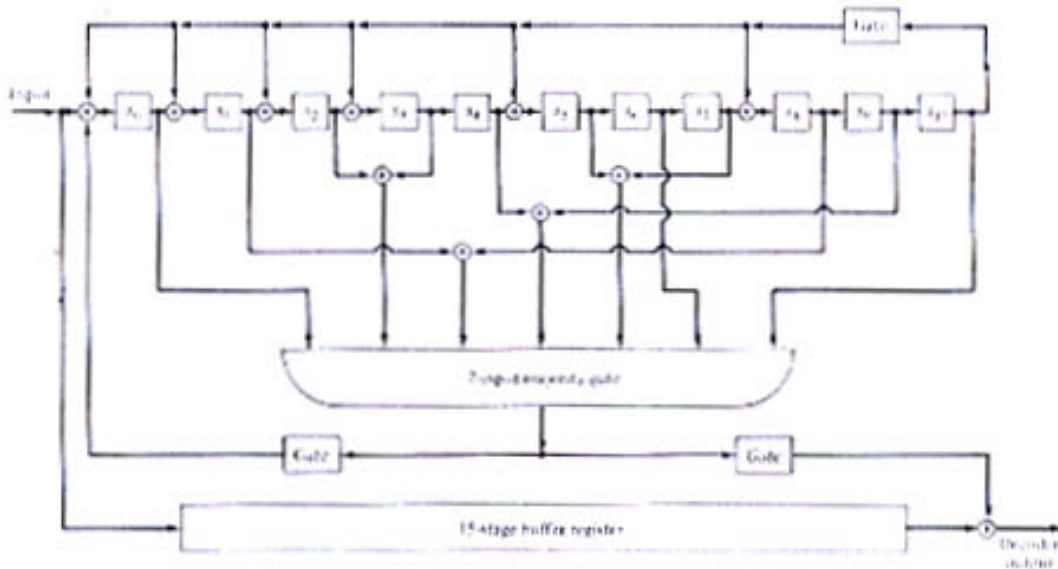


Figure 7.5 Type I majority-logic decoder for the (15, 4) maximum-length code.

type II one-step majority-logic decoders

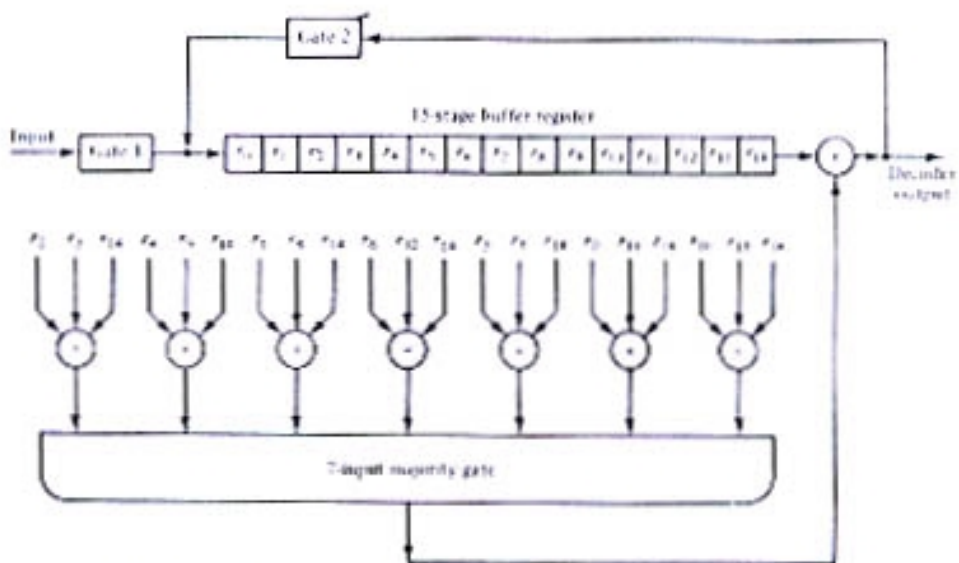


Figure 7.6 Type II majority-logic decoder for the (15, 4) maximum-length code.

Difference-Set Codes

- The formulation of difference-set codes is based on the construction of a perfect difference set. Let $P = \{l_0, l_1, l_2, \dots, l_q\}$ be a set of $q+1$ nonnegative integers such that

$$0 \leq l_0 < l_1 < l_2 < \dots < l_q \leq q(q+1)$$

From this set of integers, it is possible to form $q(q+1)$ ordered differences as follows :

$$D = \{l_j - l_i \mid j \neq i\}.$$

- Obviously, half of the differences in D are positive and the other half are negative. The set P is said to be a *perfect simple difference set of order q* if and only if it has the following properties :

1. All the positive differences in D are distinct.
2. All the negative differences in D are distinct.

3.If $l_j - l_i$ is a negative difference in D , then $q(q+1)+1+(l_j - l_i)$ is not equal to any positive difference in D .

● Clearly, it follows from the definition that

$P' = \{0, l_1 - l_0, l_2 - l_0, \dots, l_q - l_0\}$ is also a perfect simple difference set.

Example 6 :

Consider the set $P = \{0,2,7,8,11\}$ with $q = 4$. The

$4 \cdot 5 = 20$ ordered differences are

$$D = \{2,7,8,11,5,6,9,1,4,3,-2,-7,-8,-11,-5,-6,-9,-1,-4,-3\}$$

It can be checked easily that P satisfies all three properties of a perfect simple difference set.

Singer has constructed perfect difference sets for order $q = p^s$, where p is a prime and s is any positive integer. In what follows we shall only be concerned with $q = 2^s = 2^2 = 4$

Let $P = \{l_0 = 0, l_1, l_2, l_3, \dots, l_{2^s}\}$ be a perfect simple difference set of order 2^s . Define the polynomial :

$$\begin{aligned} z(X) &= 1 + X^{l_1} + X^{l_2} + \dots + X^{l_{2^s}} \\ &= 1 + X^2 + X^7 + X^8 + X^{11} \end{aligned}$$

Let $n = 2^s(2^s+1)+1$ and $h(X)$ be the greatest common

divisor of $z(X)$ and $X^n + 1$, that is ,

$$\begin{aligned} h(X) &= GCD\{z(X), X^n + 1\} \\ &= GCD\{1 + X^2 + X^7 + X^8 + X^{11}, X^{21} + 1\} \\ &= 1 + X^2 + X^7 + X^8 + X^{11} \end{aligned}$$

The generator polynomial of the difference-set code of length $n = 21$ is :

$$\begin{aligned} g(X) &= \frac{X^n + 1}{h(X)} = \frac{X^{21} + 1}{h(X)} \\ &= 1 + X^2 + X^4 + X^6 + X^7 + X^{10} \end{aligned}$$

Thus, the code is a $(21,11)$ cyclic code.

This code has the following parameters :

$$\text{Code length : } n = 2^{2s} + 2^s + 1 = 21$$

$$\text{Number of parity-check digits : } n - k = 3^s + 1 = 10$$

$$\text{Minimum distance : } d = 2^s + 2$$

Let $h^*(X) = X^k h(X^{-1})$ be the reciprocal polynomial of $h(X)$. Then the $(n, n-k)$ cyclic code generated by $h^*(X)$ is the null space of the difference-set code generated by $g(X)$. Let

$$z^*(X) = X^{11} z(X^{-1}) = 1 + X^3 + X^4 + X^9 + X^{11}$$

Since $z(X)$ is divisible by $h(X)$, $z^*(X)$ is divisible by $h^*(X)$. Thus, $z^*(X)$ is in the null space of the difference-set code generated by $g(X)$. Let

$$\begin{aligned} w_0(X) &= X^{n-1-l_2s} z^*(X) = X^9 z^*(X) \\ &= X^9 + X^{12} + X^{13} + X^{18} + X^{20} \end{aligned}$$

By shifting $w_0(X)$ cyclically to the right 2 times, 7 times, 8 times, and 11 times, we obtain

$$W_1(X) = X + X^{11} + X^{14} + X^{15} + X^{20}$$

$$W_2(X) = X^4 + X^6 + X^{16} + X^{19} + X^{20}$$

$$W_3(X) = 1 + X^5 + X^7 + X^{17} + X^{20}$$

$$W_4(X) = X^2 + X^3 + X^8 + X^{10} + X^{20}$$

Clearly, $w_0(X)$, $w_1(X)$, $w_2(X)$, $w_3(X)$, $w_4(X)$, are five polynomials orthogonal on X^{20} . From these five orthogonal polynomials, we can form the following five parity-check sums orthogonal on e_{20} :

$$A_1 = s_9 = e_9 + e_{12} + e_{13} + e_{18} + e_{20}$$

$$A_2 = s_1 = e_1 + e_{11} + e_{14} + e_{15} + e_{20}$$

$$A_3 = s_4 + s_6 = e_4 + e_6 + e_{16} + e_{19} + e_{20}$$

$$A_4 = s_0 + s_5 + s_7 = e_0 + e_5 + e_7 + e_{17} + e_{20}$$

$$A_5 = s_2 + s_3 + s_8 = e_2 + e_3 + e_8 + e_{10} + e_{20}$$

A type II majority-logic decoder for this code is shown below.

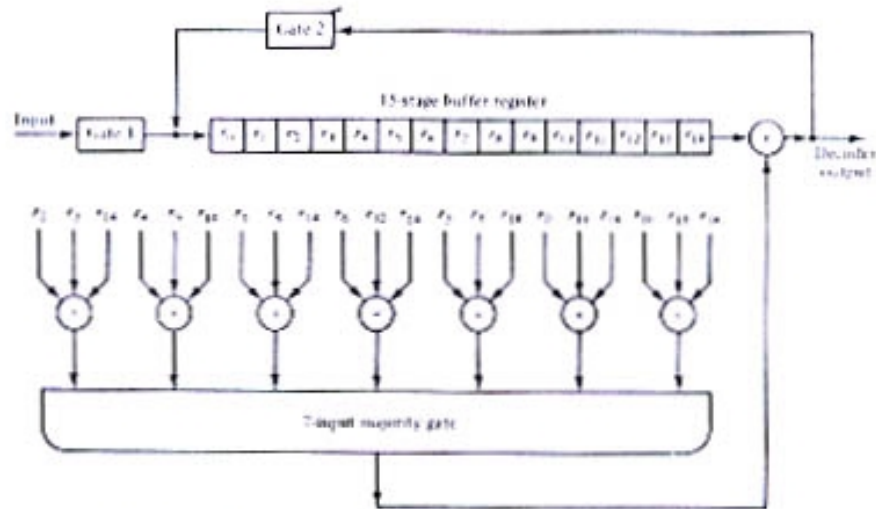


Figure 7.6 Type II majority-logic decoder for the (15, 4) maximum-length code.

List of Binary Difference-Set Cyclic codes

TABLE 7.2 LIST OF BINARY DIFFERENCE-SET CYCLIC CODES

λ	n	k	d	t	Generator polynomial, $g(X)^*$	Associated difference set
1	7	3	4	1	0, 2, 3, 4	0, 2, 3
2	21	11	6	2	0, 2, 4, 6, 7, 10	0, 2, 7, 8, 11
3	73	45	10	4	0, 2, 4, 6, 8, 12, 16, 22, 25, 28	0, 7, 10, 24, 25, 29, 36, 42, 45
4	273	191	18	8	0, 4, 10, 18, 22, 24, 34, 36, 40, 48, 52, 56, 66, 67, 71, 76, 77, 82	0, 18, 24, 46, 50, 67, 103, 112, 115, 126, 128, 159, 166, 167, 186, 196, 201
5	1057	813	34	16	0, 1, 3, 4, 5, 11, 14, 17, 18, 22, 23, 26, 27, 28, 32, 33, 35, 37, 39, 41, 43, 45, 47, 48, 51, 52, 55, 59, 62, 68, 70, 71, 72, 74, 75, 76, 79, 81, 83, 88, 95, 96, 98, 101, 103, 105, 106, 108, 111, 114, 115, 116, 120, 121, 122, 123, 124, 126, 129, 131, 132, 135, 137, 138, 141, 142, 146, 147, 149, 150, 151, 153, 154, 155, 158, 160, 161, 164, 165, 166, 167, 169, 174, 175, 176, 177, 178, 179, 180, 181, 182, 183, 184, 186, 188, 189, 191, 193, 194, 195, 198, 199, 200, 201, 202, 203, 208, 209, 210, 211, 212, 214, 216, 222, 224, 226, 228, 232, 234, 236, 242, 244	0, 1, 3, 7, 15, 31, 54, 63, 109, 127, 138, 219, 255, 272, 298, 338, 348, 439, 452, 511, 528, 555, 592, 677, 697, 702, 792, 897, 905, 924, 990, 1023

*Each generator polynomial is represented by the exponents of its nonzero terms. For example, (0, 2, 3, 4) represents $g(X) = 1 + X^2 + X^3 + X^4$.

5. Multiple-Step Majority-Logic Decoding

- The one-step majority-logic decoding for a cyclic code is based on the condition that a set of J parity-check sums orthogonal on a single error digit can be formed. This decoding method is effective for codes that are completely orthogonalizable or for codes with large J compared to their minimum distance d_{min} .

Unfortunately, there are only several small classes of cyclic codes known to be in this category. However, the concept of parity-check sums orthogonal on a single error digit can be generalized in such a way that many cyclic codes can be decoded by employing several levels of majority-logic gates.

Let $E = \{e_{i_1}, e_{i_2}, \dots, e_{i_M}\}$ be a set of M error digits where $0 \leq i_1 < i_2 < \dots < i_M < n$. The integer M is called the size of E .

● **Definition :** A set of J paritycheck sums A_1, A_2, \dots, A_J is said to be orthogonal on the set E if and only if (1) every error digit e_{il} in E is checked by every check sum A_j for $1 \leq j \leq J$, and (2) no other error digit is checked by more than one check sum.

For example, the following four parity-check sums are orthogonal on the set $E = \{e_6, e_8\}$:

$$A_1 = e_0 + e_2 + e_6 + e_8 ,$$

$$A_2 = e_3 + e_4 + e_6 + e_8 ,$$

$$A_3 = e_1 + e_6 + e_7 + e_8 ,$$

$$A_4 = e_5 + e_6 + e_8 ,$$

● Following the same argument employed for one-step majority-logic decoding, the sum of error digits in E , $e_{i1} + e_{i2} + \dots + e_{iM}$, can be determined correctly from the check sums, A_1, A_2, \dots, A_j , orthogonal on E provided that there are $[J/2]$ or fewer errors in the error pattern e . This sum of error digits in E may be regarded as an

additional check sum and so can be used for decoding.

- Consider an (n,k) cyclic code C which is used for error control purpose in a communication system. Let $e = (e_0, e_1, \dots, e_{n-1})$ denote the error vector that occurs during the transmission of a code vector v in C . Let $E_1^1, E_2^1, \dots, E_i^1, \dots$ be some properly selected sets of error digits of e . Let $S(E_i^1)$ denote the modulo-2 sum of the error digits in E_i^1 . Suppose that, for each set E_i^1 , it is possible to form at least J parity-check sums orthogonal on it. Then the sum $S(E_i^1)$ can be estimated from these J orthogonal check sums. The estimation can be done by a J -input majority-logic gate with the J orthogonal check sums as inputs. The estimated value of $S(E_i^1)$ is the output of a majority-logic gate, which is 1 if and only if more than one-half of the input are 1; otherwise, it is 0. The estimation is correct provided that there are $[J/2]$ or fewer errors in the error vector e .

The sums, $S(E_1^1), S(E_2^1), \dots, S(E_i^1), \dots$ are then used to estimate the sums of error digits in the second selected sets, $E_1^2, E_2^2, \dots, E_i^2, \dots$ with size smaller than that of the first selected sets. Suppose that, for each set E_i^2 , it is possible to form J or more check sums orthogonal on it. Then the sum $S(E_i^2)$ can be determined correctly from the check sums orthogonal on E_i^2 provided that there are no more than $\lfloor J/2 \rfloor$ errors in e .

- A code is said to be L -step orthogonalizable (or L -step majority-logic decodable) if L -steps of orthogonalization are required to make a decoding decision on an error digit. The decoding process is called L -step majority-logic decoding.
- A code is said to be completely L -step orthogonalizable if J is one less than the minimum distance of the code (i.e., $J = d_{min} - 1$). Since

majority-logic gates are used to estimate selected sums of error digits at each step of orthogonalization, a total of L levels of majority-logic gates are required for decoding. The number of gates required at each level depends on the structure of the code.

● General L -step majority-logic decoder circuit :

Type I L -step majority-logic decoder

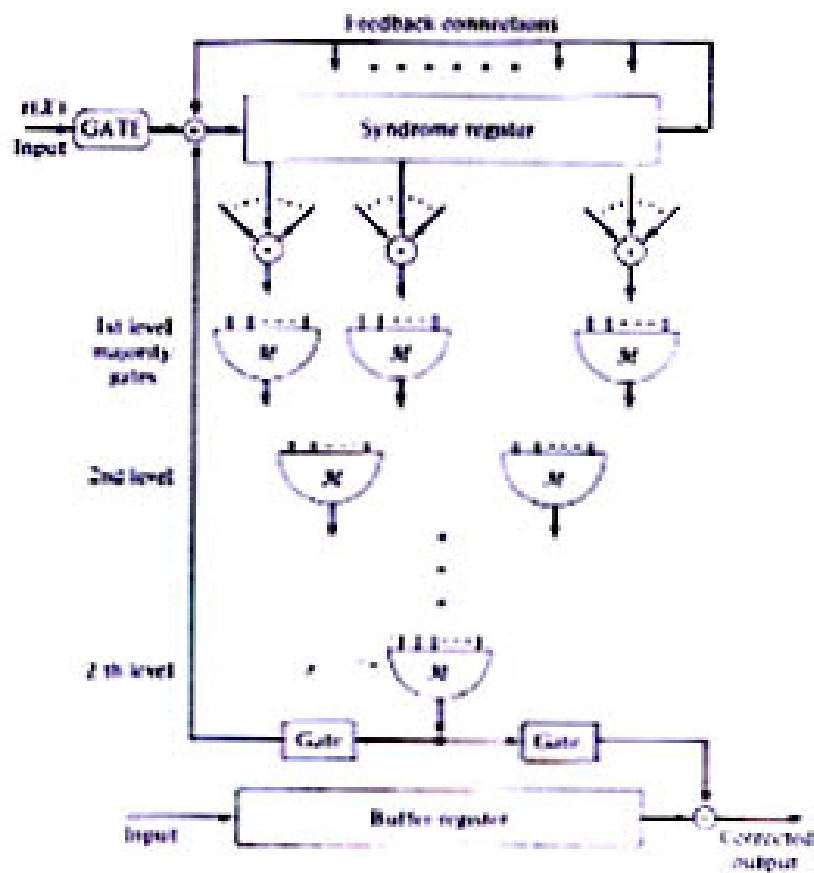


Figure 7.12 General type I L -step majority-logic decoder.

Type II L-step majority-logic decoder

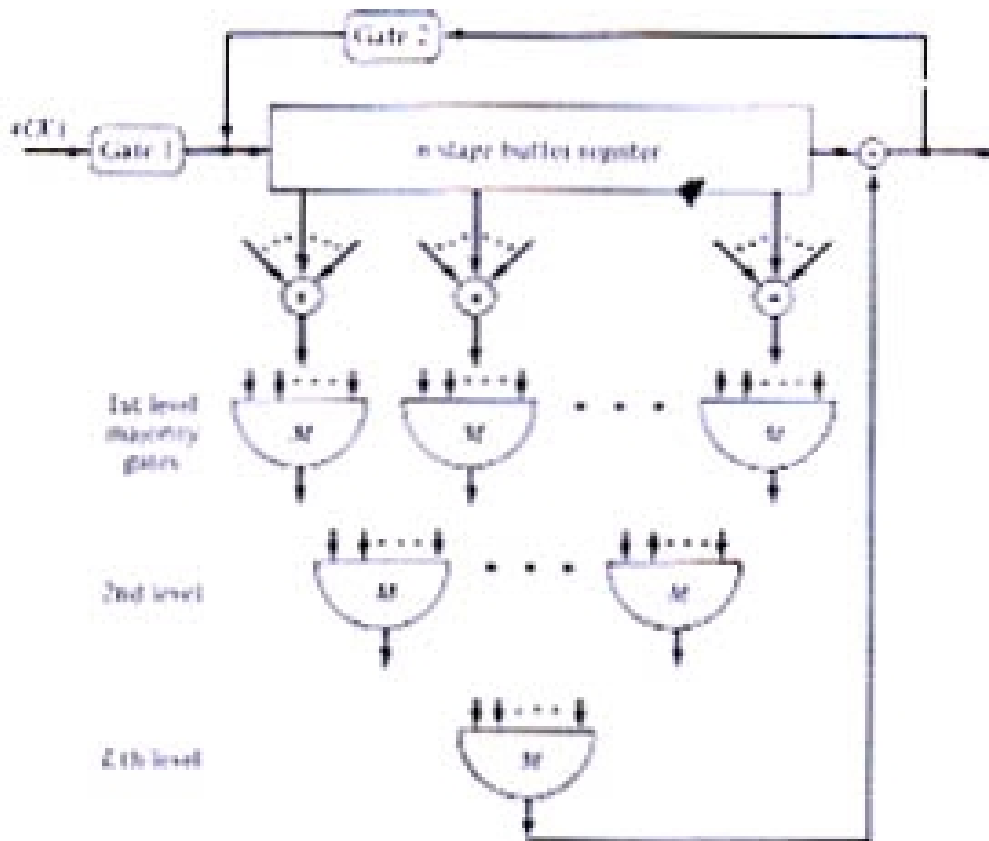


Figure 7.11 General type II L-step majority-logic decoder

Example 7 :

Consider the (7,4) cyclic code generated by $g(X)=1+X+X^3$, This is a Hamming code. The parity-check matrix (in systematic form) is found as follows :

$$H = \begin{bmatrix} h_0 \\ h_1 \\ h_2 \end{bmatrix} = \begin{bmatrix} 1001011 \\ 0101110 \\ 0010111 \end{bmatrix}$$

We see that the vectors, h_0 and h_2 are orthogonal on digit position 5 and 6 (or X^5 and X^6). We also see that the vectors, $h_0 + h_1$ and h_2 , are orthogonal on digit positions 4 and 6. Let $E_1^1 = \{e_5, e_6\}$ and $E_2^1 = \{e_4, e_6\}$ be two selected sets. Let $r = (r_0, r_1, r_2, r_3, r_4, r_5, r_6)$ be the received vector. Then the parity-check sums formed from h_0 and h_2 are :

$$A_1 = r \cdot h_0 = e_0 + e_3 + e_5 + e_6$$

$$A_2 = r \cdot h_2 = e_2 + e_4 + e_5 + e_6$$

and the parity-check sums formed from $h_0 + h_1$ and h_2 are

$$B_1 = r \cdot (h_0 + h_1) = e_0 + e_1 + e_4 + e_6$$

$$B_2 = r \cdot h_0 = e_2 + e_4 + e_5 + e_6$$

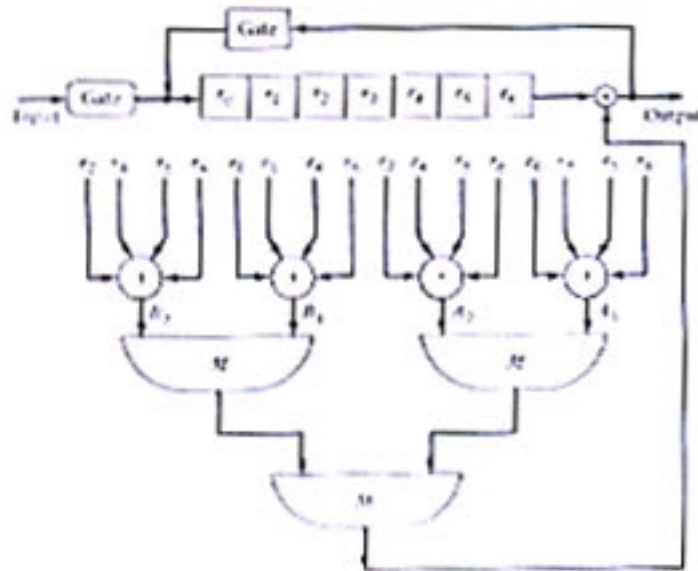
The parity-check sums, A_1 and A_2 , are orthogonal on the set $E_1^1 = \{e_5, e_6\}$ and the parity-check sums, B_1 and B_2 , are orthogonal on the set $E_2^1 = \{e_4, e_6\}$.

Therefore, the sum $S(E_1^1) = e_5 + e_6$ can be estimated from A_1 and A_2 , and the sum $S(E_2^1) = e_4 + e_6$ can be estimated from B_1 and B_2 . The sums $S(E_1^1)$ and $S(E_2^1)$ would be correctly estimated provided that there is no more than one error in the error vector e .

Now let $E_1^2 = \{e_6\}$. We see that $S(E_1^1)$ and $S(E_2^1)$ are orthogonal on e_6 . Hence, e_6 can be estimated from $S(E_1^1)$ and $S(E_2^1)$. The value of e_6 will be estimated correctly provided that there are no more than one error in e . Therefore, its minimum distance is 3 and $J = 2$, it is two step completely orthogonalizable. A type II decoder for this code is shown in below.

Type II two-step majority-logic decoder for the (7,4) Hamming code.

Figure 7.8 Type II two-step majority-logic decoder for the (7,4) Hamming code.



Let $s = (s_0, s_1, s_2) = r \bullet H^T$ be the syndrome of the received vector r . Then we can form the parity-check sums $A_1, A_2, B_1,$ and B_2 from the syndrome digits as follows :

$$A_1 = s_0 \quad , \quad A_2 = s_2 \quad ,$$

$$B_1 = s_0 + s_1 \quad , \quad B_2 = s_2$$

Based on these check sums, one may construct a type I majority-logic decoder for the (7,4) Hamming code.

Example 8 :

Consider the triple-error-correcting (15,5) BCH code whose generator polynomial is :

$$g(X) = 1 + X + X^2 + X^4 + X^5 + X^8 + X^{10}$$

Type II two-step majority-logic decoder for the (15,5) BCH code.

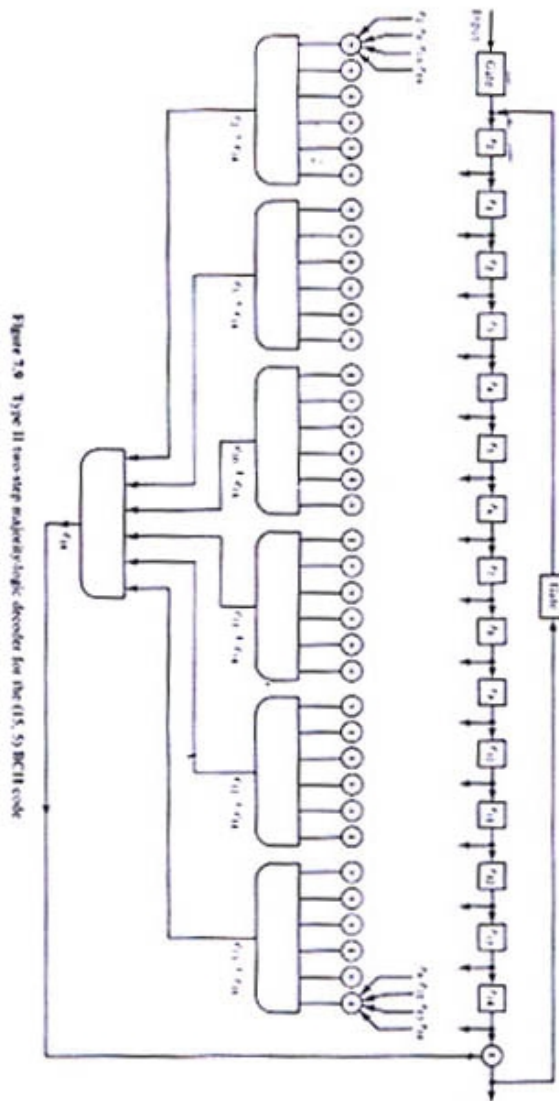


Figure 7.9 Type II two-step majority-logic decoder for the (15, 5) BCH code

Type II one-step majority-logic decoder for the (15,5) BCH code.

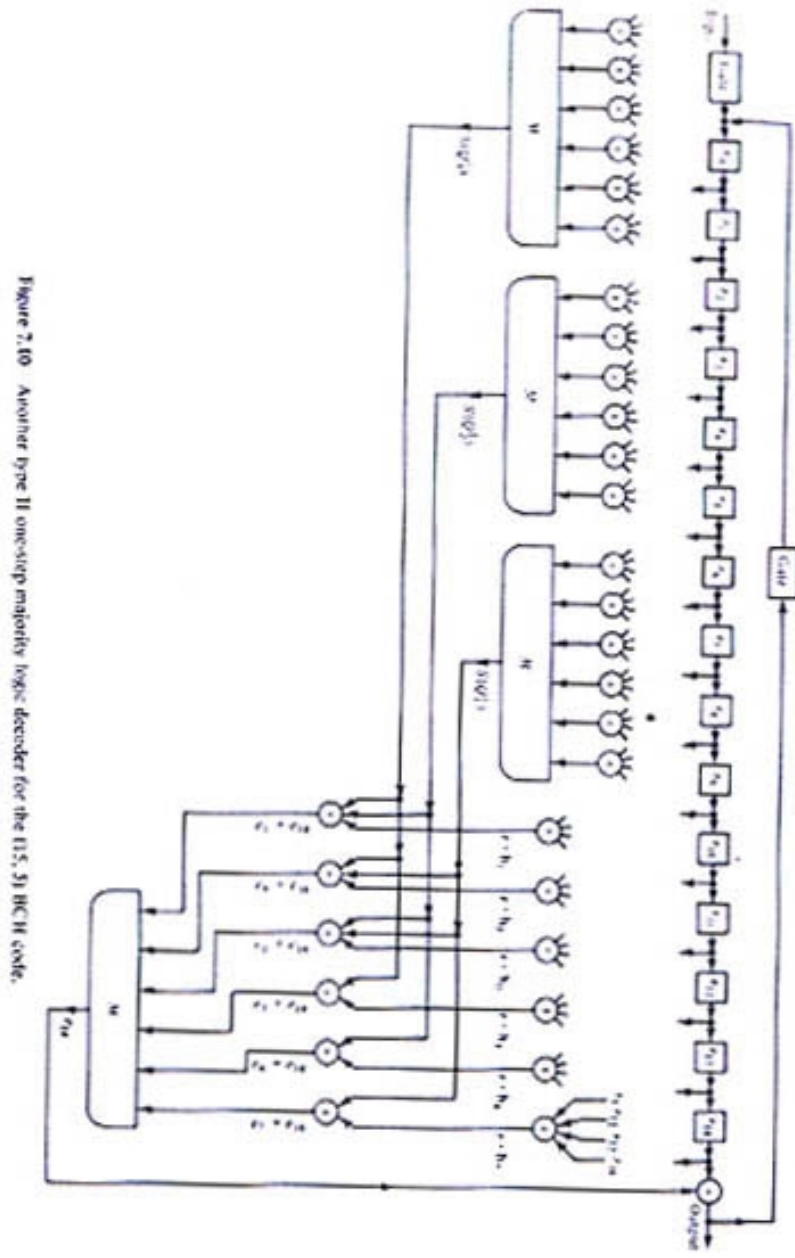


Figure 7.10 Another Type II one-step majority logic decoder for the (15, 5) BCH code.

Books :

Lin , p.184 ~ p.222

Reporter :

8614067 張德威

8611018 蔡志侃