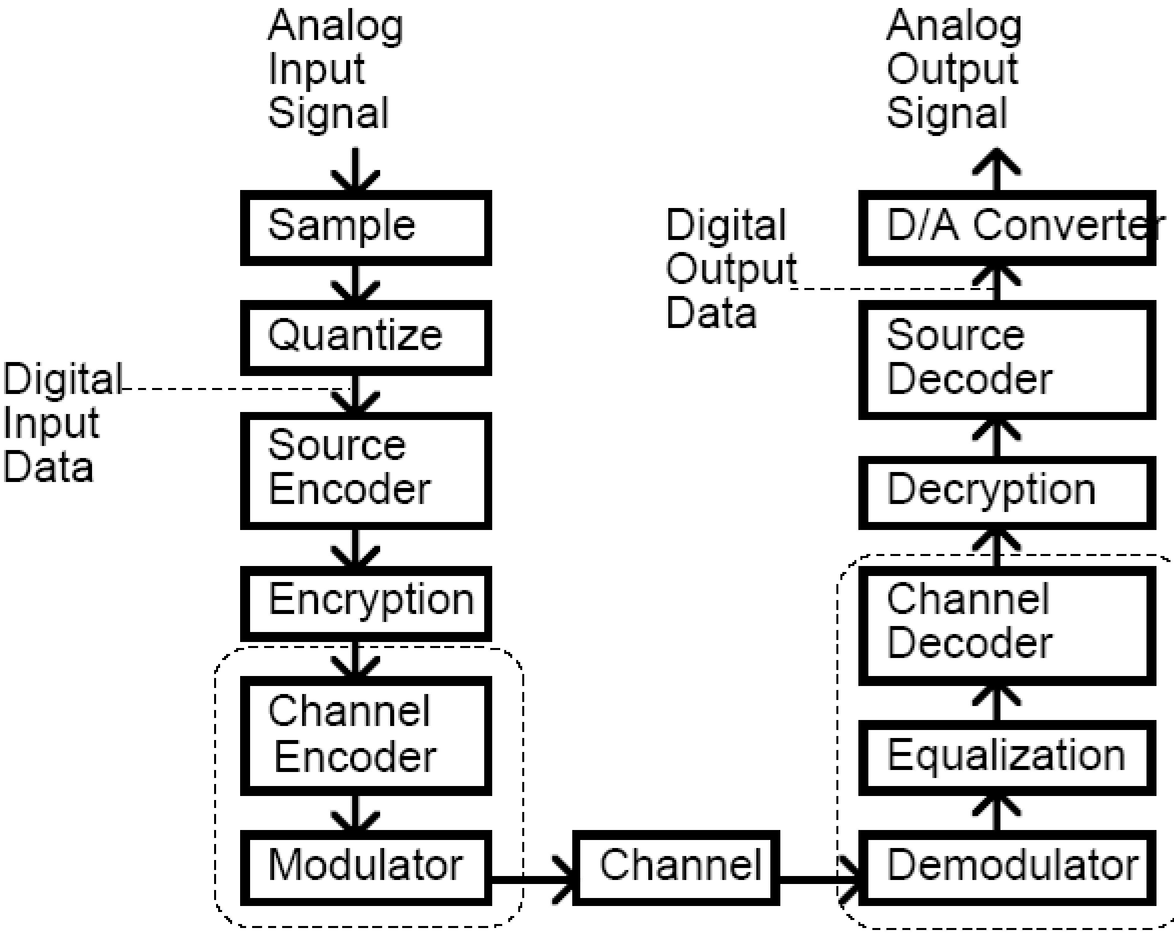
EENG-373

Channel coding
(Error correcting codes)

Dr. Mohab A. Mangoud
Associate Professor of Wireless Communications
University of Bahrain, College of Engineering,
Department of Electrical and Electronics Engineering,
P.O. Box 32038, Isa Town,Kingdom of Bahrain
Office: +973 17876033/6261 Mobile +973 33332771
Fax: + 973 17680924 Email : mangoud@eng.uob.bh
URL: http://userspages.uob.edu.bh/mangoud

Block Diagram of Digital Communications System



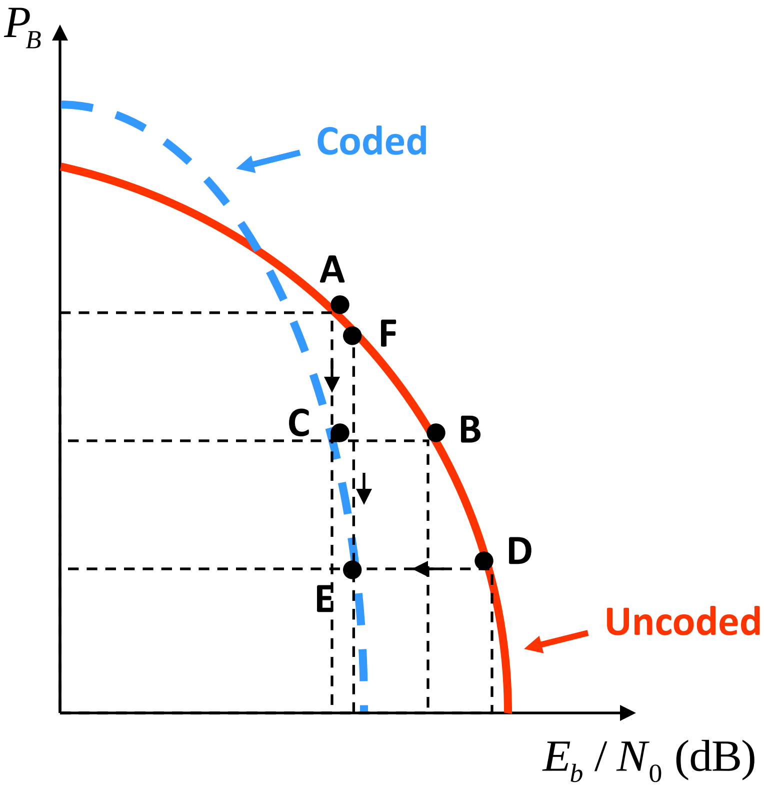
Why using error correction coding?

- Error performance vs. bandwidth
- Power vs. bandwidth
- Data rate vs. bandwidth
- Capacity vs. bandwidth

Coding gain:

For a given bit-error probability, the reduction in the Eb/N0 that can be realized through the use of code:

$$G\,[\mathrm{dB}] = \left(\frac{E_b}{N_0}\right)_{\mathrm{u}} [\mathrm{dB}] - \left(\frac{E_b}{N_0}\right)_{\mathrm{c}} [\mathrm{dB}]$$



- We will focus on two coding types:
  - (1) Block codes: mapping of information source into channel inputs done independently: Encoder output depends only on the current *block* of input sequence
  - (2) Convolutional codes: *each source bit* influences *n(L+1)* channel input bits. *n(L+1)* is the constraint length and *L* is the memory depth. These codes are denoted by *(n,k,L).*

Block Codes

- The encoder for a block code accepts blocks of k input symbols and produces blocks of n output symbols.
- Usually the input symbols and output symbols, are both bits {0,1}, but there is one notable exception (Reed-Solomon Codes) for which input and output symbols are *M-ary*.
- Note that there will be $n - k$ redundant symbols.
- The ratio $r = k/n \le 1$ is called the code rate.
  - Small r -> lots of redundancy
  - Large r -> little redundancy

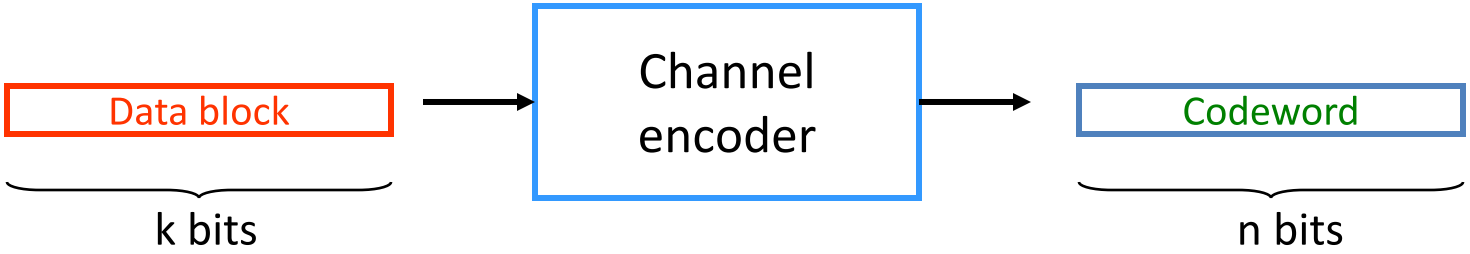
Typical Values

- For a practical block codes, $20 \le n \le 1000$ is a typical range of values.
- For practical block codes, $\frac{1}{4} \le r \le 1$ is typical.

- We will refer to an "(n,k)" block code.

Linear block codes – cont’d

- The information bit stream is chopped into blocks of k bits.
- Each block is encoded to a larger block of n bits.
- The coded bits are modulated and sent over channel.
- The reverse procedure is done at the receiver.



$n\text{-}k$ Redundant bits

$$R_c = \frac{k}{n} \quad \text{Code rate}$$

Linear block codes – cont'd

- The Hamming weight of vector **c**, denoted by w(**c**), is the number of non-zero elements in **c**.
- The Hamming distance between two vectors **ci** and **cj**, is the number of elements in which they differ.

- The minimum distance of a block code is

$$d(ci,cj) = w(ci \oplus cj)$$

$$d_{\min} = \min_{i \neq j} d(c_i, c_j) = \min_i w(c_i)$$

Linear block codes – cont’d

- Error detection capability is given by

$$e = d_{\min} - 1$$

- Error correcting-capability **t** of a code, which is defined as the maximum number of guaranteed correctable errors per codeword, is

$$t = \left\lfloor \frac{d_{\min} - 1}{2} \right\rfloor$$

Linear block codes – cont’d

- Encoding in (n,k) block code

$$\boxed{c = \mathbf{mG}}$$

$$(c_1, c_2, \ldots, c_n) = (m_1, m_2, \ldots, m_k) \cdot \begin{bmatrix} \mathbf{V}_1 \\ \mathbf{V}_2 \\ \vdots \\ \mathbf{V}_k \end{bmatrix}$$

$$(c_1, c_2, \ldots, c_n) = m_1 \cdot \mathbf{V}_1 + m_2 \cdot \mathbf{V}_2 + \ldots + m_2 \cdot \mathbf{V}_k$$

  - The rows of G, are linearly independent.

Linear block codes – cont'd

- Example: Block code (6,3)

$$\mathbf{G} = \begin{bmatrix} \mathbf{V}_1 \\ \mathbf{V}_2 \\ \mathbf{V}_3 \end{bmatrix} = \begin{bmatrix} 110100 \\ 011010 \\ 101001 \end{bmatrix}$$

Message vector	Codeword
000	000000
100	110100
010	011010
110	101110
001	101001
101	011101
011	110011
111	000111

Linear block codes – cont'd

- Systematic block code (n,k)
  - For a systematic code, the first (or last) k elements in the codeword are information bits.

$$\mathbf{G} = [\mathbf{P} \;\vdots\; \mathbf{I}_k]$$

$$\mathbf{I}_k = k \times k \;\; \text{identity matrix}$$

$$\mathbf{P}_k = k \times (n-k) \quad \text{matrix}$$

$$\mathbf{U} = (u_1, u_2, \ldots, u_n) = (\underbrace{p_1, p_2, \ldots, p_{n-k}}_{\text{parity bits}}, \underbrace{m_1, m_2, \ldots, m_k}_{\text{message bits}})$$

Linear block codes – cont’d

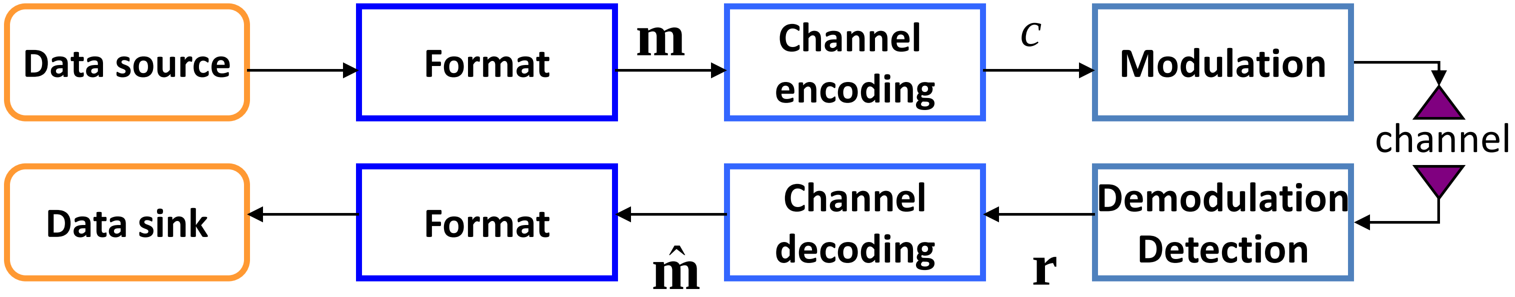
- For any linear code we can find an matrix $\mathbf{H}_{(n-k)\times n}$, which its rows are orthogonal to rows of $\mathbf{G}$:

$$\boxed{\mathbf{G}\mathbf{H}^T = \mathbf{0}}$$

- **H** is called the parity check matrix and its rows are linearly independent.
- For systematic linear block codes:

$$\boxed{\mathbf{H} = [\mathbf{I}_{n-k} \,\vdots\, \mathbf{P}^T]}$$

Linear block codes – cont'd



$$\mathbf{r} = \mathbf{U} + \mathbf{e}$$

$\mathbf{r} = (r_1, r_2, ..., r_n)$ received codeword or vector

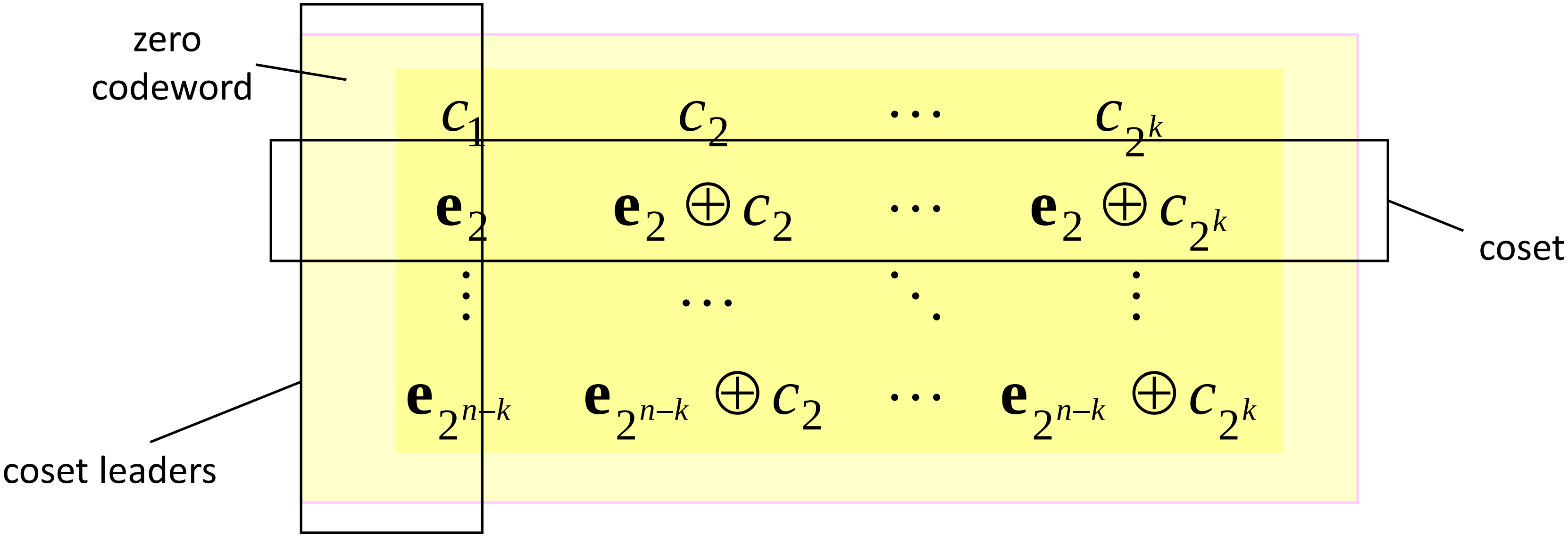
$\mathbf{e} = (e_1, e_2, ..., e_n)$ error patternor vector

- Syndrome testing:
  - **S** is syndrome of **r**, corresponding to the error pattern **e**.

$$\mathbf{S} = \mathbf{r}\mathbf{H}^T = \mathbf{e}\mathbf{H}^T$$

Linear block codes – cont'd

- Standard array
  1. For row $i = 2,3,\ldots,2^{n-k},$ find a vector in of minimum weight which is not already listed in the array.
  2. Call this pattern $\mathbf{e}_i$ and form the $i\text{th}$ row as the corresponding coset



zero codeword

$$\begin{array}{cccc} c_1 & c_2 & \cdots & c_{2^k} \\ \mathbf{e}_2 & \mathbf{e}_2 \oplus c_2 & \cdots & \mathbf{e}_2 \oplus c_{2^k} \\ \vdots & \cdots & \ddots & \vdots \\ \mathbf{e}_{2^{n-k}} & \mathbf{e}_{2^{n-k}} \oplus c_2 & \cdots & \mathbf{e}_{2^{n-k}} \oplus c_{2^k} \end{array}$$

coset

coset leaders

Linear block codes – cont’d

- Standard array and syndrome table decoding

1. Calculate $\mathbf{S}=\mathbf{r}\mathbf{H}^T$
2. Find the coset leader, $\hat{\mathbf{e}}=\mathbf{e}_i$, corresponding to $\mathbf{S}$.
3. Calculate $\hat{\mathbf{U}}=\mathbf{r}+\hat{\mathbf{e}}$ and corresponding .

– Note that $\hat{\mathbf{U}}=\mathbf{r}+\hat{\mathbf{e}}=(\mathbf{U}+\mathbf{e})+\hat{\mathbf{e}}=\mathbf{U}+(\mathbf{e}+\hat{\mathbf{e}})$

  - If $\hat{\mathbf{e}}=\mathbf{e}$, error is corrected.
  - If $\hat{\mathbf{e}}\neq\mathbf{e}$, undetectable decoding error occurs.

Linear block codes – cont’d

- Example: Standard array for the (6,3) code

codewords

000000	110100	011010	101110	101001	011101	110011	000111
000001	110101	011011	101111	101000	011100	110010	000110
000010	110110	011000	101100	101011	011111	110001	000101
000100	110000	011100	101010	101101	011010	110111	000110
001000	111100	⋮			⋮		⋮
010000	100100						
100000	010100				⋮		
010001	100101		⋯			⋯	010110

coset

Coset leaders

Linear block codes – cont'd

Error pattern	Syndrome
000000	000
000001	101
000010	011
000100	110
001000	001
010000	010
100000	100
010001	111

$\mathbf{U} = (101110)$ transmitted.

$\mathbf{r} = (001110)$ is received.

The syndrome of $\mathbf{r}$ is computed:

$$\mathbf{S} = \mathbf{r}\mathbf{H}^T = (001110)\mathbf{H}^T = (100)$$

Error pattern corresponding to this syndrome is

$$\hat{\mathbf{e}} = (100000)$$

The corrected vector is estimated

$$\hat{\mathbf{U}} = \mathbf{r} + \hat{\mathbf{e}} = (001110) + (100000) = (101110)$$

Hamming codes

- Hamming codes
  - Hamming codes are a subclass of linear block codes and belong to the category of *perfect codes*.
  - Hamming codes are expressed as a function of a single integer . $m \geq 2$

Code length :	$n = 2^m - 1$
Number of information bits :	$k = 2^m - m - 1$
Number of parity bits :	$n\text{-}k = m$
Error correction capability :	$t = 1$

  - The columns of the parity check matrix, H, consist of all non-zero binary m-tuples.

Hamming codes

- Example: Systematic Hamming code (7,4)

$$\mathbf{H} = \left[\begin{array}{ccc:cccc} 1 & 0 & 0 & 0 & 1 & 1 & 1 \\ 0 & 1 & 0 & 1 & 0 & 1 & 1 \\ 0 & 0 & 1 & 1 & 1 & 0 & 1 \end{array}\right] = [\mathbf{I}_{3\times3} \;\vdots\; \mathbf{P}^T]$$

$$\mathbf{G} = \left[\begin{array}{ccc:cccc} 0 & 1 & 1 & 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 & 1 & 0 \\ 1 & 1 & 1 & 0 & 0 & 0 & 1 \end{array}\right] = [\mathbf{P} \;\vdots\; \mathbf{I}_{4\times4}]$$

Properties of Block Codes

- Because decoding is generally the difficult problem, most block codes of interest have structure to them.
- A code $C = \left\{\underline{c}_1, \underline{c}_2, \ldots, \underline{c}_{\cancel{2}}^{k}\right\}$ is <u>linear</u> if:

$$\underline{c}_1 \in C, \underline{c}_2 \in C \Rightarrow \underline{c}_1 \oplus \underline{c}_2 \in C \quad ,$$

where $\oplus$ denotes modulo-2 bitwise addition.

- Example: The (7,4) Hamming Code is linear.

$$\underline{x} = (0001) \Rightarrow \underline{y} = (1010001)$$

$$\underline{x} = (0110) \Rightarrow \underline{y} = (0010110)$$

$$\underline{x} = (0111) \Rightarrow \underline{y} = (1000111)$$

Properties of Block Codes (continued)

- A code $C = \left\{\underline{c}_1, \underline{c}_2, \ldots, \underline{c}_{\cancel{2}^k}\right\}$ is <u>systematic</u> if there are k bits of the codeword which correspond directly to information bits.
- Example, the (7,4) Hamming Code is systematic:

$$c_1 = x_1, c_2 = x_2, c_3 = x_3, c_4 = x_4$$

- We can think of the remaining bits as just a fancy system of parity checks:

$$c_5 = x_1 \oplus x_2 \oplus x_3$$

$$c_6 = x_2 \oplus x_3 \oplus x_4$$

$$c_7 = x_1 \oplus x_2 \oplus x_4$$

Properties of Block Codes (continued)

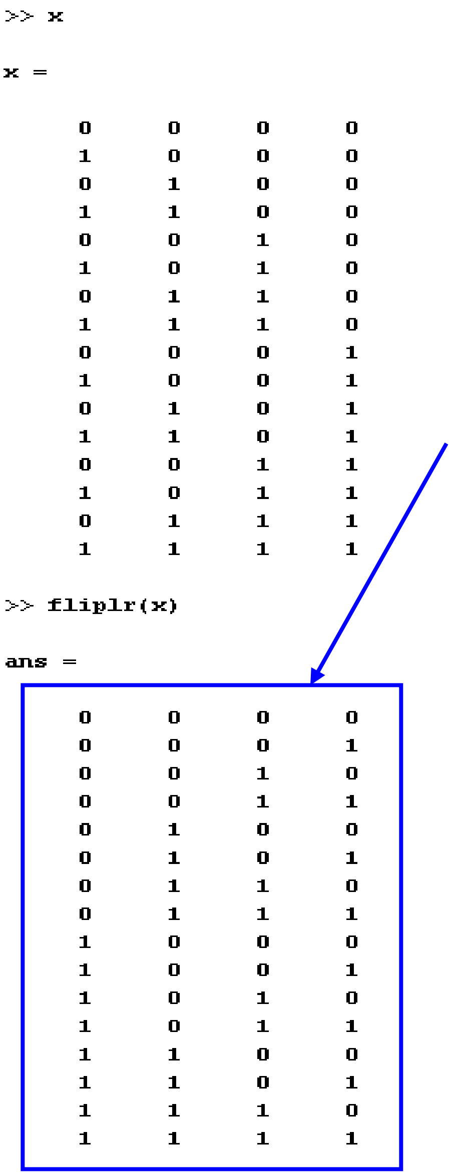
- A code $C = \{\underline{c}_1, \underline{c}_2, \ldots, \underline{c}_{\cancel{2}^k}\}$ is <u>cyclic</u> if:

$$(c_1, c_2, \ldots, c_n) \in C \Rightarrow (c_n, c_1, \ldots, c_{n-1}) \in C$$

- Example: the (7,4) Hamming Code is cyclic.

$$\underline{x} = (0001) \Rightarrow \underline{c} = (1010001)$$
$$\underline{x} = (1000) \Rightarrow \underline{c} = (1101000)$$
$$\underline{x} = (0100) \Rightarrow \underline{c} = (0110100)$$
$$\underline{x} = (1010) \Rightarrow \underline{c} = (0011010)$$
$$\underline{x} = (1101) \Rightarrow \underline{c} = (0001101)$$
$$\underline{x} = (0110) \Rightarrow \underline{c} = (1000110)$$
$$\underline{x} = (0011) \Rightarrow \underline{c} = (0100011)$$

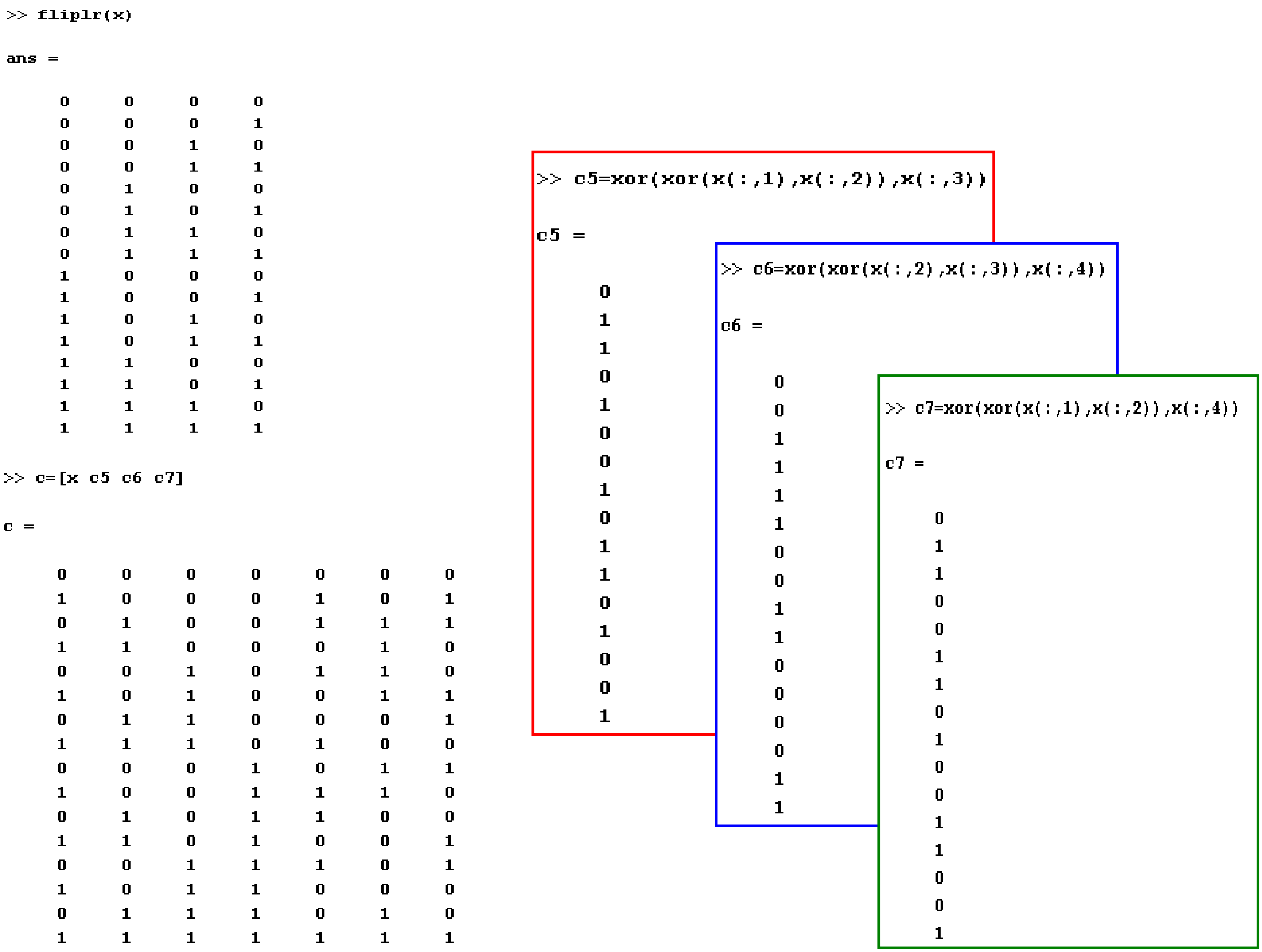
- Most practical block codes are linear and cyclic.



```
>> x

x =

     0     0     0     0
     1     0     0     0
     0     1     0     0
     1     1     0     0
     0     0     1     0
     1     0     1     0
     0     1     1     0
     1     1     1     0
     0     0     0     1
     1     0     0     1
     0     1     0     1
     1     1     0     1
     0     0     1     1
     1     0     1     1
     0     1     1     1
     1     1     1     1

>> fliplr(x)

ans =

     0     0     0     0
     0     0     0     1
     0     0     1     0
     0     0     1     1
     0     1     0     0
     0     1     0     1
     0     1     1     0
     0     1     1     1
     1     0     0     0
     1     0     0     1
     1     0     1     0
     1     0     1     1
     1     1     0     0
     1     1     0     1
     1     1     1     0
     1     1     1     1
```

Input Bits				The output bits						
x_4	x_3	x_2	x_1	c7	c6	c5	c4	c3	c2	c1
0	0	0	0	0	0	0	0	0	0	0
0	0	0	1	1	0	1	0	0	0	1
0	0	1	0	1	1	1	0	0	1	0
0	0	1	1	0	1	0	0	0	1	1
0	1	0	0	0	1	1	0	1	0	0
0	1	0	1	1	1	0	0	1	0	1
0	1	1	0	1	0	0	0	1	1	0
0	1	1	1	0	0	1	0	1	1	1
1	0	0	0	1	1	0	1	0	0	0
1	0	0	1	0	1	1	1	0	0	1
1	0	1	0	0	0	1	1	0	1	0
1	0	1	1	1	0	0	1	0	1	1
1	1	0	0	1	0	1	1	1	0	0
1	1	0	1	0	0	0	1	1	0	1
1	1	1	0	0	1	0	1	1	1	0
1	1	1	1	1	1	1	1	1	1	1

Example Block Code: (7,4) Hamming Code



```
>> fliplr(x)

ans =

     0     0     0     0
     0     0     0     1
     0     0     1     0
     0     0     1     1
     0     1     0     0
     0     1     0     1
     0     1     1     0
     0     1     1     1
     1     0     0     0
     1     0     0     1
     1     0     1     0
     1     0     1     1
     1     1     0     0
     1     1     0     1
     1     1     1     0
     1     1     1     1

>> c=[x c5 c6 c7]

c =

     0     0     0     0     0     0     0
     1     0     0     0     1     0     1
     0     1     0     0     1     1     1
     1     1     0     0     0     1     0
     0     0     1     0     1     1     0
     1     0     1     0     0     1     1
     0     1     1     0     0     0     1
     1     1     1     0     1     0     0
     0     0     0     1     0     1     1
     1     0     0     1     1     1     0
     0     1     0     1     1     0     0
     1     1     0     1     0     0     1
     0     0     1     1     1     0     1
     1     0     1     1     0     0     0
     0     1     1     1     0     1     0
     1     1     1     1     1     1     1
```

```
>> c5=xor(xor(x(:,1),x(:,2)),x(:,3))

c5 =

     0
     1
     1
     0
     1
     0
     0
     1
     0
     1
     1
     0
     1
     0
     0
     1
```

```
>> c6=xor(xor(x(:,2),x(:,3)),x(:,4))

c6 =

     0
     0
     1
     1
     1
     1
     0
     0
     1
     1
     0
     0
     0
     0
     1
     1
```

```
>> c7=xor(xor(x(:,1),x(:,2)),x(:,4))

c7 =

     0
     1
     1
     0
     0
     1
     1
     0
     1
     0
     0
     1
     1
     0
     0
     1
```

Distance Properties of a Block Code

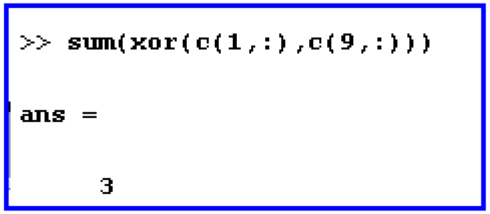
- The Hamming Distance between two codewords $\underline{c}_1 \in C$ and $\underline{c}_2 \in C$ is the number of bits in which they differ:

$$d_H(\underline{c}_1, \underline{c}_2) = \sum_{i=1}^{n} c_{1,i} \oplus c_{2,i}$$

- Example: $d_H((1010001),(1101000)) = 4$
- The minimum distance $d_{H,\min}$ of a code is the smallest distance separating any two codewords:

$$d_{H,\min} = \min_{i \neq j} \left\{ d_H\left(\underline{c}_i, \underline{c}_j\right) \right\}$$

- Example: $d_{H,\min} = 3$ for (7,4) Hamming Code



```
>> sum(xor(c(1,:),c(9,:)))

ans =

     3
```

Decoding of Block Codes

- For a linear error correction code, $d_{H,\min}$ is the smallest <u>weight</u> of any nonzero codeword.
- Just as we wanted to make Euclidean distance large for modulation, we want to make Hamming distance large for our block codes.
- The decoder's job will be to choose the codeword most closely resembling the received sequence of bits.
  - Example: Suppose we transmit: $\underline{c} = (0100011)$ but receive $\underline{y} = (0100001)$ The closest match is: $\underline{\hat{c}} = (0100011)$ so we estimate that our data bits were: $\underline{\hat{x}} = (0011)$

Error Correction Capability

- Any code with minimum distance $d_{H,\min}$, can correct any combination of up to errors.

$$\boxed{t = \left\lfloor \frac{d_{H,\min} - 1}{2} \right\rfloor}$$

- We call t the error correcting capability of the code.
- There is at least one combination of $t+1$ errors which will cause an error.
- Any code with minimum distance $d_{H,\min}$ can detect any combination of up to $d_{H,\min} - 1$ errors by the channel.

Block Code Error Detection and Correction

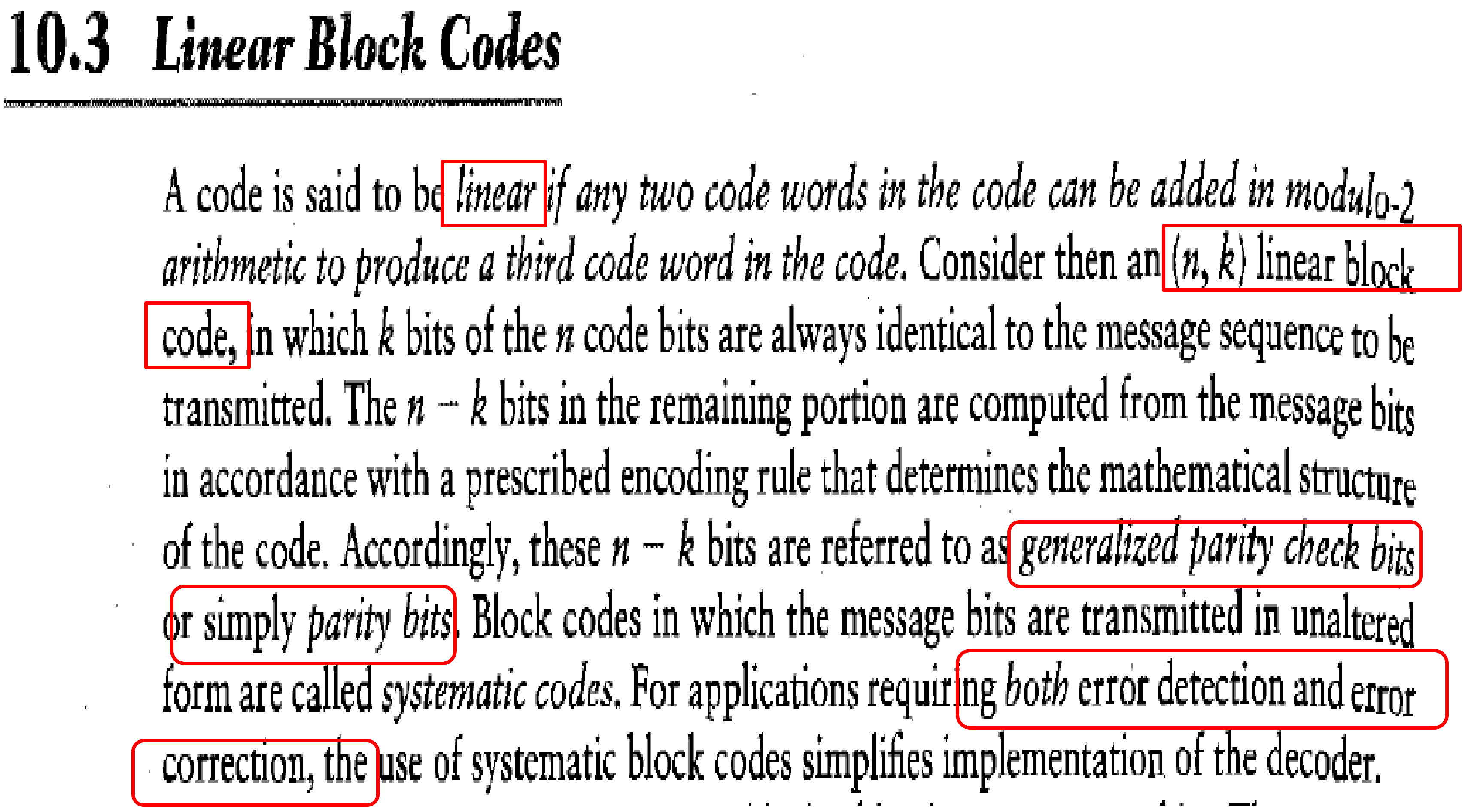
- (6,3) code $2^3 \Rightarrow 2^6$, $d_{min}=3$
- Can detect 2 bit errors, correct 1 bit
  - 110100 sent; 110101 received
- Erasure: Suppose code word 110011 sent but two digits were erased (xx0011), correct code word has smallest Hamming distance

Messag e	Code-word	1	2
000	000000	4	2
100	110100	**1**	3
010	011010	3	2
110	101110	3	3
001	101001	3	2
101	011101	2	3
011	110011	2	**0**
111	000111	3	1

Haykin/Communication Systems, 4th Ed

Linear Block codes

Chapter 10

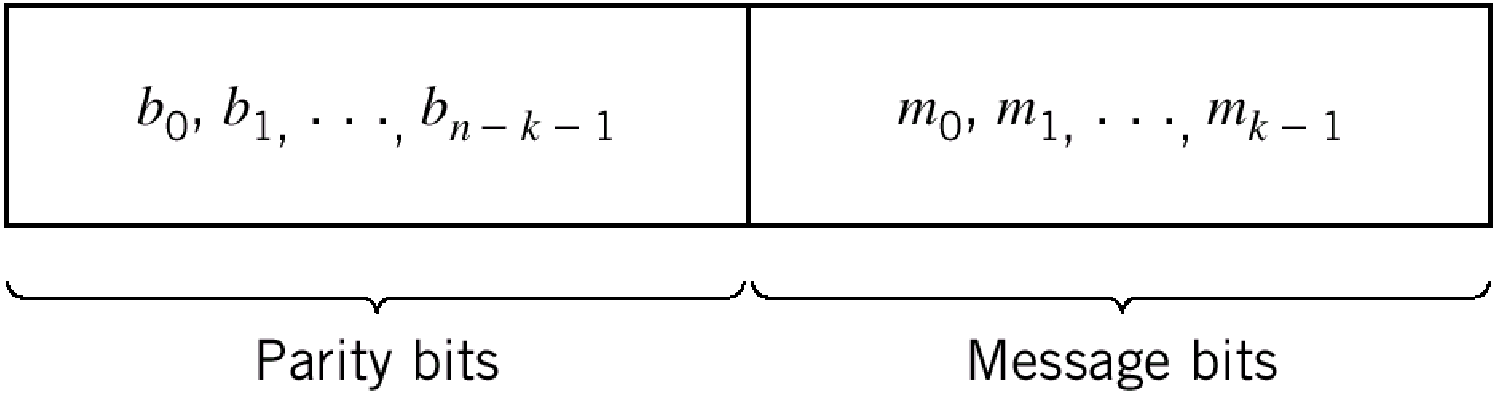


10.3 *Linear Block Codes*

A code is said to be *linear if any two code words in the code can be added in modulo-2 arithmetic to produce a third code word in the code.* Consider then an (n, k) linear block code, in which k bits of the n code bits are always identical to the message sequence to be transmitted. The $n - k$ bits in the remaining portion are computed from the message bits in accordance with a prescribed encoding rule that determines the mathematical structure of the code. Accordingly, these $n - k$ bits are referred to as *generalized parity check bits* or simply *parity bits*. Block codes in which the message bits are transmitted in unaltered form are called *systematic codes*. For applications requiring *both* error detection and error correction, the use of systematic block codes simplifies implementation of the decoder.

Figure 10.4

Structure of systematic code word.



Haykin/Communication Systems, 4th Ed

According to the representation of Figure 10.4, the $(n - k)$ left-most bits of a code word are identical to the corresponding parity bits, and the k right-most bits of the code word are identical to the corresponding message bits. We may therefore write

$$c_i = \begin{cases} b_i, & i = 0, 1, \ldots, n - k - 1 \\ m_{i+k-n}, & i = n - k, n - k + 1, \ldots, n - 1 \end{cases} \tag{10.1}$$

The $(n - k)$ parity bits are *linear sums* of the k message bits, as shown by the generalized relation

$$b_i = p_{0i}m_0 + p_{1i}m_1 + \cdots + p_{k-1,i}m_{k-1} \tag{10.2}$$

where the coefficients are defined as follows:

$$p_{ij} = \begin{cases} 1 & \text{if } b_i \text{ depends on } m_j \\ 0 & \text{otherwise} \end{cases} \tag{10.3}$$

The coefficients p_{ij} are chosen in such a way that the rows of the generator matrix are linearly independent and the parity equations are *unique*.

The system of Equations (10.1) and (10.2) defines the mathematical structure of the (n, k) linear block code. This system of equations may be rewritten in a compact form using matrix notation. To proceed with this reformulation, we define the 1-by-k *message vector*, or *information vector*, $\mathbf{m}$, the 1-by-$(n-k)$ parity vector $\mathbf{b}$, and the 1-by-n code vector $\mathbf{c}$ as follows:

$$\boxed{\begin{aligned} \mathbf{m} &= [m_0, m_1, \ldots, m_{k-1}] \\ \mathbf{b} &= [b_0, b_1, \ldots, b_{n-k-1}] \\ \mathbf{c} &= [c_0, c_1, \ldots, c_{n-1}] \end{aligned}} \qquad \begin{aligned} &(10.4) \\ &(10.5) \\ &(10.6) \end{aligned}$$

Note that all three vectors are *row vectors.* The use of row vectors is adopted in this chapter for the sake of being consistent with the notation commonly used in the coding literature. We may thus rewrite the set of simultaneous equations defining the parity bits in the compact matrix form:

$$\boxed{\mathbf{b} = \mathbf{mP}} \tag{10.7}$$

where $\mathbf{P}$ is the k-by-$(n-k)$ *coefficient matrix* defined by

$$\boxed{\mathbf{P} = \begin{bmatrix} p_{00} & p_{01} & \cdots & p_{0,n-k-1} \\ p_{10} & p_{11} & \cdots & p_{1,n-k-1} \\ \vdots & \vdots & & \vdots \\ p_{k-1,0} & p_{k-1,1} & \cdots & p_{k-1,n-k-1} \end{bmatrix}} \tag{10.8}$$

where p_{ij} is 0 or 1.

From the definitions given in Equations (10.4)–(10.6), we see that c may be expressed as a partitioned row vector in terms of the vectors m and b as follows:

$$\mathbf{c} = [\mathbf{b} \vdots \mathbf{m}] \tag{10.9}$$

Hence, substituting Equation (10.7) into Equation (10.9) and factoring out the common message vector m, we get

$$\boxed{\mathbf{c} = \mathbf{m}[\mathbf{P} : \mathbf{I}_k]} \tag{10.10}$$

where $\mathbf{I}_k$ is the k-by-k *identity matrix*:

$$\boxed{\mathbf{I}_k = \begin{bmatrix} 1 & 0 & \cdots & 0 \\ 0 & 1 & \cdots & 0 \\ \vdots & \vdots & & \vdots \\ 0 & 0 & \cdots & 1 \end{bmatrix}} \tag{10.11}$$

Define the k-by-n *generator matrix*

$$\boxed{\mathbf{G} = [\mathbf{P} \vdots \mathbf{I}_k]} \tag{10.12}$$

The generator matrix G of Equation (10.12) is said to be in the *canonical form* in that its k rows are linearly independent; that is, it is not possible to express any row of the matrix G as a linear combination of the remaining rows. Using the definition of the generator matrix G, we may simplify Equation (10.10) as

$$\mathbf{c} = \mathbf{m}\mathbf{G} \tag{10.13}$$

The full set of code words, referred to simply as *the code*, is generated in accordance with Equation (10.13) by letting the message vector $\mathbf{m}$ range through the set of all 2^k binary k-tuples (1-by-k vectors). Moreover, the sum of any two code words is another code word. This basic property of linear block codes is called *closure*. To prove its validity, consider a pair of code vectors $\mathbf{c}_i$ and $\mathbf{c}_j$ corresponding to a pair of message vectors $\mathbf{m}_i$ and $\mathbf{m}_j$, respectively. Using Equation (10.13) we may express the sum of $\mathbf{c}_i$ and $\mathbf{c}_j$ as

$$\begin{aligned} \mathbf{c}_i + \mathbf{c}_j &= \mathbf{m}_i\mathbf{G} + \mathbf{m}_j\mathbf{G} \\ &= (\mathbf{m}_i + \mathbf{m}_j)\mathbf{G} \end{aligned}$$

The modulo-2 sum of $\mathbf{m}_i$ and $\mathbf{m}_j$ represents a new message vector. Correspondingly, the modulo-2 sum of $\mathbf{c}_i$ and $\mathbf{c}_j$ represents a new code vector.

modulo-2 sum of c_i and c_j, respec.

There is another way of expressing the relationship between the message bits and parity-check bits of a linear block code. Let **H** denote an $(n-k)$-by-n matrix, defined as

$$\mathbf{H} = [\mathbf{I}_{n-k} \vdots \mathbf{P}^T] \tag{10.14}$$

where $\mathbf{P}^T$ is an $(n-k)$-by-k matrix, representing the transpose of the coefficient matrix **P**, and $\mathbf{I}_{n-k}$ is the $(n-k)$-by-$(n-k)$ identity matrix. Accordingly, we may perform the following multiplication of partitioned matrices:

$$\begin{aligned} \mathbf{H}\mathbf{G}^T &= [\mathbf{I}_{n-k} \vdots \mathbf{P}^T]\begin{bmatrix} \mathbf{P}^T \\ \cdots \\ \mathbf{I}_k \end{bmatrix} \\ &= \mathbf{P}^T + \mathbf{P}^T \end{aligned}$$

where we have used the fact that multiplication of a rectangular matrix by an identity matrix of compatible dimensions leaves the matrix unchanged. In modulo-2 arithmetic, we have $\mathbf{P}^T + \mathbf{P}^T = \mathbf{0}$, where $\mathbf{0}$ denotes an $(n-k)$-by-k null matrix (i.e., a matrix that has zeros for all of its elements). Hence,

$$\mathbf{H}\mathbf{G}^T = \mathbf{0} \tag{10.15}$$

Equivalently, we have $\mathbf{G}\mathbf{H}^T = \mathbf{0}$, where $\mathbf{0}$ is a new null matrix. Postmultiplying both sides of Equation (10.13) by $\mathbf{H}^T$, the transpose of **H**, and then using Equation (10.15), we get

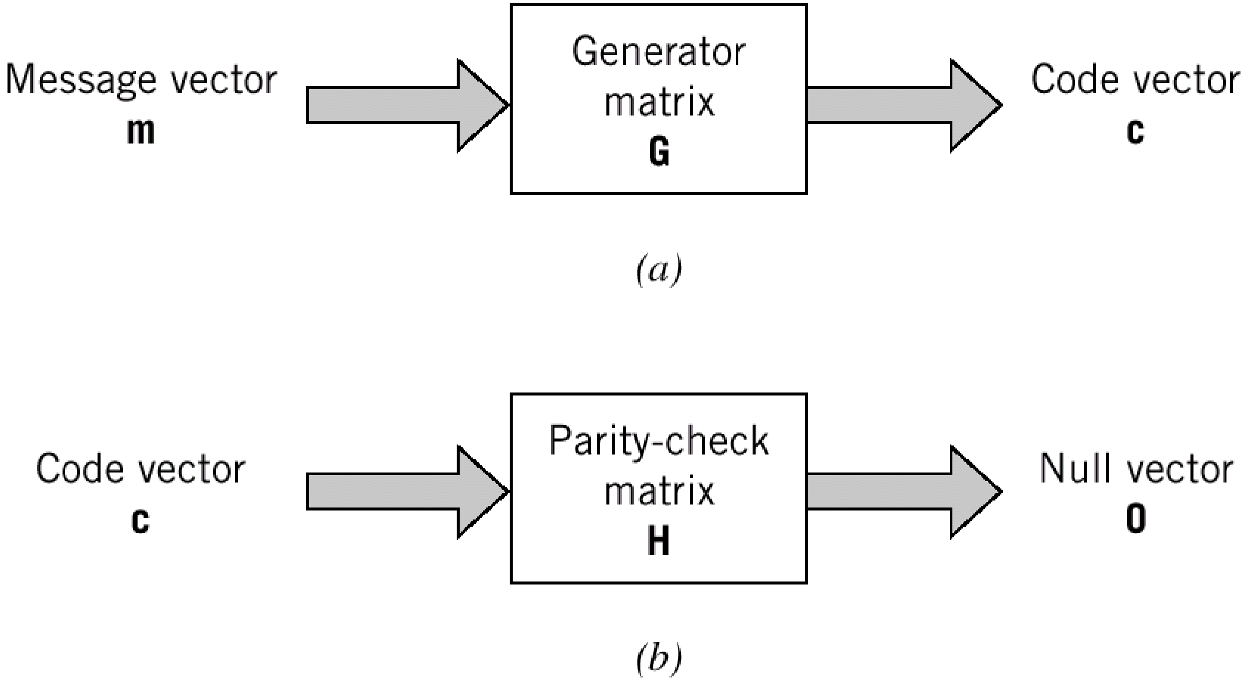
$$\begin{aligned} \mathbf{c}\mathbf{H}^T &= \mathbf{m}\mathbf{G}\mathbf{H}^T \\ &= \mathbf{0} \end{aligned} \tag{10.16}$$

The matrix **H** is called the *parity-check matrix* of the code, and the set of equations specified by Equation (10.16) are called *parity-check equations*.

The generator equation (10.13) and the parity-check detector equation (10.16) are basic to the description and operation of a linear block code. These two equations are depicted in the form of block diagrams in Figure 10.5*a* and 10.5*b*, respectively.

Figure 10.5

Block diagram representations of the generator equation (10.13) and the parity-check equation (10.16).



▶ Example 10.1 Repetition Codes

Repetition codes represent the simplest type of linear block codes. In particular, a single message bit is encoded into a block of n identical bits, producing an $(n, 1)$ block code. Such a code allows provision for a variable amount of redundancy. There are only two code words in the code: an all-zero code word and an all-one code word.

Consider, for example, the case of a repetition code with $k = 1$ and $n = 5$. In this case, we have four parity bits that are the same as the message bit. Hence, the identity matrix $\mathbf{I}_k = 1$, and the coefficient matrix $\mathbf{P}$ consists of a 1-by-4 vector that has 1 for all of its elements. Correspondingly, the generator matrix equals a row vector of all 1s, as shown by

$$\mathbf{G} = \left[\begin{array}{cccc:c} 1 & 1 & 1 & 1 & 1 \end{array}\right]$$

The transpose of the coefficient matrix $\mathbf{P}$, namely, matrix $\mathbf{P}^T$, consists of a 4-by-1 vector that has 1 for all of its elements. The identity matrix $\mathbf{I}_{n-k}$ consists of a 4-by-4 matrix. Hence, the parity-check matrix equals

$$\mathbf{H} = \left[\begin{array}{cccc:c} 1 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 1 \end{array}\right]$$

Since the message vector consists of a single binary symbol, 0 or 1, it follows from Equation (10.13) that there are only two code words: 00000 and 11111 in the (5, 1) repetition code, as expected. Note also that $\mathbf{H}\mathbf{G}^T = 0$, modulo-2, in accordance with Equation (10.15). ◀

Property 1

The syndrome depends only on the error pattern, and not on the transmitted code word.

To prove this property, we first use Equations (10.17) and (10.19), and then Equation (10.16) to obtain

$$\boxed{\begin{aligned} \mathbf{s} &= (\mathbf{c} + \mathbf{e})\mathbf{H}^T \\ &= \mathbf{c}\mathbf{H}^T + \mathbf{e}\mathbf{H}^T \\ &= \mathbf{e}\mathbf{H}^T \end{aligned}} \tag{10.20}$$

Hence, the parity-check matrix H of a code permits us to compute the syndrome s, which depends only upon the error pattern e.

Property 2

All error patterns that differ by a code word have the same syndrome.

For k message bits, there are 2^k distinct code vectors denoted as $\mathbf{c}_i$, $i = 0, 1, \ldots, 2^k - 1$. Correspondingly, for any error pattern $\mathbf{e}$, we define the 2^k distinct vectors $\mathbf{e}_i$ as

$$\mathbf{e}_i = \mathbf{e} + \mathbf{c}_i, \qquad i = 0, 1, \ldots, 2^k - 1 \tag{10.21}$$

The set of vectors $\{\mathbf{e}_i, i = 0, 1, \ldots, 2^k - 1\}$ so defined is called a *coset* of the code. In other words, a coset has exactly 2^k elements that differ at most by a code vector. Thus, an (n, k) linear block code has 2^{n-k} possible cosets. In any event, multiplying both sides of Equation (10.21) by the matrix $\mathbf{H}^T$, we get

$$\boxed{\begin{aligned} \mathbf{e}_i\mathbf{H}^T &= \mathbf{e}\mathbf{H}^T + \mathbf{c}_i\mathbf{H}^T \\ &= \mathbf{e}\mathbf{H}^T \end{aligned}} \tag{10.22}$$

which is independent of the index i. Accordingly, we may state that each coset of the code is characterized by a unique syndrome.

We may put Properties 1 and 2 in perspective by expanding Equation (10.20). Specifically, with the matrix **H** having the systematic form given in Equation (10.14), where the matrix **P** is itself defined by Equation (10.8), we find from Equation (10.20) that the $(n - k)$ elements of the syndrome **s** are linear combinations of the n elements of the error pattern **e**, as shown by

$$\begin{aligned}
s_0 &= e_0 + e_{n-k}p_{00} + e_{n-k+1}p_{10} + \cdots + e_{n-1}p_{k-1,0} \\
s_1 &= e_1 + e_{n-k}p_{01} + e_{n-k+1}p_{11} + \cdots + e_{n-1}p_{k-1,1} \\
&\vdots \\
s_{n-k-1} &= e_{n-k-1} + e_{n-k}p_{0,n-k-1} + \cdots + e_{n-1}p_{k-1,n-k-1}
\end{aligned} \tag{10.23}$$

This set of $(n - k)$ linear equations clearly shows that the syndrome contains information about the error pattern and may therefore be used for error detection. However, it should be noted that the set of equations is *underdetermined* in that we have more unknowns than equations. Accordingly, there is *no* unique solution for the error pattern. Rather, there are 2^n error patterns that satisfy Equation (10.23) and therefore result in the same syndrome, in accordance with Property 2 and Equation (10.22). In particular, with 2^{n-k} possible syndrome vectors, the information contained in the syndrome **s** about the error pattern **e** is *not* enough for the decoder to compute the exact value of the transmitted code vector. Nevertheless, knowledge of the syndrome **s** reduces the search for the true error pattern **e** from 2^n to 2^{n-k} possibilities. Given these possibilities, the decoder has the task of making the best selection from the cosets corresponding to **s**.

MINIMUM DISTANCE CONSIDERATIONS

Consider a pair of code vectors $\mathbf{c}_1$ and $\mathbf{c}_2$ that have the same number of elements. The *Hamming distance* $d(\mathbf{c}_1, \mathbf{c}_2)$ between such a pair of code vectors is defined as the number of locations in which their respective elements differ.

The *Hamming weight* $w(\mathbf{c})$ of a code vector $\mathbf{c}$ is defined as the number of nonzero elements in the code vector. Equivalently, we may state that the Hamming weight of a code vector is the distance between the code vector and the all-zero code vector.

The *minimum distance* $d_{\min}$ of a linear block code is defined as the smallest Hamming distance between any pair of code vectors in the code. That is, the minimum distance is the same as the smallest Hamming weight of the difference between any pair of code vectors. From the closure property of linear block codes, the sum (or difference) of two code vectors is another code vector. Accordingly, we may state that *the minimum distance of a linear block code is the smallest Hamming weight of the nonzero code vectors in the code.*

The minimum distance $d_{\min}$ is related to the structure of the parity-check matrix $\mathbf{H}$ of the code in a fundamental way. From Equation (10.16) we know that a linear block code is defined by the set of all code vectors for which $\mathbf{c}\mathbf{H}^T = \mathbf{0}$, where $\mathbf{H}^T$ is the transpose of the parity-check matrix $\mathbf{H}$. Let the matrix $\mathbf{H}$ be expressed in terms of its columns as follows:

$$\mathbf{H} = [\mathbf{h}_1, \mathbf{h}_2, \ldots, \mathbf{h}_n] \tag{10.24}$$

of the parity-check matrix **H**. Let the matrix **H** be expressed in terms of its columns as follows:

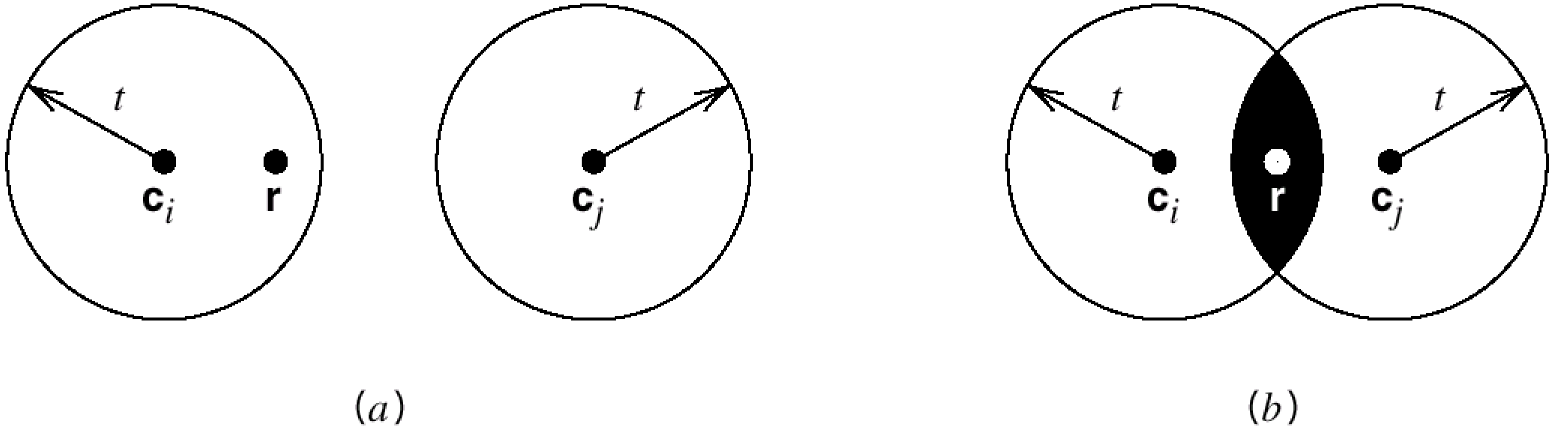
$$\mathbf{H} = [\mathbf{h}_1, \mathbf{h}_2, \ldots, \mathbf{h}_n] \tag{10.24}$$

Then, for a code vector **c** to satisfy the condition $\mathbf{c}\mathbf{H}^T = \mathbf{0}$, the vector **c** must have 1s in such positions that the corresponding rows of $\mathbf{H}^T$ sum to the zero vector **0**. However, by definition, the number of 1s in a code vector is the Hamming weight of the code vector. Moreover, the smallest Hamming weight of the nonzero code vectors in a linear block code equals the minimum distance of the code. Hence, *the minimum distance of a linear block code is defined by the minimum number of rows of the matrix* $\mathbf{H}^T$ whose sum is equal to the zero vector.

The minimum distance of a linear block code, $d_{\min}$, is an important parameter of the code. Specifically, it determines the error-correcting capability of the code. Suppose an (n, k) linear block code is required to detect and correct all error patterns (over a binary symmetric channel), and whose Hamming weight is less than or equal to t. That is, if a code vector $\mathbf{c}_i$ in the code is transmitted and the received vector is $\mathbf{r} = \mathbf{c}_i + \mathbf{e}$, we require that the decoder output $\hat{\mathbf{c}} = \mathbf{c}_i$, whenever the error pattern $\mathbf{e}$ has a Hamming weight $w(\mathbf{e}) \leq t$. We assume that the 2^k code vectors in the code are transmitted with equal probability. The best strategy for the decoder then is to pick the code vector closest to the received vector $\mathbf{r}$, that is, the one for which the Hamming distance $d(\mathbf{c}_i, \mathbf{r})$ is the smallest. With such a strategy, the decoder will be able to detect and correct all error patterns of Hamming weight $w(\mathbf{e}) \leq t$, provided that the minimum distance of the code is equal to or greater than $2t + 1$. We may demonstrate the validity of this requirement by adopting a geometric interpretation of the problem. In particular, the 1-by-n code vectors and the 1-by-n received vector are represented as points in an n-dimensional space. Suppose that we construct two spheres, each of radius t, around the points that represent code vectors $\mathbf{c}_i$ and $\mathbf{c}_j$. Let these two spheres be disjoint, as depicted in Figure 10.6*a*. For this condition to be satisfied, we require that $d(\mathbf{c}_i, \mathbf{c}_j) \geq 2t + 1$. If then the code vector $\mathbf{c}_i$ is transmitted and the Hamming distance $d(\mathbf{c}_i, \mathbf{r}) \leq t$, it is clear that the decoder will pick $\mathbf{c}_i$ as it is the

Figure 10.6

(*a*) Hamming distance $d(\mathbf{c}_i, \mathbf{c}_j) \geq 2t + 1$. (*b*) Hamming distance $d(\mathbf{c}_i, \mathbf{c}_j) < 2t$. The received vector is denoted by **r**.



Haykin/Communication Systems, 4th Ed

code vector closest to the received vector **r**. If, on the other hand, the Hamming distance $d(\mathbf{c}_i, \mathbf{c}_j) \leq 2t$, the two spheres around $\mathbf{c}_i$ and $\mathbf{c}_j$ intersect, as depicted in Figure 10.6*b*. Here we see that if $\mathbf{c}_i$ is transmitted, there exists a received vector **r** such that the Hamming distance $d(\mathbf{c}_i, \mathbf{r}) \leq t$, and yet **r** is as close to $\mathbf{c}_j$ as it is to $\mathbf{c}_i$. Clearly, there is now the possibility of the decoder picking the vector $\mathbf{c}_j$, which is wrong. We thus conclude that *an (n, k) linear block code has the power to correct all error patterns of weight t or less if, and only if,*

$$d(\mathbf{c}_i, \mathbf{c}_j) \geq 2t + 1 \qquad \text{for all } \mathbf{c}_i \text{ and } \mathbf{c}_j$$

By definition, however, the smallest distance between any pair of code vectors in a code is the minimum distance of the code, $d_{\min}$. We may therefore state that *an (n, k) linear block code of minimum distance $d_{\min}$ can correct up to t errors if, and only if,*

$$t \leq \left\lfloor \tfrac{1}{2}(d_{\min} - 1) \right\rfloor \tag{10.25}$$

where $\lfloor \ \rfloor$ *denotes the largest integer* less than or equal to the enclosed quantity. Equation (10.25) gives the error-correcting capability of a linear block code a quantitative meaning.

Syndrome Decoding

We are now ready to describe a syndrome-based decoding scheme for linear block codes. Let $\mathbf{c}_1, \mathbf{c}_2, \dots, \mathbf{c}_{2^k}$ denote the 2^k code vectors of an (n, k) linear block code. Let $\mathbf{r}$ denote the received vector, which may have one of 2^n possible values. The receiver has the task of partitioning the 2^n possible received vectors into 2^k disjoint subsets $\mathscr{D}_1, \mathscr{D}_2, \dots, \mathscr{D}_{2^k}$ in such a way that the *i*th subset $\mathscr{D}_i$ corresponds to code vector $\mathbf{c}_i$ for $1 \leq i \leq 2^k$. The received vector $\mathbf{r}$ is decoded into $\mathbf{c}_i$ if it is in the *i*th subset. For the decoding to be correct, $\mathbf{r}$ must be in the subset that belongs to the code vector $\mathbf{c}_i$ that was actually sent.

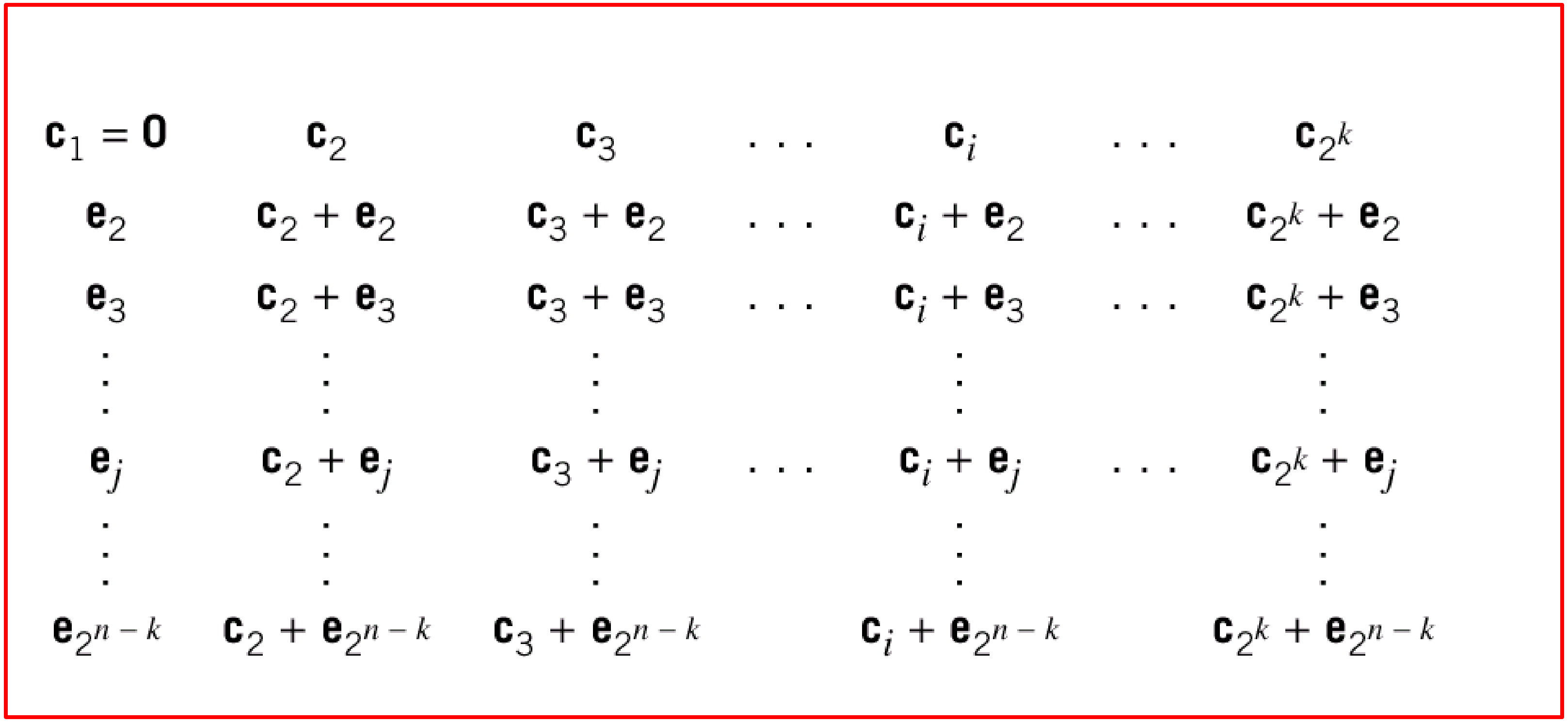
The 2^k subsets described herein constitute a *standard array* of the linear block code. To construct it, we may exploit the linear structure of the code by proceeding as follows:

1. The 2^k code vectors are placed in a row with the all-zero code vector $\mathbf{c}_1$ as the left-most element.
2. An error pattern $\mathbf{e}_2$ is picked and placed under $\mathbf{c}_1$, and a second row is formed by adding $\mathbf{e}_2$ to each of the remaining code vectors in the first row; it is important that the error pattern chosen as the first element in a row not have previously appeared in the standard array.
3. Step 2 is repeated until all the possible error patterns have been accounted for.

Figure 10.7 illustrates the structure of the standard array so constructed. The 2^k columns of this array represent the disjoint subsets $\mathscr{D}_1, \mathscr{D}_2, \dots, \mathscr{D}_{2^k}$. The 2^{n-k} rows of the array

Figure 10.7

Standard array for an (*n*, *k*) block code.



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

Haykin/Communication Systems, 4th Ed

represent the cosets of the code, and their first elements $\mathbf{e}_2, \ldots, \mathbf{e}_{2^{n-k}}$ are called *coset leaders.*

For a given channel, the probability of decoding error is minimized when the most likely error patterns (i.e., those with the largest probability of occurrence) are chosen as the coset leaders. In the case of a binary symmetric channel, the smaller the Hamming weight of an error pattern the more likely it is to occur. Accordingly, the standard array should be constructed with each coset leader having the minimum Hamming weight in its coset.

We may now describe a decoding procedure for a linear block code:

1. For the received vector $\mathbf{r}$, compute the syndrome $\mathbf{s} = \mathbf{r}\mathbf{H}^T$.
2. Within the coset characterized by the syndrome $\mathbf{s}$, identify the coset leader (i.e., the error pattern with the largest probability of occurrence); call it $\mathbf{e}_0$.
3. Compute the code vector

$$\boxed{\mathbf{c} = \mathbf{r} + \mathbf{e}_0} \tag{10.26}$$

as the decoded version of the received vector $\mathbf{r}$.

This procedure is called *syndrome decoding.*

EXAMPLE 10.2 Hamming Codes[4]

Consider a family of (n, k) linear block codes that have the following parameters:

Block length:	$n = 2^m - 1$
Number of message bits:	$k = 2^m - m - 1$
Number of parity bits:	$n - k = m$

where $m \geq 3$. These are the so-called Hamming codes.

Consider, for example, the (7, 4) Hamming code with $n = 7$ and $k = 4$, corresponding to $m = 3$. The generator matrix of the code must have a structure that conforms to Equation (10.12). The following matrix represents an appropriate generator matrix for the (7, 4) Hamming code:

$$\mathbf{G} = \left[\; \underbrace{\begin{matrix} 1 & 1 & 0 \\ 0 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 0 & 1 \end{matrix}}_{\mathbf{P}} \;\vdots\; \underbrace{\begin{matrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{matrix}}_{\mathbf{I}_k} \;\right]$$

TABLE 10.1 *Code words of a (7, 4) Hamming code*

Message Word	Code Word	Weight of Code Word	*Message Word*	Code Word	Weight of Code Word
0000	0000000	0	1000	1101000	3
0001	1010001	3	1001	0111001	4
0010	1110010	4	1010	0011010	3
0011	0100011	3	1011	1001011	4
0100	0110100	3	1100	1011100	4
0101	1100101	4	1101	0001101	3
0110	1000110	3	1110	0101110	4
0111	0010111	4	1111	1111111	7

The corresponding parity-check matrix is given by

$$\mathbf{H} = \left[\begin{array}{ccc:cccc} 1 & 0 & 0 & 1 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 & 1 & 1 & 0 \\ \underbrace{0 \quad 0 \quad 1}_{\mathbf{I}_{n-k}} & & & \underbrace{0 \quad 1 \quad 1 \quad 1}_{\mathbf{P}^T} & & & \end{array}\right]$$

With $k = 4$, there are $2^k = 16$ distinct message words, which are listed in Table 10.1. For a given message word, the corresponding code word is obtained by using Equation (10.13). Thus, the application of this equation results in the 16 code words listed in Table 10.1.

In Table 10.1, we have also listed the Hamming weights of the individual code words in the (7, 4) Hamming code. Since the smallest of the Hamming weights for the nonzero code words is 3, it follows that the minimum distance of the code is 3. Indeed, Hamming codes have the property that the minimum distance $d_{\min} = 3$, independent of the value assigned to the number of parity bits m.

To illustrate the relation between the minimum distance $d_{\min}$ and the structure of the parity-check matrix H, consider the code word 0110100. In the matrix multiplication defined by Equation (10.16), the nonzero elements of this code word "sift" out the second, third, and fifth columns of the matrix H yielding

$$\begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} + \begin{bmatrix} 0 \\ 1 \\ 1 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}.$$

We may perform similar calculations for the remaining 14 nonzero code words. We thus find that the smallest number of columns in H that sums to zero is 3, confirming the earlier statement that $d_{\min} = 3$.

An important property of Hamming codes is that they satisfy the condition of Equation (10.25) with the equality sign, assuming that $t = 1$. This means that Hamming codes are *single-error correcting binary perfect codes.*

Assuming single-error patterns, we may formulate the seven coset leaders listed in the right-hand column of Table 10.2. The corresponding 2^3 syndromes, listed in the left-hand column, are calculated in accordance with Equation (10.20). The zero syndrome signifies no transmission errors.

Suppose, for example, the code vector [1110010] is sent, and the received vector is

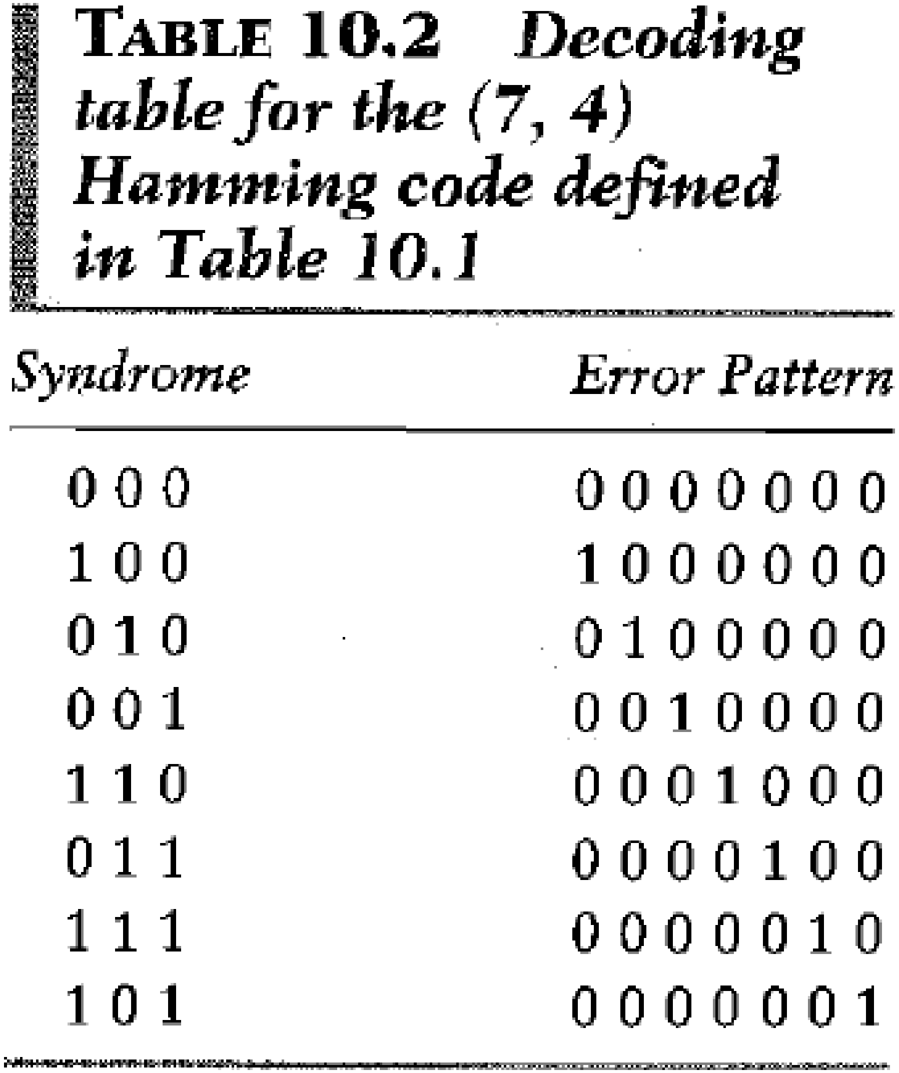


TABLE 10.2 ***Decoding table for the (7, 4) Hamming code defined in Table 10.1***

Syndrome	*Error Pattern*
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

[1100010] with an error in the third bit. Using Equation (10.19), the syndrome is calculated to be

$$\mathbf{s} = [1100010]\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 1 & 0 \\ 0 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 0 & 1 \end{bmatrix}$$

$$= [0 \quad 0 \quad 1]$$

From Table 10.2 the corresponding coset leader (i.e., error pattern with the highest probability of occurrence) is found to be [0010000], indicating correctly that the third bit of the received vector is erroneous. Thus, adding this error pattern to the received vector, in accordance with Equation (10.26), yields the correct code vector actually sent.