CHAPTER 3

Linear Block Codes

In this chapter we introduce basic concepts of block codes. For ease of code synthesis and implementation, we restrict our attention to a subclass of the class of all block codes, the *linear block codes*. Because information in most current digital data communication and storage systems is coded in the binary digits 0 and 1, we discuss only the linear block codes with symbols from the binary field $GF(2)$. It is straightforward to generalize the theory developed for the binary codes to codes with symbols from a nonbinary field.

First, linear block codes are defined and described in terms of *generator* and *parity-check* matrices. The parity-check equations for a *systematic* code are derived. Encoding of linear block codes is discussed. In Section 3.2 the concept of *syndrome* is introduced. The use of syndrome for error detection and correction is discussed. In Sections 3.3 and 3.4 we define the *minimum distance* of a block code and show that the random-error-detecting and random-error-correcting capabilities of a code are determined by its minimum distance. Probabilities of a decoding error are discussed. In Section 3.5 the *standard array* and its application to the decoding of linear block codes are presented. A general decoder based on the *syndrome decoding* scheme is given.

References [1] through [6] contain excellent treatments of linear block codes.

3.1 INTRODUCTION TO LINEAR BLOCK CODES

We assume that the output of an information source is a sequence of the binary digits 0 and 1. In block coding, this binary information sequence is segmented into *message blocks* of fixed length; each message block, denoted by $\mathbf{u}$, consists of k information digits. There are a total of 2^k distinct messages. The encoder, *according to certain rules*, transforms each input message $\mathbf{u}$ into a binary n-tuple $\mathbf{v}$ with $n > k$. This binary n-tuple $\mathbf{v}$ is referred to as the *codeword* (or *code vector*) of the message $\mathbf{u}$. Therefore, corresponding to the 2^k possible messages, there are 2^k codewords. This set of 2^k codewords is called a *block code*. For a block code to be useful, the 2^k codewords must be distinct. Therefore, there should be a one-to-one correspondence between a message $\mathbf{u}$ and its codeword $\mathbf{v}$.

For a block code with 2^k codewords and length n, unless it has a certain special structure, the encoding apparatus will be prohibitively complex for large k and n, since it has to store the 2^k codewords of length n in a dictionary. Therefore, we restrict our attention to block codes that can be mechanized in a practical manner. A desirable structure for a block code to possess is *linearity*, which greatly reduces the encoding complexity, as we shall see.

DEFINITION 3.1 A block code of length n and 2^k codewords is called a *linear* (n, k) code if and only if its 2^k codewords form a k-dimensional subspace of the vector space of all the n-tuples over the field $GF(2)$.

In fact, a binary block code is linear if and only if the modulo-2 sum of two codewords is also a codeword. The block code given in Table 3.1 is a (7, 4) linear code. One can easily check that the sum of any two codewords in this code is also a codeword.

Because an (n, k) linear code C is a k-dimensional subspace of the vector space V_n of all the binary n-tuples, it is possible to find k linearly independent codewords, $\mathbb{g}_0, \mathbb{g}_1, \cdots, \mathbb{g}_{k-1}$, in C such that every codeword $\mathbb{v}$ in C is a linear combination of these k codewords; that is,

$$\mathbb{v} = u_0\mathbb{g}_0 + u_1\mathbb{g}_1 + \cdots + u_{k-1}\mathbb{g}_{k-1}, \tag{3.1}$$

where $u_i = 0$ or 1 for $0 \leq i < k$. We arrange these k linearly independent codewords as the rows of a $k \times n$ matrix as follows:

$$\mathbb{G} = \begin{bmatrix} \mathbb{g}_0 \\ \mathbb{g}_1 \\ \vdots \\ \mathbb{g}_{k-1} \end{bmatrix} = \begin{bmatrix} g_{00} & g_{01} & g_{02} & \cdots & g_{0,n-1} \\ g_{10} & g_{11} & g_{12} & \cdots & g_{1,n-1} \\ \vdots & \vdots & \vdots & & \vdots \\ g_{k-1,0} & g_{k-1,1} & g_{k-1,2} & \cdots & g_{k-1,n-1} \end{bmatrix}, \tag{3.2}$$

where $\mathbb{g}_i = (g_{i0}, g_{i1}, \cdots, g_{i,n-1})$ for $0 \leq i < k$. If $\mathbb{u} = (u_0, u_1, \cdots, u_{k-1})$ is the message to be encoded, the corresponding codeword can be given as follows:

$$\begin{aligned} \mathbb{v} &= \mathbb{u} \cdot G \\ &= (u_0, u_1, \cdots, u_{k-1}) \cdot \begin{bmatrix} \mathbb{g}_0 \\ \mathbb{g}_1 \\ \vdots \\ \mathbb{g}_{k-1} \end{bmatrix} \\ &= u_0\mathbb{g}_0 + u_1\mathbb{g}_1 + \cdots + u_{k-1}\mathbb{g}_{k-1}. \end{aligned} \tag{3.3}$$

Clearly, the rows of $\mathbb{G}$ *generate* (or *span*) the (n, k) linear code C. For this reason, the matrix $\mathbb{G}$ is called a *generator* matrix for C. Note that any k linearly independent codewords of an (n, k) linear code can be used to form a generator matrix for the code. It follows from (3.3) that an (n, k) linear code is completely specified by the k rows of a generator matrix $\mathbb{G}$. Therefore, the encoder has only to store the k rows of $\mathbb{G}$ and to form a linear combination of these k rows based on the input message $\mathbb{u} = (u_0, u_1, \cdots, u_{k-1})$.

EXAMPLE 3.1

The (7, 4) linear code given in Table 3.1 has the following matrix as a generator matrix:

$$\mathbb{G} = \begin{bmatrix} \mathbb{g}_0 \\ \mathbb{g}_1 \\ \mathbb{g}_2 \\ \mathbb{g}_3 \end{bmatrix} = \begin{bmatrix} 1 & 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 1 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 & 1 \end{bmatrix}.$$

TABLE 3.1: Linear block code with $k = 4$ and $n = 7$.

Messages	Codewords
(0 0 0 0)	(0 0 0 0 0 0 0)
(1 0 0 0)	(1 1 0 1 0 0 0)
(0 1 0 0)	(0 1 1 0 1 0 0)
(1 1 0 0)	(1 0 1 1 1 0 0)
(0 0 1 0)	(1 1 1 0 0 1 0)
(1 0 1 0)	(0 0 1 1 0 1 0)
(0 1 1 0)	(1 0 0 0 1 1 0)
(1 1 1 0)	(0 1 0 1 1 1 0)
(0 0 0 1)	(1 0 1 0 0 0 1)
(1 0 0 1)	(0 1 1 1 0 0 1)
(0 1 0 1)	(1 1 0 0 1 0 1)
(1 1 0 1)	(0 0 0 1 1 0 1)
(0 0 1 1)	(0 1 0 0 0 1 1)
(1 0 1 1)	(1 0 0 1 0 1 1)
(0 1 1 1)	(0 0 1 0 1 1 1)
(1 1 1 1)	(1 1 1 1 1 1 1)

If $\mathbf{u} = (1\ 1\ 0\ 1)$ is the message to be encoded, its corresponding codeword, according to (3.3), will be

$$\begin{aligned}\mathbf{v} &= 1 \cdot \mathbf{g}_0 + 1 \cdot \mathbf{g}_1 + 0 \cdot \mathbf{g}_2 + 1 \cdot \mathbf{g}_3 \\ &= (1\ 1\ 0\ 1\ 0\ 0\ 0) + (0\ 1\ 1\ 0\ 1\ 0\ 0) + (1\ 0\ 1\ 0\ 0\ 0\ 1) \\ &= (0\ 0\ 0\ 1\ 1\ 0\ 1).\end{aligned}$$

A desirable property for a linear block code to possess is the *systematic structure* of the codewords, as shown in Figure 3.1, in which a codeword is divided into two parts, the message part and the redundant checking part. The message part consists of k unaltered information (or message) digits, and the redundant checking part consists of $n - k$ *parity-check* digits, which are *linear sums* of the information digits. A linear block code with this structure is referred to as a *linear systematic block code*. The (7, 4) code given in Table 3.1 is a linear systematic block

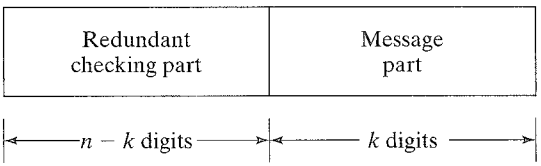

FIGURE 3.1: Systematic format of a codeword.

code; the rightmost four digits of each codeword are identical to the corresponding information digits.

A linear systematic (n, k) code is completely specified by a $k \times n$ matrix $\mathbb{G}$ of the following form:

$$\mathbb{G} = \begin{bmatrix} \mathbb{g}_0 \\ \mathbb{g}_1 \\ \mathbb{g}_2 \\ \vdots \\ \mathbb{g}_{k-1} \end{bmatrix} = \left[\begin{array}{cccc|ccccc} \overbrace{p_{00} & p_{01} & \cdots & p_{0,n-k-1}}^{p \text{ matrix}} & \overbrace{1 & 0 & 0 & \cdots & 0}^{k \times k \text{ identity matrix}} \\ p_{10} & p_{11} & \cdots & p_{1,n-k-1} & 0 & 1 & 0 & \cdots & 0 \\ p_{20} & p_{21} & \cdots & p_{2,n-k-1} & 0 & 0 & 1 & \cdots & 0 \\ & & & & & & & & \\ p_{k-1,0} & p_{k-1,1} & \cdots & p_{k-1,n-k-1} & 0 & 0 & 0 & \cdots & 1 \end{array}\right]. \tag{3.4}$$

where $p_{ij} = 0$ or 1. Let $\mathbb{I}_k$ denote the $k \times k$ identity matrix. Then, $\mathbb{G} = [\mathbb{P}\ \mathbb{I}_k]$. Let $\mathbb{u} = (u_0, u_1, \cdots, u_{k-1})$ be the message to be encoded. The corresponding codeword is

$$\begin{aligned} \mathbb{v} &= (v_0, v_1, v_2, \cdots, v_{n-1}) \\ &= (u_0, u_1, \cdots, u_{k-1}) \cdot \mathbb{G}. \end{aligned} \tag{3.5}$$

It follows from (3.4) and (3.5) that the components of $\mathbb{v}$ are

$$v_{n-k+i} = u_i \quad \text{for } 0 \leq i < k \tag{3.6a}$$

and

$$v_j = u_0 p_{0j} + u_1 p_{1j} + \cdots + u_{k-1} p_{k-1,j} \tag{3.6b}$$

for $0 \leq j < n - k$. Equation (3.6a) shows that the rightmost k digits of a codeword $\mathbb{v}$ are identical to the information digits $u_0, u_1, \cdots, u_{k-1}$ to be encoded, and (3.6b) shows that the leftmost $n - k$ redundant digits are linear sums of the information digits. The $n - k$ equations given by (3.6b) are called *parity-check equations* of the code.

EXAMPLE 3.2

The matrix $\mathbb{G}$ given in Example 3.1 is in systematic form. Let $\mathbb{u} = (u_0, u_1, u_2, u_3)$ be the message to be encoded, and let $\mathbb{v} = (v_0, v_1, v_2, v_3, v_4, v_5, v_6)$ be the corresponding codeword. Then,

$$\mathbb{v} = (u_0, u_1, u_2, u_3) \cdot \begin{bmatrix} 1 & 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 1 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 & 1 \end{bmatrix}.$$

By matrix multiplication, we obtain the following digits of the codeword $\mathbb{v}$:

$$v_6 = u_3$$

$$v_5 = u_2$$

$$v_4 = u_1$$

$$v_3 = u_0$$

$$v_2 = u_1 + u_2 + u_3$$

$$v_1 = u_0 + u_1 + u_2$$

$$v_0 = u_0 + u_2 + u_3.$$

The codeword corresponding to the message (1 0 1 1) is (1 0 0 1 0 1 1).

There is another useful matrix associated with every linear block code. As stated in Chapter 2, for any $k \times n$ matrix $\mathbb{G}$ with k linearly independent rows, there exists an $(n-k) \times n$ matrix $\mathbb{H}$ with $n-k$ linearly independent rows such that any vector in the row space of $\mathbb{G}$ is orthogonal to the rows of $\mathbb{H}$, and any vector that is orthogonal to the rows of $\mathbb{H}$ is in the row space of $\mathbb{G}$. Hence, we can describe the (n, k) linear code C generated by $\mathbb{G}$ in an alternative way as follows: *An n-tuple* $\mathbf{v}$ *is a codeword in the code C generated by* $\mathbb{G}$ *if and only if* $\mathbf{v} \cdot \mathbb{H}^T = \mathbb{0}$. The code is said to be the null space of $\mathbb{H}$. This matrix $\mathbb{H}$ is called a *parity-check matrix* of the code. The 2^{n-k} linear combinations of the rows of matrix $\mathbb{H}$ form an $(n, n-k)$ linear code C_d. This code is the null space of the (n, k) linear code C generated by matrix $\mathbb{G}$ (i.e., for any $\mathbf{v} \in C$ and any $\mathbf{w} \in C_d$, $\mathbf{v} \cdot \mathbf{w} = 0$). C_d is called the *dual code* of C. Therefore, a parity-check matrix for a linear code C is a generator matrix for its dual code C_d.

If the generator matrix of an (n, k) linear code is in the systematic form of (3.4), the parity-check matrix may take the following form:

$$\mathbb{H} = [\mathbb{I}_{n-k} \mathbb{P}^T] = \begin{bmatrix} 1 & 0 & 0 & \cdots & 0 & p_{00} & p_{10} & \cdots & p_{k-1,0} \\ 0 & 1 & 0 & \cdots & 0 & p_{01} & p_{11} & \cdots & p_{k-1,1} \\ 0 & 0 & 1 & \cdots & 0 & p_{02} & p_{12} & \cdots & p_{k-1,2} \\ \vdots & & & & & & & & \\ 0 & 0 & 0 & \cdots & 1 & p_{0,n-k-1} & p_{1,n-k-1} & \cdots & p_{k-1,n-k-1} \end{bmatrix}, \tag{3.7}$$

where $\mathbb{P}^T$ is the transpose of the matrix $\mathbb{P}$. Let $\mathbb{h}_j$ be the jth row of $\mathbb{H}$. We can readily check that the inner product of the ith row of $\mathbb{G}$ given by (3.4) and the jth row of $\mathbb{H}$ given by (3.7) is

$$\mathbb{g}_i \cdot \mathbb{h}_j = p_{ij} + p_{ij} = 0$$

for $0 \leq i < k$ and $0 \leq j < n-k$. This implies that $\mathbb{G} \cdot \mathbb{H}^T = \mathbb{0}$. Also, the $n-k$ rows of $\mathbb{H}$ are linearly independent. Therefore, the $\mathbb{H}$ matrix of (3.7) is a parity-check matrix of the (n, k) linear code generated by the matrix $\mathbb{G}$ of (3.4).

The parity-check equations given by (3.6b) also can be obtained from the parity-check matrix $\mathbb{H}$ of (3.7). Let $\mathbf{u} = (u_0, u_1, \cdots, u_{k-1})$ be the message to be encoded. In systematic form the corresponding codeword will be

$$\mathbf{v} = (v_0, v_1, \cdots, v_{n-k-1}, u_0, u_1, \cdots, u_{k-1}).$$

Using the fact that $\mathbf{v} \cdot \mathbb{H}^T = \mathbb{0}$, we obtain

$$v_j + u_0 p_{0j} + u_1 p_{1j} + \cdots + u_{k-1} p_{k-1,j} = 0 \tag{3.8}$$

for $0 \leq j < n-k$. Rearranging the equations of (3.8), we obtain the same parity-check equations of (3.6b). Therefore, an (n, k) linear code is completely specified by its parity-check matrix.

EXAMPLE 3.3

Consider the generator matrix of the (7, 4) linear code given in Example 3.1. The corresponding parity-check matrix is

$$\mathbb{H} = \begin{bmatrix} 1 & 0 & 0 & 1 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 & 1 & 1 \end{bmatrix}.$$

At this point we summarize the foregoing results: For any (n, k) linear block code C there exists a $k \times n$ matrix $\mathbb{G}$ whose row space gives C. Furthermore, there exists an $(n-k) \times n$ matrix $\mathbb{H}$ such that an n-tuple $\mathbb{v}$ is a codeword in C if and only if $\mathbb{v} \cdot \mathbb{H}^T = \mathbb{0}$. If $\mathbb{G}$ is of the form given by (3.4), then $\mathbb{H}$ may take the form given by (3.7), and vice versa.

Based on the equations of (3.6a) and (3.6b), the encoding circuit for an (n, k) linear systematic code can easily be implemented. The encoding circuit is shown in Figure 3.2, where $\rightarrow\boxed{\ }\rightarrow$ denotes a shift-register stage (e.g., a flip-flop), $\oplus$ denotes a modulo-2 adder, and $\longrightarrow (p_{ij}) \longrightarrow$ denotes a connection if $p_{ij} = 1$, and no connection if $p_{ij} = 0$. The encoding operation is very simple. The message $\mathbb{u} = (u_0, u_1, \cdots, u_{k-1})$ to be encoded is shifted into the message register and simultaneously into the channel. As soon as the entire message has entered the message register the $n-k$ parity-check digits are formed at the outputs of the $n-k$ modulo-2 adders. These parity-check digits are then serialized and shifted into the channel. We see that the complexity of the encoding circuit is linearly proportional to the block length of the code. The encoding circuit for the (7, 4) code given in Table 3.1 is shown in Figure 3.3, where the connection is based on the parity-check equations given in Example 3.2.

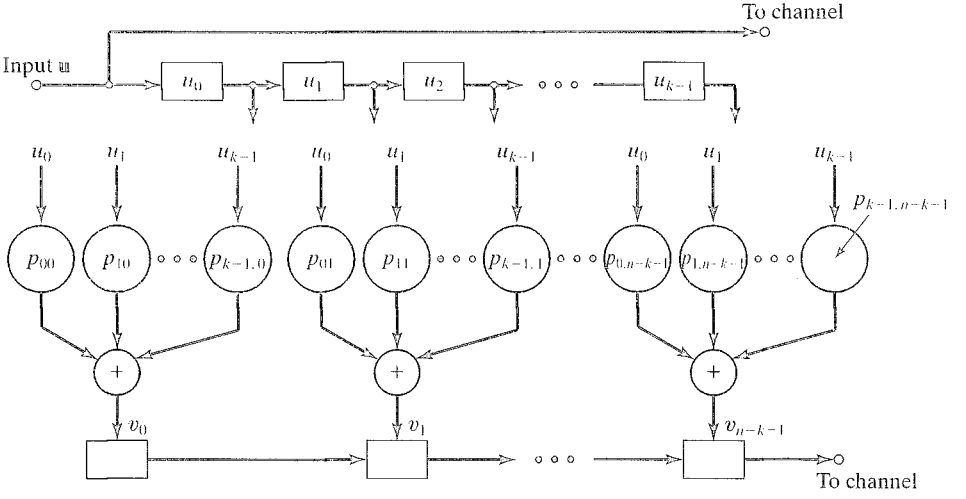

FIGURE 3.2: Encoding circuit for a linear systematic (n, k) code.

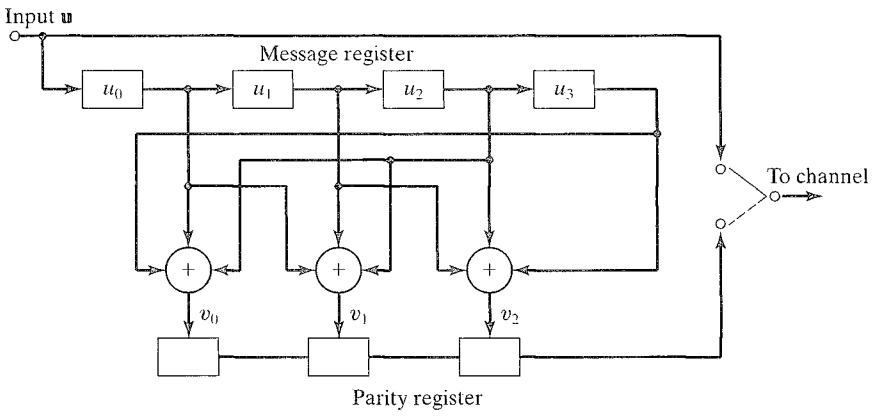

FIGURE 3.3: Encoding circuit for the (7, 4) systematic code given in Table 3.1.

3.2 SYNDROME AND ERROR DETECTION

Consider an (n, k) linear code with generator matrix $\mathbb{G}$ and parity-check matrix $\mathbb{H}$. Let $\mathbb{v} = (v_0, v_1, \cdots, v_{n-1})$ be a codeword that was transmitted over a noisy channel. Let $\mathbb{r} = (r_0, r_1, \cdots, r_{n-1})$ be the received vector at the output of the channel. Because of the channel noise, $\mathbb{r}$ may be different from $\mathbb{v}$. The vector sum

$$\begin{aligned} \mathbb{e} &= \mathbb{r} + \mathbb{v} \\ &= (e_0, e_1, \cdots, e_{n-1}) \end{aligned} \tag{3.9}$$

is an n-tuple, where $e_i = 1$ for $r_i \neq v_i$, and $e_i = 0$ for $r_i = v_i$. This n-tuple is called the *error vector* (or *error pattern*), which simply displays the positions where the received vector $\mathbb{r}$ differ from the transmitted codeword $\mathbb{v}$. The 1's in $\mathbb{e}$ are the *transmission errors* caused by the channel noise. It follows from (3.9) that the received vector $\mathbb{r}$ is the vector sum of the transmitted codeword and the error vector; that is,

$$\mathbb{r} = \mathbb{v} + \mathbb{e}.$$

Of course, the receiver does not know either $\mathbb{v}$ or $\mathbb{e}$. On receiving $\mathbb{r}$, the decoder must first determine whether $\mathbb{r}$ contains transmission errors. If the presence of errors is detected, the decoder will either take actions to locate the errors and correct them (FEC) or request a retransmission of $\mathbb{v}$ (ARQ).

When $\mathbb{r}$ is received, the decoder computes the following $(n - k)$-tuple:

$$\begin{aligned} \mathbb{s} &= \mathbb{r} \cdot \mathbb{H}^T \\ &= (s_0, s_1, \cdots, s_{n-k-1}). \end{aligned} \tag{3.10}$$

which is called the *syndrome* of $\mathbb{r}$. Then, $\mathbb{s} = \mathbb{0}$ if and only if $\mathbb{r}$ is a codeword, and $\mathbb{s} \neq \mathbb{0}$ if and only if $\mathbb{r}$ is not a codeword. Therefore, when $\mathbb{s} \neq \mathbb{0}$, we know that $\mathbb{r}$ is not a codeword and the presence of errors has been detected. When $\mathbb{s} = \mathbb{0}$, $\mathbb{r}$ is a codeword, and the receiver accepts $\mathbb{r}$ as the transmitted codeword. It is possible that the errors in certain error vectors are not detectable (i.e., $\mathbb{r}$ contains errors but $\mathbb{s} = \mathbb{r} \cdot \mathbb{H}^T = \mathbb{0}$).

This happens when the error pattern $\mathbb{e}$ is identical to a nonzero codeword. In this event, $\mathbb{r}$ is the sum of two codewords, which is a codeword, and consequently $\mathbb{r} \cdot \mathbb{H}^T = \mathbb{0}$. Error patterns of this kind are called *undetectable* error patterns. Because there are $2^k - 1$ nonzero codewords, there are $2^k - 1$ undetectable error patterns. When an undetectable error pattern occurs, the decoder makes a *decoding error*. In a later section of this chapter we derive the probability of an undetected error for a BSC and show that this error probability can be made very small.

Based on (3.7) and (3.10), the syndrome digits are as follows:

$$\begin{aligned} s_0 &= r_0 + r_{n-k}p_{00} + r_{n-k+1}p_{10} + \cdots + r_{n-1}p_{k-1,0} \\ s_1 &= r_1 + r_{n-k}p_{01} + r_{n-k+1}p_{11} + \cdots + r_{n-1}p_{k-1,1} \\ &\vdots \\ s_{n-k-1} &= r_{n-k-1} + r_{n-k}p_{0,n-k-1} + r_{n-k+1}p_{1,n-k-1} + \cdots + r_{n-1}p_{k-1,n-k-1}. \end{aligned} \tag{3.11}$$

If we examine the preceding equations carefully, we find that the syndrome $\mathbb{s}$ is simply the vector sum of the received parity digits $(r_0, r_1, \cdots, r_{n-k-1})$ and the parity-check digits recomputed from the received information digits $(r_{n-k}, r_{n-k+1}, \cdots, r_{n-1})$. Therefore, the syndrome can be formed by a circuit similar to the encoding circuit. A general syndrome circuit is shown in Figure 3.4.

EXAMPLE 3.4

Consider the (7, 4) linear code whose parity-check matrix is given in Example 3.3. Let $\mathbb{r} = (r_0, r_1, r_2, r_3, r_4, r_5, r_6)$ be the received vector. Then, the syndrome is given by

$$\begin{aligned} \mathbb{s} &= (s_0, s_1, s_2) \\ &= (r_0, r_1, r_2, r_3, r_4, r_5, r_6) \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 1 & 0 \\ 0 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 0 & 1 \end{bmatrix}. \end{aligned}$$

The syndrome digits are

$$\begin{aligned} s_0 &= r_0 + r_3 + r_5 + r_6 \\ s_1 &= r_1 + r_3 + r_4 + r_5 \\ s_2 &= r_2 + r_4 + r_5 + r_6. \end{aligned}$$

The syndrome circuit for this code is shown in Figure 3.5.

The syndrome $\mathbb{s}$ computed from the received vector $\mathbb{r}$ depends only on the error pattern $\mathbb{e}$ and not on the transmitted codeword $\mathbb{v}$. Because $\mathbb{r}$ is the vector sum of $\mathbb{v}$ and $\mathbb{e}$, it follows from (3.10) that

$$\mathbb{s} = \mathbb{r} \cdot \mathbb{H}^T = (\mathbb{v} + \mathbb{e})\mathbb{H}^T = \mathbb{v} \cdot \mathbb{H}^T + \mathbb{e} \cdot \mathbb{H}^T;$$

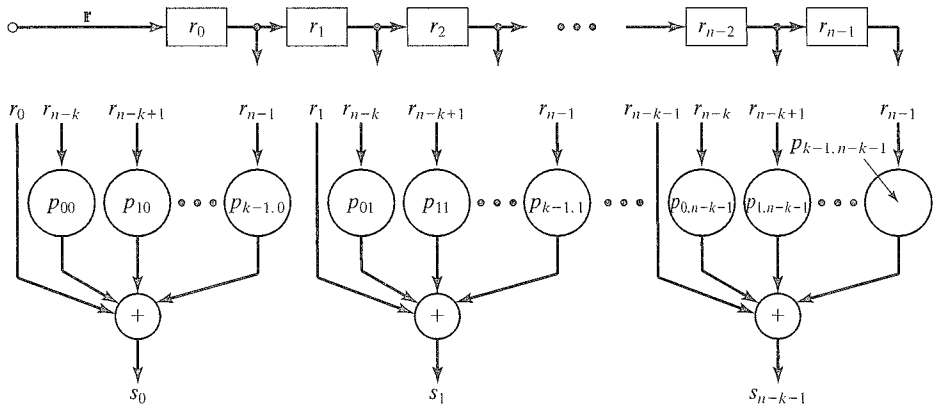

FIGURE 3.4: Syndrome circuit for a linear systematic (n, k) code.

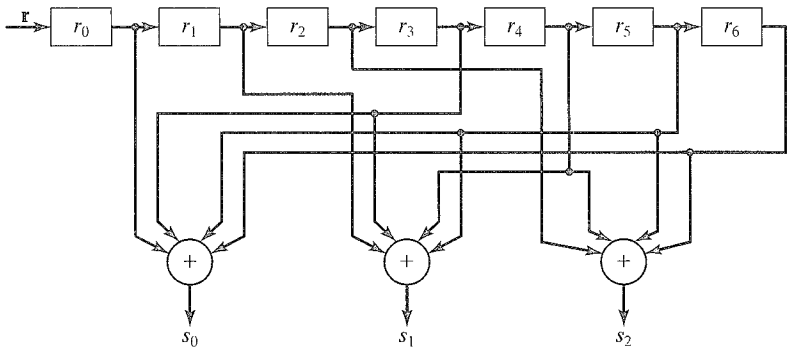

FIGURE 3.5: Syndrome circuit for the (7, 4) code given in Table 3.1.

however, $\mathbb{v} \cdot \mathbb{H}^T = \mathbb{0}$. Consequently, we obtain the following relation between the syndrome and the error pattern:

$$\mathbb{s} = \mathbb{e} \cdot \mathbb{H}^T. \tag{3.12}$$

If the parity-check matrix $\mathbb{H}$ is expressed in the systematic form as given by (3.7), multiplying out $\mathbb{e} \cdot \mathbb{H}^T$ yields the following linear relationship between the syndrome digits and the error digits:

$$\begin{aligned} s_0 &= e_0 + e_{n-k}p_{00} + e_{n-k+1}p_{10} + \cdots + e_{n-1}p_{k-1,0} \\ s_1 &= e_1 + e_{n-k}p_{01} + e_{n-k+1}p_{11} + \cdots + e_{n-1}p_{k-1,1} \\ &\vdots \\ s_{n-k-1} &= e_{n-k-1} + e_{n-k}p_{0,n-k-1} + e_{n-k+1}p_{1,n-k-1} + \cdots + e_{n-1}p_{k-1,n-k-1}. \end{aligned} \tag{3.13}$$

The syndrome digits are simply linear combinations of the error digits. Clearly, they provide information about the error digits and therefore can be used for error correction.

At this point, one would think that any error correction scheme would be a method of solving the $n - k$ linear equations of (3.13) for the error digits. Once the error pattern $\mathbf{e}$ was found, the vector $\mathbf{r} + \mathbf{e}$ would be taken as the actual transmitted codeword. Unfortunately, determining the true error vector $\mathbf{e}$ is not a simple matter. This is because the $n - k$ linear equations of (3.13) do not have a unique solution but have 2^k solutions (this will be proved in Theorem 3.6). In other words, there are 2^k error patterns that result in the same syndrome, and the true error pattern $\mathbf{e}$ is just one of them. Therefore, the decoder has to determine the true error vector from a set of 2^k candidates. To minimize the probability of a decoding error, the most *probable* error pattern that satisfies the equations of (3.13) is chosen as the true error vector. If the channel is a BSC, the most probable error pattern is the one that has the smallest number of nonzero digits.

The notion of using syndrome for error correction may be clarified by an example.

EXAMPLE 3.5

Again, we consider the (7, 4) code whose parity-check matrix is given in Example 3.3. Let $\mathbf{v} = (1\,0\,0\,1\,0\,1\,1)$ be the transmitted codeword and $\mathbf{r} = (1\,0\,0\,1\,0\,0\,1)$ be the received vector. On receiving $\mathbf{r}$, the receiver computes the syndrome:

$$\mathbf{s} = \mathbf{r} \cdot \mathbf{H}^T = (1\,1\,1).$$

Next, the receiver attempts to determine the true error vector $\mathbf{e} = (e_0, e_1, e_2, e_3, e_4, e_5, e_6)$, which yields the given syndrome. It follows from (3.12) or (3.13) that the error digits are related to the syndrome digits by the following linear equations:

$$1 = e_0 + e_3 + e_5 + e_6$$
$$1 = e_1 + e_3 + e_4 + e_5$$
$$1 = e_2 + e_4 + e_5 + e_6.$$

There are $2^4 = 16$ error patterns that satisfy the preceding equations, namely,

(0000010),	(1010011),
(1101010),	(0111011),
(0110110),	(1100111),
(1011110),	(0001111),
(1110000),	(0100001),
(0011000),	(1001001),
(1000100),	(0010101),
(0101100),	(1111101).

The error vector $\mathbf{e} = (0\,0\,0\,0\,0\,1\,0)$ has the smallest number of nonzero components. If the channel is a BSC, $\mathbf{e} = (0\,0\,0\,0\,0\,1\,0)$ is the most probable error vector that satisfies the preceding equations. Taking $\mathbf{e} = (0\,0\,0\,0\,0\,1\,0)$ as the true error vector, the

receiver decodes the received vector $\mathbf{r} = (1\,0\,0\,1\,0\,0\,1)$ into the following codeword:

$$\begin{aligned}\mathbf{v}^* &= \mathbf{r} + \mathbf{e} \\ &= (1\,0\,0\,1\,0\,0\,1) + (0\,0\,0\,0\,0\,1\,0) \\ &= (1\,0\,0\,1\,0\,1\,1).\end{aligned}$$

We see that $\mathbf{v}^*$ is the actual transmitted codeword. Hence, the receiver has performed a correct decoding. Later we show that the (7, 4) linear code considered in this example is capable of correcting any single error over a span of seven digits; that is, if a codeword is transmitted and if only one digit is changed by the channel noise, the receiver will be able to determine the true error vector and to perform a correct decoding.

More discussion on error correction based on syndrome is given in Section 3.5. Various methods of determining the true error pattern from the $n-k$ linear equations of (3.13) are presented in later chapters.

3.3 THE MINIMUM DISTANCE OF A BLOCK CODE

In this section we introduce an important parameter of a block code called the *minimum distance*. This parameter determines the random-error-detecting and random-error-correcting capabilities of a code. Let $\mathbf{v} = (v_0, v_1, \cdots, v_{n-1})$ be a binary n-tuple. The *Hamming weight* (or simply *weight*) of $\mathbf{v}$, denoted by $w(\mathbf{v})$, is defined as the number of nonzero components of $\mathbf{v}$. For example, the Hamming weight of $\mathbf{v} = (1\,0\,0\,1\,0\,1\,1)$ is 4. Let $\mathbf{v}$ and $\mathbf{w}$ be two n-tuples. The *Hamming distance* (or simply *distance*) between $\mathbf{v}$ and $\mathbf{w}$, denoted $d(\mathbf{v}, \mathbf{w})$, is defined as the number of places where they differ. For example, the Hamming distance between $\mathbf{v} = (1\,0\,0\,1\,0\,1\,1)$ and $\mathbf{w} = (0\,1\,0\,0\,0\,1\,1)$ is 3; they differ in the zeroth, first, and third places. The Hamming distance is a metric function that satisfies the *triangle inequality*. Let $\mathbf{v}$, $\mathbf{w}$, and $\mathbf{x}$ be three n-tuples. Then,

$$d(\mathbf{v}, \mathbf{w}) + d(\mathbf{w}, \mathbf{x}) \geq d(\mathbf{v}, \mathbf{x}). \tag{3.14}$$

(The proof of this inequality is left as a problem.) It follows from the definition of Hamming distance and the definition of modulo-2 addition that the Hamming distance between two n-tuples $\mathbf{v}$ and $\mathbf{w}$ is equal to the Hamming weight of the sum of $\mathbf{v}$ and $\mathbf{w}$; that is,

$$d(\mathbf{v}, \mathbf{w}) = w(\mathbf{v} + \mathbf{w}). \tag{3.15}$$

For example, the Hamming distance between $\mathbf{v} = (1\,0\,0\,1\,0\,1\,1)$ and $\mathbf{w} = (1\,1\,1\,0\,0\,1\,0)$ is 4, and the weight of $\mathbf{v} + \mathbf{w} = (0\,1\,1\,1\,0\,0\,1)$ is also 4.

Given a block code C, one can compute the Hamming distance between any two distinct codewords. The *minimum distance* of C, denoted by d_{min}, is defined as

$$d_{min} \triangleq \min\{d(\mathbf{v}, \mathbf{w}) : \mathbf{v}, \mathbf{w} \in C, \mathbf{v} \neq \mathbf{w}\}. \tag{3.16}$$

If C is a linear block code, the sum of two codewords is also a codeword. It follows from (3.15) that the Hamming distance between two codewords in C is equal to the

Hamming weight of a third codeword in C. Then, it follows from (3.16) that

$$\begin{aligned} d_{min} &= \min\{w(\mathbb{v}+\mathbb{w}) : \mathbb{v}, \mathbb{w} \in C, \mathbb{v} \neq \mathbb{w}\} \\ &= \min\{w(\mathbb{x}) : \mathbb{x} \in C, \mathbb{x} \neq \mathbb{0}\} \\ &\triangleq w_{min}. \end{aligned} \tag{3.17}$$

The parameter $w_{min} \triangleq \{w(\mathbb{x}) : \mathbb{x} \in C, \mathbb{x} \neq \mathbb{0}\}$ is called the *minimum weight* of the linear code C. Summarizing the preceding result, we have the following theorem.

THEOREM 3.1 The minimum distance of a linear block code is equal to the minimum weight of its nonzero codewords and vice versa.

Therefore, for a linear block code, determining the minimum distance of the code is equivalent to determining its minimum weight. The (7, 4) code given in Table 3.1 has minimum weight 3; thus, its minimum distance is 3. Next, we prove a number of theorems that relate the weight structure of a linear block code to its parity-check matrix.

THEOREM 3.2 Let C be an (n, k) linear code with parity-check matrix $\mathbb{H}$. For each codeword of Hamming weight l, there exist l columns of $\mathbb{H}$ such that the vector sum of these l columns is equal to the zero vector. Conversely, if there exist l columns of $\mathbb{H}$ whose vector sum is the zero vector, there exists a codeword of Hamming weight l in C.

Proof. We express the parity-check matrix in the following form:

$$\mathbb{H} = [\mathbb{h}_0, \mathbb{h}_1, \cdots, \mathbb{h}_{n-1}],$$

where $\mathbb{h}_i$ represents the ith column of $\mathbb{H}$. Let $\mathbb{v} = (v_0, v_1, \cdots, v_{n-1})$ be a codeword of weight l. Then, $\mathbb{v}$ has l nonzero components. Let $v_{i_1}, v_{i_2}, \cdots, v_{i_l}$ be the l nonzero components of $\mathbb{v}$, where $0 \leq i_1 < i_2 < \cdots < i_l \leq n-1$. Then, $v_{i_1} = v_{i_2} = \cdots = v_{i_l} = 1$. Because $\mathbb{v}$ is a codeword, we must have

$$\begin{aligned} \mathbb{0} &= \mathbb{v} \cdot \mathbb{H}^T \\ &= v_0\mathbb{h}_0 + v_1\mathbb{h}_1 + \cdots + v_{n-1}\mathbb{h}_{n-1} \\ &= v_{i_1}\mathbb{h}_{i_1} + v_{i_2}\mathbb{h}_{i_2} + \cdots + v_{i_l}\mathbb{h}_{i_l} \\ &= \mathbb{h}_{i_1} + \mathbb{h}_{i_2} + \cdots + \mathbb{h}_{i_l}. \end{aligned}$$

This proves the first part of the theorem.

Now, suppose that $\mathbb{h}_{i_1}, \mathbb{h}_{i_2}, \cdots, \mathbb{h}_{i_l}$ are l columns of $\mathbb{H}$ such that

$$\mathbb{h}_{i_1} + \mathbb{h}_{i_2} + \cdots + \mathbb{h}_{i_l} = \mathbb{0}. \tag{3.18}$$

We form a binary n-tuple $\mathbb{x} = (x_0, x_1, \cdots, x_{n-1})$ whose nonzero components are $x_{i_1}, x_{i_2}, \cdots, x_{i_l}$. The Hamming weight of $\mathbb{x}$ is l. Consider the product

$$\begin{aligned} \mathbb{x} \cdot \mathbb{H}^T &= x_0\mathbb{h}_0 + x_1\mathbb{h}_1 + \cdots + x_{n-1}\mathbb{h}_{n-1} \\ &= x_{i_1}\mathbb{h}_{i_1} + x_{i_2}\mathbb{h}_{i_2} + \cdots + x_{i_l}\mathbb{h}_{i_l} \\ &= \mathbb{h}_{i_1} + \mathbb{h}_{i_2} + \cdots + \mathbb{h}_{i_l}. \end{aligned}$$

It follows from (3.18) that $\mathbf{x} \cdot \mathbf{H}^T = \mathbf{0}$. Thus, $\mathbf{x}$ is a codeword of weight l in C. This proves the second part of the theorem. Q.E.D.

The following two corollaries follow from Theorem 3.2.

COROLLARY 3.2.1 Let C be a linear block code with parity-check matrix $\mathbf{H}$. If no $d-1$ or fewer columns of $\mathbf{H}$ add to $\mathbf{0}$, the code has minimum weight at least d.

COROLLARY 3.2.2 Let C be a linear code with parity-check matrix $\mathbf{H}$. The minimum weight (or the minimum distance) of C is equal to the smallest number of columns of $\mathbf{H}$ that sum to $\mathbf{0}$.

Consider the (7, 4) linear code given in Table 3.1. The parity-check matrix of this code is

$$\mathbf{H} = \begin{bmatrix} 1 & 0 & 0 & 1 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 & 1 & 1 \end{bmatrix}.$$

We see that all columns of $\mathbf{H}$ are nonzero and that no two of them are alike. Therefore, no two or fewer columns sum to $\mathbf{0}$. Hence, the minimum weight of this code is at least 3; however, the zeroth, second, and sixth columns sum to $\mathbf{0}$. Thus, the minimum weight of the code is 3. From Table 3.1 we see that the minimum weight of the code is indeed 3. It follows from Theorem 3.1 that the minimum distance is 3.

Corollaries 3.2.1 and 3.2.2 are generally used to determine the minimum distance or to establish a lower bound on the minimum distance of a linear block code.

3.4 ERROR-DETECTING AND ERROR-CORRECTING CAPABILITIES OF A BLOCK CODE

When a codeword $\mathbf{v}$ is transmitted over a noisy channel, an error pattern of l errors will result in a received vector $\mathbf{r}$ that differs from the transmitted codeword $\mathbf{v}$ in l places [i.e., $d(\mathbf{v}, \mathbf{r}) = l$]. If the minimum distance of a block code C is d_{min}, any two distinct codewords of C differ in at least d_{min} places. For this code C, no error pattern of $d_{min} - 1$ or fewer errors can change one codeword into another. Therefore, any error pattern of $d_{min} - 1$ or fewer errors will result in a received vector $\mathbf{r}$ that is not a codeword in C. When the receiver detects that the received vector is not a codeword of C, we say that errors are detected. Hence, a block code with minimum distance d_{min} is capable of detecting all the error patterns of $d_{min} - 1$ or fewer errors. However, it cannot detect all the error patterns of d_{min} errors because there exists at least one pair of codewords that differ in d_{min} places, and there is an error pattern of d_{min} errors that will carry one into the other. The same argument applies to error patterns of more than d_{min} errors. For this reason we say that the random-error-detecting capability of a block code with minimum distance d_{min} is $d_{min} - 1$.

Even though a block code with minimum distance d_{min} guarantees detection of all the error patterns of $d_{min} - 1$ or fewer errors, it is also capable of detecting a large fraction of error patterns with d_{min} or more errors. In fact, an (n, k) linear code is capable of detecting $2^n - 2^k$ error patterns of length n. This can be shown as follows. Among the $2^n - 1$ possible nonzero error patterns, there are $2^k - 1$ error

patterns that are identical to the $2^k - 1$ nonzero codewords. If any of these $2^k - 1$ error patterns occurs, it alters the transmitted codeword $\mathbb{v}$ into another codeword $\mathbb{w}$. Thus, $\mathbb{w}$ is received, and its syndrome is zero. In this case, the decoder accepts $\mathbb{w}$ as the transmitted codeword and thus performs an incorrect decoding. Therefore, there are $2^k - 1$ *undetectable* error patterns. If an error pattern is not identical to a nonzero codeword, the received vector $\mathbb{r}$ will not be a codeword and the syndrome will not be zero. In this case, an error will be detected. There are exactly $2^n - 2^k$ error patterns that are not identical to the codewords of an (n, k) linear code. These $2^n - 2^k$ error patterns are *detectable* error patterns. For large n, $2^k - 1$ is, in general, much smaller than 2^n. Therefore, only a small fraction of error patterns pass through the decoder without being detected.

Let C be an (n, k) linear code. Let A_i be the number of codewords of weight i in C. The numbers $A_0, A_1, \cdots, A_n$ are called the *weight distribution* of C. If C is used only for error detection on a BSC, the probability that the decoder will fail to detect the presence of errors can be computed from the weight distribution of C. Let $P_u(E)$ denote the probability of an undetected error. Because an undetected error occurs only when the error pattern is identical to a nonzero codeword of C,

$$P_u(E) = \sum_{i=1}^{n} A_i p^i (1-p)^{n-i}, \tag{3.19}$$

where p is the transition probability of the BSC. If the minimum distance of C is d_{min}, then A_1 to $A_{d_{min}-1}$ are zero.

Consider the $(7, 4)$ code given in Table 3.1. The weight distribution of this code is $A_0 = 1$, $A_1 = A_2 = 0$, $A_3 = 7$, $A_4 = 7$, $A_5 = A_6 = 0$, and $A_7 = 1$. The probability of an undetected error is

$$P_u(E) = 7p^3(1-p)^4 + 7p^4(1-p)^3 + p^7.$$

If $p = 10^{-2}$, this probability is approximately 7×10^{-6}. In other words, if 1 million codewords are transmitted over a BSC with $p = 10^{-2}$, on average seven erroneous codewords pass through the decoder without being detected.

If a block code C with minimum distance d_{min} is used for random-error correction, one would like to know how many errors the code is able to correct. The minimum distance d_{min} is either odd or even. Let t be a positive integer such that

$$2t + 1 \leq d_{min} \leq 2t + 2. \tag{3.20}$$

Next, we show that the code C is capable of correcting all the error patterns of t or fewer errors. Let $\mathbb{v}$ and $\mathbb{r}$ be the transmitted codeword and the received vector, respectively. Let $\mathbb{w}$ be any other codeword in C. The Hamming distances among $\mathbb{v}$, $\mathbb{r}$, and $\mathbb{w}$ satisfy the triangle inequality:

$$d(\mathbb{v}, \mathbb{r}) + d(\mathbb{w}, \mathbb{r}) \geq d(\mathbb{v}, \mathbb{w}). \tag{3.21}$$

Suppose that an error pattern of t' errors occurs during the transmission of $\mathbb{v}$. Then, the received vector $\mathbb{r}$ differs from $\mathbb{v}$ in t' places, and therefore $d(\mathbb{v}, \mathbb{r}) = t'$. Because $\mathbb{v}$ and $\mathbb{w}$ are codewords in C, we have

$$d(\mathbb{v}, \mathbb{w}) \geq d_{min} \geq 2t + 1. \tag{3.22}$$

Combining (3.21) and (3.22) and using the fact that $d(\mathbf{v}, \mathbf{r}) = t'$, we obtain the following inequality:

$$d(\mathbf{w}, \mathbf{r}) \geq 2t + 1 - t'.$$

If $t' \leq t$,

$$d(\mathbf{w}, \mathbf{r}) > t.$$

The preceding inequality says that if an error pattern of t or fewer errors occurs, the received vector $\mathbf{r}$ is closer (in Hamming distance) to the transmitted codeword $\mathbf{v}$ than to any other codeword $\mathbf{w}$ in C. For a BSC, this means that the conditional probability $P(\mathbf{r}|\mathbf{v})$ is greater than the conditional probability $P(\mathbf{r}|\mathbf{w})$ for $\mathbf{w} \neq \mathbf{v}$. Based on the maximum likelihood decoding scheme, $\mathbf{r}$ is decoded into $\mathbf{v}$, which is the actual transmitted codeword. The result is a correct decoding, and thus errors are corrected.

In contrast, the code is not capable of correcting all the error patterns of l errors with $l > t$, for there is at least one case in which an error pattern of l errors results in a received vector that is closer to an incorrect codeword than to the transmitted codeword. To show this, let $\mathbf{v}$ and $\mathbf{w}$ be two codewords in C such that

$$d(\mathbf{v}, \mathbf{w}) = d_{min}.$$

Let $\mathbf{e}_1$ and $\mathbf{e}_2$ be two error patterns that satisfy the following conditions:

i. $\mathbf{e}_1 + \mathbf{e}_2 = \mathbf{v} + \mathbf{w}$.
ii. $\mathbf{e}_1$ and $\mathbf{e}_2$ do not have nonzero components in common places.

Obviously, we have

$$w(\mathbf{e}_1) + w(\mathbf{e}_2) = w(\mathbf{v} + \mathbf{w}) = d(\mathbf{v}, \mathbf{w}) = d_{min}. \tag{3.23}$$

Now, suppose that $\mathbf{v}$ is transmitted and is corrupted by the error pattern $\mathbf{e}_1$. Then, the received vector is

$$\mathbf{r} = \mathbf{v} + \mathbf{e}_1.$$

The Hamming distance between $\mathbf{v}$ and $\mathbf{r}$ is

$$d(\mathbf{v}, \mathbf{r}) = w(\mathbf{v} + \mathbf{r}) = w(\mathbf{e}_1). \tag{3.24}$$

The Hamming distance between $\mathbf{w}$ and $\mathbf{r}$ is

$$d(\mathbf{w}, \mathbf{r}) = w(\mathbf{w} + \mathbf{r}) = w(\mathbf{w} + \mathbf{v} + \mathbf{e}_1) = w(\mathbf{e}_2). \tag{3.25}$$

Now, suppose that the error pattern $\mathbf{e}_1$ contains more than t errors (i.e., $w(\mathbf{e}_1) > t$). Because $2t + 1 \leq d_{min} \leq 2t + 2$, it follows from (3.23) that

$$w(\mathbf{e}_2) \leq t + 1.$$

Combining (3.24) and (3.25) and using the fact that $w(\mathbf{e}_1) > t$ and $w(\mathbf{e}_2) \leq t + 1$, we obtain the following inequality:

$$d(\mathbf{v}, \mathbf{r}) \geq d(\mathbf{w}, \mathbf{r}).$$

This inequality says that there exists an error pattern of $l (l > t)$ errors that results in a received vector that is closer to an incorrect codeword than to the transmitted codeword. Based on the maximum likelihood decoding scheme, an incorrect decoding would be performed.

In summary, a block code with minimum distance $d_{\min}$ *guarantees* correction of all the error patterns of $t = \lfloor (d_{min} - 1)/2 \rfloor$ or fewer errors, where $\lfloor (d_{min} - 1)/2 \rfloor$ denotes the largest integer no greater than $(d_{min} - 1)/2$. The parameter $t = \lfloor (d_{min} - 1)/2 \rfloor$ is called the *random-error-correcting capability* of the code. The code is referred to as a t-error-correcting code. The (7, 4) code given in Table 3.1 has minimum distance 3 and thus $t = 1$. The code is capable of correcting any error pattern of single error over a block of seven digits.

A block code with random-error-correcting capability t is usually capable of correcting many error patterns of $t + 1$ or more errors. A t-error-correcting (n, k) linear code is capable of correcting a total of 2^{n-k} error patterns, including those with t or fewer errors (this will be seen in the next section). If a t-error-correcting block code is used strictly for error correction on a BSC with transition probability p, the probability that the decoder commits an erroneous decoding is upper bounded by

$$P(E) \leq \sum_{i=t+1}^{n} \binom{n}{i} p^i (1-p)^{n-i}. \tag{3.26}$$

In practice, a code is often used for correcting λ or fewer errors and simultaneously detecting $l (l > \lambda)$ or fewer errors. That is, when λ or fewer errors occur, the code is capable of correcting them; when more than λ but fewer than $l + 1$ errors occur, the code is capable of detecting their presence without making a decoding error. For this purpose, the minimum distance d_{min} of the code is at least $\lambda + l + 1$ (left as a problem). Thus, a block code with $d_{min} = 10$ is capable of correcting three or fewer errors and simultaneously detecting six or fewer errors.

So far, we have considered only the case in which the receiver makes a hard-decision for each received symbol; however, a receiver may be designed to declare a symbol erased when it is received ambiguously (or unreliably). In this case, the received sequence consists of zeros, ones, or erasures. A code can be used to correct combinations of errors and erasures. A code with minimum distance d_{min} is capable of correcting any pattern of v errors and e erasures provided the following condition

$$d_{min} \geq 2v + e + 1$$

is satisfied. To see this, delete from all the codewords the e components where the receiver has declared erasures. This deletion results in a shortened code of length $n - e$. The minimum distance of this shortened code is at least $d_{min} - e \geq 2v + 1$. Hence, v errors can be corrected in the unerased positions. As a result, the shortened codeword with e components erased can be recovered. Finally, because $d_{min} \geq e + 1$, there is one and only one codeword in the original code that agrees with the unerased components. Consequently, the entire codeword can be recovered. This correction of combinations of errors and erasures is often used in practice.

From the preceding discussion we see that the random-error-detecting and random-error-correcting capabilities of a block code are determined by the code's minimum distance. Clearly, for a given n and k, one would like to construct a block

code with as large a minimum distance as possible, in addition to the implementation considerations.

Often an (n, k) linear block code with minimum distance d_{min} is denoted by (n, k, d_{min}). The code given by Table 3.1 is a (7, 4, 3) code.

3.5 STANDARD ARRAY AND SYNDROME DECODING

In this section we present a scheme for decoding linear block codes. Let C be an (n, k) linear code. Let $\mathbf{v}_1, \mathbf{v}_2, \cdots, \mathbf{v}_{2^k}$ be the codewords of C. No matter which codeword is transmitted over a noisy channel, the received vector $\mathbf{r}$ may be any of the 2^n n-tuples over $GF(2)$. Any decoding scheme used at the receiver is a rule to partition the 2^n possible received vectors into 2^k disjoint subsets $D_1, D_2, \cdots, D_{2^k}$ such that the codeword $\mathbf{v}_i$ is contained in the subset D_i for $1 \leq i \leq 2^k$. Thus, each subset D_i is one-to-one correspondence to a codeword $\mathbf{v}_i$. If the received vector $\mathbf{r}$ is found in the subset D_i, $\mathbf{r}$ is decoded into $\mathbf{v}_i$. Decoding is correct if and only if the received vector $\mathbf{r}$ is in the subset D_i that corresponds to the codeword transmitted.

A method to partition the 2^n possible received vectors into 2^k disjoint subsets such that each subset contains one and only one codeword is described here. The partition is based on the linear structure of the code. First, we place the 2^k codewords of C in a row with the all-zero codeword $\mathbf{v}_1 = (0, 0, \cdots, 0)$ as the first (leftmost) element. From the remaining $2^n - 2^k$ n-tuples, we choose an n-tuple $\mathbf{e}_2$ and place it under the zero vector $\mathbf{v}_1$. Now, we form a second row by adding $\mathbf{e}_2$ to each codeword $\mathbf{v}_i$ in the first row and placing the sum $\mathbf{e}_2 + \mathbf{v}_i$ under $\mathbf{v}_i$. Having completed the second row, we choose an unused n-tuple $\mathbf{e}_3$ from the remaining n-tuples and place it under $\mathbf{v}_1$. Then, we form a third row by adding $\mathbf{e}_3$ to each codeword $\mathbf{v}_i$ in the first row and placing $\mathbf{e}_3 + \mathbf{v}_i$ under $\mathbf{v}_i$. We continue this process until we have used all the n-tuples. Then, we have an array of rows and columns, as shown in Figure 3.6. This array is called a *standard array* of the given linear code C.

It follows from the construction rule of a standard array that the sum of any two vectors in the same row is a codeword in C. Next, we prove some important properties of a standard array.

THEOREM 3.3 No two n-tuples in the same row of a standard array are identical. Every n-tuple appears in one and only one row.

Proof. The first part of the theorem follows from the fact that all the codewords of C are distinct. Suppose that two n-tuples in the lth row are identical, say

$$
\begin{array}{ccccccc}
\mathbf{v}_1 = 0 & \mathbf{v}_2 & \cdots & \mathbf{v}_i & \cdots & \mathbf{v}_{2^k} \\
\mathbf{e}_2 & \mathbf{e}_2 + \mathbf{v}_2 & \cdots & \mathbf{e}_2 + \mathbf{v}_i & \cdots & \mathbf{e}_2 + \mathbf{v}_{2^k} \\
\mathbf{e}_3 & \mathbf{e}_3 + \mathbf{v}_2 & \cdots & \mathbf{e}_3 + \mathbf{v}_i & \cdots & \mathbf{e}_3 + \mathbf{v}_{2^k} \\
\vdots & & & & & \vdots \\
\mathbf{e}_l & \mathbf{e}_l + \mathbf{v}_2 & \cdots & \mathbf{e}_l + \mathbf{v}_i & \cdots & \mathbf{e}_l + \mathbf{v}_{2^k} \\
\vdots & & & & & \vdots \\
\mathbf{e}_{2^{n-k}} & \mathbf{e}_{2^{n-k}} + \mathbf{v}_2 & \cdots & \mathbf{e}_{2^{n-k}} + \mathbf{v}_i & \cdots & \mathbf{e}_{2^{n-k}} + \mathbf{v}_{2^k}
\end{array}
$$

FIGURE 3.6: Standard array for an (n, k) linear code.

$\mathbb{e}_l + \mathbb{v}_i = \mathbb{e}_l + \mathbb{v}_j$ with $i \neq j$. This means that $\mathbb{v}_i = \mathbb{v}_j$, which is impossible. Therefore, no two n-tuples in the same row are identical.

It follows from the construction rule of the standard array that every n-tuple appears at least once. Now, suppose that an n-tuple appears in both the lth row and the mth row with $l < m$. Then this n-tuple must be equal to $\mathbb{e}_l + \mathbb{v}_i$ for some i and equal to $\mathbb{e}_m + \mathbb{v}_j$ for some j. As a result, $\mathbb{e}_l + \mathbb{v}_i = \mathbb{e}_m + \mathbb{v}_j$. From this equality we obtain $\mathbb{e}_m = \mathbb{e}_l + (\mathbb{v}_i + \mathbb{v}_j)$. Because $\mathbb{v}_i$ and $\mathbb{v}_j$ are codewords in C, $\mathbb{v}_i + \mathbb{v}_j$ is also a codeword in C, say $\mathbb{v}_s$. Then $\mathbb{e}_m = \mathbb{e}_l + \mathbb{v}_s$. This implies that the n-tuple $\mathbb{e}_m$ is in the lth row of the array, which contradicts the construction rule of the array that $\mathbb{e}_m$, the first element of the mth row, should be unused in any previous row. Therefore, no n-tuple can appear in more than one row of the array. This concludes the proof of the second part of the theorem. Q.E.D.

From Theorem 3.3 we see that there are $2^n/2^k = 2^{n-k}$ disjoint rows in the standard array, and that each row consists of 2^k distinct elements. The 2^{n-k} rows are called the *cosets* of the code C, and the first n-tuple $\mathbb{e}_j$ of each coset is called a *coset leader* (or coset representative). The coset concept for a subgroup was presented in Section 2.1. Any element in a coset can be used as its coset leader. This does not change the elements of the coset; it simply permutes them.

EXAMPLE 3.6

Consider the (6, 3) linear code generated by the following matrix:

$$\mathbb{G} = \begin{bmatrix} 0 & 1 & 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 0 & 0 & 1 \end{bmatrix}.$$

The standard array of this code is shown in Figure 3.7.

A standard array of an (n, k) linear code C consists of 2^k disjoint columns. Each column consists of 2^{n-k} n-tuples, with the topmost one as a codeword in C. Let D_j denote the jth column of the standard array. Then,

$$D_j = \{\mathbb{v}_j, \mathbb{e}_2 + \mathbb{v}_j, \mathbb{e}_3 + \mathbb{v}_j, \cdots, \mathbb{e}_{2^{n-k}} + \mathbb{v}_j\}, \tag{3.27}$$

Coset leader							
000000	011100	101010	110001	110110	101101	011011	000111
100000	111100	001010	010001	010110	001101	111011	100111
010000	001100	111010	100001	100110	111101	001011	010111
001000	010100	100010	111001	111110	100101	010011	001111
000100	011000	101110	110101	110010	101001	011111	000011
000010	011110	101000	110011	110100	101111	011001	000101
000001	011101	101011	110000	110111	101100	011010	000110
100100	111000	001110	010101	010010	001001	111111	100011

FIGURE 3.7: Standard array for a (6, 3) code.

where $\mathbf{v}_j$ is a codeword of C, and $\mathbf{e}_2, \mathbf{e}_3, \cdots, \mathbf{e}_{2^{n-k}}$ are the coset leaders. The 2^k disjoint columns $D_1, D_2, \cdots, D_{2^k}$ can be used for decoding the code C as described earlier in this section. Suppose that the codeword $\mathbf{v}_j$ is transmitted over a noisy channel. From (3.27) we see that the received vector $\mathbf{r}$ is in D_j if the error pattern caused by the channel is a coset leader. In this event, the received vector $\mathbf{r}$ will be decoded correctly into the transmitted codeword $\mathbf{v}_j$; however, if the error pattern caused by the channel is not a coset leader, an erroneous decoding will result. This can be seen as follows. The error pattern $\mathbf{x}$ caused by the channel must be in some coset and under some nonzero codeword, say in the lth coset and under the codeword $\mathbf{v}_i \neq \mathbf{0}$. Then, $\mathbf{x} = \mathbf{e}_l + \mathbf{v}_i$, and the received vector is

$$\mathbf{r} = \mathbf{v}_j + \mathbf{x} = \mathbf{e}_l + (\mathbf{v}_i + \mathbf{v}_j) = \mathbf{e}_l + \mathbf{v}_s.$$

The received vector $\mathbf{r}$ is thus in D_s and is decoded into $\mathbf{v}_s$, which is not the transmitted codeword. This results in an erroneous decoding. Therefore, the decoding is correct if and only if the error pattern caused by the channel is a coset leader. For this reason, the 2^{n-k} coset leaders (including the zero vector $\mathbf{0}$) are called the *correctable error patterns*. Summarizing the preceding results, we have the following theorem:

THEOREM 3.4 Every (n, k) linear block code is capable of correcting 2^{n-k} error patterns.

To minimize the probability of a decoding error, the error patterns that are most likely to occur for a given channel should be chosen as the coset leaders. For a BSC, an error pattern of smaller weight is more probable than an error pattern of larger weight. Therefore, when a standard array is formed, each coset leader should be chosen to be a vector of *least weight* from the remaining available vectors. If coset leaders are chosen in this manner, each coset leader has minimum weight in its coset. As a result, the decoding based on the standard array is the minimum distance decoding (i.e., the maximum likelihood decoding). To see this, let $\mathbf{r}$ be the received vector. Suppose that $\mathbf{r}$ is found in the ith column D_i and lth coset of the standard array. Then, $\mathbf{r}$ is decoded into the codeword $\mathbf{v}_i$. Because $\mathbf{r} = \mathbf{e}_l + \mathbf{v}_i$, the distance between $\mathbf{r}$ and $\mathbf{v}_i$ is

$$d(\mathbf{r}, \mathbf{v}_i) = w(\mathbf{r} + \mathbf{v}_i) = w(\mathbf{e}_l + \mathbf{v}_i + \mathbf{v}_i) = w(\mathbf{e}_l). \tag{3.28}$$

Now, consider the distance between $\mathbf{r}$ and any other codeword, say $\mathbf{v}_j$,

$$d(\mathbf{r}, \mathbf{v}_j) = w(\mathbf{r} + \mathbf{v}_j) = w(\mathbf{e}_l + \mathbf{v}_i + \mathbf{v}_j).$$

Because $\mathbf{v}_i$ and $\mathbf{v}_j$ are two different codewords, their vector sum, $\mathbf{v}_i + \mathbf{v}_j$, is a nonzero codeword, say $\mathbf{v}_s$. Thus,

$$d(\mathbf{r}, \mathbf{v}_j) = w(\mathbf{e}_l + \mathbf{v}_s). \tag{3.29}$$

Because $\mathbf{e}_l$ and $\mathbf{e}_l + \mathbf{v}_s$ are in the same coset and since $w(\mathbf{e}_l) \leq w(\mathbf{e}_l + \mathbf{v}_s)$, it follows from (3.28) and (3.29) that

$$d(\mathbf{r}, \mathbf{v}_i) \leq d(\mathbf{r}, \mathbf{v}_j).$$

This says that the received vector is decoded into a closest codeword. Hence, if each coset leader is chosen to have minimum weight in its coset, the decoding based on the standard array is the minimum distance decoding, or MLD.

Let α_i denote the number of coset leaders of weight i. The numbers $\alpha_0, \alpha_1, \cdots, \alpha_n$ are called the *weight distribution* of the coset leaders. Knowing these numbers, we can compute the probability of a decoding error. Because a decoding error occurs if and only if the error pattern is not a coset leader, the error probability for a BSC with transition probability p is

$$P(E) = 1 - \sum_{i=0}^{n} \alpha_i p^i (1-p)^{n-i}. \tag{3.30}$$

EXAMPLE 3.7

Consider the (6, 3) code given in Example 3.6. The standard array for this code is shown in Figure 3.7. The weight distribution of the coset leaders is $\alpha_0 = 1, \alpha_1 = 6, \alpha_2 = 1$, and $\alpha_3 = \alpha_4 = \alpha_5 = \alpha_6 = 0$. Thus,

$$P(E) = 1 - (1-p)^6 - 6p(1-p)^5 - p^2(1-p)^4.$$

For $p = 10^{-2}$, we have $P(E) \approx 1.37 \times 10^{-3}$.

An (n, k) linear code is capable of detecting $2^n - 2^k$ error patterns; however, it is capable of correcting only 2^{n-k} error patterns. For large n, 2^{n-k} is a small fraction of $2^n - 2^k$. Therefore, the probability of a decoding error is much higher than the probability of an undetected error.

THEOREM 3.5 For an (n, k) linear code C with minimum distance $d_{\min}$, all the n-tuples of weight $t = \lfloor (d_{\min} - 1)/2 \rfloor$ or less can be used as coset leaders of a standard array of C. If all the n-tuples of weight t or less are used as coset leaders, there is at least one n-tuple of weight $t + 1$ that cannot be used as a coset leader.

Proof. Because the minimum distance of C is $d_{\min}$, the minimum weight of C is also $d_{\min}$. Let $\mathbb{x}$ and $\mathbb{y}$ be two n-tuples of weight t or less. Clearly, the weight of $\mathbb{x} + \mathbb{y}$ is

$$w(\mathbb{x} + \mathbb{y}) \leq w(\mathbb{x}) + w(\mathbb{y}) \leq 2t < d_{\min}.$$

Suppose that $\mathbb{x}$ and $\mathbb{y}$ are in the same coset; then, $\mathbb{x} + \mathbb{y}$ must be a nonzero codeword in C. This is impossible, because the weight of $\mathbb{x} + \mathbb{y}$ is less than the minimum weight of C. Therefore, no two n-tuples of weight t or less can be in the same coset of C, and all the n-tuples of weight t or less can be used as coset leaders.

Let $\mathbb{v}$ be a minimum weight codeword of C [i.e., $w(\mathbb{v}) = d_{\min}$]. Let $\mathbb{x}$ and $\mathbb{y}$ be two n-tuples that satisfy the following two conditions:

i. $\mathbb{x} + \mathbb{y} = \mathbb{v}$.
ii. $\mathbb{x}$ and $\mathbb{y}$ do not have nonzero components in common places.

It follows from the definition that $\mathbb{x}$ and $\mathbb{y}$ must be in the same coset and

$$w(\mathbb{x}) + w(\mathbb{y}) = w(\mathbb{v}) = d_{\min}.$$

Suppose we choose $\mathbf{y}$ such that $w(\mathbf{y}) = t + 1$. Because $2t + 1 \leq d_{\min} \leq 2t + 2$, we have $w(\mathbf{x}) = t$ or $t + 1$. If $\mathbf{x}$ is used as a coset leader, then $\mathbf{y}$ cannot be a coset leader. **Q.E.D.**

Theorem 3.5 reconfirms the fact that an (n, k) linear code with minimum distance $d_{\min}$ is capable of correcting all the error patterns of $\lfloor (d_{\min} - 1)/2 \rfloor$ or fewer errors, but it is not capable of correcting all the error patterns of weight $t + 1$.

A standard array has an important property that can be used to simplify the decoding process. Let $\mathbf{H}$ be the parity-check matrix of the given (n, k) linear code C.

THEOREM 3.6 All the 2^k n-tuples of a coset have the same syndrome. The syndromes for different cosets are different.

Proof. Consider the coset whose coset leader is $\mathbf{e}_l$. A vector in this coset is the sum of $\mathbf{e}_l$ and some codeword $\mathbf{v}_i$ in C. The syndrome of this vector is

$$(\mathbf{e}_l + \mathbf{v}_i)\mathbf{H}^T = \mathbf{e}_l\mathbf{H}^T + \mathbf{v}_i\mathbf{H}^T = \mathbf{e}_l\mathbf{H}^T$$

(since $\mathbf{v}_i\mathbf{H}^T = \mathbf{0}$). The preceding equality says that the syndrome of any vector in a coset is equal to the syndrome of the coset leader. Therefore, all the vectors of a coset have the same syndrome.

Let $\mathbf{e}_j$ and $\mathbf{e}_l$ be the coset leaders of the jth and lth cosets, respectively, where $j < l$. Suppose that the syndromes of these two cosets are equal. Then,

$$\mathbf{e}_j\mathbf{H}^T = \mathbf{e}_l\mathbf{H}^T,$$

$$(\mathbf{e}_j + \mathbf{e}_l)\mathbf{H}^T = \mathbf{0}.$$

This implies that $\mathbf{e}_j + \mathbf{e}_l$ is a codeword in C, say $\mathbf{v}_i$. Thus, $\mathbf{e}_j + \mathbf{e}_l = \mathbf{v}_i$, and $\mathbf{e}_l = \mathbf{e}_j + \mathbf{v}_i$. This implies that $\mathbf{e}_l$ is in the jth coset, which contradicts the construction rule of a standard array that a coset leader should be previously unused. Therefore, no two cosets have the same syndrome. **Q.E.D.**

We recall that the syndrome of an n-tuple is an $(n - k)$-tuple, and there are 2^{n-k} distinct $(n - k)$-tuples. It follows from Theorem 3.6 that there is a one-to-one correspondence between a coset and an $(n - k)$-tuple syndrome; or there is a one-to-one correspondence between a coset leader (a correctable error pattern) and a syndrome. Using this one-to-one correspondence relationship, we can form a decoding table, which is much simpler to use than a standard array. The table consists of 2^{n-k} coset leaders (the correctable error patterns) and their corresponding syndromes. This table is either stored or wired in the receiver. The decoding of a received vector consists of three steps:

1. Compute the syndrome of $\mathbf{r}$, $\mathbf{r} \cdot \mathbf{H}^T$.
2. Locate the coset leader $\mathbf{e}_l$ whose syndrome is equal to $\mathbf{r} \cdot \mathbf{H}^T$. Then $\mathbf{e}_l$ is assumed to be the error pattern caused by the channel.
3. Decode the received vector $\mathbf{r}$ into the codeword $\mathbf{v}^* = \mathbf{r} + \mathbf{e}_l$.

The described decoding scheme is called the *syndrome decoding* or *table-lookup decoding*. In principle, table-lookup decoding can be applied to any (n, k) linear code. It results in minimum decoding delay and minimum error probability;

however, for large $n - k$, the implementation of this decoding scheme becomes impractical, and either a large storage or a complicated logic circuitry is needed. Several practical decoding schemes that are variations of table-lookup decoding are discussed in subsequent chapters. Each of these decoding schemes requires additional properties of a code other than the linear structure.

EXAMPLE 3.8

Consider the (7, 4) linear code given in Table 3.1. The parity-check matrix, as given in Example 3.3, is

$$\mathbb{H} = \begin{bmatrix} 1 & 0 & 0 & 1 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 & 1 & 1 \end{bmatrix}.$$

The code has $2^3 = 8$ cosets, and therefore there are eight correctable error patterns (including the all-zero vector). Because the minimum distance of the code is 3, it is capable of correcting all the error patterns of weight 1 or 0. Hence, all the 7-tuples of weight 1 or 0 can be used as coset leaders. There are $\binom{7}{0} + \binom{7}{1} = 8$ such vectors. We see that for the (7, 4) linear code considered in this example, the number of correctable error patterns guaranteed by the minimum distance is equal to the total number of correctable error patterns. The correctable error patterns and their corresponding syndromes are given in Table 3.2.

Suppose that the codeword $\mathbb{v} = (1\,0\,0\,1\,0\,1\,1)$ is transmitted, and $\mathbb{r} = (1\,0\,0\,1\,1\,1\,1)$ is received. For decoding $\mathbb{r}$, we compute the syndrome of $\mathbb{r}$:

$$\mathbb{s} = (1\,0\,0\,1\,1\,1\,1) \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 1 & 0 \\ 0 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 0 & 1 \end{bmatrix} = (0\,1\,1).$$

TABLE 3.2: Decoding table for the (7, 4) linear code given in Table 3.1.

Syndrome	Coset leaders
(1 0 0)	(1 0 0 0 0 0 0)
(0 1 0)	(0 1 0 0 0 0 0)
(0 0 1)	(0 0 1 0 0 0 0)
(1 1 0)	(0 0 0 1 0 0 0)
(0 1 1)	(0 0 0 0 1 0 0)
(1 1 1)	(0 0 0 0 0 1 0)
(1 0 1)	(0 0 0 0 0 0 1)

From Table 3.2 we find that (0 1 1) is the syndrome of the coset leader $\mathbf{e} = (0000100)$. Thus, (0000100) is assumed to be the error pattern caused by the channel, and $\mathbf{r}$ is decoded into

$$\begin{aligned}\mathbf{v}^* &= \mathbf{r} + \mathbf{e} \\ &= (1001111) + (0000100) \\ &= (1001011),\end{aligned}$$

which is the codeword transmitted. The decoding is correct, since the error pattern caused by the channel is a coset leader.

Now, suppose that $\mathbf{v} = (0000000)$ is transmitted, and $\mathbf{r} = (1000100)$ is received. We see that two errors have occurred during the transmission of $\mathbf{v}$. The error pattern is not correctable and will cause a decoding error. When $\mathbf{r}$ is received, the receiver computes the syndrome:

$$\mathbf{s} = \mathbf{r} \cdot \mathbf{H}^T = (1\,1\,1).$$

From the decoding table we find that the coset leader $\mathbf{e} = (0000010)$ corresponds to the syndrome $\mathbf{s} = (1\,1\,1)$. As a result, $\mathbf{r}$ is decoded into the codeword

$$\begin{aligned}\mathbf{v}^* &= \mathbf{r} + \mathbf{e} \\ &= (1000100) + (0000010) \\ &= (1000110).\end{aligned}$$

Because $\mathbf{v}^*$ is not the codeword transmitted, a decoding error is committed.

Using Table 3.2, we see that the code is capable of correcting any single error over a block of seven digits. When two or more errors occur, a decoding error will be committed.

The table-lookup decoding of an (n, k) linear code may be implemented as follows. The decoding table is regarded as the truth table of n switching functions:

$$\begin{aligned}e_0 &= f_0(s_0, s_1, \cdots, s_{n-k-1}), \\ e_1 &= f_1(s_0, s_1, \cdots, s_{n-k-1}), \\ &\vdots \\ e_{n-1} &= f_{n-1}(s_0, s_1, \cdots, s_{n-k-1}),\end{aligned}$$

where $s_0, s_1, \cdots, s_{n-k-1}$ are the syndrome digits, which are regarded as switching variables, and $e_0, e_1, \cdots, e_{n-1}$ are the estimated error digits. When these n switching functions are derived and simplified, a combinational logic circuit with the $n - k$ syndrome digits as inputs and the estimated error digits as outputs can be realized. The implementation of the syndrome circuit was discussed in Section 3.2. The general decoder for an (n, k) linear code based on the table-lookup scheme is shown

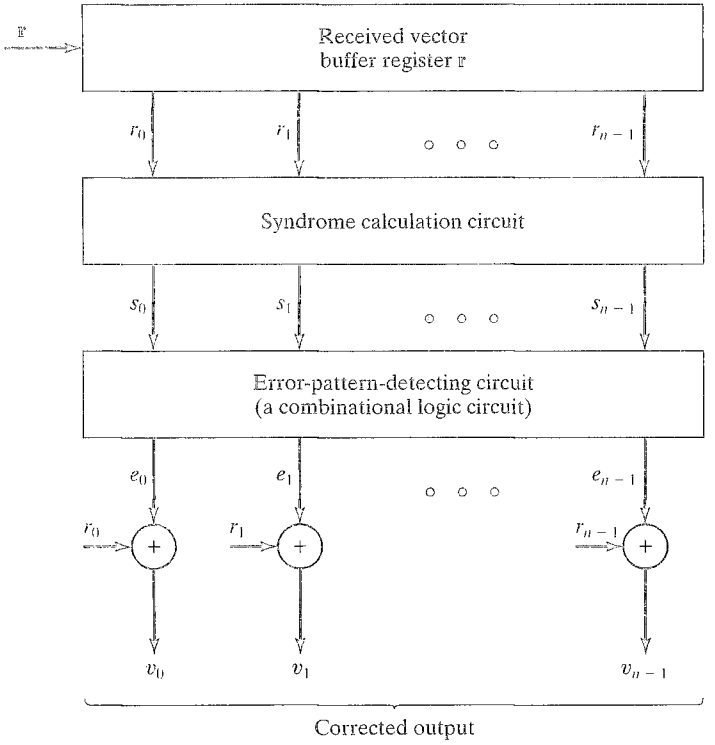

FIGURE 3.8: General decoder for a linear block code.

in Figure 3.8. The cost of this decoder depends primarily on the complexity of the combinational logic circuit.

EXAMPLE 3.9

Again, we consider the (7, 4) code given in Table 3.1. The syndrome circuit for this code is shown in Figure 3.5. The decoding table is given by Table 3.2. From this table we form the truth table (Table 3.3). The switching expressions for the seven error digits are

$$e_0 = s_0 \Lambda s_1' \Lambda s_2', \qquad e_1 = s_0' \Lambda s_1 \Lambda s_2',$$
$$e_2 = s_0' \Lambda s_1' \Lambda s_2, \qquad e_3 = s_0 \Lambda s_1 \Lambda s_2',$$
$$e_4 = s_0' \Lambda s_1 \Lambda s_2, \qquad e_5 = s_0 \Lambda s_1 \Lambda s_2,$$
$$e_6 = s_0 \Lambda s_1' \Lambda s_2,$$

where Λ denotes the logic-AND operation and s' denotes the logic-COMPLEMENT of s. These seven switching expressions can be realized by seven 3-input AND gates. The complete circuit of the decoder is shown in Figure 3.9.

TABLE 3.3: Truth table for the error digits of the correctable error patterns of the (7, 4) linear code given in Table 3.1.

Syndromes			Correctable error patterns (coset leaders)						
s_0	s_1	s_2	e_0	e_1	e_2	e_3	e_4	e_5	e_6
0	0	0	0	0	0	0	0	0	0
1	0	0	1	0	0	0	0	0	0
0	1	0	0	1	0	0	0	0	0
0	0	1	0	0	1	0	0	0	0
1	1	0	0	0	0	1	0	0	0
0	1	1	0	0	0	0	1	0	0
1	1	1	0	0	0	0	0	1	0
1	0	1	0	0	0	0	0	0	1

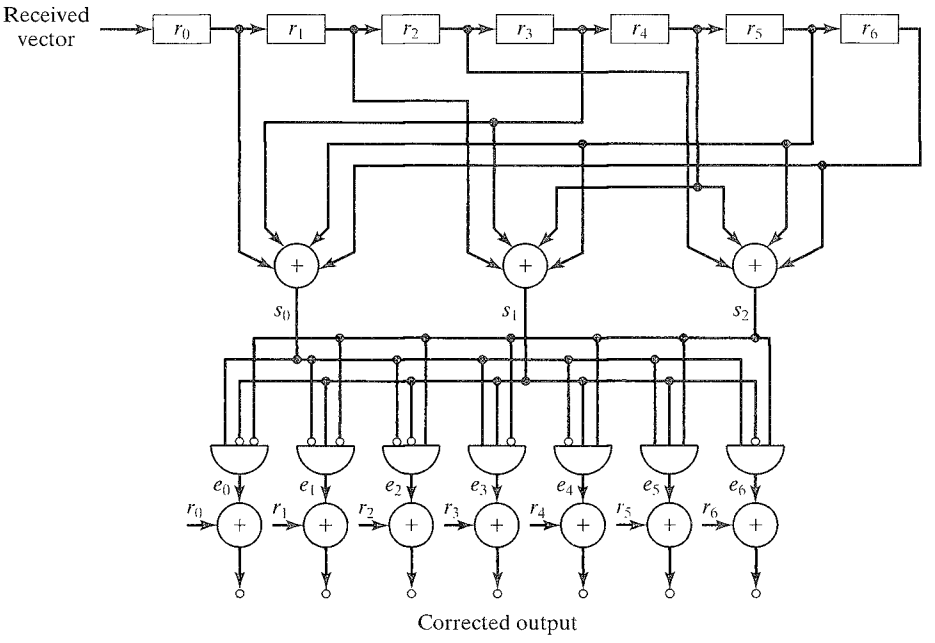

FIGURE 3.9: Decoding circuit for the (7, 4) code given in Table 3.1.

3.6 PROBABILITY OF AN UNDETECTED ERROR FOR LINEAR CODES OVER A BSC

If an (n, k) linear code is used only for error detection over a BSC, the probability of an undetected error $P_u(E)$ can be computed from (3.19) if the weight distribution of the code is known. There exists an interesting relationship between the weight distribution of a linear code and the weight distribution of its dual code. This relationship often makes the computation of $P_u(E)$ much easier. Let $\{A_0, A_1, \cdots, A_n\}$

be the weight distribution of an (n, k) linear code C, and let $\{B_0, B_1, \cdots, B_n\}$ be the weight distribution of its dual code C_d. Now, we represent these two weight distributions in polynomial form as follows:

$$\begin{aligned} A(z) &= A_0 + A_1 z + \cdots + A_n z^n, \\ B(z) &= B_0 + B_1 z + \cdots + B_n z^n. \end{aligned} \tag{3.31}$$

Then, $A(z)$ and $B(z)$ are related by the following identity:

$$A(z) = 2^{-(n-k)}(1+z)^n B\left(\frac{1-z}{1+z}\right). \tag{3.32}$$

This identity is known as the *MacWilliams identity* [13]. The polynomials $A(z)$ and $B(z)$ are called the *weight enumerators* for the (n, k) linear code C and its dual C_d. From the MacWilliams identity, we see that if the weight distribution of the dual of a linear code is known, the weight distribution of the code itself can be determined. As a result, this gives us more flexibility in computing the weight distribution of a linear code.

Using the MacWilliams identity, we can compute the probability of an undetected error for an (n, k) linear code from the weight distribution of its dual. First, we put the expression of (3.19) into the following form:

$$\begin{aligned} P_u(E) &= \sum_{i=1}^{n} A_i p^i (1-p)^{n-i} \\ &= (1-p)^n \sum_{i=1}^{n} A_i \left(\frac{p}{1-p}\right)^i. \end{aligned} \tag{3.33}$$

Substituting $z = p/(1-p)$ in $A(z)$ of (3.31) and using the fact that $A_0 = 1$, we obtain the following identity:

$$A\left(\frac{p}{1-p}\right) - 1 = \sum_{i=1}^{n} A_i \left(\frac{p}{1-p}\right)^i. \tag{3.34}$$

Combining (3.33) and (3.34), we have the following expression for the probability of an undetected error:

$$P_u(E) = (1-p)^n \left[A\left(\frac{p}{1-p}\right) - 1\right]. \tag{3.35}$$

From (3.35) and the MacWilliams identity of (3.32), we finally obtain the following expression for $P_u(E)$:

$$P_u(E) = 2^{-(n-k)} B(1-2p) - (1-p)^n, \tag{3.36}$$

where

$$B(1-2p) = \sum_{i=0}^{n} B_i (1-2p)^i.$$

Hence, there are two ways for computing the probability of an undetected error for a linear code; often, one is easier than the other. If $n-k$ is smaller than k, it is much easier to compute $P_u(E)$ from (3.36); otherwise, it is easier to use (3.35).

EXAMPLE 3.10

Consider the (7, 4) linear code given in Table 3.1. The dual of this code is generated by its parity-check matrix,

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

(see Example 3.3). Taking the linear combinations of the rows of $\mathbb{H}$, we obtain the following eight vectors in the dual code:

$$(0\,0\,0\,0\,0\,0\,0), \qquad (1\,1\,0\,0\,1\,0\,1),$$

$$(1\,0\,0\,1\,0\,1\,1), \qquad (1\,0\,1\,1\,1\,0\,0),$$

$$(0\,1\,0\,1\,1\,1\,0), \qquad (0\,1\,1\,1\,0\,0\,1),$$

$$(0\,0\,1\,0\,1\,1\,1), \qquad (1\,1\,1\,0\,0\,1\,0).$$

Thus, the weight enumerator for the dual code is $B(z) = 1 + 7z^4$. Using (3.36), we obtain the probability of an undetected error for the (7, 4) linear code given in Table 3.1:

$$P_u(E) = 2^{-3}[1 + 7(1 - 2p)^4] - (1 - p)^7.$$

This probability was also computed in Section 3.4 using the weight distribution of the code itself.

Theoretically, we can compute the weight distribution of an (n, k) linear code by examining its 2^k codewords or by examining the 2^{n-k} codewords of its dual and then applying the MacWilliams identity; however, for large n, k, and $n - k$, the computation becomes practically impossible. Except for some short linear codes and a few small classes of linear codes, the weight distributions for many known linear codes are still unknown. Consequently, it is very difficult, if not impossible, to compute their probability of an undetected error.

Although it is difficult to compute the probability of an undetected error for a specific (n, k) linear code for large n and k, it is quite easy to derive an upper bound on the average probability of an undetected error for the ensemble of all (n, k) linear systematic codes. As we showed earlier, an (n, k) linear systematic code is completely specified by a matrix $\mathbb{G}$ of the form given by (3.4). The submatrix $\mathbb{P}$ consists of $k(n - k)$ entries. Because each entry p_{ij} can be either a 0 or a 1, there are $2^{k(n-k)}$ distinct matrices $\mathbb{G}'s$ of the form given by (3.4). Let Γ denote the ensemble of codes generated by these $2^{k(n-k)}$ matrices. Suppose that we choose a code randomly from Γ and use it for error detection. Let C_j be the chosen code. Then, the probability that C_j will be chosen is

$$P(C_j) = 2^{-k(n-k)}. \tag{3.37}$$

Let A_{ji} denote the number of codewords in C_j with weight i. It follows from (3.19) that the probability of an undetected error for C_j is given by

$$P_u(E|C_j) = \sum_{i=1}^{n} A_{ji} p^i (1-p)^{n-i}. \tag{3.38}$$

The average probability of an undetected error for a linear code in Γ is defined as

$$\mathbb{P}_u(\mathbb{E}) = \sum_{j=1}^{|\Gamma|} P(C_j) P_u(E|C_j), \tag{3.39}$$

where $|\Gamma|$ denotes the number of codes in Γ. Substituting (3.37) and (3.38) into (3.39), we obtain

$$\mathbb{P}_u(\mathbb{E}) = 2^{-k(n-k)} \sum_{i=1}^{n} p^i (1-p)^{n-i} \sum_{j=1}^{|\Gamma|} A_{ji}. \tag{3.40}$$

A nonzero n-tuple is contained in either exactly $2^{(k-1)(n-k)}$ codes in Γ or in none of the codes (left as a problem). Because there are $\binom{n}{i}$ n-tuples of weight i, we have

$$\sum_{j=1}^{|\Gamma|} A_{ji} \leq \binom{n}{i} 2^{(k-1)(n-k)}. \tag{3.41}$$

Substituting (3.41) into (3.40), we obtain the following upper bound on the average probability of an undetected error for an (n, k) linear systematic code:

$$\begin{aligned} \mathbb{P}_u(\mathbb{E}) &\leq 2^{-(n-k)} \sum_{i=1}^{n} \binom{n}{i} p^i (1-p)^{n-i} \\ &= 2^{-(n-k)} [1 - (1-p)^n]. \end{aligned} \tag{3.42}$$

Because $[1 - (1-p)^n] \leq 1$, it is clear that $\mathbb{P}_u(\mathbb{E}) \leq 2^{-(n-k)}$.

The preceding result says that there exist (n, k) linear codes with the probability of an undetected error, $P_u(E)$, upper bounded by $2^{-(n-k)}$. In other words, there exist (n, k) linear codes with $P_u(E)$ *decreasing exponentially with the number of parity-check digits*, $n - k$. Even for moderate $n - k$, these codes have a very small probability of an undetected error. For example, let $n - k = 30$. There exist (n, k) linear codes for which $P_u(E)$ is upper bounded by $2^{-30} \approx 10^{-9}$. Many classes of linear codes have been constructed for the past five decades; however, only a few small classes of linear codes have been proved to have a $P_u(E)$ that satisfies the upper bound $2^{-(n-k)}$. It is still not known whether the other known linear codes satisfy this upper bound.

Good codes for error detection and their applications in error control will be presented in later chapters. An excellent treatment of error-detecting codes can be found in [12].

3.7 SINGLE-PARITY-CHECK CODES, REPETITION CODES, AND SELF-DUAL CODES

A single-parity-check (SPC) code is a linear block code with a single parity-check digit. Let $\mathbf{u} = (u_0, u_1, \ldots, u_{k-1})$ be the message to be encoded. The single parity-check digit is given by

$$p = u_0 + u_1 + \cdots + u_{k-1} \tag{3.43}$$

which is simply the modulo-2 sum of all the message digits. Adding this parity-check digit to each k-digit message results in a $(k+1, k)$ linear block code. Each codeword is of the form

$$\mathbf{v} = (p, u_0, u_1, \ldots, u_{k-1}).$$

From (3.43), we readily see that $p = 1$ if the weight of message $\mathbf{u}$ is odd, and $p = 0$ if the weight of message $\mathbf{u}$ is even. Therefore, all the codewords of a SPC code have even weights, and the minimum weight (or minimum distance) of the code is 2. The generator of the code in systematic form is given by

$$\mathbb{G} = \left[\begin{array}{c:ccccc} 1 & 1 & 0 & 0 & 0 & \cdots 0 \\ 1 & 0 & 1 & 0 & 0 & \cdots 0 \\ 1 & 0 & 0 & 1 & 0 & \cdots 0 \\ \vdots & & & \vdots & & \\ 1 & 0 & 0 & 0 & 0 & \cdots 1 \end{array}\right] = \left[\begin{array}{c:c} 1 & \\ 1 & \\ 1 & \mathbb{I}_k \\ \vdots & \\ 1 & \end{array}\right]. \tag{3.44}$$

From (3.44) we find that the parity-check matrix of the code is

$$\mathbb{H} = [1\ 1\ \cdots\ 1]. \tag{3.45}$$

Because all the codewords have even weights, a SPC code is also called an even-parity-check code. SPC codes are often used for simple error detection. Any error pattern with an odd number of errors will change a codeword into a received vector of odd weight that is not a codeword. Hence, the syndrome of the received vector is not equal to zero. Consequently, all the error patterns of odd weight are detectable.

A repetition code of length n is an $(n, 1)$ linear block code that consists of only two codewords, the all-zero codeword $(0\ 0\ \cdots\ 0)$ and the all-one codeword $(1\ 1\ \cdots\ 1)$. This code is obtained by simply repeating a single message bit n times. The generator matrix of the code is

$$\mathbb{G} = [1\ 1\ \cdots\ 1]. \tag{3.46}$$

From (3.44) through (3.46), we readily see that the $(n, 1)$ repetition code and the $(n, n-1)$ SPC code are dual codes to each other.

SPC and repetition codes are often used as component codes for constructing long, powerful codes as will be seen in later chapters.

A linear block code C that is equal to its dual code C_d is called a *self-dual code*. For a self-dual code, the code length n must be even, and the dimension k of the code must be equal to $n/2$. Therefore, its rate R is equal to $\frac{1}{2}$. Let $\mathbb{G}$ be a generator matrix of a self-dual code C. Then, $\mathbb{G}$ is also a generator matrix of its dual code C_d and hence is a parity-check matrix of C. Consequently,

$$\mathbb{G} \cdot \mathbb{G}^T = \mathbb{0} \tag{3.47}$$

Suppose $\mathbb{G}$ is in systematic form, $\mathbb{G} = [\mathbb{P}\ \mathbb{I}_{n/2}]$. From (3.47), we can easily see that

$$\mathbb{P} \cdot \mathbb{P}^T = \mathbb{I}_{n/2}. \tag{3.48}$$

Conversely, if a rate-$\frac{1}{2}$ $(n, n/2)$ linear block code C satisfies the condition of (3.47) [or (3.48)], then it is a self-dual code (the proof is left as a problem).

EXAMPLE 3.11

Consider the (8, 4) linear block code generated by the matrix

$$\mathbb{G} = \begin{bmatrix} 1\,1\,1\,1\,1\,1\,1\,1 \\ 0\,0\,0\,0\,1\,1\,1\,1 \\ 0\,0\,1\,1\,0\,0\,1\,1 \\ 0\,1\,0\,1\,0\,1\,0\,1 \end{bmatrix}.$$

The code has a rate $R = \frac{1}{2}$. It is easy to check that $\mathbb{G} \cdot \mathbb{G}^T = \mathbb{0}$. Therefore, it is a self-dual code.

There are many good self-dual codes but the most well known self-dual code is the (24, 12) Golay code, which will be discussed in Chapter 4.

PROBLEMS

3.1 Consider a systematic (8, 4) code whose parity-check equations are

$$\begin{aligned} v_0 &= u_1 + u_2 + u_3, \\ v_1 &= u_0 + u_1 + u_2, \\ v_2 &= u_0 + u_1 + u_3, \\ v_3 &= u_0 + u_2 + u_3. \end{aligned}$$

where u_0, u_1, u_2, and u_3, are message digits, and v_0, v_1, v_2, and v_3 are parity-check digits. Find the generator and parity-check matrices for this code. Show analytically that the minimum distance of this code is 4.

3.2 Construct an encoder for the code given in Problem 3.1.

3.3 Construct a syndrome circuit for the code given in Problem 3.1.

3.4 Let $\mathbb{H}$ be the parity-check matrix of an (n, k) linear code C that has both odd- and even-weight codewords. Construct a new linear code C_1 with the following parity-check matrix:

$$\mathbb{H}_1 = \left[\begin{array}{c|ccc} 0 & & & \\ 0 & & & \\ \vdots & & \mathbb{H} & \\ 0 & & & \\ \hline 1 & 1 & 1 \cdots & 1 \end{array}\right].$$

(Note that the last row of $\mathbb{H}_1$ consists of all 1's.)

a. Show that C_1 is an $(n + 1, k)$ linear code. C_1 is called an *extension* of C.

b. Show that every codeword of C_1 has even weight.

c. Show that C_1 can be obtained from C by adding an extra parity-check digit, denoted by v_∞, to the left of each codeword $\mathbf{v}$ as follows: (1) if $\mathbf{v}$ has odd weight, then $v_\infty = 1$, and (2) if $\mathbf{v}$ has even weight, then $v_\infty = 0$. The parity-check digit v_∞ is called an *overall parity-check* digit.

3.5 Let C be a linear code with both even- and odd-weight codewords. Show that the number of even-weight codewords is equal to the number of odd-weight codewords.

3.6 Consider an (n, k) linear code C whose generator matrix $\mathbf{G}$ contains no zero column. Arrange all the codewords of C as rows of a 2^k-by-n array.

a. Show that no column of the array contains only zeros.

b. Show that each column of the array consists of 2^{k-1} zeros and 2^{k-1} ones.

c. Show that the set of all codewords with zeros in a particular component position forms a subspace of C. What is the dimension of this subspace?

3.7 Prove that the Hamming distance satisfies the triangle inequality; that is, let $\mathbf{x}$, $\mathbf{y}$, and $\mathbf{z}$ be three n-tuples over $GF(2)$, and show that

$$d(\mathbf{x}, \mathbf{y}) + d(\mathbf{y}, \mathbf{z}) \geq d(\mathbf{x}, \mathbf{z}).$$

3.8 Prove that a linear code is capable of correcting λ or fewer errors and simultaneously detecting $l (l > \lambda)$ or fewer errors if its minimum distance $d_{\min} \geq \lambda + l + 1$.

3.9 Determine the weight distribution of the (8, 4) linear code given in Problem 3.1. Let the transition probability of a BSC be $p = 10^{-2}$. Compute the probability of an undetected error of this code.

3.10 Because the (8, 4) linear code given in Problem 3.1 has minimum distance 4, it is capable of correcting all the single-error patterns and simultaneously detecting any combination of double errors. Construct a decoder for this code. The decoder must be capable of correcting any single error and detecting any double errors.

3.11 Let Γ be the ensemble of all the binary systematic (n, k) linear codes. Prove that a nonzero binary n-tuple $\mathbf{v}$ is contained in either exactly $2^{(k-1)(n-k)}$ codes in Γ or in none of the codes in Γ.

3.12 The (8, 4) linear code given in Problem 3.1 is capable of correcting 16 error patterns (the coset leaders of a standard array). Suppose that this code is used for a BSC. Devise a decoder for this code based on the table-lookup decoding scheme. The decoder is designed to correct the 16 most probable error patterns.

3.13 Let C_1 be an (n_1, k) linear systematic code with minimum distance d_1 and generator matrix $\mathbf{G}_1 = [\mathbf{P}_1 \ \mathbf{I}_k]$. Let C_2 be an (n_2, k) linear systematic code with minimum distance d_2 and generator matrix $\mathbf{G}_2 = [\mathbf{P}_2 \ \mathbf{I}_k]$. Consider an $(n_1 + n_2, k)$ linear code with the following parity-check matrix:

$$\mathbf{H} = \begin{bmatrix} & & \mathbf{P}_1^T \\ \mathbf{I}_{n_1+n_2-k} & & \mathbf{I}_k \\ & & \mathbf{P}_2^T \end{bmatrix}.$$

Show that this code has a minimum distance of at least $d_1 + d_2$.

3.14 Show that the (8, 4) linear code C given in Problem 3.1 is self-dual.

3.15 For any binary (n, k) linear code with minimum distance (or minimum weight) $2t+1$ or greater, show that the number of parity-check digits satisfies the following inequality:

$$n - k \geq \log_2 \left[1 + \binom{n}{1} + \binom{n}{2} + \cdots + \binom{n}{t} \right].$$

The preceding inequality gives an upper bound on the random-error-correcting capability t of an (n, k) linear code. This bound is known as the *Hamming*

bound [14]. (*Hint:* For an (n, k) linear code with minimum distance $2t + 1$ or greater, all the n-tuples of weight t or less can be used as coset leaders in a standard array.)

3.16 Show that the minimum distance $d_{\min}$ of an (n, k) linear code satisfies the following inequality:

$$d_{\min} \leq \frac{n \cdot 2^{k-1}}{2^k - 1}.$$

(*Hint:* Use the result of Problem 3.6(b). This bound is known as the *Plotkin bound* [1-3].)

3.17 Show that there exists an (n, k) linear code with a minimum distance of at least d if

$$\sum_{i=1}^{d-1} \binom{n}{i} < 2^{n-k}.$$

(*Hint:* Use the result of Problem 3.11 and the fact that the nonzero n-tuples of weight $d - 1$ or less can be at most in

$$\left\{ \sum_{i=1}^{d-1} \binom{n}{i} \right\} \cdot 2^{(k-1)(n-k)}$$

(n, k) systematic linear codes.)

3.18 Show that there exists an (n, k) linear code with a minimum distance of at least $d_{\min}$ that satisfies the following inequality:

$$\sum_{i=1}^{d_{\min}-1} \binom{n}{i} < 2^{n-k} \leq \sum_{i=1}^{d_{\min}} \binom{n}{i}.$$

(*Hint:* See Problem 3.17. The second inequality provides a lower bound on the minimum distance attainable with an (n, k) linear code. This bound is known as the *Varsharmov–Gilbert* bound [1-3].)

3.19 Consider a rate-$\frac{1}{2}$ $(n, n/2)$ linear block code C with a generator matrix $\mathbb{G}$. Prove that C is self-dual if $\mathbb{G} \cdot \mathbb{G}^T = \mathbb{0}$.

3.20 Devise an encoder for the $(n, n-1)$ SPC code with only one memory element (or flip-flop) and one X-OR gate (or modulo-2 adder).

BIBLIOGRAPHY

1. E. R. Berlekamp, *Algebraic Coding Theory*, McGraw-Hill, New York, 1968; Rev. ed., Aegean Park Press, Laguna Hills, N.Y., 1984.

2. W. W. Peterson and E. J. Weldon, Jr., *Error-Correcting Codes*, 2d ed., MIT Press, Cambridge, 1972.

3. F. J. MacWilliams and J. J. A. Sloane, *The Theory of Error-Correcting Codes*, North Holland, Amsterdam, 1977.

4. R. J. McEliece, *The Theory of Information and Coding*, Addison-Wesley, Reading, Mass., 1977.

5. G. Clark and J. Cain, *Error-Correcting Codes for Digital Communications*, Plenum Press, New York, 1981.

6. R. E. Blahut, *Algebraic Codes for Data Transmission*, Cambridge University Press, Cambridge, UK, 2003.

7. A. M. Michelson and A. H. Levesque, *Error Control Techniques for Digital Communication*, John Wiley & Sons, New York, 1985.

8. T. R. N. Rao and E. Fujiwara, *Error-Correcting Codes for Computer Systems*, Prentice Hall, Englewood Cliffs, N.J., 1989.

9. S. A. Vanstone and P. C. van Oorschot, *An Introduction to Error Correcting Codes with Applications*, Kluwer Academic, Boston, Mass., 1989.

10. A. Poli and L. Huguet, *Error Correcting Codes, Theory and Applications*, Prentice Hall, Hemel Hempstead, UK, 1992.

11. S. B. Wicker, *Error Control Systems for Digital Communication and Storage*, Prentice Hall, Englewood Cliffs, N.J., 1995.

12. T. Klove and V. I. Korzhik, *Error Detecting Codes*, Kluwer Academic, Boston, Mass., 1995.

13. F. J. MacWilliams, "A Theorem on the Distribution of Weights in a Systematic Codes," *Bell Syst. Tech. J.* 42: 79–94, 1963.

14. R. W. Hamming, "Error Detecting and Error Correcting Codes," *Bell Syst. Tech. J.* 29: 147–160, April 1950.